Volume

3

DOE-2.2

Building Energy Use and Cost Analysis Program

Volume 3: Topics

July 2006

LAWRENCE BERKELEY NATIONAL LABORATORY
JAMES J. HIRSCH & ASSOCIATES

DOE-2.2 BUILDING ENERGY USE AND COST ANALYSIS PROGRAM

Volume 3: Topics

E. O. Lawrence Berkeley National Laboratory
Simulation Research Group
Berkeley, California 94720

James J. Hirsch & Associates
12185 Presilla Road.
Camarillo, CA 93012-9243
Phone 805.553.9000 • Fax 805.532.2401

Acknowledgements

DOE-2.2, both the program and its documentation, are based upon earlier versions of DOE-2. The DOE-2 family of programs was created primarily through a partnership between James J. Hirsch & Associates (JJH) and Lawrence Berkeley National Laboratory (LBNL) with additional contributions, over a twenty five year period, from a large number of individuals and institutions around the world. Support for the continued development of DOE-2, over its two decades of wide distribution, has come from many public and private agencies, companies and educational institutions around the world. The primary support for DOE-2 development, however, has come from public funds provided by the United States Department of Energy (USDOE) and the United States electric and gas utility industry; particularly the USDOE Office of Energy Efficiency and Renewable Energy Building Technologies Program, Southern California Edison Company's Energy Efficiency Division, and the Electric Power Research Institute's Customer Systems Division.

Authorship of the DOE-2.2 program components and documentation is an ongoing team effort that has its roots in previous versions going back over twenty-five years and we expect will continue into future decades. The contributions to DOE-2, both directly as authors and indirectly in the form of advice, comment and testing or feedback, are too numerous to catalog here; however, the primary authors are mentioned below in alphabetical order. Currently, and over the past decade, Marlin Addison, Scott Criswell, Steve Gates, Jeff Hirsch, and Kevin Madison, as consulting staff for JJH, are the major contributors to DOE-2.2. Fred Buhl, Ender Erdem, Kathy Ellington and Fred Winkelmann, as staff members of the Environmental Energy Technologies Division's Simulation Research Group at LBNL, were major contributors to the initial version of DOE-2.2. The primary contributors to the previous versions of DOE-2 (2.1E, 2.1D, 2.1C, etc) were Fred Buhl, Ender Erdem, Kathy Ellington, Steve Gates, Jeff Hirsch and Fred Winkelmann, as LBNL staff and Steve Gates and Jeff Hirsch as consulting staff for JJH.

The authors of DOE-2.2 also wish to acknowledge many persons who, apart from the financial support provided by their organizations, have provided vision and insight that has been instrumental to the ongoing support of the DOE-2 family of products, including DOE-2.1, DOE-2.2, PowerDOE and eQUEST. In particular we express our thanks to Gregg Ander, and his staff, and Janith Johnson, and her staff, at Southern California Edison Company.

Table of Contents

Section

1

Overview and Converting From 2.1E

This manual describes how to use the program to model the wide range of components and systems that are found in residential and commercial buildings. It also discusses a variety of special analysis features of the program, such as "Input Functions" and "Parametric Runs." Unlike the *DOE-2.2 Dictionary*, which is meant to be used with this manual and which describes individual input commands and associated keywords, this manual is organized by "topics." Some topics, such as "Daylighting" or "Circulation Loops," often span a number of different commands and, in some cases like "Sunspaces," invoke concepts from both the LOADS and HVAC parts of the program.

For users familiar with the concepts of the previous version of DOE-2, version 2.1, the first section of this manual will discuss converting BDL input from 2.1E for use in DOE-2.2. Since DOE-2.2 is under continuous development, this first section will only describe the basic changes relating to the initial release of DOE-2.2; it is expected that added changes will be needed to fully convert to use the current features of DOE-2..2 at the time the conversion is made.

CONVERTING FROM DOE-2.1E

Using existing DOE-2.1E input files (BDL files) with this program requires some restructuring of the BDL input. Once you have made these conversions, you can use the new input file in exactly the same way as with DOE-2.1E. That is, you can run the program and generate similar standard reports. You can use parametric runs, macros and input functions as with DOE-2.1E.

Combined SYSTEMS and PLANT

The most significant difference between DOE-2.1E and this program is the combination of the old SYSTEM and PLANT programs into a new, combined HVAC simulation program. One reason for this change is to improve the connectivity between the loads incurred by the secondary HVAC systems (air handler coils, reheat coils, etc.) and the primary HVAC equipment (boilers, chillers, etc.) This is accomplished using *circulation loops*, which are defined using the new CIRCULATION-LOOP command.

INPUT, END, COMPUTE, STOP

Because SYSTEMS and PLANT have been combined, the old DOE-2.1E commands INPUT LOADS, INPUT SYSTEMS, INPUT PLANT and INPUT ECONOMICS; and COMPUTE LOADS, COMPUTE SYSTEMS, COMPUTE PLANT and COMPUTE ECONOMICS are no longer necessary or permitted in this program. The input/compute/stop/end sequence is now:

```
INPUT  ..

  LOADS, HVAC and ECONOMICS input

END ..

COMPUTE ..

STOP ..
```

INPUT ECONOMICS and COMPUTE ECONOMICS can still be used; they are only useful for multiple economics calculations or for parametric runs. See "Parametric Runs" in this manual.

Multiple SYSTEM and PLANT Simulations

In previous versions of the program it was possible to do a LOADS simulation followed by multiple SYSTEMS and PLANT simulations, or a LOADS and SYSTEMS simulation followed by multiple PLANT simulations. This is no longer possible now, since the LOADS, SYSTEMS, and PLANT inputs are no longer separate. But it is still possible to do a LOADS and HVAC simulation followed by multiple ECONOMICS calculations. See "Parametric Runs" in this manual.

Libraries

The program has two libraries: the *standard library* and the *user library*. The standard library is provided with the program. It contains entries for equipment performance curves, wall/roof/floor materials, wall/roof/floor constructions, window glass layers, window gap layers, window blind layers, window layer assemblies, luminaires, and lamps. Unlike the DOE-2.1E wall/roof/floor construction library, which contains precalculated response factors in binary form, the construction library contains only the basic layer properties in ASCII form; the corresponding response factors are recalculated each time a run is made.

U-Names

U-names of up to 256 characters were previously permitted; however, only the first 16 characters were significant. Now, U-names can be up to 32 characters long and all 32 characters are significant. U-names may now include embedded blanks, provided that the U-name begins and ends with double quotes. For example, Heating-Loop and "Heating Loop" are both legal U-names.

Schedules

The format of *all* schedules has changed. All schedules must have TYPE as a keyword and the type specified must be compatible with the keyword that later references that schedule. The schedule TYPE must be specified in the DAY-SCHEDULE, WEEK-SCHEDULE and SCHEDULE commands. The TYPE keyword must be the *first* keyword, other than LIKE, in these commands. If a SCHEDULE, WEEK-SCHEDULE or DAY-SCHEDULE is LIKE'd to another, the other must be of the same TYPE. As in DOE-2.1E, it is possible to define an entire annual schedule using only the SCHEDULE command (i.e., as a "nested" schedule).

Use the following conversion as a guideline:

DOE-2.1E version:

```
SHADE-MULT = SCHEDULE
   THRU DEC 31        (ALL) (1,24) (.5)  ..
```

Converted:

```
SHADE-MULT = SCHEDULE
   TYPE               = MULTIPLIER
   THRU DEC 31          (ALL) (1,24) (.5) ..
```

OR

```
SHADE-DAY = DAY-SCHEDULE
   TYPE               = MULTIPLIER
                        (1,24) (.5) ..

SHADE-WEEK = WEEK-SCHEDULE
   TYPE               = MULTIPLIER
                        (ALL) SHADE-DAY ..

SHADE-MULT = SCHEDULE
   TYPE               = MULTIPLIER
   THRU DEC 31          SHADE-WEEK  ..
```

OR

```
SHADE-DAY = DAY-SCHEDULE
   TYPE               = MULTIPLIER (1,24) (.5) ..

SHADE-MULT = SCHEDULE
   TYPE               = MULTIPLIER
   THRU DEC 31          (ALL) SHADE-DAY ..
```

Note that the WEEK-SCHEDULE command now allows for *ten-day* weeks: the days are now the eight original days (i.e., Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday and Holiday) plus heating and cooling design days. The code-words HDD and CDD specify the heating and cooling design days, respectively. These design days default to the day schedule specified for Monday if not explicitly assigned. When DESIGN-DAYs are

defined, they are run for the single month and day specified within the DESIGN-DAY command and all schedules use the HDD or CDD day schedule based upon the TYPE of the DESIGN-DAY being run.

The allowed TYPEs for each keyword that accept a schedule are listed below

Table 1 Allowable types for each schedule

Schedule	Allowable Types	Schedule	Allowable Types
Loads Schedules			
SHADE-SCHEDULE	Fraction or Multiplier	SHADING-SCHEDULE	Fraction or Multiplier
PEOPLE-SCHEDULE	Fraction or Multiplier	CONDUCT-SCHEDULE	Fraction or Multiplier
LIGHTING-SCHEDUL	Fraction or Multiplier	MAX-SOLAR-SCH	Radiation
TASK-LIGHT-SCH	Fraction or Multiplier	VIS-TRANS-SCH	Fraction or Multiplier
EQUIP-SCHEDULE	Fraction or Multiplier	CONDUCT-TMIN-SCH	Temperature
SOURCE-SCHEDULE	Fraction or Multiplier	OPEN-SHADE-SCH	Fraction or Multiplier
INF-SCHEDULE	Fraction or Multiplier	SOL-TRANS-SCH	Fraction or Multiplier
DAYLIGHT-REP-SCH	On/Off	SWITCH-SCH	On/Off
PEOPLE-SCHEDULE	Fraction or Multiplier	SLAT-SCHEDULE	Exp-fraction or Fraction
LIGHTING-SCHEDULE	Fraction or Multiplier	SLAT-TRIGG-SCH	Temperature or Radiation
TASK-LIGHT-SCH	Fraction or Multiplier	BLIND-SCHEDULE	Fraction or Multiplier
EQUIP-SCHEDULE	Fraction or Multiplier	BLIND-TRIGG-SCH	Temperature or Radiation
SOURCE-SCHEDULE	Fraction or Multiplier	REPORT-SCHEDULE	On/Off
DAYLIGHT-REP-SCH	On/Off	SHADE-SCHEDULE	Fraction or Multiplier
HVAC Schedules			
HEATING-SCHEDULE	On/Off/Temp or On/Off or Temperature	TROM-VENT-SCH	On/Off
HEAT-RESET-SCH	Reset Temp	REFG-SENS-SCH	Fraction or Multiplier
HEAT-SET-SCH	Temperature	REFG-LAT-SCH	Fraction or Multiplier
COOLING-SCHEDULE	On/Off/Temp or On/Off or Temperature	REFG-AUX-SCH	Fraction or Multiplier or On/Off
COOL-RESET-SCH	Reset Temp	HMIN-FLOW-SCH	Fraction or Frac/Design
COOL-SET-SCH	Temperature	CMIN-FLOW-SCH	Fraction or Frac/Design
BASEBOARD-SCH	Reset Ratio	HEATING-SCHEDULE	On/Off
MIN-AIR-SCH	Fraction or Frac/Design	COOLING-SCHEDULE	On/Off
VENT-TEMP-SCH	Temperature	PUMP-SCHEDULE	On/Off
NATURAL-VENT-SCH	On/Off or On/Off/Flag	HEAT-RESET-SCH	Reset Temp
OPEN-VENT-SCH	Fraction	HEAT-SETPT-SCH	Temperature
FAN-SCHEDULE	On/Off/Flag or On/Off	COOL-RESET-SCH	Reset Temp
NIGHT-VENT-SCH	On/Off	COOL-SETPT-SCH	Temperature
EXHAUST-FAN-SCH	On/Off	LOOP-TUNNEL-SCH	Temperature (new)
HFAN-SCHEDULE	On/Off or On/Off/Flag	PROCESS-LOAD-SCH	Fraction or Multiplier
INDUC-MODE-SCH	On/Off	LOOP-AUX-SCH	Fraction or Multiplier
MIN-SUPPLY-SCH	Temperature	DHW-INLET-T-SCH	Temperature
EVAP-PCC-SCH	On/Off or On/Off/Temp or On/Off/Flag	AUX-SCHEDULE	Fraction or Multiplier
RECOV-SCH	On/Off or On/Off/Temp or On/Off/Flag	ASSIGN-SCH-1	Flag
MIN-FLOW-SCH	Fraction or Frac/Design	ASSIGN-SCH-2	Flag
SS-FLOW-T-SCH	Temperature	ASSIGN-SCH-3	Flag
SS-FLOW-SCH	Fraction	ASSIGN-SCH-4	Flag
SS-VENT-T-SCH	Temperature	ASSIGN-SCH-5	Flag
SS-VENT-SCH	On/Off	INTERIOR-SCH	Fraction or Multiplier (new)
HEAT-TEMP-SCH	Temperature	EXTERIOR-SCH	Fraction or Multiplier (new)
COOL-TEMP-SCH	Temperature		
ZONE-FAN-T-SCH	Temperature		
Economics Schedules			
QUAL-SCH	Flag		
ENERGY-CHG-SCH	Fraction or Multiplier		
ENERGY-ADJ-SCH	Fraction or Multiplier		
BLOCK-SCH	Flag		
RATCHET-SCH	Flag		

Keyword Defaulting

Previously, all keywords had a single, or fixed, default value. The SYSTEM command allowed keyword defaults to vary by system type, but all other commands had a single default for each keyword.

Now, the majority of keywords do not have fixed defaults, but instead have defaults defined by *Keyword Expressions*. An expression uses language constructs similar to C++ or FORTRAN to define the default value of a keyword as a function of other keywords. The keyword relationships may be either within the same component, or may relate across two or more components. For example, a zone's THERMOSTAT-TYPE now defaults according to the type of SYSTEM that serves the zone. This type of defaulting can be described as *dynamic defaulting*. Refer to "Keyword Expressions" for more information.

Because the majority of keywords now default dynamically, it is not possible to present tables of keyword defaults in these documents; there are simply too many possible combinations of defaults. Therefore, if you are using DOE-2 in the batch mode, we *strongly* recommend that you specify DIAGNOSTIC COMMENTS at the beginning of your input. Doing so will cause all of the active keywords to display, together with their value, whether input by you, fixed in BDL, or calculated by an expression. All inactive keywords (keywords not needed in your model), will be suppressed.

The eQUEST and PowerDOE graphical user interfaces for DOE-2 overcome this problem by interactively communicating with BDL, and retrieving the dynamic default for each keyword. This allows you to review the default is for each keyword in question. These interfaces also indicate which keywords are unused, which are required, etc. For this reason, we recommend that you use one of these programs when assembling your input.

LOADS

Key concepts in Loads remain the same, but there are a number of command and keyword changes.

Run Period

Unlike DOE-2.1E, RUN-PERIODs are not specified for DESIGN-DAYs. Rather, the DESIGN-DAY command now includes MONTH and DAY keywords that identify the design day dates (June 21 and Dec 21 is the default for the cooling and heating design day, respectively).

Location

The DOE-2.1E LOCATION command and its associated keywords have been replaced with two new commands: BUILD-PARAMETERS and SITE-PARAMETERS. As the names imply, BUILD-PARAMETERS is concerned with the old BUILDING-LOCATION keywords dealing with the building and SITE-PARAMETERS includes the old BUILDING-LOCATION keywords dealing with the site. For example:

DOE-2.1E version:

```
BUILDING-LOCATION
   LATITUDE          = 42
   LONGITUDE         = 88
   TIME-ZONE         = 6
   GROSS-AREA        = 10000
   AZIMUTH           = 30   ..
```

Converted:

```
BUILD-PARAMETERS
   GROSS-AREA        = 10000
   AZIMUTH           = 30   ..

SITE-PARAMETERS
   LATITUDE          = 42
   LONGITUDE         = 88
   TIME-ZONE         = 6     ..
```

The following is a complete listing of the keywords (and their abbreviations) associated with the new commands. The keyword descriptions remain the same, except that ATM-MOISTURE keyword, previously in BUILDING-LOCATION, is no longer used.

```
BUILD-PARAMETERS
   AZIMUTH
   HOLIDAYS
   GROSS-AREA
   HEAT-PEAK-PERIOD
   COOL-PEAK-PERIOD
   X-REF
   Y-REF
   FUNCTION
   DAYL-FUNCTION
```

```
SITE-PARAMETERS
   LATITUDE
   LONGITUDE
   TIME-ZONE
   ALTITUDE
   DAYLIGHT-SAVINGS
   GROUND-T
   CLEARNESS-NUMBER
   ATM-TURBIDITY
   SHIELDING-COEF
   TERRAIN-PAR1
   TERRAIN-PAR2
   WS-TERRAIN-PAR1
   WS-TERRAIN-PAR2
   WS-HEIGHT
   WS-HEIGHT-LIST
```

Design Days

The syntax of design day specifications has changed slightly to use common data sources such as ASHRAE. The design day is designated as either a heating or cooling day. This information is required so that the proper schedule day can be used during the design calculations. The day to be used for the design calculations is specified within this command rather than in the RUN-PERIOD command as in DOE-2.1E. Other differences include: wetbulb temperatures are now used in place of dewpoint temperatures; DRYBULB-LO is replaced with RANGE; the humidity ratio is now assumed to be constant throughout the day (unless the drybulb temperature falls below the dewpoint); CLEARNESS and CLOUD-TYPE are now replaced by CLIMATE and VISIBILIY. Use the following conversion as a guideline:

DOE-2.1E version:

```
HOT-CLEAR-SUMMER = DESIGN-DAY
   DRYBULB-HI   = 91    DRYBULB-LO = 71
   HOUR-HI      = 15    HOUR-LO    = 7
   DEWPT-HI     = 65.5  DEWPT-LO   = 60
   DHOUR-HI     = 16    DHOUR-LO   = 8
   WIND-SPEED   = 7.5   WIND-DIR   = 10
   CLOUD-AMOUNT = 0     CLOUD-TYPE = 0
   CLEARNESS    = 1     GROUND-T   = 61  ..

COLD-CLOUDY-WINTER = DESIGN-DAY
   DRYBULB-HI   = 6     DRYBULB-LO = -4
   HOUR-HI      = 15    HOUR-LO    = 2
   DEWPT-HI     = 6     DEWPT-LO   = -4
   DHOUR-HI     = 16    DHOUR-LO   = 3
   WIND-SPEED   = 15    WIND-DIR   = 14
   CLOUD-AMOUNT = 5     CLOUD-TYPE = 2
   CLEARNESS    = 1     GROUND-T   = 46  ..
```

Converted:

```
"Chicago CDD" = DESIGN-DAY
   TYPE            = COOLING
   DRYBULB-HIGH    = 91    DRYBULB-RANGE   = 20
   HOUR-HIGH       = 15    HOUR-LOW        = 7
   WETBULB-AT-HIGH = 77
   WIND-SPEED      = 7.5   WIND-DIR        = SW
   CLOUD-AMOUNT    = 0     CLIMATE         = MIDLATITUDE
   VISIBILITY      = HIGH  GROUND-T        = 61
   ..

"Chicago HDD" = DESIGN-DAY
   TYPE            = HEATING
   DRYBULB-HIGH    = 6     DRYBULB-RANGE  = 10
   HOUR-HIGH       = 15    HOUR-LOW       = 2
   WIND-SPEED      = 15    WIND-DIR       = NW
   CLOUD-AMOUNT    = 5     CLIMATE         = MIDLATITUDE
   VISIBILITY      = LOW   GROUND-T       = 46
   ..
```

The following is a complete listing of the keywords associated with the new DESIGN-DAY command. Many of the keyword descriptions remain the same. New keywords or code words are shown in bold.

```
DESIGN-DAY
   TYPE              = HEATING or COOLING   (required)
   DRYBULB-HIGH      =
   DRYBULB-RANGE     =
   WETBULB-AT-HIGH   =
   HOUR-HIGH         =
   HOUR-LOW          =
   WIND-SPEED        =
   WIND-DIR          = N, NNE, NE, ....., NNW  (new)
   CLOUD-AMOUNT      =
   MONTH             = (month of year to run;
                        COOLING default = 6,
                        HEATING = 12)
   DAY (integer)     = (day of month to run;
                        default is 21)
   GROUND-T          =
   ..
```

Unlike DOE-2.1E, all keywords for DESIGN-DAY are not required. Minimal input for heating and cooling design days for Los Angeles looks like this:

```
LOSANGELES-CDD = DESIGN-DAY
   TYPE              = COOLING
   DRYBULB-HIGH      = 89
   DRYBULB-RANGE     = 20
   WETBULB-AT-HIGH   = 70  ..

LOSANGELES-HDD = DESIGN-DAY
   TYPE              = HEATING
   DRYBULB-HIGH      = 40  ..
```

Space Conditions

SPACE-CONDITIONS, while it still exists in the simulation engine, is not supported in eQUEST or PowerDOE, and may not be supported in future releases of the engine. Instead, we recommend that all SPACE-CONDITIONS keywords be input directly into the SPACE command.

SPACE

The keywords used to describe area lighting and equipment can now accept lists (within parenthesis) of up to five values or a single value (with or without parenthesis). This allows a space to have up to a mixture of up to five different lighting and equipment usages and corresponding schedules. The parenthesis rule means, for example, that LIGHTING-SCHEDULE = (LT1) and LIGHTING-SCHEDULE = LT1 are both permitted, which simplifies converting DOE-2.1E inputs, which were constrained to one lighting (or equipment) schedule per space.

Density-related keywords used to be named in terms of square feet (e.g., LIGHTING-W/SQFT), but now refer to "area" instead (e.g., LIGHTING-W/AREA). Use the following conversion as a guideline (changes are in bold):

DOE-2.1E version:

```
OFFICE = SPACE-CONDITIONS
   TEMPERATURE        = (75)
   FLOOR-WEIGHT       = 70
   LIGHTING-W/SQFT    = 1.5
   EQUIPMENT-W/SQFT   = 1
   PEOPLE-HEAT-GAIN   = 450
   EQUIP-SCHEDULE     = EQ1
   LIGHTING-SCHEDULE  = LT1
   PEOPLE-SCHEDULE    = OCCUP
   AIR-CHANGES/HR     = .3
   INF-SCHEDULE       = INF1
   LIGHT-TO-SPACE     = .80
   INF-METHOD         = AIR-CHANGE
   LIGHTING-TYPE      = REC-FLUOR-RV
   ..
```

Converted:

```
OFFICE = SPACE
   TEMPERATURE        = (75)
   FLOOR-WEIGHT       = 70
   LIGHTING-W/AREA    = (1.0,0.75)
   EQUIPMENT-W/AREA   = (1,.25)
   PEOPLE-HEAT-GAIN   = 450
   EQUIP-SCHEDULE     = (EQ1,EQ2)
   LIGHTING-SCHEDULE  = (LT1,LT2)
   PEOPLE-SCHEDULE    = OCCUP
   AIR-CHANGES/HR     = .3
   INF-SCHEDULE       = INF1
   LIGHT-TO-SPACE     = .80
   INF-METHOD         = AIR-CHANGE
   LIGHTING-TYPE      = (REC-FLUOR-RV, REC-FLUOR-RV)
   etc.
   ..
```

Glass Type

The GLASS-TYPE-CODE (G-T-C) values of 1 to 11 are no longer valid. Allowed values must now be in the range 1000 to 9999 and must correspond to a G-T-C value in window portion of the library. The alternative method of specifying GLASS-CONDUCT and SHADING-COEF can still be used. PANES is no longer a valid keyword, as this is incorporated into the window library entry.

Additionally, the GLASS-TYPE command now requires the specification of the TYPE keyword. As in other "TYPE" commands, the TYPE keyword is required and must be the first keyword specified. The values for TYPE are either GLASS-TYPE-CODE or SHADING-COEF; these values specify which input keywords are to be used (are required) for this GLASS-TYPE command. Examples of each TYPE are as follows:

```
"Single Grey" = GLASS-TYPE
   TYPE              = GLASS-TYPE-CODE
   GLASS-TYPE-CODE   = 1205
   ..
```

OR

```
"Ref-A-M" = GLASS-TYPE
   TYPE              = SHADING-COEF
   SHADING-COEF      = 0.86
   ..
```

MATERIAL

The MATERIAL command now requires the specification of the TYPE keyword. As in other "TYPE" commands, the TYPE keyword must be present and be the first keyword specified. The values for TYPE are either PROPERTIES or RESISTANCE; these values specify which input keywords are to be used (are required) for this MATERIAL command. Examples of each TYPE are as follows:

```
"Mat 2ft Gnd" = MATERIAL
   TYPE              = PROPERTIES
   THICKNESS         = 2
   CONDUCTIVITY      = 0.5
   DENSITY           = 100
   SPECIFIC-HEAT     = 0.25
   ..
```

OR

```
"Mat R11M" = MATERIAL
   TYPE              = RESISTANCE
   RESISTANCE        = 5.5
   ..
```

All materials previously contained in the DOE2.1E materials library are also contained in the materials library. However, each 4-character or 5-character material code word, such as IN36, previously used in DOE2.1E has been replaced with a more descriptive name. Therefore, it is necessary to convert the old DOE2.1E material code words to the new material names. An alphabetical listing of the old DOE2.1E material code words and the new names is shown below. Note that new material names containing blanks must be surrounded in double quotes when included in BDL lists, as illustrated below.

DOE-2.1E version:

```
Blt-Up-Roof-LA = LAYERS
   MATERIAL           = ( BR01, IN36, AS01 )
   ..
```

Converted:

```
"Built Up Roof Layer" = LAYERS
   MATERIAL           = ("Blt-Up Roof 3/8in (BR01)",
                         "Polystyrene 3in (IN36)",
                         "Steel Siding (AS01)" )
   ..
```

Table 2 Materials Library

DOE-2.1E Code Word	Library Item	Library Category
AB01	AbsCem Bd 1/8in (AB01)	Asbestos Cement
AB02	AbsCem Bd 1/4in (AB02)	Asbestos Cement
AB03	AbsCem Shingle (AB03)	Asbestos Cement
AB04	AbsCem Siding (AB04)	Asbestos Cement
AC01	AcousTile 3/8in (AC01)	Acoustic Tile
AC02	AcousTile 1/2in (AC02)	Acoustic Tile
AC03	AcousTile 3/4in (AC03)	Acoustic Tile
AL01	Surf Air Film Vert (AL01)	Surface Air Film
AL02	Surf Air Film Slope (AL02)	Surface Air Film
AL03	Surf Air Film Horiz (AL03)	Surface Air Film
AL11	Air Lay <3/4in Vert (AL11)	Air Layer
AL12	Air Lay <3/4in Slope (AL12)	Air Layer
AL13	Air Lay <3/4in Horiz (AL13)	Air Layer
AL21	Air Lay <4in Vert (AL21)	Air Layer
AL22	Air Lay <4in Slope (AL22)	Air Layer
AL23	Air Lay <4in Horiz (AL23)	Air Layer
AL31	Air Lay >4in Vert (AL31)	Air Layer
AL32	Air Lay >4in Slope (AL32)	Air Layer
AL33	Air Lay >4in Horiz (AL33)	Air Layer
AR01	Asph Roll Roof (AR01)	Asphalt
AR02	Asph Siding (AR02)	Asphalt
AR03	Asph Tile (AR03)	Asphalt
AS01	Steel Siding (AS01)	Steel Siding
AV01	AbsVinyl Tile (AV01)	Asbestos Vinyl
BK01	Com Brick 4in (BK01)	Brick
BK02	Com Brick 8in (BK02)	Brick
BK03	Com Brick 12in (BK03)	Brick
BK04	Face Brick 3in (BK04)	Brick
BK05	Face Brick 4in (BK05)	Brick
BP01	Bldg Paper Felt (BP01)	Building Paper
BP02	Bldg Paper Seal (BP02)	Building Paper
BP03	Plastic Film Seal (BP03)	Building Paper
BR01	Blt-Up Roof 3/8in (BR01)	Built-Up Roofing
CB01	CMU HW 4in Hollow (CB01)	Conc Blk Hvy Wt
CB02	CMU HW 4in ConcFill (CB02)	Conc Blk Hvy Wt
CB03	CMU HW 4in PerlFill (CB03)	Conc Blk Hvy Wt
CB04	CMU HW 4in PartFill (CB04)	Conc Blk Hvy Wt
CB05	CMU HW 4in Conc/Perl (CB05)	Conc Blk Hvy Wt
CB06	CMU HW 6in Hollow (CB06)	Conc Blk Hvy Wt
CB07	CMU HW 6in ConcFill (CB07)	Conc Blk Hvy Wt
CB08	CMU HW 6in PerlFill (CB08)	Conc Blk Hvy Wt
CB09	CMU HW 6in PartFill (CB09)	Conc Blk Hvy Wt
CB10	CMU HW 6in Conc/Perl (CB10)	Conc Blk Hvy Wt
CB11	CMU HW 8in Hollow (CB11)	Conc Blk Hvy Wt
CB12	CMU HW 8in ConcFill (CB12)	Conc Blk Hvy Wt
CB13	CMU HW 8in PerlFill (CB13)	Conc Blk Hvy Wt
CB14	CMU HW 8in PartFill (CB14)	Conc Blk Hvy Wt
CB15	CMU HW 8in Conc/Perl (CB15)	Conc Blk Hvy Wt

DOE-2.1E Code Word	Library Item	Library Category
CB16	CMU HW 12in Hollow (CB16)	Conc Blk Hvy Wt
CB17	CMU HW 12in ConcFill (CB17)	Conc Blk Hvy Wt
CB18	CMU HW 12in PartFill (CB18)	Conc Blk Hvy Wt
CB21	CMU MW 4in Hollow (CB21)	Conc Blk Med Wt
CB22	CMU MW 4in ConcFill (CB22)	Conc Blk Med Wt
CB23	CMU MW 4in PerlFill (CB23)	Conc Blk Med Wt
CB24	CMU MW 4in PartFill (CB24)	Conc Blk Med Wt
CB25	CMU MW 4in Conc/Perl (CB25)	Conc Blk Med Wt
CB26	CMU MW 6in Hollow (CB26)	Conc Blk Med Wt
CB27	CMU MW 6in ConcFill (CB27)	Conc Blk Med Wt
CB28	CMU MW 6in PerlFill (CB28)	Conc Blk Med Wt
CB29	CMU MW 6in PartFill (CB29)	Conc Blk Med Wt
CB30	CMU MW 6in Conc/Perl (CB30)	Conc Blk Med Wt
CB31	CMU MW 8in Hollow (CB31)	Conc Blk Med Wt
CB32	CMU MW 8in ConcFill (CB32)	Conc Blk Med Wt
CB33	CMU MW 8in PerlFill (CB33)	Conc Blk Med Wt
CB34	CMU MW 8in PartFill (CB34)	Conc Blk Med Wt
CB35	CMU MW 8in Conc/Perl (CB35)	Conc Blk Med Wt
CB36	CMU MW 12in Hollow (CB36)	Conc Blk Med Wt
CB37	CMU MW 12in ConcFill (CB37)	Conc Blk Med Wt
CB38	CMU MW 12in PartFill (CB38)	Conc Blk Med Wt
CB41	CMU LW 4in Hollow (CB41)	Conc Blk Lt Wt
CB42	CMU LW 4in ConcFill (CB42)	Conc Blk Lt Wt
CB43	CMU LW 4in PerlFill (CB43)	Conc Blk Lt Wt
CB44	CMU LW 4in PartFill (CB44)	Conc Blk Lt Wt
CB45	CMU LW 4in Conc/Perl (CB45)	Conc Blk Lt Wt
CB46	CMU LW 6in Hollow (CB46)	Conc Blk Lt Wt
CB47	CMU LW 6in ConcFill (CB47)	Conc Blk Lt Wt
CB48	CMU LW 6in PerlFill (CB48)	Conc Blk Lt Wt
CB49	CMU LW 6in PartFill (CB49)	Conc Blk Lt Wt
CB50	CMU LW 6in Conc/Perl (CB50)	Conc Blk Lt Wt
CB51	CMU LW 8in Hollow (CB51)	Conc Blk Lt Wt
CB52	CMU LW 8in ConcFill (CB52)	Conc Blk Lt Wt
CB53	CMU LW 8in PerlFill (CB53)	Conc Blk Lt Wt
CB54	CMU LW 8in PartFill (CB54)	Conc Blk Lt Wt
CB55	CMU LW 8in Conc/Perl (CB55)	Conc Blk Lt Wt
CB56	CMU LW 12in Hollow (CB56)	Conc Blk Lt Wt
CB57	CMU LW 12in ConcFill (CB57)	Conc Blk Lt Wt
CB58	CMU LW 12in PartFill (CB58)	Conc Blk Lt Wt
CC01	Conc HW 140lb 1.25in (CC01)	Concrete 140 lbs
CC02	Conc HW 140lb 2in (CC02)	Concrete 140 lbs
CC03	Conc HW 140lb 4in (CC03)	Concrete 140 lbs
CC04	Conc HW 140lb 6in (CC04)	Concrete 140 lbs
CC05	Conc HW 140lb 8in (CC05)	Concrete 140 lbs
CC06	Conc HW 140lb 10in (CC06)	Concrete 140 lbs
CC07	Conc HW 140lb 12in (CC07)	Concrete 140 lbs
CC11	Conc HW 140lb 3/4in (CC11)	Concrete 140 lbs
CC12	Conc HW 140lb 1-1/8in (CC12)	Concrete 140 lbs
CC13	Conc HW 140lb 1.75in (CC13)	Concrete 140 lbs
CC14	Conc HW 140lb 4in (CC14)	Concrete 140 lbs

DOE-2.1E Code Word	Library Item	Library Category
CC15	Conc HW 140lb 6in (CC15)	Concrete 140 lbs
CC16	Conc HW 140lb 8in (CC16)	Concrete 140 lbs
CC21	Conc LW 80lb 3/4in (CC21)	Concrete 80 lbs
CC22	Conc LW 80lb 1.25in (CC22)	Concrete 80 lbs
CC23	Conc LW 80lb 2in (CC23)	Concrete 80 lbs
CC24	Conc LW 80lb 4in (CC24)	Concrete 80 lbs
CC25	Conc LW 80lb 6in (CC25)	Concrete 80 lbs
CC26	Conc LW 80lb 8in (CC26)	Concrete 80 lbs
CC31	Conc LW 30lb 3/4in (CC31)	Concrete 30 lbs
CC32	Conc LW 30lb 1.25in (CC32)	Concrete 30 lbs
CC33	Conc LW 30lb 2in (CC33)	Concrete 30 lbs
CC34	Conc LW 30lb 4in (CC34)	Concrete 30 lbs
CC35	Conc LW 30lb 6in (CC35)	Concrete 30 lbs
CC36	Conc LW 30lb 8in (CC36)	Concrete 30 lbs
CM01	Cmt Mortar 1in (CM01)	Cement Mortar
CM02	Cmt Mortar 1.75in (CM02)	Cement Mortar
CM03	Cmt Plaster 1in (CM03)	Cement Mortar
CP01	Carpet & Fiber Pad (CP01)	Carpet
CP02	Carpet & Rubber Pad (CP02)	Carpet
CT01	Hol ClayTile 3in (CT01)	Clay Tile
CT02	Hol ClayTile 4in (CT02)	Clay Tile
CT03	Hol ClayTile 6in (CT03)	Clay Tile
CT04	Hol ClayTile 8in (CT04)	Clay Tile
CT05	Hol ClayTile 10in (CT05)	Clay Tile
CT06	Hol ClayTile 12in (CT06)	Clay Tile
CT11	ClayTile Paver 3/8in (CT11)	Clay Tile
GL01	Glass Wool 1/4in (GL01)	Board Insul
GP01	GypBd 1/2in (GP01)	Gypsum
GP02	GypBd 5/8in (GP02)	Gypsum
GP03	GypBd 3/4in (GP03)	Gypsum
GP04	Gypsum LW Agg 3/4in (GP04)	Gypsum
GP05	Gypsum LW Agg 1in (GP05)	Gypsum
GP06	Gypsum Sand Agg 3/4in (GP06)	Gypsum
GP07	Gypsum Sand Agg 1in (GP07)	Gypsum
HB01	Hd Bd 3/4in Md Dens (HB01)	Hard Board
HB02	Hd Bd 3/4in Md Dens (HB02)	Hard Board
HB03	Hd Bd 3/4in Std Temp (HB03)	Hard Board
HB04	Hd Bd 3/4in Srv Temp (HB04)	Hard Board
HF-A1	Stucco 1in (HF-A1)	Stucco
HF-A2	Face Brick 4in (HF-A2)	Brick
HF-A3	Steel Siding (HF-A3)	Steel Siding
HF-A6	Finish (HF-A6)	Finish
HF-A7	Face Brick 4in (HF-A7)	Brick
HF-B1	Air Space (HF-B1)	Air Layer
HF-B2	Insul Bd 1in (HF-B2)	Board Insul
HF-B3	Insul Bd 2in (HF-B3)	Board Insul
HF-B4	Insul Bd 3in (HF-B4)	Board Insul
HF-B5	Insul Bd 1in (HF-B5)	Board Insul
HF-B6	Insul Bd 2in (HF-B6)	Board Insul
HF-B7	Wood 1in (HF-B7)	Wood

DOE-2.1E Code Word	Library Item	Library Category
HF-B8	Wood 2.5in (HF-B8)	Wood
HF-B9	Wood 4in (HF-B9)	Wood
HF-B10	Wood 2in (HF-B10)	Wood
HF-B11	Wood 3in (HF-B11)	Wood
HF-B12	Insul Bd 3in (HF-B12)	Board Insul
HF-C1	ClayTile 4in (HF-C1)	Clay Tile
HF-C2	CMU LW 4in (HF-C2)	Conc Blk Lt Wt
HF-C3	CMU HW 4in (HF-C3)	Conc Blk Hvy Wt
HF-C4	Com Brick 4in (HF-C4)	Brick
HF-C5	Conc HW 140lb 4in (HF-C5)	Concrete 140 lbs
HF-C6	ClayTile 8in (HF-C6)	Clay Tile
HF-C7	CMU LW 8in (HF-C7)	Conc Blk Lt Wt
HF-C8	CMU HW 8in (HF-C8)	Conc Blk Hvy Wt
HF-C9	Com Brick 8in (HF-C9)	Brick
HF-C10	Conc HW 8in (HF-C10)	Concrete 140 lbs
HF-C11	Conc HW 12in (HF-C11)	Concrete 140 lbs
HF-C12	Conc HW 140lb 2in (HF-C12)	Concrete 140 lbs
HF-C13	Conc HW 6in (HF-C13)	Concrete 140 lbs
HF-C14	Conc LW 40lb 4in (HF-C14)	Concrete 40 lbs
HF-C15	Conc LW 40lb 6in (HF-C15)	Concrete 40 lbs
HF-C16	Conc LW 40lb 8in (HF-C16)	Concrete 40 lbs
HF-E1	GypBd 3/4in (HF-E1)	Gypsum
HF-E2	Stone 1/2in (HF-E2)	Stone
HF-E3	Felt 3/8in (HF-E3)	Felt
HF-E4	Clg Air Space (HF-E4)	Air Layer
HF-E5	AcousTile (HF-E5)	Acoustic Tile
IN01	MinWool Batt R7 (IN01)	Batt Insulation
IN02	MinWool Batt R11 (IN02)	Batt Insulation
IN03	MinWool Batt R19 (IN03)	Batt Insulation
IN04	MinWool Batt R24 (IN04)	Batt Insulation
IN05	MinWool Batt R30 (IN05)	Batt Insulation
IN11	MinWool Fill 3.5in R11 (IN11)	Fill Insulation
IN12	MinWool Fill 5.5in R19 (IN12)	Fill Insulation
IN13	Cellulose 3.5in R-13 (IN13)	Fill Insulation
IN14	Cellulose 5.5in R-20 (IN14)	Fill Insulation
IN21	MinBd 7/8in R-3 (IN21)	Board Insul
IN22	MinBd 1in R-3.5 (IN22)	Board Insul
IN23	MinBd 2in R-6.9 (IN23)	Board Insul
IN24	MinBd 3in R-10.3 (IN24)	Board Insul
IN31	Polystyrene 1/2in (IN31)	Polystyrene
IN32	Polystyrene 3/4in (IN32)	Polystyrene
IN33	Polystyrene 1in (IN33)	Polystyrene
IN34	Polystyrene 1.25in (IN34)	Polystyrene
IN35	Polystyrene 2in (IN35)	Polystyrene
IN36	Polystyrene 3in (IN36)	Polystyrene
IN37	Polystyrene 4in (IN37)	Polystyrene
IN41	Polyurethane 1/2in (IN41)	Polyurethane
IN42	Polyurethane 3/4in (IN42)	Polyurethane
IN43	Polyurethane 1in (IN43)	Polyurethane
IN44	Polyurethane 1.25in (IN44)	Polyurethane

DOE-2.1E Code Word	Library Item	Library Category
IN45	Polyurethane 2in (IN45)	Polyurethane
IN46	Polyurethane 3in (IN46)	Polyurethane
IN47	Polyurethane 4in (IN47)	Polyurethane
IN51	Urea Formald 3.5in R19 (IN51)	Urea Formaldehyde
IN52	Urea Formald 5.5in R30 (IN52)	Urea Formaldehyde
IN61	Insul Bd 1/2in (IN61)	Board Insul
IN62	Insul Bd 3/4in (IN62)	Board Insul
IN63	Insul Bd 3/8in (IN63)	Board Insul
IN64	Insul Bd 1/2in (IN64)	Board Insul
IN71	Roof Insul 1/2in (IN71)	Board Insul
IN72	Roof Insul 1in (IN72)	Board Insul
IN73	Roof Insul 1.5in (IN73)	Board Insul
IN74	Roof Insul 2in (IN74)	Board Insul
IN75	Roof Insul 2.5in (IN75)	Board Insul
IN76	Roof Insul 3in (IN76)	Board Insul
LT01	Linoleum Tile (LT01)	Linoleum Tile
PB01	PartBd Lo Dens 3/4in (PB01)	Particle Board
PB02	PartBd Md Dens 3/4in (PB02)	Particle Board
PB03	PartBd Hi Dens 3/4in (PB03)	Particle Board
PB04	PartBd Underlay 5/8in (PB04)	Particle Board
PW01	Plywd 1/4in (PW01)	Plywood
PW02	Plywd 3/8in (PW02)	Plywood
PW03	Plywd 1/2in (PW03)	Plywood
PW04	Plywd 5/8in (PW04)	Plywood
PW05	Plywd 3/4in (PW05)	Plywood
PW06	Plywd 1in (PW06)	Plywood
RG01	Gravel 1/2in (RG01)	Roof Gravel
RG02	Gravel 1in (RG02)	Roof Gravel
RT01	Rubber Tile (RT01)	Rubber Tile
SC01	Stucco 1in (SC01)	Stucco
SL01	Slate 1/2in (SL01)	Slate
	Soil 12in	Soil
ST01	Stone 1in (ST01)	Stone
TZ01	Terrazzo 1in (TZ01)	Terrazzo
WD01	Wood Sft 3/4in (WD01)	Wood
WD02	Wood Sft 1.5in (WD02)	Wood
WD03	Wood Sft 2.5in (WD03)	Wood
WD04	Wood Sft 3.5in (WD04)	Wood
WD05	Wood Sft 4in (WD05)	Wood
WD11	Wood Hd 3/4in (WD11)	Wood
WD12	Wood Hd 1in (WD12)	Wood
WS01	Wood Shingle (WS01)	Wood
WS02	Wood Shingle (WS02)	Wood

CONSTRUCTION

The CONSTRUCTION command is also now a TYPE command. As in other "TYPE" commands, the TYPE keyword must be present and be the first keyword specified. The value for TYPE is either LAYERS or U-VALUE;

this value specifies which input keywords are to be used (are required) for this CONSTRUCTION command. Examples of each TYPE are as follows:

```
"Cons Roof" = CONSTRUCTION
   TYPE              = LAYERS
   LAYERS            = "Lay Roof"
   ..
```

or

```
"Cons Ceiling" = CONSTRUCTION
   TYPE              = U-VALUE
   U-VALUE           = 0.5
   ..
```

All LAYERS previously contained in the DOE2.1E library are also contained in the library, however, each 6-digit or 7-digit code word previously used in DOE2.1E has been replaced with a more descriptive name. Therefore, it is necessary to convert the old DOE2.1E layers code words to the new names. An alphabetical listing of the old DOE2.1E layers code words and the new names is provided below. Note that new layers names containing blanks must be surrounded in double quotes, as illustrated below.

DOE-2.1E version:

```
Roof-Deck-Cons = CONSTRUCTION
   ROUGHNESS         = 1
   LAYERS            = ASHR-17   ..
```

Converted:

```
“Built Up Roof Cons” = CONSTRUCTION
   TYPE              = LAYERS
   ROUGHNESS         = 1
   LAYERS            = "ASH Roof-17 lay"
   ..
```

Table 3 Layers Library

DOE-2.1E Code Word	Library Item Name	Library Category
ASHI-1	ASH Int Wall-1 lay	I-Clay Tile
ASHI-2	ASH Int Wall-2 lay	I-Concrete, Lt
ASHI-3	ASH Int Wall-3 lay	I-Concrete, Hvy
ASHI-4	ASH Int Wall-4 lay	I-Brick
ASHI-5	ASH Int Wall-5 lay	I-Concrete, Hvy
ASHI-6	ASH Int Wall-6 lay	I-Clay Tile
ASHI-7	ASH Int Wall-7 lay	I-Conc Blk, Lt
ASHI-8	ASH Int Wall-8 lay	I-Conc Blk, Hvy
ASHI-9	ASH Int Wall-9 lay	I-Brick
ASHI-10	ASH Int Wall-10 lay	I-Concrete, Hvy
ASHI-11	ASH Int Wall-11 lay	I-Concrete, Hvy
ASHI-12	ASH Int Wall-12 lay	I-Clay Tile
ASHI-13	ASH Int Wall-13 lay	I-Conc Blk, Lt
ASHI-14	ASH Int Wall-14 lay	I-Conc Blk, Lt
ASHI-15	ASH Int Wall-15 lay	I-Brick
ASHI-16	ASH Int Wall-16 lay	I-Concrete, Hvy
ASHI-17	ASH Int Wall-17 lay	I-Clay Tile
ASHI-18	ASH Int Wall-18 lay	I-Conc Blk, Lt
ASHI-19	ASH Int Wall-19 lay	I-Conc Blk, Hvy
ASHI-20	ASH Int Wall-20 lay	I-Brick
ASHI-21	ASH Int Wall-21 lay	I-Concrete, Hvy
ASHI-22	ASH Int Wall-22 lay	I-Concrete, Hvy
ASHI-23	ASH Int Wall-23 lay	I-Frame
ASHI-24	ASH Int Wall-24 lay	I-Wood
ASHI-25	ASH Int Wall-25 lay	I-Wood
ASHI-26	ASH Int Wall-26 lay	I-Wood
ASHI-27	ASH Int Wall-27 lay	I-Wood
ASHI-28	ASH Int Wall-28 lay	I-Wood
ASHI-29	ASH Int Wall-29 lay	I-Wood
ASHI-30	ASH Int Wall-30 lay	I-Wood
ASHI-31	ASH Int Wall-31 lay	I-Concrete, Hvy
ASHI-32	ASH Int Wall-32 lay	I-Concrete, Hvy
ASHI-33	ASH Int Wall-33 lay	I-Concrete, Lt
ASHI-34	ASH Int Wall-34 lay	I-Concrete, Hvy
ASHI-35	ASH Int Wall-35 lay	I-Concrete, Lt
ASHI-36	ASH Int Wall-36 lay	I-Wood
ASHI-37	ASH Int Wall-37 lay	I-Wood
ASHI-38	ASH Int Wall-38 lay	I-Concrete, Hvy
ASHI-39	ASH Int Wall-39 lay	I-Concrete, Hvy
ASHI-40	ASH Int Wall-40 lay	I-Concrete, Lt
ASHI-41	ASH Int Wall-41 lay	I-Concrete, Hvy
ASHI-42	ASH Int Wall-42 lay	I-Concrete, Lt
ASHI-43	ASH Int Wall-43 lay	I-Wood
ASHI-44	ASH Int Wall-44 lay	I-Wood
ASHI-45	ASH Int Wall-45 lay	I-Concrete, Hvy
ASHI-46	ASH Int Wall-46 lay	I-Wood
ASHI-47	ASH Int Wall-47 lay	I-Steel Deck
ASHR-1	ASH Roof-1 lay	R-Concrete, Hvy

DOE-2.1E Code Word	Library Item Name	Library Category
ASHR-2	ASH Roof-2 lay	R-Wood
ASHR-3	ASH Roof-3 lay	R-Wood
ASHR-4	ASH Roof-4 lay	R-Wood
ASHR-5	ASH Roof-5 lay	R-Wood
ASHR-6	ASH Roof-6 lay	R-Wood
ASHR-7	ASH Roof-7 lay	R-Wood
ASHR-8	ASH Roof-8 lay	R-Concrete, Lt
ASHR-9	ASH Roof-9 lay	R-Concrete, Lt
ASHR-10	ASH Roof-10 lay	R-Concrete, Lt
ASHR-11	ASH Roof-11 lay	R-Concrete, Hvy
ASHR-12	ASH Roof-12 lay	R-Concrete, Hvy
ASHR-13	ASH Roof-13 lay	R-Concrete, Hvy
ASHR-14	ASH Roof-14 lay	R-Concrete, Hvy
ASHR-15	ASH Roof-15 lay	R-Concrete, Hvy
ASHR-16	ASH Roof-16 lay	R-Concrete, Hvy
ASHR-17	ASH Roof-17 lay	R-Sheet Metal
ASHR-18	ASH Roof-18 lay	R-Sheet Metal
ASHR-19	ASH Roof-19 lay	R-Concrete, Hvy
ASHR-20	ASH Roof-20 lay	R-Wood
ASHR-21	ASH Roof-21 lay	R-Wood
ASHR-22	ASH Roof-22 lay	R-Wood
ASHR-23	ASH Roof-23 lay	R-Wood
ASHR-24	ASH Roof-24 lay	R-Wood
ASHR-25	ASH Roof-25 lay	R-Wood
ASHR-26	ASH Roof-26 lay	R-Concrete, Lt
ASHR-27	ASH Roof-27 lay	R-Concrete, Lt
ASHR-28	ASH Roof-28 lay	R-Concrete, Lt
ASHR-29	ASH Roof-29 lay	R-Concrete, Hvy
ASHR-30	ASH Roof-30 lay	R-Concrete, Hvy
ASHR-31	ASH Roof-31 lay	R-Concrete, Hvy
ASHR-32	ASH Roof-32 lay	R-Concrete, Hvy
ASHR-33	ASH Roof-33 lay	R-Concrete, Hvy
ASHR-34	ASH Roof-34 lay	R-Concrete, Hvy
ASHR-35	ASH Roof-35 lay	R-Sheet Metal
ASHR-36	ASH Roof-36 lay	R-Sheet Metal
ASHW-1	ASH Wall-1 lay	W-Brick/Block
ASHW-2	ASH Wall-2 lay	W-Concrete, Lt
ASHW-3	ASH Wall-3 lay	W-Brick
ASHW-4	ASH Wall-4 lay	W-Brick/Block
ASHW-5	ASH Wall-5 lay	W-Brick/Block
ASHW-6	ASH Wall-6 lay	W-Brick/Clay Tile
ASHW-7	ASH Wall-7 lay	W-Brick/Block
ASHW-8	ASH Wall-8 lay	W-Brick
ASHW-9	ASH Wall-9 lay	W-Brick/Block
ASHW-10	ASH Wall-10 lay	W-Brick/Block
ASHW-11	ASH Wall-11 lay	W-Concrete, Hvy
ASHW-12	ASH Wall-12 lay	W-Concrete, Hvy
ASHW-13	ASH Wall-13 lay	W-Concrete, Hvy
ASHW-14	ASH Wall-14 lay	W-Concrete, Hvy
ASHW-15	ASH Wall-15 lay	W-Concrete, Hvy

DOE-2.1E Code Word	Library Item Name	Library Category
ASHW-16	ASH Wall-16 lay	W-Brick
ASHW-17	ASH Wall-17 lay	W-Brick
ASHW-18	ASH Wall-18 lay	W-Brick/Block
ASHW-19	ASH Wall-19 lay	W-Stucco
ASHW-20	ASH Wall-20 lay	W-Brick
ASHW-21	ASH Wall-21 lay	W-Brick/Conc
ASHW-22	ASH Wall-22 lay	W-Brick/Conc
ASHW-23	ASH Wall-23 lay	W-Brick/Conc
ASHW-24	ASH Wall-24 lay	W-Brick/Conc
ASHW-25	ASH Wall-25 lay	W-Wood Frame
ASHW-26	ASH Wall-26 lay	W-Wood Frame
ASHW-27	ASH Wall-27 lay	W-Curtain Wall
ASHW-28	ASH Wall-28 lay	W-Curtain Wall
ASHW-29	ASH Wall-29 lay	W-Curtain Wall
ASHW-30	ASH Wall-30 lay	W-Concrete, Hvy
ASHW-31	ASH Wall-31 lay	W-Concrete, Hvy
ASHW-32	ASH Wall-32 lay	W-Concrete, Hvy
ASHW-33	ASH Wall-33 lay	W-Concrete, Hvy
ASHW-34	ASH Wall-34 lay	W-Concrete, Hvy
ASHW-35	ASH Wall-35 lay	W-Concrete, Hvy
ASHW-36	ASH Wall-36 lay	W-Wood Frame
ASHW-37	ASH Wall-37 lay	W-Wood Frame
ASHW-38	ASH Wall-38 lay	W-Wood Frame
ASHW-39	ASH Wall-39 lay	W-Wood Frame
ASHW-40	ASH Wall-40 lay	W-Concrete, Hvy
ASHW-41	ASH Wall-41 lay	W-Concrete, Hvy
ASHW-42	ASH Wall-42 lay	W-Brick
ASHW-43	ASH Wall-43 lay	W-Conc Blk, Hvy
ASHW-44	ASH Wall-44 lay	W-Conc Blk, Lt
ASHW-45	ASH Wall-45 lay	W-Clay Tile
ASHW-46	ASH Wall-46 lay	W-Conc Blk, Hvy
ASHW-47	ASH Wall-47 lay	W-Brick
ASHW-48	ASH Wall-48 lay	W-Conc Blk, Hvy
ASHW-49	ASH Wall-49 lay	W-Conc Blk, Lt
ASHW-50	ASH Wall-50 lay	W-Clay Tile
ASHW-51	ASH Wall-51 lay	W-Brick/Conc
ASHW-52	ASH Wall-52 lay	W-Brick/Conc
ASHW-53	ASH Wall-53 lay	W-Brick
ASHW-54	ASH Wall-54 lay	W-Brick/Conc
ASHW-55	ASH Wall-55 lay	W-Brick/Conc
ASHW-56	ASH Wall-56 lay	W-Brick/Block
ASHW-57	ASH Wall-57 lay	W-Brick/Block
ASHW-58	ASH Wall-58 lay	W-Brick/Clay Tile
ASHW-59	ASH Wall-59 lay	W-Brick/Conc
ASHW-60	ASH Wall-60 lay	W-Brick
ASHW-61	ASH Wall-61 lay	W-Brick/Block
ASHW-62	ASH Wall-62 lay	W-Brick
ASHW-63	ASH Wall-63 lay	W-Conc Blk, Hvy
ASHW-64	ASH Wall-64 lay	W-Conc Blk, Hvy
ASHW-65	ASH Wall-65 lay	W-Conc Blk, Lt

DOE-2.1E Code Word	Library Item Name	Library Category
ASHW-66	ASH Wall-66 lay	W-Conc Blk, Lt
ASHW-67	ASH Wall-67 lay	W-Brick/Clay Tile
ASHW-68	ASH Wall-68 lay	W-Brick/Clay Tile
ASHW-69	ASH Wall-69 lay	W-Brick/Clay Tile
ASHW-70	ASH Wall-70 lay	W-Clay Tile
ASHW-71	ASH Wall-71 lay	W-Clay Tile
ASHW-72	ASH Wall-72 lay	W-Clay Tile
ASHW-73	ASH Wall-73 lay	W-Concrete, Hvy
ASHW-74	ASH Wall-74 lay	W-Concrete, Hvy
ASHW-75	ASH Wall-75 lay	W-Concrete, Hvy
ASHW-76	ASH Wall-76 lay	W-Concrete, Hvy
ASHW-77	ASH Wall-77 lay	W-Brick
ASHW-78	ASH Wall-78 lay	W-Brick
ASHW-79	ASH Wall-79 lay	W-Brick
ASHW-80	ASH Wall-80 lay	W-Brick
ASHW-81	ASH Wall-81 lay	W-Conc Blk, Hvy
ASHW-82	ASH Wall-82 lay	W-Brick/Block
ASHW-83	ASH Wall-83 lay	W-Brick/Block
ASHW-84	ASH Wall-84 lay	W-Brick/Block
ASHW-85	ASH Wall-85 lay	W-Conc Blk, Lt
ASHW-86	ASH Wall-86 lay	W-Conc Blk, Lt
ASHW-87	ASH Wall-87 lay	W-Conc Blk, Lt
ASHW-88	ASH Wall-88 lay	W-Brick/Clay Tile
ASHW-89	ASH Wall-89 lay	W-Brick/Clay Tile
ASHW-90	ASH Wall-90 lay	W-Brick/Clay Tile
ASHW-91	ASH Wall-91 lay	W-Clay Tile
ASHW-92	ASH Wall-92 lay	W-Clay Tile
ASHW-93	ASH Wall-93 lay	W-Clay Tile
ASHW-94	ASH Wall-94 lay	W-Sheet Metal
ASHW-95	ASH Wall-95 lay	W-Sheet Metal
ASHW-96	ASH Wall-96 lay	W-Sheet Metal

FLOOR

A new command, FLOOR, is provided in this version. It is used to group SPACEs together for display purposes in graphical user interfaces. *FLOOR is a required command and a FLOOR must be defined prior to any SPACE.* Although all spaces may be assigned to the same FLOOR, for the greatest display utility care should be taken to assign SPACEs to the correct FLOOR.

FLOOR also adds a fifth coordinate system. DOE-2.1E's coordinate system hierarchy was: Reference coordinate system, Building coordinate system, Space coordinate system, and Surface coordinate system. This program's coordinate system hierarchy is: Reference coordinate system, Building coordinate system, Floor coordinate system, Space coordinate system, and Surface coordinate system. Therefore, geometry keywords X, Y, Z and AZIMUTH all apply to FLOOR and are defined relative to the Building coordinate system. Accordingly, in the space geometry keywords X, Y, Z and AZIMUTH are now defined relative to the Floor coordinate system rather than the Building coordinate system.

An example of minimum input for this new command would be as follows. (See the following section for discussion of the SHAPE keyword)

Example:

```
ALL-FLOORS = FLOOR
   SHAPE              = NO-SHAPE
   FLOOR-HEIGHT       = 12
   SPACE-HEIGHT       = 9
   ..
```

Shapes and Surface AREA vs Surface HEIGHT and WIDTH

For interfaces that graphically display the building geometry, it is important to specify the HEIGHT and WIDTH of all building surfaces, and HEIGHT, WIDTH, and DEPTH for spaces (if you wish to see the floor plan).

All FLOORs and SPACEs must have SHAPE specified as either NO-SHAPE (meaning AREA, VOLUME or HEIGHT are required), BOX (meaning HEIGHT, WIDTH, and DEPTH are required), or POLYGON (meaning POLYGON and HEIGHT are required).

The shapes of FLOORs, SPACEs and walls may be specified by referencing a previously-specified POLYGON command. The POLYGON command is used to describe a non-rectangular shape by specifying the X and Y coordinates of the shape vertices in a counter-clockwise order. A POLYGON may have up to 20 vertices so specified. A POLYGON may be referenced by multiple surfaces, or by FLOORs or SPACEs, so as to make each have the same shape. The first vertex of a POLYGON is usually the origin, but that is not required. Examples are:

```
"FlrPoly" = POLYGON
   V1                 = (0,0)
   V2                 = (100,0)
   V3                 = (100,80)
   V4                 = (0,80)
   ..
```

or

```
"LongTrapPoly" = POLYGON
   V1                 = (0,0)
   V2                 = (100,0)
   V3                 = (75,25)
   V4                 = (25,25)
   ..
```

The use of POLYGON's and LOCATION can greatly simplify the input and allow automatic relocation of FLOORs, SPACEs and their surfaces. This is accomplished by specifying the POLYGON shape for all FLOORs and SPACEs and using the LOCATION keyword in the SPACE and wall commands to specify the vertex of the "parent" command's POLYGON at which you want this item to be located. Using this technique BDL will calculate the X, Y, and AZIMUTH for the SPACE or wall as well as the WIDTH for walls. The following examples shows this use:

```
"Floor 1" = FLOOR
   SHAPE              = POLYGON
   MULTIPLIER         = 1
   POLYGON            = "FlrPoly"
   ..
```

```
"Flr:1 > North" = SPACE
   SHAPE              = POLYGON
   HEIGHT             = 18
   LOCATION           = FLOOR-V1
   POLYGON            = "LongTrapPoly"
   .....

"Flr:1 Spc:N > North Wall" = EXTERIOR-WALL
   LOCATION           = SPACE-V1
   .....
```

Underground Surfaces

The U-EFFECTIVE keyword in the UNDERGROUND-FLOOR and UNDERGROUND-WALL commands is no longer used. It has been replaced by two new keywords: PERIM-CONDUCT and PERIM-LENGTH. See "Underground Surfaces" in this manual.

MAX-SOLAR-SCH

For window shade control, the values in the SCHEDULE referred to by the MAX-SOLAR-SCH keyword in the WINDOW command have been changed from transmitted direct solar intensity in DOE-2.1E to total incident solar intensity (direct plus diffuse).

SYSTEMS

The most time-consuming aspect of converting the SYSTEMS section of the DOE-2.1E input file is likely to be changing all of the SCHEDULEs to include the TYPE specification (see above). Relatively simple changes are required of the ZONE and SYSTEM commands. The PLANT-ASSIGNMENT command no longer exists; all of the PLANT-ASSIGNMENT keywords have been moved to other sections of the program.

Meters

Metering capabilities have been greatly expanded. There are now ELEC-METER, FUEL-METER, STEAM-METER and CHW-METER commands to define each meter. A MASTER-METER command has been added to set the default meter connection for all electric and fuel uses either globally or by end use.

A MASTER-METER command must appear before any HVAC equipment description. If not, a MASTER-METER command will be created that defaults all electric use to EM1 (fetched from the library) and all FUEL use to FM1 (fetched from the library).

The form of the commands is:

```
MASTER-METER
   MSTR-ELEC-METER    = EM1
   MSTR-FUEL-METER    = FM1
   ..
```

Loops

Circulation Loops are a new concept, and replace the previous PLANT-ASSIGNMENT command. The concept is described in "Circulation Loops" in this manuall. See also the discussion on circulation loops under "PLANT," below.

ZONE

There are a few changes to the ZONE command. TYPE is now a required keyword that must be specified as CONDITIONED, UNCONDITIONED or PLENUM. TYPE must be either the first keyword or the second keyword after a LIKE keyword. Also, the ZONE U-name must now be different from the SPACE U-name that the zone refers to. This requires an additional ZONE keyword called SPACE to specify the associated space.

DOE-2.1E version:

```
RZ1 = ZONE
   DESIGN-HEAT-T      = 70.
   DESIGN-COOL-T      = 75.
   HEAT-TEMP-SCH      = "Heat Setpt Sch"
   COOL-TEMP-SCH      = "Cool Setpt Sch"
   etc.
   ..

RZ2 = ZONE
   LIKE RZ1  ..
```

Converted:

```
RZZ1 = ZONE
   TYPE              = CONDITIONED
   DESIGN-HEAT-T     = 70.
   DESIGN-COOL-T     = 75.
   HEAT-TEMP-SCH     = "Heat Setpt Sch"
   COOL-TEMP-SCH     = "Cool Setpt Sch"
   SPACE             = "Space RZZ1"
   etc.
   ..

RZZ2 = ZONE
   LIKE              = RZZ1
   SPACE             = RZ2  ..
```

ZONE Subcommands

ZONE-AIR, ZONE-CONTROL, and ZONE-FANS, while still accepted by the Building Design Language (BDL), *are no longer supported and should not be used.* Subcommands are incompatible with the newly implemented keyword expressions. Expressions cannot be entered within a subcommand, and the existing default expressions may not give correct results for keywords specified within a subcommand. Instead, keywords previously listed under a subcommand must be input directly into the SYSTEM command.

SYSTEM

Under the SYSTEM command, the keyword SYSTEM-TYPE has been changed to TYPE, and must be the first keyword listed.

The ZONE-NAMES keyword is no longer used. In analogy to the FLOOR/SPACE/walls input hierarchy, ZONEs are placed after the SYSTEM to which they are attached and a SYSTEM must be specified prior to any ZONE.

The keyword CONTOL-ZONE is used to specify the zone in which the thermostat is located for SYSTEMs having a control ZONE (SZRH, PSZ, RESYS, etc.).

The PLENUM-NAMES keyword has been removed.

For central systems, specify the heating and cooling loops. *(HEAT-SOURCE is now required for chilled water coil SYSTEMS)*:

DOE-2.1E version:

```
SYST-1 = SYSTEM
   SYSTEM-TYPE       =VAVS
   ZONE-NAMES        =(ZONE5-1Z,ZONE2-1Z,ZONE3-1Z,
                       ZONE4-1Z,ZONE1-1Z,PLENUM-1Z)
   PLENUM-NAMES      =(PLENUM—1Z)
   SYSTEM-CONTROL    = S-CONT
   SYSTEM-FANS       = S-FAN
   SYSTEM-TERMINAL   = S-TERM
   SYSTEM-FLUID      = S-FLUID
   RETURN-AIR-PATH   = PLENUM-ZONES
   HEAT-SOURCE       = HOT-WATER
   ..
```

Converted:

```
SYST-1 = SYSTEM  TYPE=VAVS
   MAX-SUPPLY-T       = 105.
   MIN-SUPPLY-T       = 55.
   FAN-SCHEDULE       = “The Fan Sch”
   etc.
   RETURN-AIR-PATH    = PLENUM-ZONES
   HEAT-SOURCE        = HOT-WATER
   HW-LOOP            = "Heating Loop"
   CHW-LOOP           = "Cooling Loop"
   ..

ZZONE5-1Z = ZONE
   TYPE               = CONDITIONED
   SPACE              = ZONE5-1Z ..

ZZONE2-1Z = ZONE
   TYPE               = CONDITIONED
   SPACE              = ZONE2-1Z ..

ZZONE3-1Z = ZONE
   TYPE               = CONDITIONED
   SPACE              = ZONE3-1Z ..

ZZONE4-1Z = ZONE
   TYPE               = CONDITIONED
   SPACE              = ZONE4-1Z ..

ZZONE1-1Z = ZONE
   TYPE               = CONDITIONED
   SPACE              = ZONE1-1Z ..

ZPLENUM-1Z = ZONE
   TYPE               = PLENUM
   SPACE              = PLENUM-1Z ..
```

Note that the zone names in the DOE-2.1E version above are changed in the conversion since the zone names are no longer the same as the space names (see Zone discussion above).

SYSTEM Subcommands

SYSTEM-CONTROL, SYSTEM-AIR, SYSTEM-FANS, SYSTEM-TERMINAL, SYSTEM-FLUID, and SYSTEM-EQUIPMENT, while still accepted by the Building Design Language (BDL), *are no longer supported and should not be used*. Subcommands are incompatible with the newly implemented keyword expressions. Expressions cannot be entered within a subcommand, and the existing default expressions may not give correct results for keywords specified within a subcommand. Instead, keywords previously listed under a subcommand must be input directly into the SYSTEM command.

Plant Assignment

The functionality of the PLANT-ASSIGNMENT concept has been expanded using CIRCULATION-LOOPS, and the PLANT-ASSIGNMENT command has been eliminated. All of the keywords associated with this command have been moved to other commands. Miscellaneous energy use is now handled with the ELEC-

METER, FUEL-METER, STEAM-METER and CHW-METER commands using the keywords INTERIOR-POWER, INTERIOR-SCH, EXTERIOR-POWER and EXTERIOR-SCH.

PLANT

The PLANT section has been almost entirely restructured into a more consistent and intuitive format. The DOE-2.1E keywords under PLANT-EQUIPMENT, PLANT-PARAMETERS, PART-LOAD-RATIO, and EQUIPMENT-QUAD have been moved to their respective equipment commands, i.e., CHILLER, BOILER, PUMP, etc.

Circulation Loops

For central systems, a *heating loop* and a *cooling loop* should be defined. If a cooling tower is included in the plant, a *condenser loop* should also be defined. Each of these loops must have an associated loop *pump*. To simulate a domestic hot water heating, a *domestic hot water loop* should also be defined. This loop may have a *pump* (to recirculate water), but a pump is not required.

Note that you can define more than one loop of any given type. For example, your project may have three chilled water loops that operate completely separately from each other. In this case, your various HVAC systems can be attached to different (or the same) loops and one or more separate chillers will have to be created and attached to each of these loops.

This capability replaces the old PLANT-ASSIGNMENT concept, but gives you far more flexibility in how you serve your loads. For example, individual zone reheat coils may be attached to different hot-water loops, even though the zones are served by the same central air handler.

Primary/secondary loop arrangements are also possible, as are various pumping arrangements. The concept of circulation loops is a major expansion of the program's capabilities, and you are urged to read the documentation on this feature before attempting to use this feature in more than its most basic form. See "Circulation Loops" in this manual and "CIRCULATION-LOOP" command in the *DOE-2.2 Dictionary*.

The following commands define the basic circulation loops that are referenced by other plant and system commands.

```
"Heating Loop" =CIRCULATION-LOOP
   TYPE               = HW
   LOOP-OPERATION     = DEMAND
   LOOP-PUMP          = "Heating Pump"
   ..

"Cooling Loop" =CIRCULATION-LOOP
   TYPE               = CHW
   LOOP-OPERATION     = DEMAND
   LOOP-PUMP          = "Cooling Pump"
   ..

"Condenser Loop" =CIRCULATION-LOOP
   TYPE               = CW
   LOOP-OPERATION     = DEMAND
   LOOP-PUMP          = "Condenser Pump"
   ..
```

All loops must now be defined, the program does not default any loops.

Chillers

Nine chiller types have been implemented as of this writing: electric-hermetic-centrifugal (ELEC-HERM-CENT), electric-open-centrifugal (ELEC-OPEN-CENT), electric-hermetic-reciprocating (ELEC-HERM-REC), electric-open-reciprocating (ELEC-OPEN-REC), heat recovery (ELEC-HTREC), single stage absorption (ABSOR-1), two stage absorption (ABSOR-2), direct gas-fired absorption (GAS-ABSOR), and gas engine-driven (ENGINE).

CONDENSER-TYPE is a required input for chillers; it can be WATER-COOLED or AIR-COOLED.

All keywords regarding the chiller are specified within the CHILLER command. In other words, chiller-related keywords that used to be specified using the PART-LOAD-RATIO, PLANT-PARAMETERS and EQUIPMENT-QUAD commands are now defined within the CHILLER command. Each CHILLER command specifies a single chiller and all chillers are specified separately even if identical (INSTALLED-NUMBER and MAX-NUMBER-AVAIL are no longer used). This allows the performance characteristics of each chiller to be different from all others. For example, previously, all open-centrifugal chillers had to have the same electric input ratio and performance curves even if they were different sizes, makes, and ages. Now, each chiller can be completely unique.

Following are the minimum specifications for a 0.7 kW/ton, 200-ton chiller connected to the defined cooling and condenser loops:

```
MAINCHLR = CHILLER
   TYPE              = ELEC-HERM-CENT
   CONDENSER-TYPE    = WATER-COOLED
   CAPACITY          = 2.4 $200 x 12000 Btuh = 2.4 MBtu/hr
   EIR               = 0.20 $(0.7/12000) x 3413
   CHW-LOOP          = "Cooling Loop"
   ..

CW-LOOP = "Condenser Loop"
   ..
```

Boilers

Following are the minimum specifications for a 7 MBtu/hr boiler connected to the defined heating loop:

```
MAINBLR = BOILER
   TYPE              = HW-BOILER
   CAPACITY          = 7.0
   HW-LOOP           = "Heating Loop"
   ..
```

Towers

Cooling towers are defined with the HEAT-REJECTION command. Following are the minimum specifications for a cooling tower connected to the defined condenser loop:

```
CTOWER = COOLING-TWR
   TYPE              = OPEN-TWR
   CW-LOOP           = "Condenser Loop"  ..
```

Thermal Storage

Most of the keywords contained in the THERMAL-STORAGE command are the same as the previous DOE-2.1E keywords that were scattered across the PLANT-EQUIPMENT, PART-LOAD-RATIO, PLANT-PARAMMETERS and ENERGY-STORAGE commands, but are now combined together in one command.

Energy Meters

The electric and fuel meters have been expanded from 5 each to 100 and 15 each, respectively. You now define these meters using the ELEC-METER and FUEL-METER commands, and may give each meter your own name. The ENERGY-RESOURCE command has been eliminated and all characteristics of the energy (consumption units, etc.) are now defined within the meter command. As before, you define the cost structure of the energy in the UTILITY-RATE command.

The ELEC-METER command is a TYPE command; the TYPE can be either UTILITY, BUILDING or SUB-METER. All ELEC-METERs with TYPE=SUB-METER must be included in the BLDG/SUB-METER list of a UTILITY meter or the SUB-METER list for an ELEC-METER with TYPE=BUILDING. Similarly, all ELEC-METERs with TYPE=BUILDING must be included in the BLDG/SUB-METER list of a UTILITY meter.

```
"EM20" = ELEC-METER
   TYPE               = UTILITY
   SOURCE-SITE-EFF    = 0.37
   ..
```

In addition, purchased steam and chilled water may be specified using the STEAM-METER and CHW-METER commands.

If you do not define any meters, you may still assign various loads to different meters using the default meters predefined in the library. For electricity, these are EM1 through EM25, and for fuel they are FM1 through FM25. You use these meter names identically to the old names (the old codewords M1, M2, M3, M4, and M5 no longer exist).

To change the default hierarchy of the meters, use the MASTER-METERS command. This command replaces the function of the MSTR-ELEC-METER, MSTR-FUEL-METER, etc., keywords that used to be in the PLANT-ASSIGNMENT command.

Pumps

Pumps are now specified with the PUMP command. See "PUMP Command" in the *DOE-2.2 Dictionary* for a description of the keywords available for pumps. For most central systems, a heating pump and a cooling pump should be defined. If a cooling tower is used a condenser pump should also be defined. The following defines the necessary pumps for a central system with a cooling tower:

```
"Heating Pump" = PUMP
   CAP-CTRL           = ONE-SPEED-PUMP  ..

"Cooling Pump" = PUMP
   CAP-CTRL           = VAR-SPEED-PUMP  ..

"Condenser Pump" = PUMP  ..
```

Other Components

The LOAD-MANAGEMENT and LOAD-ASSIGNMENT commands have been replaced with new LOAD-MANAGEMENT and EQUIP-CTRL commands that offer expanded capabilities. The HEAT-RECOVERY command has been replaced using the expanded capabilities of the CIRCULATION-LOOP and EQUIP-CTRL commands. The ELEC-GENERATOR and THERMAL-STORAGE command are used to define different types of generator sets and storage tanks.

Additional Changes

The statements: INPUT PLANT, END and COMPUTE PLANT are eliminated

ECONOMICS

Again, the most time-consuming aspect of converting the ECONOMICS section of the DOE-2.1E input file is likely to be changing all of the relevant SCHEDULEs to include the TYPE keyword (see above). Otherwise, very few of the ECONOMICS specifications have changed, with the following exceptions:

UTILITY-RATE Command

The RESOURCE keyword is changed to TYPE.

The PCT-TAX-DATA keyword is changed to PCT-TAX-BLOCKS and PCT-TAXES.

The UNIT-TAX-DATA keyword is changed to UNIT-TAX-BLOCKS and UNIT-TAXES.

The PCT-SRCHG-DATA keyword is changed to PCT-SRCHG-BLOCKS and PCT-SRCHGS.

The UNIT-SRCHG-DATA keyword is changed to UNIT-SRCHG-BLOCKS and UNIT-SRCHGS.

BLOCK-CHARGE Command

A new keyword has been added, BLOCKS-ARE. Refer to the *DOE-2.2 Dictionary* for information on this keyword.

The BLOCK1-DATA keyword, which required a list of values of different units (sets of block level, costs, and (optionally) limits) is replaced with three keywords. Use the following conversion as a guideline:

DOE-2.1E version:

```
WINTER-ON-P = BLOCK-CHARGE
   BLOCK-SCH          = SEASONS-SCH
   SCH-FLAG           = 3
   BLOCK1-TYPE        = KWH/KW
   BLOCK1-DATA        = (200,.050,0,
                         800,0.060,0)  ..
```

Converted:

```
WINTER-ON-P = BLOCK-CHARGE
   BLOCK-SCH          = SEASONS-SCH
   SCH-FLAG           = 3
   BLOCK1-TYPE        =  KWH/KW
   BLOCKS-1           = (200,800)
   COSTS-1            = (.050,.060)
   LIMITS-1           = ( 0  , 0)   ..
```

.

Section

2

Building Description Language

OVERVIEW OF BDL ELEMENTS

Using familiar English words, the Building Description Language (BDL) allows you to easily enter information required to determine the energy consumption of a building. The BDL Processor processes the instructions directed to it, which involves checking for errors, issuing diagnostic messages, and preparing the input data for the appropriate simulation program.

Several features of BDL may be used to reduce input preparation time. For instance, many variables may be omitted to allow BDL to assign values by default. These default values are described with each instruction and, if desired, are listed on various verification reports, and may be printed along with the input instructions.

All instructions are processed before any simulation takes place. If any errors are detected that are considered fatal, no simulation occurs. You may designate what severity of errors will be allowed before simulation is prohibited.

BDL input is free-format; you are not required to enter data in a rigid column format. However, experienced users will recognize the advantage of having the instructions in a format that is organized for quick reference and that is easily understood by others.

Data are input to BDL using statements called instructions. BDL instructions take the form:

```
U-name = Command
   Keyword           = Value
   Keyword           = Value
   etc.              = Value
   ..
```

where U-name is a user-specified name assigned to a particular instruction. "Command" indicates the type of instruction (and therefore the type of data to follow). "Keyword = Value" is one set of data for an instruction. The instruction terminator is defined to be two successive periods preceded and followed by a blank character. The details of each instruction are described in the *DOE-2.2 Dictionary*.

Terminator

The symbol for the terminator is .. (two periods with no space between them and with a blank space between the terminator and anything else. All BDL instructions are terminated with this BDL terminator. If a terminator is missing, BDL attempts to identify this error, but issues an ERROR as the meaning of the input can be ambiguous without properly placed terminators.

For example, the instruction

```
AIR-LAYER = MATERIAL
   TYPE               = RESISTANCE
   RESISTANCE         = 0.90
   ..
```

defines a construction layer in a wall or roof, assigns it the user-specified name of AIR-LAYER, and sets its thermal resistance to 0.90 hr-ft2-F/Btu.

Input Styles

BDL allows you to format your input with a text editor in a way that makes it most readable to you and others. Following are examples of different ways of formatting the same input for a SPACE command.

```
OFFICE = SPACE
   AREA               = 1000.
   PEOPLE-SCHEDULE    = OCCUPY-1
   AREA/PERSON        = 110
   LIGHTING-SCHEDULE  = LIGHTS-1
   LIGHTING-W/AREA    = 1.5
   ..

OFFICE = SPACE      AREA               1000
                    PEOPLE-SCHEDULE    OCCUPY-1
                    AREA/PERSON        110
                    LIGHTING-SCHEDULE  LIGHTS-1
                    LIGHTING-W/AREA    1.5 ..

OFFIC=SPACE
           AREA=1000
           PEOPLE-SCHEDULE=OCCUPY-1  AREA/PERSON=110
           LIGHTING-SCHEDULE=LIGHTS-1  LIGHTING-W/AREA=1.5 ..
```

This manual most commonly uses the first structure, as this is similar to the BDL generated by eQUEST

Rules:

1. One or more blanks must separate each element of an instruction.
2. The equals sign between a U-name and a command is optional.
3. A U-name may be required, optional, or not allowed, depending upon the particular command.
4. A valid command (or command abbreviation) must be the second item, or the first if no U-name is specified.
5. If a U-name is present, the U-name and the command must appear on the same line.
6. Commands may not be misspelled or contain embedded blanks.
7. Every instruction must end with the instruction terminator, defined to be two successive periods preceded and followed by a blank character.

U-name

A U-name is a unique name that you assign to identify a particular BDL instruction. U-names are used to distinguish multiple instructions of the same type. You should define a U-name to be whatever word most appropriately describes the data to follow. Some instructions require a U-name, other instructions permit an optional U-name, and still others do not allow a U-name at all. Optional U-names should be entered in an instruction only if that instruction is referred to in another instruction.

Rules:

1. Although a U-name may be longer than 32 characters, only the first 32 characters are significant to the program.

2. A U-name must be unique, that is, one of a kind.

3. A U-name may not be the same as any BDL command, keyword, or code-word.

4. A U-name may not contain any of the following characters: ( ) [ ] , = or a blank.

5. U-names can contain blanks if the U-name is enclosed in double quotes.

Example:

```
Valid U-names
   WALL-4
   "WALL 4"
   FAN-3.6
   BOILER-G1

Invalid U-names
   U-name            Reason
   WALL,D            Contains a comma
   END               Is a command
   FAN(3.6)          Contains parentheses
```

Data

With very few exceptions, all data entered after the command, and before the instruction terminator, take one of the following forms:

```
Keyword = Value
```

or

```
Keyword = (Value, Value,...)
```

where "Keyword" is a valid keyword, or keyword abbreviation, for the instruction being processed, and "Value" is a value to be assigned to the keyword.

Rules:

1. An equals sign following the keyword is optional

2. If the keyword does not require a list of values, the value must not appear in parentheses.

3. If the keyword requires a list of values, the value(s) must appear within parentheses and all values must be separated by commas and/or one or more blanks. Normally, a list contains two or more values but occasionally a list will contain only one value , in which case the parentheses are optional.

4. Keywords not entered will be set to their default values, if such default values exist. If default values do not exist for a mandatory keyword, a diagnostic message will be printed.

5. A keyword may not be misspelled or contain embedded blanks.

Keywords

There are several different types of keywords. Most keywords are of the numeric type and must be assigned numeric values. However, if a keyword is of the code-word, U-name, or literal type, it must be assigned the corresponding type of value. Code-words are symbolic values (YES, NO, ON, OFF, etc.) assigned to keywords of the code-word type. U-names are described in the U-Name section in this chapter. Literals are values that are interpreted "literally" by the program, such as the values assigned in the TITLE command.

Rules:

1. Numeric values may be input as integers, decimal fractions, or exponential numbers. The following three values are equivalent and may be used interchangeably: 600, 600.0, and 6E+2.
   a. Blanks may not occur after a minus sign or be embedded in the number.
   b. Plus signs (+) may appear only in an exponent.

2. Code-word values must be selected from the list of valid code-words as described with the keyword.

3. U-name values must be defined (appear in the U-name field of an instruction) before the END instruction is reached.

4. Literal values must begin and end with an asterisk (*) and may not be continued to the next line.

5. Code-words must not be misspelled or contain embedded blanks.

Comments

Comments may be inserted in or between the instructions to aid in the documentation of the input data. Comments are not processed in any way by the program, other than to print them along with the instructions.

Rules:

1. A comment must begin with a dollar sign ($).

2. If the dollar sign is not in column 1, the dollar sign must be preceded by a blank.

3. A comment is terminated by the end of the current line, or by a second dollar sign, whichever occurs first.

4. A comment may not occur within a literal value (see section on keywords).

Example:

```
$EXECUTIVE SUITE
```

```
BIG-GLASS = WINDOW $IN EXECUTIVE SUITE $
   HEIGHT             =7
   WIDTH               30
   ..
```

LIKE

The LIKE keyword is used to duplicate the user-supplied data input in a previous instruction (default values are not taken from the previous instruction but are recalculated by the program for the current instruction). The value entered for LIKE is the U-name of a previous instruction of the same type. Any data that differ from the data of the previous instruction may be specified in the normal manner. For example, the instruction

```
BLUE-ROOM = SPACE
   LIKE               = RED-ROOM EXCEPT
   WIDTH              = 30.0
   LENGTH             = 50.5
   ..
```

defines a new space named BLUE-ROOM that has all the attributes of a previouslydefined space named RED-ROOM, but with different values for its width and length.

The LIKE command will duplicate only user input keywords and keyword values, not program-calculated values. For example, in the instruction

```
BROWN-ROOM = SPACE
   LIKE               = BLUE-ROOM ..
```

the BROWN-ROOM will be assigned a WIDTH of 30.0 and an LENGTH of 50.5, not an AREA of 1515, the product of 30.0 and 50.5.

Rules:

1. The LIKE keyword may be used only in those instructions having LIKE listed as a valid keyword.

2. If the LIKE keyword is used, it must be the first keyword used in an instruction.

3. The instruction, whose name is used as the value for LIKE, must be entered before the instruction containing the LIKE keyword.

4. The word EXCEPT, after the value for LIKE, is optional.

5. Only instructions of the same type of command may use the LIKE keyword. Example: A DOOR instruction cannot use the U-name of WINDOW as a LIKE keyword value.

6. The code-words, which identify materials and walls in the Library of MATERIALs, LAYERS and CONSTRUCTIONs (BDLLIB), are not U-names. Therefore, these code-words cannot be used in a LIKE keyword.

If you have specified an incorrect value in an instruction, which is subsequently referenced in other instructions with a LIKE keyword, the error diagnostics will not be repeated in the subsequent instructions.

Example:

```
WINDOW-1 = WINDOW
   HEIGHT             = 5
   WIDTH              = 3
   DEPTH              = 4 ..
   ------------------------------------------
   ERROR                          DEPTH         UNKNOWN KEYWORD
   ------------------------------------------

WINDOW-2 = WINDOW
   LIKE WINDOW-1 ..

WINDOW-3 = WINDOW
   LIKE WINDOW-1 ..
```

DEPTH is not a valid keyword for WINDOW, therefore, it was rejected for WINDOW-1 and an error diagnostic was printed. Although WINDOW-2 and WINDOW-3 included the LIKE WINDOW-1 keyword and value, DEPTH = 4 in both cases will be rejected, but the error diagnostic will not be repeated.

Subcommands and Referenced Commands

Subcommands

Subcommands, while still accepted by the Building Design Language (BDL), *are no longer supported and should not be used*. Subcommands are incompatible with the newly implemented keyword expressions. Expressions cannot be entered within a subcommand, and the existing default expressions may not give correct results for keywords specified within a subcommand. Instead, keywords previously listed under a subcommand must be input directly into the SYSTEM command.

Referenced Commands and U-names

There are keywords that are really referenced commands and whose values are user-created names (U-names). A referenced command is a command in one instruction and a keyword in a subsequent instruction. The use of referenced commands allows you to collect data in one instruction and to use the same data in several later instructions.

RUN-PERIOD

The RUN-PERIOD command is used to specify the initial and final dates of the desired simulation period. The initial date is the first date of the simulation, given in the form: month day year. The code-words that specify the names of the months are given below. The final date is the last simulation date, specified in the same manner as the initial date. U-name is not allowed.

The code-words for the months are:

JAN	FEB	MAR	APR	MAY	JUN
JUL	AUG	SEP	OCT	NOV	DEC

SCHEDULE, WEEK-SCHEDULE AND DAY-SCHEDULE

Schedules are hourly profiles of quantities like lighting power, occupancy, and thermostat setpoint. The format of schedules is the same in LOADS, HVAC, and ECONOMICS input.

The SCHEDULE command references two related commands, DAY-SCHEDULE, which defines the hourly profile for a particular type of day (such as weekday, weekend day, or holiday), and WEEK-SCHEDULE, which defines what day schedules make up a weekly schedule. The following describes DAY-SCHEDULE and WEEK-SCHEDULE, then shows how and SCHEDULEs are built up from WEEK-SCHEDULEs and how WEEK-SCHEDULEs are built up from DAY SCHEDULEs.

Special Schedule for Design Days

DAY-SCHEDULEs can be assigned to different days of the week or to holidays. In WEEK-SCHEDULE it is also possible to assign a schedule to a heating design day (HDD) or a cooling design day (CDD). See the Loads Topic "Design Day" for a discussion and examples

DAY-SCHEDULE

In its simplest form, the input for DAY-SCHEDULE is:

```
U-name = DAY-SCHEDULE
   TYPE              = MULTIPLIER
                       (all 24 hours covered)
                       (values for each hour) ..
```

Example input:

```
LTG-1=DAY-SCHEDULE
   TYPE              = MULTIPLIER
                      (1,24)(0,0,0,0,0,0,0,0,
                             0.3,0.6,0.8,1,1,1,1,1,
                             1,1,0,0,0,0,0,0) ..
```

Optionally, this can be shortened by writing

```
LTG-1=DAY-SCHEDULE
   TYPE              = MULTIPLIER
   ( 1, 8)           (0)
   ( 9,11)           (0.3,0.6,0.8)
   (12,18)           (1)
   (19,24)           (0) ..
```

which is representative of a week-day daily profile.

Hour 1 is midnight to 1 am, hour 2 is 1 am to 2 am, etc. For example, (12,18)(1) (19,24)(0), above, means that the lights are fully on from 11 am to 6 pm and fully off from 6 pm to midnight.

In this example, and all input examples, the alignment of keywords and values is strictly arbitrary. While this particular format is easy to read, it is not mandatory.

In these examples, TYPE=MULTIPLIER indicates that the schedule value "multiplies" some quantity, such as lighting power. Other schedule types are possible, e.g. TYPE='TEMPERATURE, ON/OFF, etc.

In the following, let's assume that the day schedule for week-ends and holidays looks like:

```
LTG-2 = DAY-SCHEDULE
   TYPE              = MULTIPLIER
   (1,24)            (0) ..
```

Next we show how day schedules can be combined into week schedules.

WEEK-SCHEDULE

So far we have two DAY-SCHEDULEs -- LTG-1 represents week-days and LTG-2 represents week-ends and holidays. The form of the WEEK-SCHEDULE is:

```
U-NAME = WEEK-SCHEDULE
   (days of week covered)  (U-name of DAY-SCHEDULE) .
```

Using the previously defined DAY-SCHEDULEs, the example can be carried forward with:

```
NORMAL = WEEK-SCHEDULE
   TYPE              = MULTIPLIER
   (MON,FRI)         LTG-1
   (SAT,HOL)         LTG-2 ..
```

where (MON,FRI) includes MON,TUE,WED,THU,FRI and (SAT,HOL) includes SAT,SUN,HOL.

Optionally, this can be shortened to:

```
NORMAL = WEEK-SCHEDULE
   TYPE              = MULTIPLIER
   (WD)              LTG-1
   (WEH)             LTG-2 ..
```

where (WD) stands for week-days and (WEH) for week-ends and holidays. If Saturday is considered part of the normal week, you have to write (MON,SAT) LTG-1 and (SUN,HOL) LTG-2.

Next we combine different WEEK-SCHEDULEs into a schedule that covers the entire year.

SCHEDULE

To illustrate the purpose of SCHEDULE, assume we have a school that is closed in the summer and on week-ends and holidays. Therefore, we need another week type:

```
VACATION = WEEK-SCHEDULE
   TYPE              = MULTIPLIER
   (ALL)             LTG-2 ..
```

where (ALL) stands for all days of the week, including holidays, and LTG-2 was the DAY-SCHEDULE representing lights as being off for 24 hours.

In its simplest form, SCHEDULE takes the form:

```
U-NAME = SCHEDULE
   TYPE              = MULTIPLIER
   THRU (calendar period covered)  (U-name of WEEK-SCHEDULE) .
```

To finalize the example:

```
LIGHTS = SCHEDULE
   TYPE              = MULTIPLIER
   THRU JUN 10       NORMAL
   THRU SEP 5        VACATION
   THRU DEC 31       NORMAL ..
```

.

METRIC OPTION

The program allows input in either English or metric units. Output reports, both hourly and summary, can also be in either English or metric units. The default for the input and output is English units. To specify metric input and output, use the INPUT-UNITS and OUTPUT-UNITS keywords in the INPUT or PARAMETRIC-INPUT command:

```
INPUT
   INPUT-UNITS        = METRIC
   OUTPUT-UNITS       = METRIC ..
```

Note that units for input and output are independent. The program can accept metric input and produce English output, and vice versa.

Defaults and Limits

For metric input, the program obtains the maximum, minimum, and default values for keywords by converting the English values.

Libraries

The program is supplied with both English and metric versions of all library files. The appropriate file is used automatically.

Unit Conversion Table

The following table (Table 4)shows all the units used by the program, along with the conversion factor from English to metric.

Table 4 Unit Conversion Table

Metric Unit	English Unit	English to Metric
WH	BTU	0.293000
WATT	BTU/HR	0.293000
J/KG-K	BTU/LB-F	4183.830078
W/M2-K	BTU/HR-SQFT-F	5.674460
DEGREES	DEGREES	1.000000
C	F	0.555556
M2	SQFT	0.092903
M3	CUFT	0.028317
KG/HR	LB/HR	0.453592
KG/M3	LB/CUFT	16.018459
M/S	MPH	0.447040
W/K	BTU/HR-F	0.527178
M	FT	0.304800
W/M-K	BTU/HR-FT-F	1.729600
WATT/M2	BTU/HR-SQFT	3.152480
CM	IN	2.540000
UNITS/CM	UNITS/IN	0.393700
UNITS	UNITS	1.000000
KG	LB	0.453592
FRAC.OR MULT.	FRAC.OR MULT.	1.000000
HRS	HOURS	1.000000
PERCENT-RH	PERCENT-RH	1.000000
M3/H	CFM	1.699010
MM-WATER	IN-WATER	25.400000
KG/M2	LB/SQFT	4.882400
KW	KW	1.000000
W/M2	W/SQFT	10.763920
THERMIES	THERMS	25.000000
M/SEC	KNOTS	0.514440
M2-K/W	HR-SQFT-F/BTU	0.176228
$DOLLARS	$DOLLARS	1.000000
MWATT	MBTU/HR	0.293000
YEARS	YEARS	1.000000
$/HR	$/HR	1.000000
HRS/YEARS	HRS/YEARS	1.000000
PERCENT	PERCENT	1.000000
$/MONTH	$/MONTH	1.000000
LITERS/MIN/KW	GALLONS/MIN/TON	1.078000
WH/KG	BTU/LB	0.645683
MBAR-GAGE	LBS/SQIN-GAGE	68.947571
$/UNIT	$/UNIT	1.000000
W/PERSON	BTU/HR/PERSON	0.293000
KGS/KG	LBS/LB	1.000000
KWH/KWH	BTU/BTU	1.000000
KG/KW	LBS/KW	0.453590
REV/MIN	REV/MIN	1.000000
KW/TON	KW/TON	1.000000
MWH	MBTU	0.293000
LITER	GAL	3.785410

Metric Unit	English Unit	English to Metric
LITERS/MIN	GAL/MIN	3.785410
J/K	BTU/F	1897.800049
KWH	KWH	1.000000
$/UNIT-HR	$/UNIT-HR	1.000000
KW/M3/HR	KW/CFM	0.588500
J/M2-K	BTU/SQFT-F	20428.400391
HR/HR	HR/HR	1.000000
J/M-K	BTU/FT-F	6226.479980
K	R	0.555556
MBAR	INCH MER	33.863800
UNITS/LITER/MIN	UNITS/GAL/MIN	0.264170
(M2-K/W)2	(HR-SQFT-F/BTU)2	0.031056
KW	KBTU/HR	0.293000
KWH	KBTU	0.293000
L/S	CFM	0.471900
M3/H-M2	CFM/SQFT	18.288000
1/K	1/R	1.799900
SEC/M	1/KNOT	1.943860
LUX	FOOTCANDLES	10.763910
CANDELA/M2	FOOTLAMBERT	3.426259
LUMEN/WATT	LUMEN/WATT	1.000000
KWH/M2-YR	KBTU/SQFT-YR	3.152480
C (DELTA)	F (DELTA)	0.555556
WATT	BTU/DAY	0.012202
$/YEAR	$/YEAR	1.000000
WATT/WATT	BTU/WATT	0.293000
RADIANS	RADIANS	1.000000
WATT/WATT	WATT/BTU	3.413000
KWH	BTU	0.000293
WATT	WATT	1.000000
LUMENS	LUMENS	1.000000
W/M-K2	BTU/HR-FT-R2	3.115335
KG/M-S	LB/FT-S	1.488163
KG/M-S-K	LB/FT-S-R	2.678693
KG/M3-K	LB/CUFT-R	28.833212
W/M-K	BTU/HR-FT-R	1.730741
M3	THERM	2.831700
M3/HR	THERM/HR	2.831700
TONNE	TON	0.907180
TONNE/HR	TON/HR	0.907180
BTU/UNIT	BTU/UNIT	1.000000
$	$	1.000000
KW/LITER/MIN	KW/GAL/MIN	0.264170
M3-MIN/H-LITERS	CUFT/GAL	0.448831
MINUTES	MINUTES	1.000000
KG/MWH	LB/MBTU	1.548100

Special Cases

1. The WINDOW keywords SWITCH-SET-LO and SWITCH-SET-HI should always be in English units, even for metric runs.

2. In the CURVE-FIT command, input data is converted automatically, but you should be careful to choose the correct TYPE, since this information is used in doing the unit conversion.

Changes from DOE-2.1E

Metric input is now much easier than in DOE-2.1E, since there are far fewer special cases. The most important example of this occurs in specifying schedules: the keywords VALUES, TEMPERATURES, and RADIATIONS are no longer needed in metric input. All the schedule commands now look the same for English and metric input.

KEYWORD DEFAULTING

Each keyword in a command is either assigned a default value or is a required input keyword; i.e., you are required to provide an input. Defaults are of three types: fixed, TYPE-dependent, or expressionized (rule based).

Fixed Defaults

These defaults are quite simple. A single value (or list of values for keywords that take a list) is used in all situations. An example is AREA/PERSON in the SPACE command. This keyword is assigned a default value of 100 ft2. However, fixed defaults are relatively rare.

TYPE-Dependent Defaults

These defaults are set by classifying the object (command) being entered. The classification is set with the TYPE keyword. All commands with a TYPE keyword use TYPE-dependent defaults. One example is the DESIGN-DAY command: for TYPE=HEATING, WIND-SPEED defaults to 13 knots (about 15 mph); for TYPE=COOLING, WIND-SPEED defaults to 6.5 knots (about 7.5 mph).

TYPE-dependent defaulting is also used to make a keyword unused for one choice of TYPE, but required for another. For instance, in the MATERIAL command, the keyword DENSITY is required for TYPE=PROPERTIES, but is unused for TYPE=RESISTANCE.

Default Expressions

In many cases, TYPE-dependent defaults don't provide enough flexibility. The desired default may depend on input to other keywords, possibly in other commands. One example occurs in the WINDOW command: if you input OVERHANG –D, OVERHANG-W is defaulted to the window WIDTH; otherwise OVERHANG-W is unused.

Defaults set by expressions can be quite complex and sophisticated. The expressions are written in the keyword expression syntax, described in "Keyword Expressions" in this manual.

Special Cases

There are a few defaults that don't fall into any of the above categories. The default LATITUDE and LONGITUDE and GROUND-T are taken from the weather file, for instance. The simulation may sometimes set default values. Finally, many keyword values are filled by the design or sizing calculations in the HVAC simulation. These are not considered defaults but rather the result of a design calculation.

How Defaulting Is Done

Files

Before describing the actual defaulting process, it is necessary to describe some files that come with the program or are created by BDL or a utility program.

BDLKEY.BIN	the initial keyword file (binary); this file includes the fixed and TYPE-dependent keyword defaults. It is created by the utility program *key* using the *dkey.cmp* file (ASCII) as input.
BDLKEY.NEW	final keyword file (binary) with parsed default expressions added. This file is created by BDL using *bdldft.inp* and *bdldft.txt* as input.
BDLKEYm.NEW	is the metric version of BDLKEY.NEW, created using *bdldftm.inp* and *bdldft.txt*.

bdldft.txt	an ASCII file (BDL input format) containing the official default expressions, both metric and English. It is recommended that this file not be altered by the user.
bdldft.inp	BDL input file used in creating BDLKEY.NEW or BDLKEYm.NEW. Basically tells BDL whether to create a metric or English unit keyword file. *bdldftm.inp* is the metric version.
bdldft.dat	BDL input file read each time BDL starts up. This is the place to put permanent user-created default expressions, which override the official defaults. Temporary or one-time defaults should be set with a SET-DEFAULT command in the regular BDL input file.

In a simulation, BDL uses BDLKEY.NEW (or BDLKEYm.NEW for metric input) and *bdldft.dat*. All the other files described above (except *bdldft.dat*) are used in creating BDLKEY.NEW and BDLKEYm.NEW.

Creating the Keyword Files

The files BDLKEY.NEW and BDLKEYm.NEW are created in a two-step process. The first step consists of running the utility program *key* with the file *dkey.cmp* as input. This creates a binary keyword file *BDLKEY.BIN* that contains the command/keyword tables used by BDL to interpret your input. The file contains the fixed and TYPE-dependent keyword defaults as well as minima and maxima for each keyword. All values in this file are in English units. Executing *key* also produces the ASCII file *BDLKEY.OUT*, which shows all the commands and keywords, fixed and TYPE-dependent defaults, and minima and maxima in a human readable format. This file is in English units only.

The second step is to execute BDL with *bdldft.inp* (*bdldftm.inp* for metric), *bdldft.txt* and *BDLKEY.BIN* as input. BDL first reads *bdldft.inp* or *bdldftm.inp*. This tells BDL whether to produce a metric or English unit keyword file. BDL then switches to reading *bdldft.txt*. This file contains default expressions, for both metric and English units, in the keyword expression syntax. This file is ASCII and human readable if the expression syntax, described in "Expressions," has been mastered. BDL parses the default expressions and stores them on the new keyword file. If a metric keyword file is being created, the fixed and TYPE-dependent defaults and the minima and maxima are converted to metric units. The result is a final keyword file *BDLKEY.NEW* or *BDLKEYm.NEW* which contains all the command and keyword data in the correct units needed by BDL to process the user input.

The Defaulting Process

Defaults are assigned to keywords by BDL as it is processing your input file. The procedure is somewhat different depending on whether the simulation engine is being run in batch mode or is being run interactively with a graphical user interface (GUI). In the batch mode BDL reads the data for an entire command before assigning defaults. In other words, defaulting is not done until BDL sees the double period in the input that indicates completion of a command. Then, using the information from the keyword table, defaults are assigned to keywords that do not have user-set inputs. A keyword may have both a fixed or TYPE-dependent default and a default expression. In this case the default expression always takes precedence. The default expression for a keyword may refer to data from other commands. In this case these commands must already have been scanned by BDL for the expression to work properly. In other words, these commands must precede the command that contains expressions referring to them. For instance, the EXTERIOR-WALL command has default expressions that refer to the SPACE containing the wall. For the defaulting to work, the SPACE command should precede the EXTERIOR-WALL command in the input file. Similarly an EXTERIOR-WALL default expression may use data contained in the CONSTRUCTION of the wall. Again, this CONSTRUCTION command should precede the EXTERIOR-WALL command.

For interactive input with a GUI, defaulting has to be done somewhat differently. The GUI will force the user to fill in all the required input for an item and then assign defaults for the remaining keywords. Each time an input is added or changed, defaulting will be repeated. This is necessary because interactive input is not sequential – you can

go back and change a previous input. Of course when an input is changed, this will affect any default that is the result of an expression that refers to that input. Thus, if the HEIGHT of a SPACE is changed, the default expressions of all the walls in that space have to be re-executed. For interactive input, BDL performs all these operations automatically, without any user intervention.

User Control of Defaults

You can override any of the standard keyword defaults. User defaults always take precedence over the defaults supplied with the program. Defaults that you will use repeatedly, or are more or less permanent, should be put on the file *bdldft.dat*. The file should consist of SET-DEFAULT commands that define your choice of defaults. The defaults can be fixed, TYPE-dependent, or set by expression. Good examples of default expressions can be obtained by looking at *bdldft.txt*. Temporary or single case defaults can be defined in the normal BDL input using SET-DEFAULT.

Since any fixed or TYPE-dependent default could equally well be expressed using a default expression, the user might wonder which to use. A general rule is that default expressions should not be used unless needed. The fixed and TYPE-dependent defaults are easier to understand and require less computer processing.

Given the complexity of the keyword defaulting, how can you be sure what default is actually being used? For batch simulations, it is best to run the input through BDL with DIAGNOSTIC COMMENTS turned on. This will tell BDL to echo out all *active* keyword values. The output will indicate those values that are defaults and whether the default was a standard keyword default or was set by the user. Only *acitve* keywords will be echoed; all *inactive* keywords will be suppressed (an inactive keyword is one that BDL has determinedis not currently needed in the simulation of your specific input). For interactive input, the GUI will display the current default values.

Differences From DOE-2.1E

In DOE-2.1E there were only fixed defaults and special-case defaults. Most of the special-case defaults were hard-coded Fortran statements within the BDL program. Obviously these hard-coded defaults were inaccessible to the user. The expression capability was developed in order to replace these hard-coded defaults with expressions that could be inspected and altered by an ordinary user. TYPE-dependent defaults were not available in DOE-2.1E. Again, this capability replaces much hard-coded defaulting in DOE-2.1E.

KEYWORD EXPRESSIONS

This section describes how to specify input values through the use of the keyword expression language. An expression allows the value of a keyword to dynamically default according to the value of another keyword, or a parameter. The program utilizes expressions heavily in order to provide more intelligent defaulting.

For example, depending on whether SYSTEM:HEAT-SOURCE = HEAT-PUMP or HOT-WATER, a series of expressions will pull different curves out of the library to describe the performance of the HVAC system.

Expressions can be used for most keyword values by placing an expression within { }.

The opening **{** must be the first character, other than spaces, after the keyword. The expression code can begin immediately or on any following line. The expression must be ended with a **}**. The length of an expression cannot exceed 4096 characters (new line counts as two characters.)

An example of BDL including an expression is:

```
Flr1-SpcN-WallN-Win1 = WINDOW
   GLASS-TYPE         = Lib1205-Grey
   FRAME-WIDTH        = 0.25
   X                  = 1.5
   Y                  = 3
   HEIGHT             = 12
   WIDTH              = {LOCAL("HEIGHT") * 0.667}
   FRAME-CONDUCT      = 3.037
   ..
```

Here, the expression sets the width of the window equal to the height of the window times 0.667.

The *Expression* evaluator is a feature that lets you specify an input value in terms of other parameters with a mathematical or logical expression. The expression parsing and evaluating module supports most of the statements and operations contained in the existing Input Function feature (see "Input Functions" in this manual), with the addition of If - Else and Case statements. An expression is comprised of a single statement and will return a single value. The expression syntax is compatible with the conventions defined in both the C and Fortran programming languages.

An expression is placed into BDL input for a keyword within curly brackets (i.e., { … }) when the keyword is in a component definition command or a SET-DEFAULT command. Expressions can include elements (such as special BDL functions, math and logical operators, and logical structures) that can be used to create complex relations that are dynamically evaluated into the value for a keyword. Expressions are saved in the BDL memory structure during execution.

The expression syntax includes the following operators and functions:

Operators	Mathematical, logical and bitwise operators
Standard functions	Standard mathematical, trigonometric and logarithmic functions
BDL functions	Use these functions to obtain information from BDL objects.
If-then statements	Use for making expression output based on the evaluation of some other expression.

Switch statements	Use for keying expression output on an integer value of some other expression
Special keyword	Use the UNUSED keyword to delete an existing value for a keyword when that keyword is not required.
Comments	You can add comments to an expression.

Operators

Constants and variables may be combined to form more complex expressions by using arithmetic, logical or bitwise operators. The operators supported in Expressions are described below.

Arithmetic Operators

These operators are used to perform arithmetic operations on constants or variables.

```
*   Multiplication
/   Division
+   Addition
-   Subtraction (or Unary Minus)
%   Remainder (mod)
**  Exponential
```

Logical Operators

These operators are used to perform logical operations on constants or variables.

```
||  or  .OR.   Or
&&  or  .AND.  And
!   or  .NOT.  Not
==  or  .EQ.   Equal
!=  or  .NE.   Not equal
>   or  .GT.   Greater than
<   or  .LT.   Less than
>=  or  .GE.   Greater than or equal to
<=  or  .LE.   Less than or equal to
```

Bitwise Operators

These operators are used to query the bits of a byte or a word. They require a more detailed knowledge of the inner workings of PowerDOE and will typically only be used by very advanced users.

```
&  And
|  Or
^  Exclusive or
```

Standard Functions

The standard mathematical and trigonometric functions available for use in creating an expression are presented in alphabetical order below.

Function	Description
ABS	Returns the absolute value of a number

ACOS	Returns the arccosine of a number
ASIN	Returns the arcsine of a number
ATAN	Returns the arctangent of a number
COS	Returns the cosine of the given angle
EXP	Returns the value of e raised to the power of a given number
FTOA	Converts a floating point number to a character string
INT	Returns the nearest integer value of a number
LOG	Returns the natural logarithm of a number
LOG10	Returns the base-10 logarithm of a number
MAX	Returns the largest value of two numbers
MIN	Returns the smallest value of two numbers
MOD	Returns the floating-point remainder
POW	Returns the result of a number raised to a power
SIN	Returns the sine of the given angle
SQRT	Returns the positive square root of number
STRUPPER	Converts string to all upper-case
STRLOWER	Converts string to all lower-case
TAN	Returns the tangent of the given angle

ABS

Returns the absolute value of a number. The absolute value of a number is the number without its sign.

Syntax

```
ABS (x)
```

where x is the real number for which you want the absolute value.

Examples

```
ABS(2)              equals 2
ABS(-2)             equals 2
```

```
If A1 contains -16, then:
   SQRT(ABS(A1))      equals 4
```

ACOS

Returns the arccosine of the given angle.

Syntax

```
ACOS(x)
```

where x is the cosine of the angle you want and must be from -1 to 1. If you want to convert the result from radians to degrees, multiply it by 180/p. (p = 3.141592654)

Examples

```
ACOS(-0.5)           equals 2.094395 (2p/3 radians)

ACOS(-0.5)*180/p     equals 120 (degrees)
```

Related Functions

SIN	Returns the sine of the given angle
ASIN	Returns the arcsine of a number
COS	Returns the cosine of the given angle
ACOS	Returns the arccosine of a number
TAN	Returns the tangent of the given angle
ATAN	Returns the arctangent of a number

ASIN

Returns the arcsine of the given angle. The arcsine is the angle whose sine is x. The returned angle is given in radians in the range -p/2 to p/2.

Syntax

```
ASIN (x)
```

x is the sine of the angle you want and must be from -1 to 1. To express the arcsine in degrees, multiply the result by 180/p. (p = 3.141592654)

Examples

```
ASIN(-0.5) equals -0.5236 (-p/6 radians)

ASIN(-0.5)*180/p equals -30 (degrees)
```

Related Functions

SIN	Returns the sine of the given angle
COS	Returns the cosine of the given angle
ACOS	Returns the arccosine of a number
TAN	Returns the tangent of the given angle
ATAN	Returns the arctangent of a number

ATAN

Returns the arctangent of the given number.

Syntax

```
ATAN (x)
```

where x is the tangent of the angle you want. The arctangent is the angle whose tangent is given by x. The returned angle is given in radians in the range -p/2 to p/2. To express the arctangent in degrees, multiply the result by 180/p. (p = 3.141592654)

Examples

```
ATAN(1)              equals 0.785398 (p/4 radians)

ATAN(1)*180/p        equals 45 (degrees)
```

Related Functions

SIN	Returns the sine of the given angle
ASIN	Returns the arcsine of a number
COS	Returns the cosine of the given angle
ACOS	Returns the arccosine of a number
TAN	Returns the tangent of the given angle

COS

Returns the cosine of the given angle.

Syntax

```
COS (x)
```

where x is the angle in radians for which you want the cosine. If the angle is in degrees, multiply it by p /180 to convert it to radians (p = 3.141592654).

Examples

```
COS(1.047)          equals 0.500171

COS(60*p/180)       equals 0.5, the cosine of 60 degrees
```

Related Functions

SIN	Returns the sine of the given angle
ASIN	Returns the arcsine of a number
ACOS	Returns the arccosine of a number
TAN	Returns the tangent of the given angle
ATAN	Returns the arctangent of a number

EXP

Returns e raised to the power of x. The constant e equals 2.71828182845904, the base of the natural logarithm.

Syntax

```
EXP (x)
```

where x is the exponent applied to the base e.

Remarks

1. To calculate powers of other bases, use the exponentiation operator (**).
2. EXP is the inverse of LOG, the natural logarithm of x.

Examples

```
EXP(1)              equals 2.718282 (the approximate value of e)

EXP(2)              equals e**2, or 7.389056

EXP(LN(3))          equals 3
```

Related Functions

LOG	Returns the natural logarithm of a number
LOG10	Returns the base-10 logarithm of a number
POW	Returns the result of a number raised to a power

FTOA

Converts a floating point number to a character string.

Syntax

```
FTOA( x)
```

where x is a number you wish to convert to a character string.

Examples

```
FTOA(5.6)            creates the character string "5.6".
```

INT

Rounds a number down to its integer value.

Syntax

```
INT (x)
```

where x is the real number you want to round down to the nearest integer. Fractional values less than or equal to 0.5 are rounded down, greater than 0.5 are rounded up.

Examples

```
INT(8.9) equals 8

INT(8.5) equals 8

INT(-8.9) equals -9
```

The following formula returns the decimal part of a positive real number X1:

```
X1-INT(X1)
```

LOG

Returns the natural logarithm of a number. Natural logarithms are based on the constant e (2.71828182845904).

Syntax

```
LOG (x)
```

where x is the positive real number for which you want the natural logarithm.

Remarks

1. LOG is the inverse of the EXP function.

Examples

```
LOG(86)              equals 4.454347

LOG(2.7182818)       equals 1

LOG(EXP(3))          equals 3
```

```
EXP(LOG(4))          equals 4
```

Related Functions

LOG10	Returns the base-10 logarithm of a number
EXP	Returns the value of e raised to the power of a given number
POW	Returns the result of a number raised to a power

LOG10

Returns the base-10 logarithm of a number.

Syntax

```
LOG(x)
```

where x1 is the positive real number for which you want the base-10 logarithm.

Examples

```
LOG10(10)            equals 1

LOG10(2.7182818)     equals 0.434294....

LOG10(1E3)           equals 3
```

Related Functions

LOG	Returns the natural logariththm of a number
EXP	Returns the value of e raised to the power of a given number
POW	Returns the result of a number raised to a power

MAX

Returns the largest value of two numbers.

Syntax

```
MAX (x1, x2)
```

where x1 and x2 are two numbers from which you want to select the largest number. You can specify arguments that are numbers or text representations of numbers. If the argument contains no numbers, then the expression will not be evaluated.

Examples

If x1=5 and x2 is a variable whose value is 15, then:

```
MAX(5,x2)            equals 15
```

Related Functions

MIN	Returns the smallest value of two numbers.

MIN

Returns the smallest value of two numbers.

Syntax

```
MIN( x1, x2)
```

where x1 and x2 are two numbers from which you want to select the smallest number. You can specify arguments that are numbers or text representations of numbers. If the argument contains no numbers, then the expression will not be evaluated.

Examples

If x1=5 and x2 is a variable whose value is 15, then:

```
MIN(5,x2)            equals 5
```

Related Functions

MAX	Returns the largest value of two numbers.

MOD

C calculates the floating-point remainder (f) of x / y such that x = i * y + f, where i is an integer, f has the same sign as x, and the absolute value of f is less than the absolute value of y.

Syntax

```
MOD( x, y )
```

where x is a number you wish to find the remainder for when divided by y.

Examples

```
MOD(5.2)             returns 1.
```

POW

Returns the result of a number raised to a power.

Syntax

```
POW (x1, x2)
```

where x1 is raised to the x2 power.

Remark

1. The "**" operator can be used instead of POW to indicate to what power the base number is to be raised, such as in 5**2

Examples

```
POW(5,2) equals 25

POW(98.6,3.2) equals 2401077

POW(4,5/4) equals 5.656854
```

Related Functions

LOG10	Returns the base-10 logarithm of a number
EXP	Returns the value of e raised to the power of a given number
POW	Returns the result of a number raised to a power

SIN

Returns the sine of the given angle.

Syntax

```
SIN (x)
```

where x is the angle in radians for which you want the sine. If your argument is in degrees, multiply it by p /180 to convert it to radians (p = 3.141592654).

Examples

```
SIN(p)              equals 1.22E-16,
                    which is approximately zero.
                    The sine of p is zero.

SIN(p/2)            equals 1

SIN(30*p/180)       equals 0.5, the sine of 30 degrees
```

Related Functions

ASIN	Returns the arcsine of a number
COS	Returns the cosine of the given angle
ACOS	Returns the arccosine of a number
TAN	Returns the tangent of the given angle
ATAN	Returns the arctangent of a number

SQRT

Returns a positive square root.

Syntax

```
SQRT (x)
```

where x is the number for which you want the square root. If x is negative, the expression will not be evaluated.

Examples

```
SQRT(16)              equals 4

SQRT(ABS(-16))        equals 4
```

Related Functions

EXP	Returns the value of e raised to the power of a given number
POW	Returns the result of a number raised to a power

TAN

Returns the tangent of the given angle.

Syntax

```
TAN (x)
```

where x is the angle in radians for which you want the tangent. If your argument is in degrees, multiply it by p /180 to convert it to radians (p = 3.141592654).

Examples

```
TAN(0.785)            equals 0.99920

TAN(45*p/180)         equals 1
```

BDL Functions

BDL functions are functions that can be used to obtain information about a BDL object that can be used in an expression. This information can be of symbolic or numeric type. BDL functions available for use in creating an expression are presented in alphabetical order below. Note that BDL functions are not case sensitive.

Function	Description
CalcAz or #CA	Returns the azimuth calculated for a POLYGON segment.
CalcDegN or #CDN	Calculates the facing direction of a component based on an azimuth and tilt.
DayOfWeek or #DOW	Returns an integer index of the day of week.
Error or #E	Registers a BDL error. No return value.
Global or #G	Returns the value of a BDL global command/keyword.
Local or #L	Returns the value of a local keyword.
LocalStatus or #LST	Return the BDL "status" value of a local Keyword
LocalSym or #LS	Returns the BDL symbol table index for the symbol entry of the local component.
LocalRef or #LR	Returns the value of a keyword in a command referenced by a local keyword.
LocalRefStatus or #LRST	Returns the BDL "status" value of a keyword in a command referenced by a local keyword.
MirrorPolygon or #MP	Returns the BDL symbol table index for the symbol entry of the mirror of a POLYGON.
Parameter or #PA	Returns the value of a BDL-defined PARAMETER given its text string.
Parent or #P	Returns the value of a parent keyword.
Parent2 or #P2	Returns the value of a grand parent keyword.
Parent3 or #P3	Returns the value of a great grand parent keyword.
ParentStatus or #PST	Returns the BDL "status" value of a parent keyword.
Parent2Status or #P2ST	Returns the BDL "status" value of a grand parent keyword.
Parent3Status or #P3ST	Returns the BDL "status" value of a great grand parent keyword.
ParentSym or #PS	Returns the BDL symbol table index for the symbol entry of parent of the local component.
Parent2Sym or #P2S	Returns the BDL symbol table index for the symbol entry of grandparent of the local component.
Parent3Sym or #P3S	Returns the BDL symbol table index for the symbol entry of greatgrandparent of the local component.
ParentRef or #PR	Returns the value of a keyword in a command referenced by a parent keyword.
Parent2Ref or #P2R	Returns the value of a keyword in a command referenced by a grandparent keyword.
Parent3Ref or #P3R	Returns the value of a keyword in a command referenced by a great-grandparent keyword.
ParentRefStatus or #PRST	Returns the BDL "status" value of a keyword in a command referenced by a parent keyword.
Parent2RefStatus or #P2RT	Returns the BDL "status" value of a keyword in a command referenced by a grandparent keyword.
Parent3RefStatus or #P3RT	Returns the BDL "status" value of a keyword in a command referenced by a great-grandparent keyword.
RefIndex or #RI	Returns the BDL reference table index for the symbol which this expression defines.
SymIndex or #SI	Returns the BDL symbol table index for a given symbol.
SymValue or #SV	Returns the BDL symbol table value of the symbol in the given BDL symbol table entry position.
ResVal or #RV	Specifies whether the argument is required, unused, has no default, or is unfilled.

CalcAz or #CA

This function was developed to facilitate the expressionization of defaults for AZIMUTH of building architectural objects (SPACEs and WALLs.)

Syntax

```
CalcAz(SymIndex, VertIndex ) or "#CA(SymIndex, VertIndex)
```

or

```
CalcAz( #P("POLYGON") or #SV(#L("LOCATION"))-11 )
```

where the first argument (SymIndex)is the symbol table index of a POLYGON, and the second argument (VertIndex) is a 0-based index of the starting polygon vertex used to determine the returned azimuth.

Examples

The following is the portion of a SPACE:AZIMUTH default expression:

```
CalcAz( #P("POLYGON") or #SV(#L("LOCATION"))-11 )
```

The first argument '#P("POLYGON")' returns the symbol table index of the SPACE's Parent (FLOOR) polygon. The second argument '#SV(#L("LOCATION"))-11' returns a polygon vertex index 0-11, depending on the Value of the Symbol assigned to the SPACE's LOCATION keyword (i.e. if LOCATION = FLOOR-V1 then the second CalcAz() argument = 0, for LOCATION = FLOOR-V2 the second argument = 1, etc.).

CalcDegN or #CDN

Calculates the facing direction of a component based on an azimuth and tilt (tilts < 0 result in azimuth+180)

Syntax

```
CalcDegN( Azimuth, Tilt ) or "#CDN(Azimuth, Tilt)
```

where the first argument (Azimuth) is the azimuth of the surface, and the second argument (Tilt) is the tile of the surface.

Examples

The following is the portion of a EXTERIOR-WALL:DEG-FROM-NORTH default expression:

```
{#CDN(#L("AZIMUTH")+#P("AZIMUTH")+#P2("AZIMUTH")+
   #G("BUILD-PARAMETERS","AZIMUTH"), #L("TILT"))}
```

DayOfWeek or #DOW

Returns an integer index of the day of week corresponding to the Yr/Mo/Da entered. When CalcLeap is set to 1 or a leap year, then the calc assumes 29 days in february, else 28.

Return values:

1. Sun
2. Mon

3. Tue
4. Wed
5. Thu
6. Fri
7. Sat

Syntax

```
DayOfWeek(Year, Month, Day, CalcLeap )
   or "#CDN(Year, Month, Day, CalcLeap)
```

where the first argument (Azimuth) is the azimuth of the surface, and the second argument (Tilt) is the tile of the surface.

Error or #E

NOT YET IMPLEMENTED - Registers a BDL error. No return value.

Syntax

```
Error(type, num, "msg") or #E(type, num, "msg")
```

where type is the symbol type, num is something, and "msg" is the error message you want to have inserted into the output file.

Remarks

1. The Error() function can be inserted into any portion of an expression but has no affect on the evaluation of the expression. This feature mimics the functionality of a multiple statement expression evaluator without adding unnecessary complexity.

Example

```
Error(type,num,"msg") which will do something.

#E(type,num,"msg") which will do the same thing.
```

Global or #G

Returns the value of a BDL global command or keyword.

Syntax

```
Global(com, key, i) or #G(com, key, i)
```

where com is a global command, key is a global keyword, and i is the ith value of the global keyword.

Examples

```
Global("SITE-PARAMETERS", "LATITUDE") equals 33.9 and

#G("SITE-PARAMETERS","TIME-ZONE") equals 8
```

when the following appears in the BDL file:

```
SITE-PARAMETERS
   LATITUDE          = 33.9
   LONGITUDE         = 118.5
   TIME-ZONE         = 8 ..
```

Related Functions

Parameter or #PA	Returns the value of a BDL global command or keyword.

Local or #L

Returns the value of a local keyword. See also BDL function notes.

Syntax

```
Local(com, key, i) or #L(com, key, i)
```

where com is a command, key is a keyword, and i is the ith value of the local keyword.

Example

An expression for the keyword "AREA" is defined as:

```
Local("WIDTH") * Local("HEIGHT")
```

which equals 44, given the following definition in the BDL file:

```
"Flr:2 Spc:S Wall:S>Win6" = WINDOW
   GLASS-TYPE        = "Lib 1205 - Grey"
   FRAME WIDTH       = 0.25
   X                 = 64
   Y                 = 3
   HEIGHT            = 5.5
   WIDTH             = 8 ..
```

Related Functions

Parent or #P	Returns the value of a parent keyword.

LocalRef or #LR

Returns the ith value of a keyword of a referenced command. Returns the ith value of keyword (key) for the last command argument listed. Each command argument is referenced as a keyword of the previous command argument. See also BDL function notes.

Syntax

```
LocalRef(key1, I1, key2, i2,...,key, i) or #LR(.......)
```

where key1 is a the first keyword, key2 is the second keyword, and i values are the ith value of the local keyword.

Example

LocalRef("CONSTRUCTION","LAYERS") would yield "ASHW-6", given the following definitions in the BDL file:

```
"Cons Perim Wall" = CONSTRUCTION
   LAYERS             = "ASHW-6" ..

"Flr:2 Spc:S>South Wall" = EXTERIOR-WALL
   CONSTRUCTION       = "Cons Perim Wall"
   X                  = 0
   Y                  = 0
   Z                  = 0
   HEIGHT             = 9
   WIDTH              = 100
   AZIMUTH            = 180
   SURFACE-TYPE       = WALL ..
```

Related Functions

ParentRef or #PR	Returns the value of a keyword of a referenced command in the current command's parent.

LocalSym or #LS
ParentSym() or #PS()
Parent2Sym() or #P2S()
Parent2Sym() or #P2S()

Returns the BDL symbol table index of a component, either the local object being evaluated, its parent, grandpart or greatgrandparent.

Syntax

```
LocalSym() or #LS()
```

Example

This example show the default expression for the INDUCED-AIR-ZONE keyword in the ZONE command. This keyword defaults to the ZONE itself, for a ZONE defined to be supplied by a PIU system and having a terminal box that uses the keyword, otherwirse the keyword is not used. :

```
if (#P("TYPE") == #SI("PIU","SYSTEM","TYPE"))
   then if (#L("TERMINAL-TYPE") == #SI("SERIES-PIU","ZONE","TERMINAL-
   TYPE")
   .or. #L("TERMINAL-TYPE") == #SI("PARALLEL-PIU","ZONE","TERMINAL-
   TYPE"))
        then #LS()
        else unused
        endif
   else unused
   endif
```

LocalStatus or #LST
LocalRefStatus or #LRST
ParentStatus or #PST
ParentRefStatus or #PRST
Parent2Status or #P2ST
Parent2RefStatus or #P2RST
Parent3Status or #P3ST
Parent3RefStatus or #P3RST

Returns the BDL "status" value of a keyword. Uses the same syntax of the Local, LocalRef, Parent, ParentRef and similar functions. Return values: are

0 Undefined
1 Default Data
2 Default Expression
3 Library Data
4 Library Expression
5 User Data
6 User Expression
7 User Default Data
8 User Default Expression
9 Linked Component Data
10 Linked Component Expression
11 Compliance Ruleset Defined Data
12 Compliance Ruleset Defined Expression
13 Compliance Ruleset Library Data
14 Compliance Ruleset Library Expression
15 Compliance Ruleset Default Symbol Selection

Syntax

```
LocalStatus(com, key, i) or #LS(com, key, i)
```

Example

LocalStatus("CONSTRUCTION","LAYERS") would yield 5, given the following definitions in the BDL file:

```
"Cons Perim Wall" = CONSTRUCTION
   LAYERS            = "ASHW-6" ..
```

```
"Flr:2 Spc:S>South Wall" = EXTERIOR-WALL
   CONSTRUCTION      = "Cons Perim Wall"
   X                 = 0
   Y                 = 0
   Z                 = 0
   HEIGHT            = 9
   WIDTH             = 100
   AZIMUTH           = 180
   SURFACE-TYPE      = WALL ..
```

MirrorPolygon or #MP

Returns the integer value of the symbol table entry for a POLYGON that is the mirror image of the referenced POLYGON.

This function searches all existing POLYGONs for one that is a mirror image of the referenced POLYGON. If one is found , the symbol table index of that POLYGON is returned. If one is not found, a new POLYGON is created that is the mirror image of the referenced one and the symbol tbale index of the new POLYGON is returned.

Syntax

```
MirrorPoly(idx) or #MP(idx)
```

where idx is an integer symbol table index.

Example

Suppose you wish to set a FLOOR's POLYGON keyword to be the mirror of its parent SPACE POLYGON, then you would use and expression with logic such as:

```
if(#SV(#L("LOCATION"))==2) then #MP(#P("POLYGON"))
```

this checks the FLOOR's LOCATION to see if it is BOTTOM, and if it is it sets the return value to the mirror polygon of the parents SPACE POLYGON.

Parameter or #PA

Returns the value of a BDL-defined PARAMETER u-name.

Syntax

```
Parameter(str) or #PA(str)
```

where str is the u-name of a PARAMETER defined in the BDL file.

Example

Parameter("Param 1") equals 10 and #PA("Param 5") equals "ELECTRIC" when the following appears in the BDL file:

```
PARAMETER
   "Param 1" = 10
   "Param 5" = "ELECTRIC" ..
```

Related Functions

Global or #G	Returns the value of a BDL global command or keyword.

Parent or #P

Returns the value of a parent keyword. See also BDL function notes.

Syntax

```
Parent(com, key, i) or #P(com, key, i)
```

where com is a command, key is a keyword, and i is the ith value of the local keyword.

Example

Given the following WALL and WINDOW definitions in the BDL file:

```
"Flr:2 Spc:S>South Wall" = EXTERIOR-WALL
   CONSTRUCTION      = "Cons Perim Wall"
   X                 = 0
   Y                 = 0
   Z                 = 0
   HEIGHT            = 9
   WIDTH             = 100
   AZIMUTH           = 180
   SURFACE-TYPE      = WALL ..

"Flr:2 Spc:S Wall:S>Win6" = WINDOW
   GLASS-TYPE        = "Lib 1205 - Grey"
   FRAME WIDTH       = 0.25
   X                 = 64
   Y                 = 3
   HEIGHT            = 5.5
   WIDTH             = 8 ..
```

The expression to define the AREA of the WINDOW as 40% of the total (parent) wall area would be:

```
{Parent("WIDTH") * Parent("HEIGHT")*.40},  which equals 360
```

Related Functions

Local or #L	Returns the value of a local keyword.

Parent2 or #P2

Returns the value of a grandparent keyword. See also BDL function notes.

Syntax

```
Parent2(com, key, i) or #P2(com, key, i)
```

where com is a command, key is a keyword, and i is the ith value of the local keyword.

Example

The expression to define the AREA of the WINDOW as 40% of the total (grand parent) SPACE area would be:

```
Parent2("AREA")
```

Parent3 or #P3

Returns the value of a great grand parent keyword. See also BDL function notes.

Syntax

```
Parent3(com, key, i) or #P3(com, key, i)
```

where com is a command, key is a keyword, and i is the ith value of the local keyword.

Example

The expression to define the AREA of the WINDOW as 40% of the total (great grand parent) FLOOR area would be:

```
Parent3("AREA")
```

ParentRef or #PR

Returns the ith value of a keyword of a referenced command in the current command's parent. Returns the ith value of keyword (key) for the last building object referenced by the function arguments and starting with the current building objects' parent. Each key argument references a building object's parent. Each key argument references a building object except the last one whose value is of a symbolic or numeric type.

Syntax

```
ParentRef(key1, i1, key2, i2,...,key, i) or #PR(.......)
```

where key1 is a the first keyword, key2 is the second keyword, and i values are the ith value of the parent keyword.

Example

In the WINDOW command, ParentRef("CONSTRUCTION","LAYERS") would yield "ASHW-6", given the following definitions in the BDL file:

```
"Cons Perim Wall" = CONSTRUCTION
                      LAYERS = "ASHW-6" ..

"Flr:2 Spc:S>South Wall" = EXTERIOR-WALL
   CONSTRUCTION       = "Cons Perim Wall"
   X                  = 0
   Y                  = 0
   Z                  = 0
   HEIGHT             = 9
   WIDTH              = 100
   AZIMUTH            = 180
   SURFACE-TYPE       = WALL ..
```

Related Functions

LocalRef or #LR	Returns the value of a local keyword referenced by BDL symbol table index.

Parent2Ref or #P2R

Similar to ParentRef, but for the current objects grand parent. Returns the ith value of keyword (key) for the last building object referenced by the function arguments and starting with the current building objects' grant parent. Each key argument references a building object's grant parent. Each key argument references a building object except the last one whose value is of a symbolic or numeric type.

Syntax

```
Parent2Ref(key1, i1, key2, i2,...,key, i) or #PR(.......)
```

where key1 is a the first keyword, key2 is the second keyword, and i values are the ith value of the grand parent keyword.

Parent3Ref or #P3R

Similar to ParentRef, but for the current objects great grand parent. Returns the ith value of keyword (key) for the last building object referenced by the function arguments and starting with the current building objects' great grant parent. Each key argument references a building object's great grant parent. Each key argument references a building object except the last one whose value is of a symbolic or numeric type.

Syntax

```
Parent3Ref(key1, i1, key2, i2,...,key, i) or #PR(.......)
```

where key1 is a the first keyword, key2 is the second keyword, and i values are the ith value of the grand parent keyword.

RefIndex or #RI

Returns the BDL reference table index of the symbol (i.e., U-name) for which this expression defines the keyword.

Syntax

```
RefIndex or #RI
```

where no variables are specified. If this function is used, it will return the index number associated with the current symbol (U-name).

Example

RefIndex returns an integer which is the BDL reference table index of the current symbol (U-name).

Related Functions

SymIndex or #SI	Returns the symbol table index of the symbol named in str.
SymValue or #SV	Returns the integer value of the idxth symbol table entry.

SymIndex or #SI

Returns the symbol table index of the symbol named.

Syntax

```
SymIndex(str,com,key) or #SI(str,com,key)
```

where str is the character string of a BDL symbol, com is a command, key is a keyword. The second two arguments are optional, so the user can call this function with just str, with both str and com, or with all three.

SymIndex(str): returns the symbol index of the symbol matching the character string str and of the same symbol type as the keyword being set by the expression.

SymIndex(str, com): returns the symbol index of the symbol matching the character string str and of the same symbol type as the command specified by the com argument.

SymIndex(str, com, key): returns the symbol index of the symbol matching the character string str and of the symbol type compatible with the com : key arguments.

Example

If the user is setting the SPACE : POLYGON parameter with the expression

```
SymIndex( "Poly1" )
```

then the expression module will find the symbol matching the string "Poly1" and of the symbol type compatible with SPACE : POLYGON (this symbol type happens to = 41).

In order to use the two argument version to get the same result as the above example, the user could use the expression

```
SymIndex( "Poly1", "POLYGON" )
```

The advantage of this method is that the return index is not dependent on the type of the symbol being set by the expression, which means that the user can use this version in an if-then statement.

For instance, if the user wished to set the SPACE : X parameter to a value depending on the current SPACE : POLYGON assignment, the user could use the expression:

```
{if (Local( "POLYGON" ) == SymIndex( "Poly1", "POLYGON" ))
   then
     30
   else
     0
 endif}
```

This expression sets the space's X to 30 if the polygon assigned to the space (via the POLYGON keyword) is "Poly1". If Poly1 is not assigned to the space, this expression sets the SPACE : X value to zero.

In order to use the three argument version to get the same result as the first example above, the user could use the expression

```
SymIndex( "Poly1", "SPACE", "POLYGON" )
```

Like the two argument version, this function call ignores the symbol type of the parameter being set by the expression. Instead, this version uses the com & key arguments to determine which symbol type to search for.

Related Functions

RefIndex or #RI	Returns the BDL reference table index of the symbol for which this expression defines the keyword.
SymValue or #SV	Returns the integer value of the idxth symbol table entry.

SymValue or #SV

Returns the integer value of the idx-th symbol table entry.

Each symbol table entry has two numeric values associated with it. One is the symbol table index, which is equivalent to the position of that symbol within the list of symbol table entries. The second is a symbol Value, which is independent of the symbol table position. For non U-name symbols, the symbol value is a constant integer value. For building component symbols, the symbol value is an occurrence number for that component, in relation to all other building components of the same type.

This function can support all of the functionality of symbol searching as the SymIndex() function, by using the SymIndex() function return value as the SymValue() argument. For example, to retrieve the occurrence number of the polygon named "Poly1", use the expression: SymValue(SymIndex("Poly1", "POLYGON"))

Syntax

```
SymValue(idx) or #SV(idx)
```

where idx is an integer symbol table index.

Example

```
if (#SV(#L(“LOCATION”))>10) then
```

Related Functions

SymIndex or #SI	Returns the symbol table index of the symbol named in str.

ResVal or #RV

Returns an integer that specifies whether the argument is required, unused, has no default, or is unfilled. The integer that this function returns can be interpreted as follows:

0 if argument is not a BDL reserved value
1 if argument = -99999.0 or "required"
2 if argument = -88888.0 or "unused"
3 if argument = -77777.0 or "no default"
4 if argument = -66666.0 or "unfilled"

Syntax

```
ResVal(number) or #RV(number)
```

where number is the character string of a BDL symbol.

Example

Check whether a keyword has been entered:

```
if (#RV(#L("PEOPLE-SCHEDULE"))!=0)
  then required
  else unused
endif
```

This expression is true if PEOPLE-SCHEDULE has not been input.

BDL Function Notes

These comments apply to the Parameter, Global, Local, LocalRef, Parent, and ParentRef BDL functions.

BDL Command\Keyword\u-names case sensitivity

BDL commands and keywords used in expressions are case sensitive. Commands and key words must be entered in all caps, and u-names must be entered exactly as they were defined.

Function return value type

BDL functions will automatically return a value of the same type as the keyword which it defines. In the event that the expression is to evaluate to a symbol, the return value shall consist of an integer index to that symbols entry in the BDL symbol table.

Example

```
Local("X") returns a number

LocalRef("CONSTRUCTION","LAYERS") returns an integer index to a symbol.
```

When argument i is required

For most BDL functions the argument i is not required. The argument i is required when calling these functions if and only if the keyword argument (key) is of type numeric or symbolic and the value array length >1 (i.e. an array of numbers or symbols). Primary use of i would be when accessing schedules

Limitations on definition of argument i

In order to perform as much error checking as possible at parse-time, the definition of the argument i should be in the form of a numeric constant and not the result of an expression.

Executable Statements

Executable statements available in Expressions include the If-Then and Switch statements. The words reserved for these executable statements - if, else, endif, switch, case, default, endswitch - are case insensitive. If and Switch statements can be nested as long as the expression evaluates to a single return value.

If-then Statements

Returns one value if an expression evaluates to TRUE and another value if it evaluates to FALSE. If-then statements can be nested and each loop must be closed with an endif statement.

Structure

```
{if  (expression1)
  then statement1
  else  if  (expression2)
          then statement2
          else statement3
        endif
endif}
```

where

(expression#)	are expressions get evaluated to TRUE or FALSE
(statement#)	is the statement that is executed based on the evaluation of expression#

Examples

In the following example, assume you are trying to define a Local HEIGHT in terms of the Local WIDTH, but you don't want the HEIGHT to go above or below a certain value. This could be accomplished by using the If-then statement to define an expression for the HEIGHT as follows:

```
{if(Local("WIDTH")>15)
   then 12
   else  if(Local("WIDTH")=15)
           then 10
           else  8
         endif
 endif}
```

Switch Statements

The switch statement enables you to choose a course of action based on the value of the primary expression, which must be an integer-compatible expression.

Structure

```
{switch  (expression)
   case  (const1):  (expression1)
   case  (const2):  (expression2)
   default       :  (expression3)
 endswitch}
```

where

(expression)	is some expression that evaluates to an integer-compatible value.
(const#)	where # is a possible integer value of (expression)
(expression#)	is the expression that applies whenever (expression) evaluates to (const#)

default	is the expression that applies whenever (expression) evaluates to a value for which there is no defined case.

Examples

In the following example, assume you are trying to define a Local HEIGHT in terms of the Local WIDTH, but you don't want the HEIGHT to go above or below a certain value. This could be accomplished by using the If-then statement to define an expression for the HEIGHT as follows:

```
{switch(Local("X"))
    case 1:  2
    case 2:  4
    case 3:  6
    case 4:  8
   default:  0
 endswitch}
```

Additional Comments

1. *Each switch statement must contain a default expression.* Note that the individual case values listed as *const* must be either numeric constants or (symbols which can be directly translated into numeric constants at parse-time in order to prevent excessive evaluation-time error checking and minimize the complexity of the statement).

Putting Comments in Expressions

The following comment styles, consistent with standard C++ comment syntax, can be used in Expressions to insert comments into an expression:

Comment Style 1 *(currently not implemented)*

```
//  Use two forward-slashes for single line comments.
//  Any text between this symbol and the end of
//  the line will be considered a comment.
```

Comment Style 2

```
/*  Use a single forward-slash and asterisk for single or multi-line
comments, ending comment with an asterisk and single forward-slash as
shown.  Any text between the symbols will be considered a comment */
```

BDL Special Keywords

The Special BDL keywords required, unused, nodefault and unfilled are used to check or set keyword values against the BDL special values as follows. Note:BDL special keywords are not case sensitive.

required	means this keyword is required
unused	means this keyword is unused
nodefault	means there is no default for this keyword

unfilled	means a flag value is sent to the simulation to indicate no values was specified for this keyword

Example

The BUILDING-SHADE object has a symbolic keyword called SHADE-SCHEDULE which is not a required keyword and can therefore be set to "Undefined". Expressions for other BUILDING-SHADE keywords can check for SHADE-SCHEDULE being defined or undefined by containing an if statement such as:

```
if (ResVal(Local("SHADE-SCHEDULE")) > 0) then ....
```

Similarly, the reserved keyword can be referenced within an expression which sets the SHADE-SCHEDULE like so:

```
{if (...)
  then  required
  else  unused
 endif}
```

This causes the keyword to become required or unused depending upon the exaluation of the if clause.

Additional Examples

Using the Local function

An example of an expression which describes the space keyword X as a function of the same space's Y keyword value:

```
Local("Y") + 10
```

Using the Local function in an If-then statement

A more complex example of an expression which describes the space keyword X as a function of the same space's Y keyword value using nested if-then-else statements:

```
{if (Local("Y") > 20)
  then 10
  else if (Local("Y") > 10)
        then 5
        else 0
       endif
 endif}
```

Using the Parent function

An example of an expression which describes the wall keyword Z as a function of its parent space's Z keyword value:

```
Parent("Z") / 5
```

Using the SymIndex function

An example of an expression which describes the space keyword POLYGON as a function of its parent floor's Z keyword value:

```
{if (Parent(“Z”) > 0)
  then  SymIndex(“Upper Poly”)
  else  SymIndex(“Lower Poly”)
endif}
```

INPUT MACROS

Introduction

The Input Macros feature was added to the Building Description Language to increase its flexibility. This feature is intended for advanced users who are already familiar with preparing BDL input. The basic capabilities are:

1. Incorporating external files containing pieces of BDL into the main BDL input stream. This is also called the General Library feature.

2. Selectively accepting or skipping portions of the input.

3. Defining a block of input with parameters and later referencing this block.

4. Performing arithmetic and logical operations on the input.

5. Input macro debugging and listing control.

These capabilities are invoked in BDL by using macro commands. Macro commands are preceded by ## to distinguish them from regular BDL commands. After execution by the BDL processor, macro commands produce regular lines of BDL input that are shown in the BDL echo print. Following are descriptions of the macro commands associated with the above capabilities. A detailed example of input macros is given at the end of this section; you should review it before reading the macro command descriptions.

Input Macros vs. Keyword Expressions

While, input macros and keyword expressions can accomplish many of the same objectives, they each have features that lend themselves to different applications. In fact, a well-designed parametric input may incorporate both macos and expressions.

In general:

1. Use keyword expressions to modify set individual keywords as a function of other keywords or parameter values.

2. Use macros to activate whole blocks of keywords within a command, or to add whole commands.

For example, assume you are evaluating a packaged VAV vs. a regular VAV system. A regular VAV system will require hot and chilled-water circulation-loops, while the packaged system will not. When evaluating the regular VAVS, a macro can automatically define the circulation-loops, as well as the pumps, boilers, chillers, and cooling towers associated with the loops.

Incorporating External Files

##include {includefilename}

This command puts all of the lines in an external file into the BDL input stream starting right after the command line. The name of the file that is included is the concatenation of {prefixpathname}, entered using ##fileprefix , and {includefilename}. The lines in the external file will be listed in the BDL echo so that the user can see exactly what is being included. When all the lines in the external file have been read in, input reverts back to the original input file at the line following the ##include command.

##fileprefix {prefixpathname}

specifies a pathname that will be prefixed to the filename given in an ##include command. The ##fileprefix command allows commonly-used include files to be kept in a directory other than the directory in which the current input file resides.

Example: on VAX/VMS, the combination

```
##fileprefix          DRC2: [GUEST.LIBRARY]

##include             SCHEDULES.INP
```

will include into the BDL stream the file whose full name is

```
DRC2: [GUEST.LIBRARY] SCHEDULES.INP
```

##includesilent {includefilename}

This command is identical to ##include, except that the lines in the included file will not be listed in the BDL echo.

##nosilent

Overrides the listing suppression of ##includesilent. Used for debugging purposes only. After ##nosilent , all following ##includesilent commands are treated as ##include commands.

Example: Assume the following files contain the indicated lines:

Main input file INPUT1.INP:

```
line 1a
##include  file2.inp
line 1b
line 1c
```

External file, FILE2.INP:

```
line 2a
line 2b
line 2c
```

The end result of processing will be:

```
line 1a           (from input1.inp)
line 2a           (from file2.inp)
line 2b           (from file2.inp)
line 2c           (from file2.inp)
line 1b           (from input1.inp)
line 1c           (from input1.inp)
```

External files can also contain ##include commands, as shown in the following example:

Main input file INPUT1.INP:

```
line 1a
##include  file2.inp
line 1b
line 1c
```

External file, FILE2.INP:

```
line 2a
line 2b
##include  file3.inp
line 2c
```

External file, FILE3.INP:

```
line 3a
line 3b
line 3c
line 3d
```

The end result of processing will be

```
line 1a           (from input1.inp)
line 2a           (from file2.inp)
line 2b           (from file2.inp)
line 3a           (from file3.inp)
line 3b           (from file3.inp)
line 3c           (from file3.inp)
line 3d           (from file3.inp)
line 2c           (from file2.inp)
line 1b           (from input1.inp)
line 1c           (from input1.inp)
```

Note: Up to nine ##include commands can be nested. However, there should be no recursion. This is an example of a recursion:

file1.inp contains **##include** file2.inp
file2.inp contains **##include** file1.inp

Selectively Accepting or Skipping Lines of Input

The **##if** series of commands is used to selectively accept or skip lines of input according to the following sequence:

```
##if  {condition1}
  line1a
  line1b
..line1x
##elseif  {condition2}
  line2a
  line2b
  line2x
##elseif  {condition3}
  line3a
  line3b
  line3c
##else
```

```
  line N a
  line N b
##endif
```

Then the lines that will be included into the BDL stream are:

If {condition 1} is TRUE,

```
line1a
line1b
….
```

otherwise, if {condition 2} is TRUE,

```
line2a
line2b
….
```

otherwise, if {condition 3} is TRUE,

```
line3a
line3b
….
```

otherwise, if {condition 1}, {condition 2}, {condition 3} are all FALSE.

```
line N a
line N b
….
```

There are six different **##if**... commands:

##ifdef {macro name}

if macro name defined, include following lines

##ifndef {macro name}

if macro name NOT defined, include following lines

##if {condition}

if condition is TRUE, include following lines

##elseif {condition}

if condition is TRUE, and previous conditions are FALSE, include following lines

##else

if all previous conditions are FALSE, include following lines

##endif

indicates the end of the if block

Notes:

1. {macro name} is explained in (3), below.

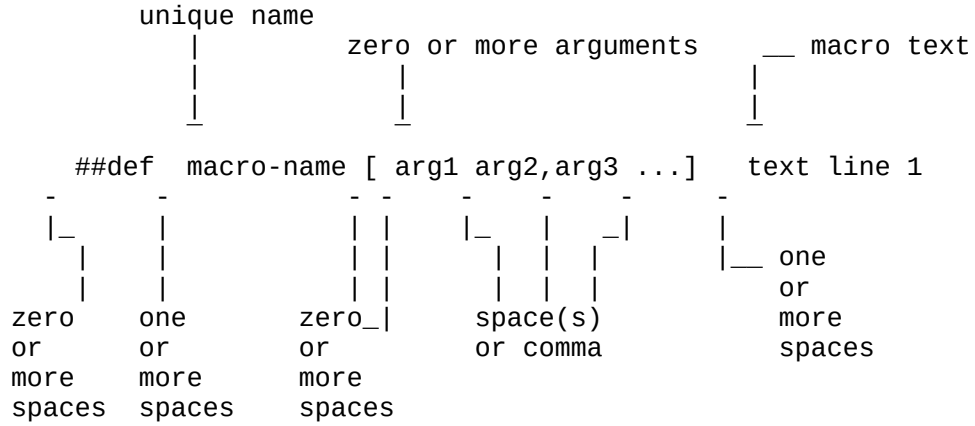
2. {condition} is 0 or BLANK meaning FALSE, and any other character meaning TRUE.

3. ##ifdef and ##ifndef do not have corresponding ##elseif commands, but they do have corresponding ##else and ##endif commands.

Defining Blocks of Input

The **##def** command allows a block of input text to be defined and given a name. The block of text can then be inserted anywhere in the BDL stream by simply referencing the name of the block. (This process is called macro expansion.) The block can have parameters (also called arguments) that can be given different values each time the block is referenced.

The syntax of the **##def** command is as follows:



Example: Define a macro with name "all_ones":

```
##def   all_ones

SCHEDULE
   TYPE              = MULTIPLIER
   THRU DEC 31         (ALL) (1,24) (1.0)  ..

##enddef
```

Then, in the BDL input stream, when we write :

```
SCHED1 = all_ones[]  ..
                ^^ the square braces are required
```

the result is equivalent to:

```
SCHED1 = SCHEDULE
   TYPE              = MULTIPLIER
   THRU DEC 31         (ALL) (1,24) (1.0) ..
```

Macro definitions may have one or more arguments; the maximum number of arguments is 32. When a macro with arguments is referenced, its arguments must be given values.

Example: Define a macro with name "sched" and argument "x":

```
##def  sched[x]

SCHEDULE
   TYPE              = MULTIPLIER
   THRU DEC 31         (ALL) (1,24) (x)  ..

##enddef
```

Then, when we put the following in the BDL input stream

```
SCHED2 = sched[.20]  ..

SCHED3 = sched[.33]  ..
```

the result is equivalent to:

```
SCHED2 = SCHEDULE
   TYPE              = MULTIPLIER
   THRU DEC 31         (ALL) (1,24) (.20)  ..

SCHED3 = SCHEDULE
   TYPE              = MULTIPLER
   THRU DEC 31         (ALL) (1,24) (.33)  ..
```

Macro names must be unique (except see ##set1 below); i.e., when a macro name is defined it cannot be defined again.

To summarize, commands you use to define macros are the following:

##def macro-name [arg1,..,argn] macro-text

Defines a macro with the name macro-name and arguments "arg1" through "argn". "Macro-text" is one or more lines of text. If there are no arguments, the syntax is **##def** macro-name macro-text.

##enddef

Indicates the end of the macro definition initiated by **##def**.

##def1 macro-name [arg1,..,argn] macro-text

This is the same as **##def** but there is only one line of text so that the terminating command **##enddef** is not required.

##set1 macro-name macro-text

Like **##def1** but has no arguments and macro-text is evaluated before storing. "Macro-text is evaluated" means that if macro-text contains other macros, these macros will be expanded, and the expanded text becomes the macro-text defined by **##set1**.

Example:

```
##def1   xx  123
##set1   yy  xx[]
```

is equivalent to:

```
##set1   yy  123
```

##set1 can also be used to redefine macro-name.

```
##set1   x   0
  .
  .
  .
##set1   x    #eval[ x[]+1 ]
```

(see Arithmetic Operations for description of the **#eval** macro.)

Arithmetic Operations

The built-in macro called **#eval[ ]** can be used to perform arithmetic, literal, and logical operations. It can be abbreviated to **#[ ]**.

#eval[X OP Y] or #[X OP Y]

gives the result X OP Y. The allowed values for X, OP, and Y, and the corresponding result, are shown in the following table.

Table 5 Macro Operations

X *	OP **	Y	Result
number	+(plus)	number	number
number	-(minus)	number	number
number	*(times)	number	number
number	/ (divided by)	number	number
number	min	number	number
number	max	number	number
number	mod	number	number
number	**	number	number
SIN	OF	number (degrees)	number
COS	OF	number (degrees)	number
TAN	OF	number (degrees)	number
SQRT	OF	number	number
ABS	OF	number	number
ASIN	OF	number	number (degrees)
ACOS	OF	number	number (degrees)
ATAN	OF	number	number
INT	OF	number	number
LOG10	OF	number	number
LOG	OF	number	number
literal1	// (concatenate)	literal2	literal "literal1 literal2"
literal1	/// (concatenate)	literal2	literal "literal1 literal2"
literal	EQS (=)	literal	logical (true or false)
literal	NES (¹)	literal	logical (true or false)
logical	AND	logical	logical (true or false)
logical	OR	logical	logical (true or false)
logical	NOT	logical	logical (true or false)
number	EQ (=)	number	logical (true or false)
number	NE (¹)	number	logical (true or false)
number	GT (>)	number	logical (true or false)
number	GE (³)	number	logical (true or false)
number	LT (<)	number	logical (true or false)
number	LE (£)	number	logical (true or false)
* Upper or lower case is allowed for SIN, COS, etc			
** Upper or lower case is allowed for OF, EQS, etc			

Example

```
#eval[  1 + 2  ]     when expanded becomes 3

#eval[ 1 +  #eval[2 * 3] ]     when expanded becomes 7
```

Example

```
#set1  city Washington

TITLE  LINE 1   #[ "large office" /// city[ ]  ]
```

gives

```
TITLE  LINE 1  "large office  Washington"
```

The following example illustrates the use of #eval inside #if commands:

```
##if  #[ city[ ] EQS  Chicago ]

##if  #[#[ city[ ] EQS  Chicago ] and #[ occup[ ] NES low ] ]
```

Notes:

1. For logical values:

   False = 0 or BLANK,

   True = any other character

2. A literal must be enclosed inside a pair of double quotes if it contains BLANKs or reserved characters like [ ] ( ) ,

   E.g., "abc *def"

   Otherwise, the quotes around the literals are optional.

3. Literal concatenation operators // and /// produce quoted literals.

   E.g., # [large /// office] gives "large office"

4. Literals are case sensitive. For example, "Chicago", "CHICAGO" and "chicago" are distinct.

Macro Debugging and Listing Control

##list

Turn on listing; echo of input lines on the OUTPUT file is enabled. This is the default condition.

##nolist

Turn off listing; echo of input lines on the output file is disabled.

##show

Start printing expanded line on output file. After this command, if a macro expansion was done, the expanded line is printed on the output file. In this way you can see the end result of macro expansions, which is the input as seen by the BDL processor.

##noshow

Stop printing expanded line on output file. This is the default condition.

##showdetail

Start printing each macro expansion. After this command, every time a macro expansion is done the result of the expansion is printed. This can produce lots of output.

##noshowdetail

Stop printing each macro expansion. This is the default condition.

##traceback

Give full traceback when printing an error message. After this command, if there is a BDL error, a full traceback of the macro expansions in progress is printed. This is the default condition.

##notraceback

Don't give full traceback when printing an error message.

##write

Start writing expanded text into file 22. This is similar to ##show except that the expanded lines are written into file 22. Therefore, file 22 will contain only the text that will be seen by the BDL processor. This file is used only for debugging purposes. It allows you to see what the macro-processed input file looks like.

##nowrite

Stop writing expanded text into file 22. This is the default condition.

##symboltable

Prints table of current macro names. All of the macro names that are defined will be printed.

##clear

Clear all macro definitions. All the macro names defined up to this point will be deleted.

##reserve TEXT k NAMES l STACK m

Allocates memory.

Reserves **k** words of space in AA array for macro definition storage.
Reserves **l** positions in macro definition names table.
Reserves **m** words of stack space.

If used, the **##reserve** command must precede all other macro commands in the BDL input. This command should be used only if one or more of the following error messages is received:

1. "Need more memory for storing macro definitions"
   Use "##reserve TEXT nnnnnn" command to get more memory. Current value of nnnnnn is: _ _ _

2. "Macro table capacity exceeded"
   Use "##reserve NAMES nnnnnn" command to get more memory. Current value of nnnnnn is: _ _ _

3. "Macro stack overflow"
   Use "##reserve STACK nnnnnn" command to get more memory. Current value of nnnnnn is: _ _ _

##$ <comment>

Allows you to enter comment lines inside a macro. <comment> is printed in the BDL echo but is not acted on by the macro processor.

Example:

This example shows the use of the **##set**, **##include**, **##eval** and **##if** commands. Let an external file called CITIES.LIB contain the following text:

```
##if      #[ city[ ]  EQS  CHICAGO ]

BUILDING-LOCATION
   LATITUDE          = 41.88
   LONGITUDE         = 87.63
   ALTITUDE          = 600
   TIME-ZONE         = 6     $Chicago$
   ..

##elseif  #[ city[ ]  EQS  WASHINGTON ]

BUILDING-LOCATION
   LATITUDE          = 38.9
   LONGITUDE         = 77
   ALTITUDE          = 50
   TIME-ZONE         = 5     $Washington$
   ..

##else

  ERROR--City Undefined

##endif
```

Then the BDL input

```
INPUT ..

##set1         city CHICAGO
##include      cities.lib

 .
 .
 .
```

will be converted, after macro processing, to:

```
INPUT ..

BUILDING-LOCATION
   LATITUDE          = 41.88
   LONGITUDE         = 87.63
   ALTITUDE          = 600
   TIME-ZONE         = 6     $Chicago$
   ..

   .
   .
   .
```

Listing Format

The format of listing in BDL gives information about the status of the input macros, as shown in the following:

```
1. 1.1  * 123 * ......  BDL input line
-  - -  ------- ---------------------------------
^  ^ ^     ^     ^
|  | |     |     echo of BDL input line
|  | |     |
|  | |    line number ( if the current line is being skipped
|  | |                  by " ##if... " etc, this is indicated
|  | |                  by printing "- 123 -" instead of
|  | |                  "* 123 *" in the line number field;
|  | |                  if the current line is part of a
|  | |                  macro command, it is indicated by
|  | |                  printing "# 123 #" )
|  | |
|  | macro expansion nesting level
|  |
|"##if" nesting level
|
"##include" nesting level
```

INPUT FUNCTIONS

The Input Function feature allows you to modify LOADS or HVAC (SYSTEMS and PLANT) calculations without recompiling the program. ***This feature is not yet implemented in DOE-2.2.*** *Because it requires familiarity with the internal calculations in the simulation engine, it should not be attempted by the beginning user.*

Input Functions are input as small, FORTRAN-like routines that are included in your regular building description. You specify the values to be calculated and where in the hourly simulation they are to be used.

There are three types of applications of Input Functions:

1. Calculation of variables that influence the program results, thus allowing you to modify or replace the algorithms used by the program without recompiling the program.

2. Calculation of variables for reporting or debugging purposes.

3. Reading in data files for use in the simulation.

To use Input Functions, you have to know (1) which simulation variables are accessible to Input Functions and (2) where the Input Functions are called in the LOADS and HVAC programs.

The simulation variables that are accessible (i.e., that can be used in, or modified by, Input Functions) are shown, along with their definitions, in the *LOADS and HVAC Global Variables Listings*. These listings are available as the text files LOADS-VARS.TXT and HVAC-VARS.TXT, which are part of the program release.

Where the Input Functions are called is shown in the program *Compiler Listings*. These are the files LDS-LIS.TXT, SYS-LIS.TXT and PLT-LIS.TXT, which are part of the program release. To find the Input Function access points look for CALLs to subroutine FINTL in LDS-LIS.TXT and subroutine FINTS in SYS-LIS.TXT and PLT.LIS-TXT. (Note that not all of the LOADS and HVAC routines have access points.) These listings can also be used to determine how and where the variables of interest are calculated by the program.The Global Variables Listings and the compiler listings should be printed out before you try to use Input Functions. These listings are essential to the use of Input Functions; if you do not fully understand the calculation sequence in the simulation engine, it is very easy to enter Functions that change the calculation results in unexpected ways.

Functions are referenced within the hourly loop of the program and, therefore, will be calculated each hour of the input run period.

Hourly Report Variables

From the Compiler Listings you can determine whether a function that changes the value of an hourly report variable makes this change before or after the program places the variable in the hourly report array for printing each hour. Only if your function changes the hourly report variable before it is put in the printing array will your change be reflected in the hourly report. See "HOURLY-REPORT" in *DOE-2.2 DOE-2.2 Libraries & Reports* for the FORTRAN name of the hourly report variables.

Commands and Keywords for Input Functions

The commands and keywords used for input functions are these:

```
LOADS-FUNCTION and HVAC-FUNCTION Commands

FUNCTION Keyword
```

```
SUBR-FUNCTIONS Command

ASSIGN Command

CALCULATE Command

END-FUNCTION Command
```

An example input sequence for a function that modifies a window calculation in LOADS looks like:

```
INPUT  ..

   .....

W-1 = WINDOW
      .....
    $ Invoke function calculation for this window $
    FUNCTION = ( *NONE* , *FNW-l* ) ..
      .....

END ..

$ Define the function $

LOADS-FUNCTION       NAME = FNW-1 ..

ASSIGN
   function_var_1    = program_variable_1
   function_var_2    = program_variable_2
   .....
   function_var_N    = program_variable_N  ..

CALCULATE ..
     [User-defined FORTRAN-like routine goes here]

END-FUNCTION ..

COMPUTE ..
```

Before reading the following descriptions you should quickly review the LOADS and HVAC Input Function examples at the end of this section. Additional examples can be found in the sample runs.

LOADS-FUNCTION, HVAC-FUNCTION Commands

These commands define the characteristics of a function. The allowable number of functions is 100.

These commands must be entered after the END command and before the COMPUTE command. The LOADS-FUNCTION and HVAC-FUNCTION commands have only one keyword, NAME, that specifies the name of the function (up to 16 alphanumeric characters).

FUNCTION Keyword

FUNCTION is a keyword in LOADS for the BUILD-PARAMETERS, SPACE, EXTERIOR-WALL, DOOR, WINDOW, and UNDERGROUND-WALL commands, and in HVAC for the ZONE and SYSTEM commands.

As a keyword, it has the form FUNCTION = (*u-name1*, *u-name2*). Assigning u-name1 means that the calculation of the function with that name will be done *before* the execution of the subroutine associated with the function (see LOADS Example 3). Assigning u-name2 means that the calculation of the function of that name will be done *after* the subroutine's execution (see LOADS Example 1). If both u-names are assigned, the function with u-name1 will be calculated *before*, and the function with u-name2 will be calculated *after* the subroutine's execution. If only one U-name is given, the other must be entered as *NONE*. The FUNCTION keyword can be used at most once in each command.

Examples:

```
FUNCTION          = (*SPXX*,*NONE*),

FUNCTION          = (*NONE*,*FNEW2*),

FUNCTION          = (*FNEWl*,*FNEW2*).
```

Special-Use Function Keywords in LOADS-FUNCTION

DAYL-FUNCTION in the BUILD-PARAMETERS command:

DAYL-FUNCTION is the special function executed in LOADS subroutine DEXTIL. This subroutine determines the hourly exterior horizontal illuminance for the daylighting simulation.

DAYL-ILLUM-FN in the SPACE command:

DAYL-ILLUM-FN is the special function executed in LOADS subroutine DINTIL. This subroutine determines the hourly daylight illuminance and glare index at each reference point in a space (see "LOADS Example 4: Using Measured Daylight Factors").

DAYL-LTCTRL-FN in the SPACE command:

DAYL-LTCTRL-FN is the special function executed in LOADS subroutine DLTSYS. This subroutine determines the electric lighting reduction in response to daylight illuminance at each reference point in a space.

WINDOW-SPEC-FN in the WINDOW command:

WINDOW-SPEC-FN is the special function used in LOADS subroutines CALWIN, DINTIL, DCOF, and DREFLT. This function is used to alter variables involved in the daylighting calculation. WINDOW-SPEC-FN takes only one u-name, surrounded by asterisks, but without parentheses. See "LOADS Example 8: Using WINDOW-SPEC-FN."

SUBR-FUNCTIONS Command

This command allows functions to be executed in user-specified subroutines in the HVAC program.

These keywords are named after HVAC subroutines; they take only one U-name (that of a function) surrounded by asterisks, but without parentheses. For example: RESYS-1Z = *RESFN-1*

Example input:

```
INPUT  ..

   .....
```

```
SUBR-FUNCTIONS
   FCOIL-0           = *FN0*
   FCOIL-1Z          = *FN1*
   FCOIL-3           = *FN3*  ..
      .....

END ..

HVAC-FUNCTION        NAME = FN0
      .....

END-FUNCTION

HVAC-FUNCTION        NAME = FN1
      .....

END-FUNCTION

HVAC-FUNCTION        NAME = FN3
      .....

END-FUNCTION  ..

COMPUTE  ..
```

The SUBR-FUNCTIONS command has the following keywords (Table 6):

Table 6 HVAC function keywords

SYSTEMS-related keywords		
BERNOU-1=* U-name*	FCOIL-2Z=* U-name*	RESYS-1Z=* U-name*
CFMINF-0=* U-name*	FCOIL-3=* U-name*	RESYS-2Z=* U-name*
CFMINF-1=* U-name*	FNSYS1-1=* U-name*	RESYS-3Z=* U-name*
CONCHN-1=* U-name*	FNSYS1-2Z=* U-name*	RESYS-4Z=* U-name*
DAYCLS-1=* U-name*	FNSYS1-3Z=* U-name*	RESYS-5=* U-name*
DAYCLS-2=* U-name*	FNSYS1-4Z=* U-name*	SDSF-0=* U-name*
DAYCLS-3=* U-name*	FNSYS1-5=* U-name*	SDSF-1=* U-name*
DAYCLS-4=* U-name*	FTDEV-1=* U-name*	SSBASB-1=* U-name*
DAYCLS-5=* U-name*	FURNAC-1=* U-name*	SSFCOR-1=* U-name*
DAYCLS-6=* U-name*	HE-1=* U-name*	SUM-1=* U-name*
DDSF-0=* U-name*	HOURIN-1=* U-name*	SUM-2Z=* U-name*
DDSF-1=* U-name*	HPUNIT-1=* U-name*	SUM-3Z=* U-name*
DESFO-0=* U-name*	HTPUMP-0Z=* U-name*	SUM-4Z=* U-name*
DESFO-1=* U-name*	HTPUMP-1Z=* U-name*	SUM-5=* U-name*
DESIGN-1=* U-name*	HTPUMP-2=* U-name*	SZCI-0=* U-name*
DESIND-0=* U-name*	HVUNIT-0=* U-name*	SZCI-1Z=* U-name*
DESIND-1=* U-name*	HVUNIT-1Z=* U-name*	SZCI-2=* U-name*
DESPIU-0=* U-name*	HVUNIT-2=* U-name*	TDVPIU-0=* U-name*
DESPIU-1=* U-name*	HVUNIT-3=* U-name*	TDVPIU-1=* U-name*
DKTEMP-0=* U-name*	INDUC-0=* U-name*	TEMDEV-0=* U-name*
DKTEMP-1=* U-name*	INDUC-1Z=* U-name*	TEMDEV-1=* U-name*
DKTEMP-2=* U-name*	INDUC-2=* U-name*	TEMDEV-2=* U-name*
DKTEMP-3=* U-name*	OPSTRT-1=* U-name*	TEMDEV-3=* U-name*
DOETRM-0=* U-name*	PANEL-0Z=* U-name*	TSOLVE-0=* U-name*
DOETRM-1=* U-name*	PANEL-1=* U-name*	TSOLVE-1=* U-name*
DOUBLE-0=* U-name*	PIU-0=* U-name*	UNITH-0=* U-name*
DOUBLE-1=* U-name*	PIU-1=* U-name*	UNITH-1Z=* U-name*
EBAL-0=* U-name*	PTAC-0=* U-name*	UNITH-2Z=* U-name*
EBAL-1=* U-name*	PTAC-1Z=* U-name*	UNITH-3=* U-name*
ECONO-1=* U-name*	PTAC-2=* U-name*	UNITV-0=* U-name*
ECONO-2=* U-name*	RESVVT-0=* U-name*	UNITV-1Z=* U-name*
ECONO-3=* U-name*	RESVVT-1=* U-name*	UNITV-2=* U-name*
ECONO-4=* U-name*	RESVVT-2Z=* U-name*	VARVOL-0=* U-name*
FANPWR-1=* U-name*	RESVVT-3=* U-name*	VARVOL-1Z=* U-name*
FCOIL-0=* U-name*	RESVVT-4=* U-name*	VARVOL-2=* U-name*
FCOIL-1Z=* U-name*	RESYS-0=* U-name*	VARVOL-3=* U-name*
PLANT-related keywords		
BOILER-1=* U-name	GASCHLR-1=* U-name	PIPEDT-1=* U-name
BOILER-2=* U-name	GASCHLR-2=* U-name	PIPEDT-2=* U-name
BOILER-3=* U-name	GASCHLR-3=* U-name	LOOP1-2=* U-name
BOILER-4=* U-name	GASCHLR-4=* U-name	ENGINEGEN-1=* U-name
ABCHLR-1=* U-name	OTWR-1=* U-name	ENGINEGEN-2=* U-name
ABCHLR-2=* U-name	FCLR-1=* U-name	ENGINEGEN-3=* U-name
ABCHLR-3=* U-name	DHW1-1=* U-name	ENGINEGEN-4=* U-name
ABCHLR-4=* U-name	DHW1-2=* U-name	GASTURBINEGEN-1=* U-name
ELCHLR-1=* U-name	DHW1-3=* U-name	GASTURBINEGEN-2=* U-name
ELCHLR-2=* U-name	DHW1-4=* U-name	GASTURBINEGEN-3=* U-name
ELCHLR-3=* U-name	DHW2-1=* U-name	GASTURBINEGEN-4=* U-name
ELCHLR-4=* U-name	DHW2-2=* U-name	STEAMGEN-1=* U-name
ENGCHLR-1=* U-name	DHW2-3=* U-name	STEAMGEN-2=* U-name
ENGCHLR-2=* U-name	DHW2-4=* U-name	STEAMGEN-3=* U-name
ENGCHLR-3=* U-name	GLHXV-1=* U-name	STEAMGEN-4=* U-name
ENGCHLR-4=* U-name	GLHXV-2=* U-name	
	GLHXS-1=* U-name	
	GLHXS-2=* U-name	

ASSIGN Command

Variables used within the function are declared through the use of the ASSIGN command. These assignments are made through the definition of names of local variables (1- to 7-character names chosen by the user) or names of table variables.

Local Variables

A local variable name may be one of the following:

1. The name of a simulation variable from the Global Variables Listings, which contain variables used in the LOADS and HVAC programs. For example, in

   ```
   ASSIGN
      WS                = WNDSPD
   ```

   WS is the local variable name that you choose and WNDSPD is the name of the simulation variable, selected from the Global Variables Listings, that corresponds to windspeed. Note that your local variable name can be the same as the global variable name; e.g., ASSIGN WNDSPD = WNDSPD is allowed.

   The simulation variables have English units even in runs with metric input. The units are shown in the Global Variables Listings. Your function can be done in metric units but, in your function, you will have to (1) convert any simulation variables you use from English to metric, and (2) convert function results from metric back to English. For an example of this, see LOADS Example 3.

2. A numeric value. For example, ASSIGN WS=12. Exponential notation can be used for small or large numbers, as in ASSIGN SIGMA = 0.1714E-8.

3. A previously-defined PARAMETER name that is set equal to a numeric constant. For example:

   ```
   PARAMETER
      SPEED             = 12.0 ..
      .....

   ASSIGN
      WS                = SPEED ..
      .....
   ```

4. A macro expansion that results in a numeric constant. For example:

   ```
   ##set1   Altitude 567

      .....

   SITE-PARAMETERS
      Altitude          = Altitude[]
         .....

   ASSIGN
        ALT             =Altitude[] ..
   ```

5. The quantity SCHEDULE-NAME (U-name of a previously-defined schedule). In this case, the schedule value will be used within the function for the date and hour in question. This overwrites the value in the original SCHEDULE for that hour for the rest of the run. For example,

```
PEOP1 = SCHEDULE (.....) ..
   .....

SOUTH = SPACE
   PEOPLE-SCHEDULE    = PEOP1
   FUNCTION           = (*SPXX*,*NONE*)
      ......

END ..

LOADS-FUNCTION        NAME = SPXX ..

ASSIGN
   Y                  = SCHEDULE-NAME(PEOP1) ..
   ......
```

6. The quantity SCHEDULE (global variable name), where "global variable name" is the pointer name (found in the Global Variables Listings) that corresponds to a previously-defined SCHEDULE. The schedule value for the hour in question will be used (without overwriting the original value). Changing the example above, the input would be:

```
ASSIGN
   Y                  = SCHEDULE(KZPPL) ..
```

7. A previously-defined PARAMETER name that is set equal to the U-name of a schedule. The schedule value for the hour in question will be used. For example,

```
PARAMETER
   VTMULT             = TVIS-SCH-1 ..

TVIS-SCH-1 = SCHEDULE
   TYPE               = MULTIPLIER
   THRU DEC 31        (ALL) (1,24) (.35)  ..

WIN-1 = WINDOW
   VIS-TRANS-SCH      = TVIS-SCH-1
   FUNCTION           = (*WINXX*, *NONE*)
   .....

END ..

LOADS-FUNCTION        NAME = WINXX

ASSIGN
   Y                  = SCHEDULE-NAME(VTMULT) ..
   .....
```

Table Variables

A table variable takes values associated with the piecewise linear interpolation of curves defined with the TABLE keyword. For example, if TAB1 is the name of a table variable, it would appear in an expression of the form

```
TAB1 = TABLE (... , ...) (... , ...) (... , ...).
```

The TABLE keyword specifies a table of x-y pairs of data points. This table is used to define a piecewise linear curve that gives y as a function of x by linear interpolation. The x-values should be in increasing order. There is no limit on

the number of pairs that define the curve. Also, each TABLE keyword should have its own ASSIGN command. Mixing of TABLE and the other ASSIGN forms is not permitted. The x-y arguments can be defined through the use of the PARAMETER technique if desired.

In the CALCULATE section of a FUNCTION, the utility routine PWL returns the y-value of the piecewise linear curve given the x-value and the table variable name, as shown in the following example. Here, the table variable name is TAB1.

```
LOADS-FUNCTION        NAME = FN-1 ..

ASSIGN
   X1                 = simulation variable from Global
                        Variables Listing ..

ASSIGN
   TAB1               = TABLE (0,10) ( 0.2,20) (0.4,30)
                              (0.6,36) (0.8,38) (1.0,40) ..

CALCULATE ..

      Y1 = PWL (TAB1 , X1)
       .....
END

END-FUNCTION ..
```

In this function, the value of Y1 is determined from the value of X1 by linear interpolation between the points (0.,10), (0.2,20), etc., defined by TABLE. For example, if X1 = 0.1, then Y1 = 15. (If X1 is outside the range of x-values in TABLE, PWL linearly extrapolates to get the corresponding Y1 value.) See LOADS Example 4 for another example of using TABLE.

CALCULATE Command

The CALCULATE command informs LOADS or HVAC that the statements that follow it, which are written in a pseudo-FORTRAN language, are to be used to define the function. The valid FORTRAN declarative and executable statements and operations are given in Table 1. Also presented is a subprogram called PWL that performs the piecewise linear interpolation discussed above under the TABLE keyword.

All statements between "CALCULATE .." and "END-FUNCTION .." should begin in column 7, except for (1) statement numbers, which begin in column 1; (2) comment lines, which begin with a "C" in column 1; or (3) an arbitrary character, placed in column 6, that designates the continuation of a statement. None of these statements should contain tabs, otherwise an error message will result. The executable statement END terminates the CALCULATE section and must be present. (Note that this is not the same as the "END .." command that terminates the input.)

Variables used in the FUNCTION are all classified as real; other types do not exist. Integers may be used, but they will be treated as real.

END-FUNCTION Command

This command informs the LOADS or SYSTEMS program that the function definition is complete.

Supported Statements, Operations and Library/DOE-2 Functions

Table 7 Valid FORTRAN Statements and Operations

Arithmetic Operators	+, –, /, *, **, =, (,)		
Logical Operators	OR, AND, NOT, EQ, NE, GT, GE, LT, LE		
Executable Statements	CONTINUE	IF	REWIND
	END	PRINT	STOP[2]
	ENDFILE	READ[1]	WRITE[1]
	GO TO	RETURN	
Declarative Statements	FORMAT	SUBROUTINE	
Standard Functions	ABS (x)	AMIN (x1, x2)	EXP (x)
	ALOG (x)	AMOD (x1, x2)	INT (x)
	ALOG10 (x)	ATAN (x)	SIN (x)
	AMAX (x1, x2)	COS (x)	SQRT (x)

Library Functions

ACCESS (ixaa)	:	Returns the value of AA(ixaa) from the program's blank common array.
GET(v, i)	:	Returns the value of v(i), where v is assigned to a global variable that is dimensioned.

GETAA (naa)	:	IX = GETAA(NAA) gets a block of memory (of NAA words) in the program's main blank common array (the AA, or IA array) for exclusive use by functions, and IX is set to the value pointing to the beginning of the block. Any value can be stored inside this block by using
		`XX = STORE( XX, IX+ccc )`
		or
		`XX = ISTORE( XX, IX+ccc )`
		and may be retrieved by using
		`XX = ACCESS( IX+ccc )`
		or
		`XX = IACCESS( IX+ccc )`
		Where $0 \le ccc \le N-1$.

GETI(v, i)	:	Like GET(), however the global variable is an integer.
H (dbt, humrat, press)	:	Returns specific enthalpy (Btu/lb) of air as a function of dry-bulb temperature (F), humidity ratio (lb-water/lb-air), pressure (in-Hg).
IACCESS (ixia)	:	Like ACCESS but returns IA(ixia).
ISTORE (val, ixia)	:	X = ISTORE(X, IXIA) will convert the value of X into integer and then store it at IA(ixia) in the program's blank common array.
PUT(x, v, i)	:	X = PUT(X, V, I) will store the value of X into V(I), where V is assigned to a global variable that is dimensioned.

[1] Unformatted or formatted.

[2] For debugging only; program stops execution without printing reports.

Table 7 Valid FORTRAN Statements and Operations (continued)

PUTI(x, v, i)	:	PUTI is like PUT(), however the global variable is an integer.
PWL (table, x)	:	Piecewise linear interpolation function.
RAND (0)	:	RAND(0) returns a random number in the range [0,1].
RHFUNC	:	Returns relative humidity (%) as a function of dry-bulb (dbt, humrat, press) temperature (F), humidity ratio (lb-water/lb-air), and pressure (in-Hg)
STORE (val, ixaa)	:	X = STORE(X, IXAA) will store the value of X at AA(ixaa) in the program's blank common array.
V (dbt, humrat, press)	:	Returns specific volume of air (lb/cuft) as a function of dry-bulb temperature (F), humidity ratio (lb-water/lb-air), and pressure (in-Hg).
WBFS (dbt, humrat, press)	:	Returns wet-bulb temperature (F) as a function of (dbt, humrat, press) dry-bulb temperature (F), humidity ratio (lb-water/lb-air), and pressure (in-Hg).
WFUNC (dbt, humrat, press)	:	Returns humidity ratio (lb-water/lb-air) as a (dbt, RelHum, press) function of dry-bulb temperature (F), relative humidity (%), and pressure (in-Hg).
Note : the following library function is available only in HVAC functions.		
CVAL (MC , x, y)	:	Evaluates the curve stored in the curve block pointed to by MC; x and y are the independent variables of the curve.

Reading From and Writing To Files

FORTRAN I/O statements can be used inside functions to read values from files, or to write files for special reports and debugging. Both binary I/O and formatted I/O are supported. The following FORTRAN I/O statements can be used:

PRINT <format number>, var1, var2, ...
Writes to the OUTPUT file.

WRITE(<unit number>, <format number>) var1, var2, ...
Writes to FORTRAN unit number, using formatted I/O.

WRITE(<unit number>) var1, var2, ...
Same as previous but uses unformatted I/O.

READ(<unit number>, <format number>) var1, var2, ...
Reads from FORTRAN unit number, using formatted I/O.

READ(<unit number>)
Same as previous but uses unformatted I/O.

REWIND <unit number>
Rewinds FORTRAN unit number.

ENDFILE <unit number>
Terminates write to FORTRAN unit number.

<unit number>
is the integer FORTRAN unit number within the range acceptable by the operating system and/or compiler. However, 1 through 40 are reserved by the program and, therefore, should not be used in functions.

<format number>
corresponds to the statement number of the FORMAT statement that is inside the function. Note that the FORMAT statement must not contain Ixx edit descriptors (e.g., I10) since all variables inside functions are REAL.

Using PRINT or WRITE statements you can print out the values of variables directly from your function, thus bypassing the hourly reports. See LOADS Example 1. It is always a good idea to print out key variables in this way to be sure that your function is performing the way you expect.

Because the functions do not support the OPEN statement, the filename corresponding to the unit number is assigned by the operating system and/or compiler. For example, "READ(50) X,Y" will read from the file named "fort.50" in UNIX systems, or "FOR050.DAT" in VAX-VMS systems. (Check your system or compiler manual to determine the naming convention.) Correspondingly, "WRITE(60) U,V" will write to "fort.60" in UNIX or "FOR060.DAT" in VAX-VMS. As a result, you will have to change the command file that runs the program so that (1) it copies your input file to a file named "fort.50" in the directory in which the program is running (assuming a UNIX system), and (2) it saves the output file by copying "fort.60" to a file named by you. For an example of using Input Functions to read files, see LOADS Example 7.

LOADS Input Function Examples

LOADS Example 1: Print Solar Radiation and Surface Temperature

This function prints out the incident solar radiation and outside surface temperature of an exterior wall.

```
EW1 = EXTERIOR-WALL
   FUNCTION          = (*NONE*,*FNEW1*)
     .

  .

END ..

LOADS-FUNCTION       NAME = FNEW1 ..

ASSIGN
   QI                = SOLI
   TO                = T
   ..

CALCULATE ..
```

(Note: Begin the FUNCTION statements in column 7, except for statement labels, which begin in column 1, or continuation characters, which go in column 6. No TAB characters are allowed until END-FUNCTION.)

```
      PRINT 100, QI,TO
100   FORMAT(1H ,2F12.3)
      END (required)

END-FUNCTION ..

COMPUTE  ..
```

LOADS Example 2: Hypothetical Ventilation System

This function models a hypothetical ventilation system in which the infiltration airflow through a window is adjusted each hour to cancel the solar and conduction heat gain through the window.

```
WINDOW-1 = WINDOW
   FUNCTION          = (*NONE*,*FNWIN1*)
   .
   .

END  ..

LOADS-FUNCTION       NAME = FNWIN1 ..

ASSIGN
   TO                = DBTR   $outside dry-bulb temperature (R)
   TS                = TZONER $zone temperature (R)
   PR                = PATM   $atmospheric pressure
   QS                = QSOLG  $solar gain through window
   QC                = QCON   $heat conduction through window
   CI                = CFMW   $window infiltration, CFM
   ..

CALCULATE  ..

C              Add solar gain and conductance through window
      X = QS + QC
C              If solar plus conduction gain is less than or
C              equal to zero, or outside warmer than inside,
C              don't alter infiltration
      IF (X .LE. 0.0)  GO TO 80
      IF (TO .GE. TS)  GO TO 80
C              Calculate infiltration CFM
      DEN = PR/(.754*TZONER)
      CI  = X/(14.4*DEN*(TO-TS))
80    CONTINUE
      END

END-FUNCTION  ..

COMPUTE  ..
```

Note in this example that CI has been assigned to CFMW, which is the internal program variable that represents the infiltration CFM through a window. Through this assignment, the value of CI calculated by the function becomes the new value for CFMW.

LOADS Example 3: Outside Film Heat Transfer Coefficient

This function redefines the outside film heat transfer coefficient using the "Kimura Algorithm" (Kimura, Scientific Basis of Air Conditioning, 1977, p.85ff)

```
EW-2 = EXTERIOR-WALL
   FUNCTION           = (*FNEW2*,*NONE*)
     .
     .

END ..

LOADS-FUNCTION        NAME = FNEW2 ..

ASSIGN
   WS                 = WNDSPD $ wind speed
   WD                 = WNDDRR $ wind direction
   WA                 = XSAZM $ surface azimuth
   FU                 = FILMU $ air film conductance
   ..

CALCULATE ..

      RWD = WA + 3.1415 - WD
      IF(RWD .GT. 3.1415) RWD = 6.283 - RWD
C             Convert global windspeed from knots to M/S
      W = .553*WS
C             Get windspeed at wall surface
      IF(RWD .LT. 1.5708 .AND. W .GT. 2.) VC = .25*W
      IF(RWD .LT. 1.5708 .AND. W .LE. 2.) VC = .5
      IF(RWD .GE. 1.5708) VC = .3 + .05*W
C             Convert back to knots
      VC = 1.808*VC
C             Combined convective plus radiative air
C             film conductance for roughness = 3
      FU = 1.90 + .38*VC
      END

END-FUNCTION ..

COMPUTE  ..
```

LOADS Example 4: Using Measured Daylight Factors

This function calculates daylight levels in a space using coefficients obtained by the user from physical scale model measurements of the ratio of interior to exterior illuminance. In the function, the coefficients are multiplied by the hourly total exterior illuminance from sun and sky to give the interior daylight illuminance. The measured coefficients for solar altitudes of 0, 10, 30, 50, and 70 degrees are entered using TABLE.

Notes:

1. This function assumes there are no movable shading devices on the windows that would alter the interior illuminance depending on whether the shades were open or closed.

2. This function does not re-calculate glare, so that the glare levels reported by the program should be ignored.

3. This function is illustrative only; the coefficients in an actual case could also depend on other factors, such as solar azimuth, cloud cover, etc.

```
SPACE1 = SPACE
   DAYLIGHTING        = YES
   ..... other daylighting-related keywords
   DAYL-ILLUM-FN      = (*NONE*,*MODEL-DATA-FN*) ..
   .
   .

END ..

LOADS-FUNCTION        NAME = MODEL-DATA-FN ..

ASSIGN
   RDNCC              = RDNCC   $ DIRECT NORMAL SOLAR, BTU/AREA  $
   BSCC               = BSCC    $ DIFFUSE HORIZ SOLAR, BTUH/SF   $
   RAYCOS3            = RAYCOS3 $ SINE OF SOLAR ALTITUDE         $
   PHSUND             = PHSUND  $ SOLAR ALTITUDE IN DEGREES      $
   ILLUM              = DAYLIGHT-ILLUM1 $ ILLUMINANCE AT REF PT 1$
                                        $ FOOTCANDLES            $
                                        $(REF PT 2 NOT USED)     $
   ..

ASSIGN
   TAB1               = TABLE ( 0,    0)
                              (10,.0050)
                              (30,.0070)
                              (50,.0085)
                              (70,.0100)
                              (90,.0100)  ..

CALCULATE ..

C              Get exterior horizontal illuminance from direct
C              sun in footcandles (Lumens/SF). Assumes that the
C              luminous efficacy of direct solar radiation
C              is 100 Lumens/Watt = 29.3 Lumens/Btuh.
      IDIRH = RDNCC * RAYCOS3 * 29.3
C              Get exterior horizontal illuminance from sky.
```

```
C              Assumes that the luminous efficacy of diffuse
C              solar radiation = 125 Lumens/Watt
C                              = 36.6 Lumens/Btuh.
      IDIFH = BSCC * 36.6
C              Get total exterior illuminance
      ITOTH = IDIRH + IDIFH
C              Get interior daylight illuminance for
C              current solar altitude
      ILLUM = PWL (TAB1,PHSUND) * ITOTH
      END

END-FUNCTION ..

COMPUTE  ..
```

LOADS Example 5: Looping Logic

```
SPACE1 = SPACE
   .
   .
   FUNCTION          = (*NONE*,*SP-FN-TEST-1*) ..

END ..

LOADS-FUNCTION       NAME = SP-FN-TEST-1 ..

ASSIGN
   DAY               = IDAY
   MZ                = MZ
   HR                = IHR
   MZEXT             = MZEXT
   LMX               = LMX
   NEXTS             = NEXTS
   LMWI              = LMWI
   QZTOT             = QZTOT
   MON               = IMO
   S                 = SCHEDULE(SCHED-1)
   MWI               = MWI
   WIAREA            = WIAREA
   MX                = MX
   XSQCOMP           = XSQCOMP
   MXWIN             = MXWIN
   YR                = IYR
   ZAREA             = ZFLRAR
   ..

CALCULATE ..

      PRINT 1
1     FORMAT(21H TEST OF SP-FN-TEST-1)
      PRINT 2, YR,MON,DAY,HR,S
2     FORMAT(1X,4F10.1,F8.2)
      PRINT 3, ZAREA,QZTOT
3     FORMAT(10X,6HZAREA=,F6.1,5X,6HQZTOT=,F8.1)
C             Initialize EXTERIOR-WALL pointer and counter.
      MX = MZEXT
      NX = 0
C             Loop through EXTERIOR-WALLs
100   NX = NX+1
C             Increment EXTERIOR-WALL count and exit
C             EXTERIOR-WALL loop it finished
      IF (NX .GT. NEXTS)  GO TO 900
      PRINT 4, NX,XSQCOMP
4     FORMAT(30X,3HNX=,F3.0,8HXSQCOMP=,F10.1)
C             Initialize window pointer and counter.
      MWI = MXWIN
      NW  = 0
C             Increment window counter and exit window loop
C             if finished
200   NW = NW+1
      IF (NW .GT. NWIN)  GO TO 400
```

```
      PRINT 5, NW, WIAREA
5     FORMAT(40X,3HNW=,F3.0,3X,7HWIAREA=,F10.1)
C             Increment window pointer to get next window
      MWI = MWI+LMWI
      GO TO 200
400   CONTINUE
C             Increment EXTERIOR-WALL pointer to get
C             next EXTERIOR-WALL.
      MX = MX+LMX
      GO TO 100
900   CONTINUE
C             Stop simulation at hour 9.
      IF (HR .EQ. 9)  STOP
      END

END-FUNCTION  ..

COMPUTE  ..
```

LOADS Example 6: Variable Shading Coefficient

This function is used to vary the shading coefficient of a window depending on the value, RTOT, of the total (direct plus diffuse) solar radiation incident on the window. If RTOT < 10 Btuh/sf, the window is "clear" (shading coefficient = 0.8). As RTOT increases from 10 to 100 Btuh/sf, the window darkens, reaching a shading coefficient of 0.2 at 100 Btuh/sf. For RTOT > 100 Btuh/sf, the shading coefficient remains at 0.2.

```
WINDOW-1 = WINDOW
        .
        .
   FUNCTION           = (*WSCSGC*,*PRINTQ*) ..

END ..

LOADS-FUNCTION        NAME = WSCSGC ..

ASSIGN
   RTOT               = RTOT
   SHACO              = GSHACO
   ..

CALCULATE  ..

C              Calculate shading coefficient
      SHACO = -0.00667*RTOT + 0.8667
      IF (RTOT.LE.10.)  SHACO = 0.8
      IF (RTOT.GE.100)  SHACO = 0.2
      END

END-FUNCTION   ..

LOADS-FUNCTION        NAME = PRINTQ   ..

ASSIGN
   HR                 = ISCHR
   IPRDFL             = IPRDFL
   SHACO              = GSHACO
   RTOT               = RTOT
   ..

CALCULATE

C              PRINTQ is used at the end of the subroutine's
C              execution; its purpose in this example is to
C              verify values generated by WSCSGC. Don't print
C              any values until building start-up is complete.
C              IPRDFL is a counter used for building start-up.
C              When the building has cycled through seven days,
C              IPRDFL goes to 0 (zero). Print values between the
C              hours of 6 and 21 only.
      IF (IPRDFL .GT. 0.)  RETURN
      IF ((HR .LT. 6.) .OR. (HR .GT. 21.))  RETURN
C              Standard FORTRAN print statement
      PRINT 20,HR,RTOT,SHACO
20    FORMAT(1X,3HHR=,F3.0,3X,5HRTOT=,F8.2,3X,6HSHACO=,F8.3/)
      END
```

```
END-FUNCTION ..

COMPUTE  ..
```

LOADS Example 7: Reading Measured Schedule Values from a File

In applications where you want to reconcile the measured performance of a building with the simulation, it may be desirable to input measured profiles (such as for lighting) rather than try to replicate these profiles using the SCHEDULE command capabilities. This might be the case, for example, if the actual lighting profile is so variable that it cannot accurately be represented by a series of different DAY-SCHEDULEs and and WEEK-SCHEDULEs.

The following example shows how to use input functions to read in 8760 hours of measured space lighting power and equipment power values into the LOADS simulation. The input function puts these values into the lighting schedule and equipment schedule, respectively, for the space.

To do this, we define "place-holder" schedules whose hourly values will be altered by the input function. We then refer to these schedules in the SPACE command.

Values are read at the beginning of every hour of the simulation using a building-level "before" function.

```
   $ Use Input Functions to read lighting and equipment
   $ profiles from a file $

INPUT  ..

BUILD-PARAMETERS
   FUNCTION          = (*READER*,*NONE*)
         .
         .

LIGHT-SC-1 = SCHEDULE
   TYPE              = MULTIPLIER
   THRU DEC 31         (ALL) (1,24)(0)  ..

EQUIP-SC-1 = SCHEDULE
   TYPE              = MULTIPLIER
   THRU DEC 31         (ALL) (1,24)(0)  ..

SP-1 = SPACE
   LIGHTING-KW       = 1.0
   LIGHTING-SCHEDULE = LIGHT-SC-1
   EQUIPMENT-KW      = 1.0
   EQUIP-SCHEDULE    = EQUIP-SC-1
   .
   .

   $ In above, LIGHTING-KW and EQUIPMENT-KW are set to 1.0
   $ since the schedule values that are read in will contain
   $ the actual KW's. Alternatively, LIGHTING-KW and
   $ EQUIPMENT-KW could be actual peak KW values, and the
   $ schedule values could be fractions

END ..
```

```
   $ The following function reads measured lighting and equipment
   $ power values every hour from FORTRAN unit 50. This unit is
   $ assigned a default filename $ by the operating system
   $ (e.g.,fort.50 in SunOS, FOR050.DAT in VAX/VMS).  Thus,if
   $ you are running VAX/VMS, your command file that runs DOE2BDL
   $ and simulation should copy the data file (that you want the
   $ function to read) to FOR050.DAT  The file that is read
   $ contains 8760 lines of measured data. Each line contains
   $ lighting power (kW) in columns 1-10 and equipment
   $ power (kW)in columns 11-20.

LOADS-FUNCTION        NAME = READER ..

ASSIGN
   LS                 = SCHEDULE-NAME( LIGHT-SC-1 )
   ES                 = SCHEDULE-NAME( EQUIP-SC-1 )
   IHR                = IHR    $ hour number
   IDAY               = IDAY   $ day number
   IMON               = IMON   $ month number
   ..

CALCULATE ..

C              We need to rewind the data file at the end of
C              the warm-up period so that when the
C              simulation begins we are at the beginning of
C              the data file. We know we are at the end
C              of the warm-up since at this time IHR and IMON
C              will have been reset to 1.
      IF (IHR + IDAY + IMON .EQ. 3 )  REWIND 50
      READ( 50, 1 ) LS, ES
l     FORMAT( 2Fl0.l )
      END

END-FUNCTION ..

COMPUTE  ..
```

LOADS Example 8: Using WINDOW-SPEC-FN

The WINDOW-SPEC-FN keyword in the WINDOW command allows an Input Function to be executed in the LOADS subroutines CALWIN, DCOF, DINTIL or DREFLT. In which of these routines the function is executed depends on the value of the variable FNTYPE, as shown in the following example. FNTYPE = 3, 4, 5 or 6 causes the function to be executed in CALWIN, DINTIL, DCOF or DREFLT, respectively.

```
WINDOW-1 = WINDOW
        .
        .
   WINDOW-SPEC-FN     = (*FN-1*) ..

END ..

LOADS-FUNCTION        NAME = FN-1 ..

ASSIGN
   FNTYPE             = FNTYPE
   (other assignments) ..

CALCULATE  ..

C              Evaluate the function only in subroutine DINTIL
      IF (FNTYPE .NE. 4)  RETURN
C              Code to be executed in DINTIL
        .
        .
      END

END-FUNCTION   ..

COMPUTE  ..
```

HVAC Input Function Examples

HVAC Example 1: Cold-Deck Supply Air Temperature Reset

In this example the return air temperature is used to reset the cold-deck supply air temperature to the reheat coils in a Reheat Fan System. Here, TRLAST is the last-hour value of the return air temperature, TR, which is stored by the function SFN1 .

```
INPUT  ..

  .......

   $ Modify subroutine DKTEMP at function access
   $ point DKTEMP-3

SUBR-FUNCTIONS
   DKTEMP-3          =*DKTEMPF* .. $

FS-SYS = SYSTEM
   TYPE              = RHFS
   FUNCTION          = (*NONE*,*SFN1*) ..

   $ systems-after-function SFNl is used to save the value
   $ of TR to be used next hour by the function DKTEMPF

END ..

   $ This function resets the cold-deck temperature (TC)
   $ according to the return air temperature (TR). The reset
   $ characteristic obeys the following piecewise linear
   $ relationship:
   $    TC = 70F if TR < 70F
   $    TC = varies linearly from 70F to 50F for 70F < TR < 80F
   $    TC = 50F for TR > 80F

HVAC-FUNCTION        NAME=DKTEMPF ..

ASSIGN
   IHR               = IHR
   IDAY              = IDAY
   IMO               = IMO
   INILZE            = INILZE
   NWRMUP            = NWRMUP
   ..

ASSIGN
   TR                = TR             $ return air temp $
   TC                = TC  ..         $ cold-deck temp  $
   ..

ASSIGN
   TRLAST            = F-SYS-VARI  .. $ where last-hour TR$
                                      $ value is stored $
   ..
```

```
ASSIGN
   TCTR              = TABLE  ( 0,70)
                              (70,70)
                              (80,50)
                             (100,50) .. $for the PWL function $
                                        $ of TC vs TR $
   ..

CALCULATE  ..

c              If TRLAST not available, do nothing
      IF (TRLAST .EQ. 0.)  RETURN
c              Compute cold-deck temperature
      TC = PWL(TCTR, TRLAST)
c              The following four executable lines can
c              be un-commented to get debug print
c              Skip in initialization cycle
c     IF (INILZE .LT. NWRMUP)  RETURN
c              Print only one day – July 7
c     IF ((IDAY .NE 7) .OR. (IMO .NE. 7))  RETURN
c     PRINT 1, IMO, IDAY, IHR, TC, TR
c1    FORMAT(' DKTEMPF -- IMo,IDAY,IHR=',3f3.0,
     &   ' TC=',f7.2,' TR=',f7.2)
      END

END-FUNCTION ..

   $ This function stores the value of the return air
   $ temperature (TR) so that the function 'DKTEMPF'
   $ can use it next hour.

HVAC-FUNCTION         NAME=SFN1 ..

ASSIGN
   TR                = TR            $ return air temp $
   TRLAST            = F-SYS-VARI .. $ stores last-hour TR$
   ..

CALCULATE ..

c              Store return air temperature
      TRLAST = TR
      END

END-FUNCTION  ..

COMPUTE  ..

STOP ..
```

HVAC Example 2: Dry-bulb Economizer

A dry-bulb economizer is modeled that sets the outside air fraction to minimum if the outside air temperature exceeds the return air temperature.

```
INPUT  ..

 .....

   $ Modify subroutine ECONO at function access point ECONO-2

SUBR-FUNCTIONS
   ECONO-2           =*econoFNa* ..

 .....

END ..

HVAC-FUNCTION        NAME = econoFNa ..

ASSIGN
   IHR               = IHR
   IDAY              = IDAY
   IMO               = IMO
   INILZE            = INILZE
   NWRMUP            = NWRMUP
   PO                = PO               $ OA fraction        $
   TR                = TR               $ return-air T       $
   POMXXX            = POMXXX           $ min OA fraction    $
   TAPPXX            = TAPPXX           $ mixed-air setpoint $
   DBT               = DBT              $ OA drybulb         $
   ECONOLT           = DRY-BULB-LIMIT   $ from BDL input     $
   ECONOLL           = ECONO-LOW-LIMIT  $ from BDL input     $
   MAXOA             = MAX-DA-FRACTION  $ from BDL input     $
   ..

CALCULATE ..

c              Compute PO to satisfy TAPPXX and limit it
c              depending on bounds
      PO = POMXX
      IF (ABS( DBT-TR ) .gt. 0.1)
     &  PO = AMAX(POMXXX, (TAPPXX-TR)/(DBT-TR))
      PO = AMIN1(PO, MAXOA)
      IF ((ECONOLL .NE. 0) .and. (DBT .LT. ECONOLL))  PO = POMXXX
      IF ((ECONOLT .NE. 0) .and. (DBT .GE. ECONOLT))  PO = POMXXX
c              compare against return temperature
      IF (DBT .GT. TR)  PO = POMXXX
c
c              The following six executable lines can be
c              un-commented to get debug print
c              Skip if in initialization
c     IF (INILZE .LT.  NWRMUP)  RETURN
c     PRINT 1, IMO,IDAY,IHR, POMXXX,TAPPXX, ECONOLT, ECONOLL,
c    &      MAXOA, PO, TR, DBT
c1    FORMAT( ' econoFN-- ',3f3.0,' POMXXX,TAPPXX=',f5.2,f6.2,
```

```
c    & ' ECONOLT=',f5.2,' ECONOLL=',f5.2,' MAXOA=',f5.2'
c    & ' PO=',f5.2,'TR=',f5.2,'DBT=',f5.2 )
      END

END-FUNCTION  ..

COMPUTE  ..

STOP  ..
```

HVAC Example 3: Enthalpy Economizer

We model an enthalpy economizer that sets the outside air fraction to minimum if the outside air enthalpy is greater than a setpoint value.

```
INPUT  ..

 .....

   $ Modify subroutine ECONO at function access point ECONO-2

SUBR-FUNCTIONS
   ECONO-2           =*econoFNb* ..

 .....

END ..

HVAC-FUNCTION        NAME = econoFNb ..

ASSIGN
   IHR               = IHR
   IDAY              = IDAY
   IMO               = IMO
   INILZE            = INILZE
   NWRMUP            = NWRMUP
   ENTHSET           = 30               $ OA enthalpy setpt  $
   ENTHAL            = ENTHAL           $ OA enthalpy        $
   PO                = PO               $ OA fraction        $
   TR                = TR               $ return-air T       $
   POMXXX            = POMXXX           $ min OA fraction    $
   TAPPXX            = TAPPXX           $ mixed-air setpt    $
   DBT               = DBT              $ OA drybulb         $
   ECONOLT           = DRY-BULB-LIMIT   $ from BDL input     $
   ECONOLL           = ECONO-LOW-LIMIT  $ from BDL input     $
   MAXOA             = MAX-DA-FRACTION  $ from BDL input     $
   ..

CALCULATE ..

c              Compute PO to satisfy TAPPXX and limit it
c              depending on bounds
      PO = POMXXX
      IF (ABS( DBT-TR ) .GT. 0.1)
     &  PO = AMAX(POMXXX, (TAPPXX-TR)/(DBT-TR))
      PO = AMIN(PO, MAXOA)
      IF ((ECONOLL .NE. 0) .AND. (DBT .LT. ECONOLL))  PO = POMXXX
      IF ((ECONOLT .NE. 0) .AND. (DBT .GE. ECONOLT))  PO = POMXXX
c              Set oa fraction to minimum if oa enthalpy
c              exceeds setpoint
        IF (ENTHAL .GT. ENTHSET)  PO = POMXXX
c
c              Uncomment the following six lines for debug
c              Skip if in initialization
c     IF (INILZE .LT. NWRMUP )  RETURN
c     PRINT 1, IMO,IDAY,IHR, POMXXX,TAPPXX,ECONOLT, ECONOLL,
```

```
c    &         MAXOA, PO TR, DBT
c1    FORMAT( ' econoFN-- ',3f3.0,' POMXXX,TAPPXX=',f5.2,f6.2,
c    & ' ECONOLT=',f5.2,' EcoNoLL=~,f5~2~, MAxoA=~,f5~2'
c    & ' Po=~,f5~2~, TR=',f5.2,' DBT=',f5.2 )
      END

END-FUNCTION  ..

COMPUTE  ..

STOP ..
```

PARAMETRIC-INPUT PARAMETER

The parametric run capability helps you make multiple runs in order to study the effect of varying one or more input parameters. The commands used are PARAMETER and PARAMETRIC-INPUT. A short example will illustrate how it is done.

Example:

```
TITLE
   LINE-1            * Centrifugal Chiller*
   LINE-2            * One-speed pumps * ..

INPUT ..

     $ Define Parameters

PARAMETER
   ChillerType       = ELEC-OPEN-CENT
   PumpCtrl          = ONE-SPEED-PUMP
   ..

.......................................

.......................................

.......................................

“CHW Pump” = PUMP
   CAP-CTRL          = PumpCtrl
   ..

“HW Pump” = PUMP
   CAP-CTRL          = PumpCtrl
   ..

etc.

“Chiller 1” = CHILLER
   TYPE              = ChillerType
   CAPACITY          = 1.2
   CHW-LOOP          = “The CHW Loop”
   . . .

“Chiller 2” = CHILLER
   TYPE              = ChillerType
   CAPACITY          = 2.3
   CHW-LOOP          = “The CHW Loop”
   . . .

END ..

COMPUTE ..
```

```
PARAMETRIC-INPUT FOR DOE2 ..

TITLE
   LINE-1            * Reciprocating Chillers *  ..

PARAMETER
   ChillerType       = ELEC-OPEN-REC
   PumpCtrl          = ONE-SPEED-PUMP
   ..

END ..

COMPUTE ..

PARAMETRIC-INPUT FOR DOE2 ..

TITLE
   LINE-1            * Reciprocating Chillers *  ..
   LINE-2            * Variable-speed pumps *  ..

PARAMETER
   ChillerType       = ELEC-OPEN-REC
   PumpCtrl          = VAR-SPEED-PUMP
   ..

END ..

COMPUTE ..

STOP ..
```

The example shows three runs (three COMPUTE commands) in which the chillers are varied from centrifugal to reciprocating, and the pumps from constant-speed to variable-speed. The input for the first run is normal except for the use of the PARAMETER command to define the chiller type and pump capacity control. Setting TYPE equal to ChillerType in the CHILLER command results in the TYPE being set to ELEC-OPEN-CENT. That is, BDL substitutes ELEC-OPEN-CENT for ChillerType. One other unusual feature is that the first TITLE command occurs before the INPUT command. This is necessary if TITLE is to be varied in the PARAMETRIC-INPUT sections of the input.

The second and third runs are described completely by PARAMETRIC-INPUT, TITLE, and PARAMETER commands. Notice that the complete input describing the building and HVAC equipment doesn't need to be repeated – the program takes care of this for you. Only the things that change need to be specified: the new title and the changed parameter values.

PARAMETER

The syntax for the PARAMETER command is simply PARAMETER followed by parameter-name = value pairs, terminated of course with a double period. The parameter-name can be up to 32 characters long and there can be up to 500 such names defined. The value can be a numeric value or a symbol such as the u-name of a command.

PARAMETRIC-INPUT

The PARAMETRIC-INPUT command is used to specify changes in parameters used in a previous complete input. The only commands that can be used after a PARAMETRIC-INPUT and before the subsequent END command are PARAMETER, TITLE, and LIST or DIAGNOSTIC.

Multiple Runs

It is possible to concatenate inputs together, separated by COMPUTE commands and terminated with a STOP command, to perform multiple runs. As described above, this becomes really useful when PARAMETER and PARAMETRIC-RUN are used to create parametric runs. In DOE-2.1E it was possible to do a LOADS simulation followed by multiple SYSTEM and PLANT simulations, or a LOADS and SYSTEMS simulation, followed by multiple PLANT simulations. This is no longer possible since the LOADS, SYSTEMS, and PLANT inputs are no longer separate. But it is still possible to do a LOADS and HVAC simulation followed by multiple ECONOMICS calculations. To do this it is necessary to define the ECONOMICS input as a separate input section. The building envelope and HVAC equipment are specified as usual between an INPUT and an END command. The economics data is specified between an INPUT FOR ECONOMICS and an END command. The building simulation is triggered by a COMPUTE command as usual; the economics calculation is triggered by a COMPUTE ECONOMICS. Any number of economics inputs and calculations can follow a building envelope and HVAC simulation. And finally, the parametric input feature can be used with the multiple economics runs. An example is the best way to illustrate all this.

Example

```
INPUT ..

   Envelope and HVAC input

END ..

COMPUTE..

INPUT FOR ECONOMICS ..

PARAMETER
   SumECost          = .0829
   WinECost          = .0778  ..

E-SM = BLOCK-CHARGE
   BLOCK-SCH         = SEASON
   SCH-FLAG          = 2
   BLOCK1-TYPE       = ENERGY
   BLOCKS-1          = (1250)
   COSTS-1           = (SumECost)
   BLOCK2-TYPE       = KWH/KW
   BLOCKS-2          = (125,1)
   COSTS-2           = (.0829,.0514)
   LIMITS-2          = (0,0) ..

E-WN = BLOCK-CHARGE
   BLOCK-SCH         = SEASON
   SCH-FLAG           1
   BLOCK1-TYPE       = ENERGY
   BLOCKS-1          = (1250)
   COSTS-1           = (WinECost)
   BLOCK2-TYPE       = KWH/KW
   BLOCKS-2          = (125,1)
   COSTS-2           = (.0778,.0514)
   LIMITS-2          = (0,0 ..
```

```
ELEC-COST = UTILITY-RATE
   TYPE               = ELECTRICITY
   MONTH-CHGS         = (21.75)
   DEMAND-CHGS        = (.81)
   BLOCK-CHARGES      = (E-SM,E-WN) ..

SEASON = SCHEDULE
   TYPE               = FLAG
   THRU APR 30        (ALL) (1,24) (1)
   THRU OCT 31        (ALL) (1,24) (2)
   THRU DEC 31        (ALL) (1,24) (1) ..

GAS-CH = BLOCK-CHARGE
   BLOCK1-TYPE        = ENERGY
   BLOCKS-1           = (1200,1)
   COSTS-1            = (.22,.15) ..

GAS-COST = UTILITY-RATE
   TYPE               = NATURAL-GAS
   MONTH-CHGS         = (10.73)
   BLOCK-CHARGES      = (GAS-CH) ..

END ..

COMPUTE ECONOMICS ..

PARAMETRIC-INPUT FOR ECONOMICS ..

TITLE
   LINE-2             * Vary Energy Costs * ..

PARAMETER
   SumECost           = .0674
   WinECost           = .0554 ..

END ..

COMPUTE ECONOMICS ..

STOP ..
```

The legal commands between an INPUT FOR ECONOMICS and the subsequent END command are:

SCHEDULE	WEEK-SCHEDULE
DAY-SCHEDULE	SET-DEFAULT
DIAGNOSTIC or LIST	ABORT
END	PARAMETER
COMPONENT-COST	BASELINE
ECONOMICS-REPORT	UTILITY-RATE
BLOCK-CHARGE	RATCHET
POLLUTANT-COEFFS	

Section

3

Envelope Components

OVERVIEW OF LOADS CALCULATION METHODOLOGY

The LOADS program calculates the sensible and latent components of the hourly thermal load for each space in the building. In the program, *load* is defined as the constant-temperature extraction rate, which is the amount of heat that must be removed from the space air each hour to keep its temperature constant at the value you specify for TEMPERATURE in the SPACE command (the default value for TEMPERATURE is 70F or 21C). Given this constant-temperature extraction rate and the temperature at which it was calculated, the HVAC program then determines the actual space temperature and the actual extraction rate needed to satisfy the thermostat set point.

Calculating Loads from Heat Gains

A two-step procedure is used to calculate the load. First, the *heat gain* to the space is calculated. Heat gain is the amount of heat that goes into the air or is absorbed by the walls and furnishings in the space. Second, *room weighting factors* are applied to the heat gain to determine the load. Only in spaces with no thermal mass are the heat gain and load the same. This is because the part of the heat gain that is absorbed by the walls and furnishings (which are the space's thermal mass) acts to raise the temperature of the thermal mass over a number of hours. The warmer thermal mass eventually adds heat to the air by convection, adding to the load. The room weighting factors, therefore, account for the time delay between a heat gain and when it appears as a load on the space air. In general, the smaller the thermal mass, the faster the heat gain appears as a load.

Calculating Loads from Extraction Rates

The Loads program sums the loads from each type of heat gain into a total load, which it passes to the HVAC program. Applying air temperature weighting factors to the total load, and using the thermostat equation for the space, the HVAC program determines the extraction rate and actual space temperature. The procedure used to go from heat gains, to loads, to extraction rates, is shown schematically in Figure 1.

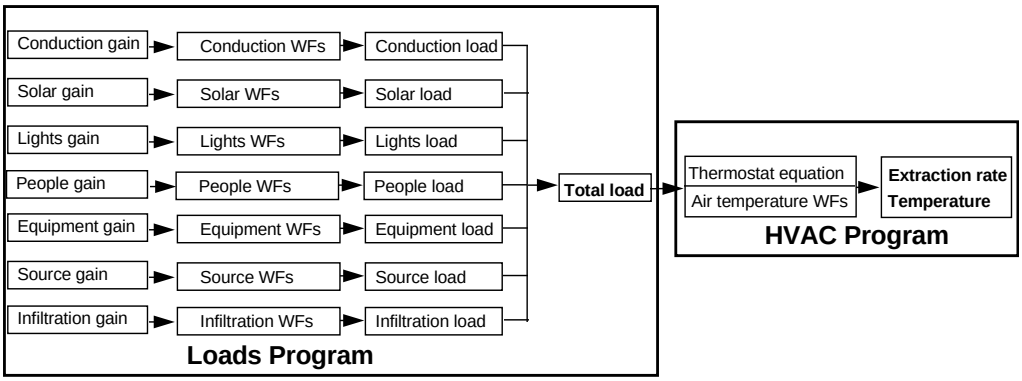


Figure 1 Calculation of extraction rate and temperature from heat gains

Types of Heat Gain

The program considers seven different types of space heat gain:

- Conduction
- Solar radiation through windows
- Infiltration
- Lights
- People
- Equipment
- Sources

The program calculates room weighting factors separately for each type of heat gain. This is required since the weighting factors depend on how much of the gain is radiative and how much is convective. This *convective/radiative split* varies from 100%/0% for infiltration, to 40%/60% for people and equipment, to 0%/100% for solar radiation. The higher the convective percentage, the faster a heat gain appears as a load.

The load produced by each type of heat gain is calculated by applying room weighting factors to current-hour and previous-hour heat gains. These component loads are then summed and passed to the HVAC program.

In the following we briefly describe each type of heat gain.

Conduction

Conduction occurs through solid surfaces, such as exterior walls and roofs, interior walls, exterior windows, interior windows and surfaces in contact with the ground (underground walls and floors). For surfaces in contact with the outside air, conduction is determined by outside conditions (air temperature, sky temperature, ground surface temperature, wind speed, wind direction and solar radiation), orientation, inside conditions (room air temperature, inside air film conductance), and the material properties of the wall or window. For surfaces in contact with the

ground, conduction is determined by ground temperature, room air temperature and material properties of the wall or floor. In general, conduction is highest when the outside-inside temperature difference is large and the U-value of the wall or window is large. Conduction heat gain in a given hour can be positive or negative. It is generally positive in the summer and negative in the winter. Conduction heat gain is considered to be 100% sensible. The latent component, associated with moisture transport and storage in the walls, is neglected.

Solar Radiation through Windows

Solar radiation can be transmitted through the glazing of windows, or it can be absorbed by the glazing. Part of the absorbed radiation conducts into the space, adding to the heat gain. The amount of solar gain through a window depends on the intensity of solar radiation on the window, and the transmittance and conductance of the window. The program considers three sources of incident solar radiation on windows (and walls): directly from sun, directly from sky, and reflected from ground (reflection from other external surfaces, like neighboring buildings, is not considered). Solar heat gain is always positive and is 100% sensible.

Infiltration

Infiltration is due to outside air entering the space through cracks or other openings. The heat gain from infiltration is proportional to the infiltration air flow rate and the outside-inside air temperature difference. Depending on the infiltration calculation method, the infiltration air flow rate can depend on wind speed,

opening area, and/or outside-inside temperature difference. Infiltration heat gain in a given hour can be positive or negative. It is generally positive in the summer and negative in the winter. It has sensible component, which depends on the outside-inside temperature difference, and a latent component, which depends on outside-inside humidity ratio difference.

Lights

The heat gain from lights depends on the kW of lighting in the space, the type of lighting (incandescent, fluorescent, etc.) and the lighting schedule. Depending on the option chosen, all of the heat gain from lights can go into the space or part of it can go into the plenum or return air.

People

Occupants produce both sensible and latent heat gain. People heat gain depends on the number of occupants, the occupancy schedule and the activity level. People heat gain is always positive.

Equipment

Equipment heat gain is produced by electrical equipment such as computers and copy machines. It can have both sensible and latent components. Equipment heat gain depends on the kW of equipment and the equipment schedule. Equipment heat gain is always positive, but can have sensible and latent components.

Sources

A source is a device, other than lighting or equipment, that produces an internal heat gain. Examples are oven and hot-water heaters. Heat gains from sources can be positive or negative, sensible and/or latent. Source heat gain depends on the intensity and schedule of the source.

Using Multipliers in LOADS Input Conduction

In the LOADS input, floors, spaces, walls, windows and doors have multiplier keywords that allow you to easily replicate these components. Using a multiplier on a window, for example, multiplies the area of the window and multiplies the conduction, solar and infiltration gains through the window. Table 8 shows the building components

on which multipliers can be used. It also indicates the effect of the multiplier, and gives warnings about cases in which multipliers should be avoided.

Notes:

1. Multipliers are multiplicative! For example, assume a window with a multiplier is in an exterior wall with a multiplier Mwall, which in turn is in a space with a multiplier Mspace. Then:

   Total spaces = Mspace

   Total walls in original space = Mwall

   Total walls in multiplied spaces = Mwall x Mspace

   Total windows in original wall = Mwin

   Total windows in multiplied walls in original space = Mwin x Mwall

   Total windows in multiplied spaces = Mwin x Mwall x Mspace

2. For all components except exterior walls and underground walls the multiplier should be an integer greater than or equal to one. For exterior walls and underground walls, fractional multipliers can be used. For example you could divide a single exterior stud wall in two walls, each with the same area as the original wall, but one with MULTIPLIER of 0.15 to represent the stud portion of the original wall and one with a MULTIPLIER of 0.85 to represent the between-stud portion. This allows you to assign different constructions to the two walls. A better approach, which avoids multipliers, is to divide the wall into two walls, one with 15% of the original area, and one with 85% of the original area.

3. FLOOR-MULTIPLIER can be used to simplify the input for multi-storied buildings where a number of the floors are thermodynamically identical and where there is negligible heat transfer from floor to floor. An alternative to using FLOOR-MULTIPLIER on a space is to use MULTIPLIER on the space's parent floor.

4. Interior walls cannot have multipliers.

Table 8 Building Components on which Multipliers Can be Used

Building Component	Keyword	Primary Effect	Secondary Effect	Warnings
FLOOR	MULTIPLIER	Multiplies total floor load.		
SPACE	MULTIPLIER	Multiplies total space load.	Multiplies heat transfer through standard and air-type interior walls.	Avoid when spaces differ in thermal behavior, external shading or daylighting. Spaces sharing an interior wall should not both have multipliers.
	FLOOR-MULTIPLIER	Multiplies total space load.	Does not multiply heat transfer through interior walls.	Avoid when spaces differ in thermal behavior, external shading or daylighting. Avoid when parent floor has a multiplier. Do not use if the space's parent floor has a multiplier.
EXTERIOR-WALL	MULTIPLIER	Multiplies conduction heat transfer.	Increased wall area changes room weighting factors.	Avoid when walls differ in thermal properties or external shading. Avoid in daylit space when wall contains windows.
UNDERGROUND-WALL	MULTIPLIER	Multiplies conduction heat transer.	Increased area gives higher thermal mass.	Avoid when walls differ in thermal properties.
WINDOW	MULTIPLIER	Multiplies conduction heat transfer, solar gain, and crack infiltration.	Multiplied window area is subtracted from area of parent wall.	Avoid for windows in spaces with daylighting (gives incorrect illuminance). Avoid for windows (exterior or interior) in sunspaces. Avoid when windows differ in external shading.
DOOR	MULTIPLIER	Multiplies conduction heat transfer and crack infiltration.	Multiplied door area is subtracted from area of parent wall.	Avoid when doors differ in thermal properties or external shading.

DESIGN-DAY

Design Day input allows the user to specify weather data that will be used in a Loads simulation to obtain peak heating and cooling loads for sizing the building heating and cooling equipment. DESIGN-DAY is a command that can be given a user selected name and uses the special TYPE keyword. The minimal (required keywords only) input for Los Angeles might look like:

```
LosAngelesCDD = DESIGN-DAY
   TYPE              = COOLING
   DRYBULB-HIGH      = 89
   DRYBULB-RANGE     = 20
   WETBULB-AT-HIGH   = 70  ..

LosAngelesHDD = DESIGN-DAY
   TYPE              = HEATING
   DRYBULB-HIGH      = 40  ..
```

The values in this input are available from *1993 ASHRAE Handbook Fundamentals*, Chapter 24, Tables 1, 2, and 3. The full set of keywords for DESIGN-DAY are:

Keyword	Description
TYPE = HEATING	Denotes heating design day.
TYPE = COOLING	Denotes cooling design day.
DRYBULB-HIGH	The outside drybulb design temperature (F or C).
DRYBULB-RANGE	The daily temperature range (F or C)
WETBULB-AT-HIGH	The outside wetbulb temperature coincident with the design drybulb (F or C)
HOUR-HIGH	The hour of the day of the highest Temperature (1-24).
HOUR-LOW	The hour of the day of the lowest temperature (1-24).
WIND-SPEED	The design wind speed (knots or m/s).
WIND-DIR = N or NNE or NE or NNW	The wind direction in compass points.
CLOUD-AMOUNT	The cloud amount in tenths of total sky Cover (0-10; 0=clear, 10=completely cloudy).
MONTH	The month of the design day (integer, 1-12).
DAY	The day of the month (integer, 1-31).
GROUND-T	The ground temperature (F or C) – this can usually be allowed to default

There can be one cooling and one heating design day. This data is used to generate one day (24 hours) of heating or cooling design weather. A one-day drybulb temperature cycle is generated from the DRYBULB-HIGH, DRYBULB-RANGE, HOUR-HIGH, and HOUR-LOW. A humidity ratio is calculated from DRYBULB-HIGH and WETBULB-AT-HIGH and this is held fixed for the day unless the drybulb falls below the dew-point. If no DRYBULB-RANGE is input, the drybulb is held constant for the entire day. Solar radiation values are generated from a clear sky model using the specified MONTH and DAY for calculating solar position. The clear sky values are then modified by the CLOUD-AMOUNT. The Loads portion of is then run with a 1 day run period for each design day. The resulting peak cooling load (for a cooling design day) and peak heating load (for a heating design day) is

used to size HVAC equipment. If no design days are input, the program sizes the HVAC equipment from the heating and cooling peak loads from the annual run.

Note that the default value for CLOUD-AMOUNT (10 for heating and 0 for cooling) minimizes the solar radiation for the heating design day and maximizes it for the cooling design day.

Design Internal Loads

For sizing the cooling equipment, the user might want peak cooling load calculated using the maximum internal load from lights, occupants, and equipment. Similarly for sizing heating equipment, the user might want these loads to be small or zero. It is possible to do this by assigning special day schedules to the heating or cooling design day. When the design day simulation is done, the program will use these schedules instead of the normal ones for that date and day of the week. The following example shows how this is done.

```
$ Design Schedules

Always-On = DAY-SCHEDULE
   TYPE               = FRACTION
                        (1,24) (1.0) ..

Always-Off = DAY-SCHEDULE
   TYPE               = FRACTION
                        (1,24) (0.0) ..

$ Occupancy Schedule

OC-1 = DAY-SCHEDULE
   TYPE               = FRACTION
                        ( 1, 8) (0.0)
                        ( 9,11) (1.0)
                        (12,14) (0.8,0.4,0.8)
                        (15,18) (1.0)
                        (19,21) (0.5,0.1,0.1)
                        (22,24) (0.0) ..

OC-2 = DAY-SCHEDULE
   TYPE               = FRACTION
                        (1,24) (0.0) ..

OC-WEEK = WEEK-SCHEDULE
   TYPE               = FRACTION
                         (WD) OC-1
                        (WEH) OC-2
                        (HDD) Always-Off
                        (CDD) Always-On ..

OCCUPY-1 = SCHEDULE
   TYPE               = FRACTION
   THRU DEC 31          OC-WEEK ..

$ Lighting Schedule
```

```
LT-1 = DAY-SCHEDULE
   TYPE               = FRACTION
                        ( 1, 8) (0.05)
                        ( 9,14) (0.9,0.95,1.0,0.95,0.8,0.9)
                        (15,18) (1.0)
                        (19,21) (0.6,0.2,0.2)
                        (22,24) (0.05) ..

LT-2 = DAY-SCHEDULE
   TYPE               = FRACTION
                        ( 1,24) (0.05)  ..

LT-WEEK = WEEK-SCHEDULE
   TYPE               = FRACTION
                        (MON,FRI) LT-1
                            (WEH) LT-2
                            (HDD) Always-Off
                            (CDD) Always-On ..

LIGHTS-1 = SCHEDULE
   TYPE               = FRACTION
   THRU DEC 31          LT-WEEK ..

$ Office Equipment Schedule

EQ-1 = DAY-SCHEDULE
   TYPE               = FRACTION
                        ( 1, 8) (0.02)
                        ( 9,14) (0.4,0.9,0.9,0.9,0.9,0.9)
                        (15,20) (0.8,0.7,0.5,0.5,0.3,0.3)
                        (21,24) (0.02) ..

EQ-2 = DAY-SCHEDULE
   TYPE               = FRACTION
                        (1,24) (0.2) ..

EQ-WEEK = WEEK-SCHEDULE
   TYPE               = FRACTION
                        (MON,FRI) EQ-1
                            (WEH) EQ-2
                            (HDD) Always-Off
                            (CDD) Always-On ..

EQUIP-1 = SCHEDULE
   TYPE               = FRACTION
   THRU DEC 31          EQ-WEEK ..
```

Essentially there are two new "days of the week" HDD and CDD which can be assigned special day schedules. In this case CDD is assigned day schedules that assign lighting and equipment to be fully on for 24 hours and for the building to be fully occupied for 24 hours. This then becomes the design condition for the design day simulation used to obtain the peak cooling load which is used to size the cooling system. Similarly HDD is assigned day schedules that have equipment and lights off and the building unoccupied. If CDD and HDD aren't explicitly assigned day schedules, their day schedules default to Monday day schedules.

Changes from DOE-2.1E

The keywords and data for the DESIGN-DAY command have changed somewhat from DOE-2.1E. An attempt was made to use data that was more readily available. And the use of special day schedules for the heating and cooling design days is a new feature. But the most striking change is that in DOE-2.1E each design day was associated with a run period, whereas now it is not. In DOE-2.1E the design day could be run for one day, a week, or whatever period was desired. Often a week was chosen, since it was felt that this would span the weekend and allow the peak to reflect the Monday morning pickup or pulldown load. In fact this was not the case. The design day simulation is a Loads only simulation. It is done at a fixed space temperature and knows nothing about the system, such as whether there is a night or weekend setback, or even whether the system is turned off at night or on weekends. The only result of running a design day for a week in DOE-2.1E was that it assured that the full range of scheduled internal loads was used. Now the capability of using special heating and cooling design day schedules obviates this need, and so the design day is run for one day only. The program has no automatic way of sizing HVAC equipment to take into account the pickup and pulldown loads. The only way to do this is to do multiple runs and size the equipment by hand.

SPACE WEIGHTING FACTORS

The program uses weighting factors in the Loads and HVAC programs. The Loads program transforms space heat gains into loads by applying *room weighting factors* (see "Loads Calculation Methodology). The HVAC program transforms loads into extraction rates by applying *air temperature weighting factors*, which are also used to determine the actual space air temperature. Basically, weighting factors account for the storage and release of heat in the thermal mass of a space. This thermal mass effect causes a delay between when a heat gain occurs and when it produces a convective heat transfer to the space air. Accurate calculation of space air temperature and of energy use and peaks depends on using correct weighting factors.

You can choose from two classes of weighting factors: *custom weighting factors* (CWFs) and *ASHRAE weighting factors* (AWFs). The program calculates CWFs based on your description of a space; because they are customized to the actual space being modeled, they give the most accurate results. On the other hand, AWFs are generic; they were precalculated for a space that has roughly the same heat capacity as the actual space but may differ from the actual space in terms of geometry and construction. Therefore, AWFs are less accurate than CWFs. Also, unlike CWFs, AWFs assume that all of the heat gain in a space eventually appears as a load. This is a poor assumption for highly conductive spaces (for example poorly insulated spaces or spaces with a lot of glazing), for which 10-30% or more of the heat gain is conducted back out of the space and so does not contribute to the space load. Because of this, AWFs overpredict both heating and cooling loads. Also, the AWF calculation assumes that all of the solar radiation entering a space stays in the space, but the CWF calculation accounts for solar that is reflected back out the windows.

For these reasons we recommend using CWFs (which is the default). AWFs should be used in only two cases:

- lWhen the details of the geometry and construction of space are poorly known (during early design, for example).

- When the CWF calculation fails.

Guidelines for Using Custom Weighting Factors

The program automatically calculates CWFs for a space when FLOOR-WEIGHT = 0 (the default) in the SPACE command. The program uses AWFs only if you input FLOOR-WEIGHT > 0. Following are some guidelines to get the most accurate calculation of CWFs:

1. Specify all walls (exterior, interior and underground) with *delayed* (i.e., response-factor type) constructions (see CONSTRUCTION command). If a wall has a quick construction, its thermal mass will *not* be accounted for in the CWF calculation.

2. Input *all of the bounding surfaces of a space*. This includes interior walls, floors, and ceilings across which there is negligible heat transfer. Such surfaces contribute to the radiation balance that is performed in calculating the CWFs. Also, interior surfaces, particularly concrete floors, often contribute substantially to the space's thermal mass.

3. The thermal mass of furnishings (furniture, bookcases, appliances, warehouse contents, etc.) can be accounted for by specifying FURNITURE-TYPE, FURN-FRACTION and FURN-WEIGHT in the SPACE command.

4. The CWFs for solar gain depend on the distribution of solar radiation in a space. For example, if most of the incoming solar is absorbed by a concrete floor the space will behave as though it has a high thermal mass in calculating the solar load. But if most of the solar is absorbed by light-weight walls, the space will behave as though it has a small thermal mass in calculating the solar load. Therefore, we

recommend that you pay attention to the SOLAR-FRACTION keyword for EXTERIOR-WALL, INTERIOR-WALL and UNDERGROUND-WALL. This keyword gives the fraction of incoming solar that is absorbed by the surface. If no SOLAR-FRACTIONs are specified, the program will distribute the solar so that 60% is absorbed by the floor and the remaining 40% is distributed to the other surfaces according to the product of their area and inside surface absorptance. This 60%/40% split is reasonable for most spaces, but for some configurations it may be unrealistic, in which case you should estimate better annual-average values based on the position and orientation of the space's windows.

5. Interior walls with TYPE = INTERNAL will contribute to the thermal mass of the space if they have delayed constructions; but no heat transfer will occur across such walls. You can enter a wall of this type to approximate the thermal mass of interior walls in the case where multiple rooms of similar temperature are aggregated into a single space.

6. Pay particular attention to quick interior walls where a U-value was chosen to be large to account for heat exchange caused by air movement through openings in the wall. For CWFs, an a*dditional interior wall* should be specified if an actual wall (as opposed to an imaginary partition) separates the spaces. The additional wall should be of delayed type of construction and its area should be exclusive of openings. The result will be the following. The original wall will affect the air temperature weighting factor calculation. The added wall will affect the detailed thermal balance calculation that produces the other weighting factors (i.e., the solar, lighting, people/equipment, and conduction weighting factors). Both walls will contribute to heat exchange (thermal conduction) in Loads.

ASHRAE Weighting Factors

The program will use AWFs if you input a FLOOR-WEIGHT that is greater than zero. To calculate the FLOOR-WEIGHT for a space, divide the weight of the materials in the space (walls, ceilings, floors and furnishings) by the floor area of the space to arrive at a lb/ft2 or kg/m2 figure. Only the weight of materials on the room side of the space's insulating layers should be counted. For example, for concrete block walls insulated on the outside, count the weight of the blocks; but if the insulation is on the inside, do not count the weight of the blocks. For interior walls, only half the weight of the wall should be included.

If the CWF calculation fails a warning will be printed and the program will revert to AWFs with FLOOR-WEIGHT = 30 lb/ft2 (146.4 kg/m2). If this floor weight is reasonable, no action is needed. If not, you should enter a better value.

FLOOR, SPACE, EXTERIOR-WALL, INTERIOR-WALL, WINDOW, AND DOOR GEOMETRY

This section describes how to specify the location, orientation and size of a building's floors, spaces, walls, windows and doors. It also describes how to locate the building itself and its exterior shading surfaces.

Coordinate Systems

The LOADS program uses a hierarchy of four right-handed Cartesian coordinate systems:

- the *building* coordinate system
- the *floor* coordinate system
- the *space* coordinate system
- the *wall* coordinate system

These systems are related to each other by translation and rotation. In addition, a reference coordinate system is used to locate the building and fixed shades when fixed shades are present (see FIXED-SHADE command).

A floor coordinate system is defined relative to the building coordinate system. A space coordinate system is defined relative to a floor coordinate system; a wall coordinate system is defined relative to a space coordinate system. Windows and doors are located with respect to a wall coordinate system.

There is only one building coordinate system, but there is a floor coordinate system for each floor, a space coordinate system for each space, and a wall coordinate system for each exterior wall, interior wall or underground wall. You don't have to keep track of the overall position of each wall and window/door relative to the building as a whole. Instead, the position of walls is specified relative to the space in which they occur, and the positions of windows and doors are specified relative to the wall on which they occur. The program then automatically transforms the walls and windows to the overall building coordinate system.

You define the building coordinate system in the SITE-PARAMETERS command by assigning values to the keywords AZIMUTH, LATITUDE, LONGITUDE and ALTITUDE. The other coordinate systems are located as follows:

1. A floor and its coordinate system are located in the building coordinate system by assigning values to X, Y, Z and AZIMUTH in the FLOOR command.

2. A space and its coordinate system are located in the floor coordinate system by assigning values to X, Y, Z and AZIMUTH in the SPACE instruction.

3. A wall and its coordinate system are located in the space coordinate system by giving values to X, Y, Z, AZIMUTH and TILT in the EXTERIOR-WALL, INTERIOR-WALL or UNDERGROUND-WALL command.

4. A window and its coordinate system is located in an exterior wall or interior wall coordinate system by assigning values to X and Y in the WINDOW command. Windows that are recessed from the surface of exterior walls can be further located by assigning a value to SETBACK in the WINDOW command. A door and its coordinate system are located in the exterior wall coordinate sytem by assigning values to X and Y in the DOOR command.

5. Fins and overhangs are located in the window or door coordinate system by assigning values to OVERHANG-A, etc. in the WINDOW or DOOR command.

6. Building shades are located in the building coordinate system by assigning values to X, Y, Z, AZIMUTH and TILT in the BUILDING-SHADE command.

Fixed shades are located in the reference coordinate system by assigning values to X-REF, Y-REF, AZIMUTH and TILT in the FIXED-SHADE command. When fixed shades are present, the building is located in the reference coordinate system by assigning values to X-REF and Y-REF in the BUILD-PARAMETERS command (see FIXED-SHADE command).

It is highly recommended that all building elements be properly positioned. If not, inaccuracies will result in calculating incident solar radiation, solar shading, and daylighting. Also, a 3-D display of the building will be meaningful only if the building elements are in the right place.

FLOORs and BUILDING-SHADEs are specified relative to the building coordinate system, which you define with the SITE-PARAMETERS command. The origin of the building coordinate system is fixed by the values of LATITUDE, LONGITUDE and ALTITUDE. The orientation of the building coordinate system is such that its z-axis is vertical and its x-y plane is horizontal. You rotate the coordinate system about the z-axis with the AZIMUTH keyword. The building AZIMUTH is defined as the clockwise angle between true north and the building y-axis. Thus AZIMUTH = 0 means that the y-axis points north, AZIMUTH = 90 means that the y-axis points east, and so forth.

Building Coordinate System

For a rectangular building we recommend that you orient the building coordinate system so that the x-axis runs along the front of the building, and the origin is located at the lower left-hand corner of the front wall, as viewed from the outside (see Figure 2). For more complicated building shapes, any convenient location of the origin and orientation of the AZIMUTH may be chosen.

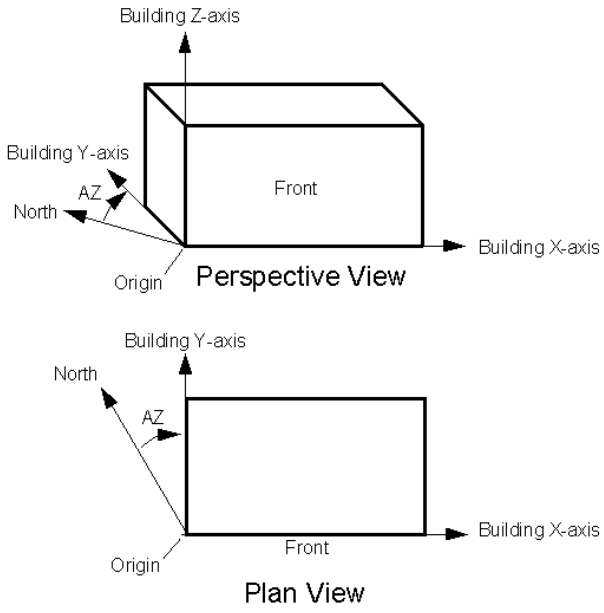


Figure 2 Example of a Building Coordinate System

Floor Coordinate System

The location of a floor and its associated coordinate system are defined relative to the building coordinate system by the keywords X, Y, Z, and AZIMUTH in the FLOOR command. The location of the origin of the floor coordinate system is specified by assigning values to X, Y, and Z.The orientation of the floor coordinate system is such that its z-axis is vertical and its x-y plane is horizontal. The floor and its coordinate system may be rotated about the z-axis by giving a value to the keyword AZIMUTH. The floor AZIMUTH is defined as the clockwise angle between the building y-axis and the floor y-axis. If X, Y, Z and AZIMUTH are not specified, they all default to zero, and the floor coordinate system becomes identical to the building coordinate system.

Space Coordinate System

The location of a space and its associated coordinate system are defined relative to the floor coordinate system by the keywords X, Y, Z, and AZIMUTH in the SPACE command (see Figure 3 and Figure 4). The location of the origin of the space coordinate system is specified by assigning values to X, Y, and Z. The orientation of the space coordinate system is such that its z-axis is vertical and its x-y plane is horizontal. The space and its coordinate system may be rotated about the z-axis by giving a value to the keyword AZIMUTH. The space AZIMUTH is defined as the clockwise angle between the building y-axis and the floor y-axis. If the space's X, Y, Z and AZIMUTH are not specified, they all default to zero, and the space coordinate system becomes identical to its parent floor's coordinate system.

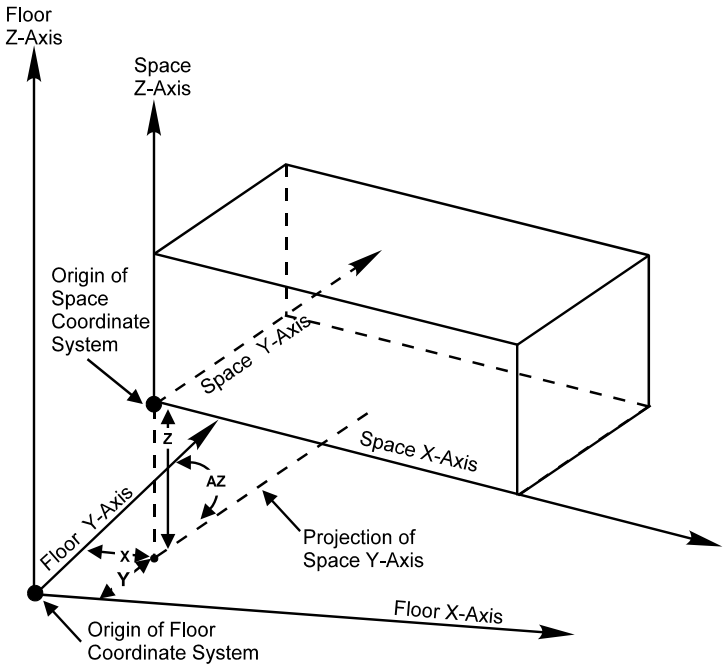


Figure 3 View of Space Located in the Floor Coordinate System

Wall Coordinate System

A wall and its coordinate system are located relative to the space coordinate system by the keywords X, Y, Z, AZIMUTH, and TILT in the EXTERIOR-WALL, INTERIOR-WALL or UNDERGROUND-WALL command.. You locate the origin of the wall coordinate system relative to the origin of the space coordinate system is by specifying X, Y, and Z. The origin of the wall coordinate system is the lower left-hand corner of the wall when you view the wall such that the wall's "outward normal" is pointing towards you. (The "outward normal" is given by the cross product of the wall's x and y axes. For an exterior wall it is best to have the outward normal pointing out of the building rather than into the building. For an interior wall it is best to have the outward normal pointing out of the space in which the wall is defined, i.e., pointing into the adjacent space.)

The orientation of the wall's coordinate system is given by the keywords AZIMUTH and TILT. The wall AZIMUTH is the clockwise angle between the y-axis of the space coordinate system and the horizontal projection of the wall's outward normal (see Figure 5). To specify AZIMUTH for a horizontal wall (a roof, ceiling or floor surface), see the discussion associated with Figure 9.

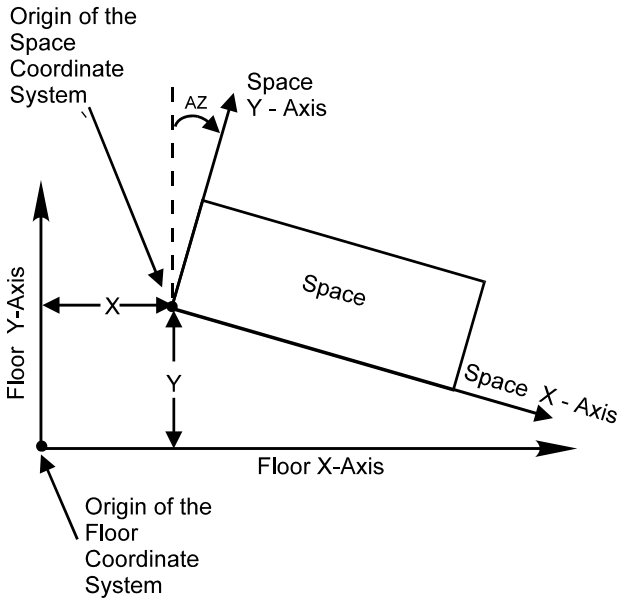


Figure 4 Plan view of space located in the floor coordinate system

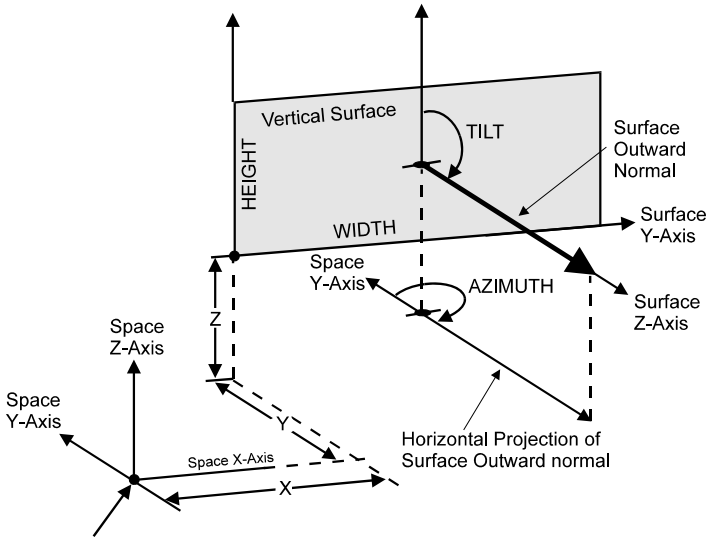


Figure 5 A wall located in a space coordinate system

For a rectangular wall described with keywords HEIGHT and WIDTH, the x-axis of the wall coordinate system runs along the width of the wall, the y-axis runs along the height of the wall, and the z-axis is parallel to the surface outward normal. The TILT is then the angle between vertical and the surface outward normal, and the AZIMUTH is the clockwise angle between the y-axis of the space coordinate system and the projection of the surface outward normal on the horizontal plane. For walls that are entered as polygons, see "Polygons" for a discussion of wall axes and wall orientation.

Shape = Box

There is an easy method for locating and orienting the walls of box-shaped spaces (rectangular parallelopipeds). You input SHAPE = BOX and specify HEIGHT, WIDTH and DEPTH in the SPACE command. Then, in the

EXTERIOR-WALL, INTERIOR-WALL or UNDERGROUND-WALL commands you specify LOCATION = LEFT, RIGHT, FRONT, BACK, TOP or BOTTOM to indicate which face of the box the wall is on (when the box is viewed along its y-axis, as shown in Figure 6). The program will then automatically assign the correct X, Y, Z, AZIMUTH and TILT for the wall.

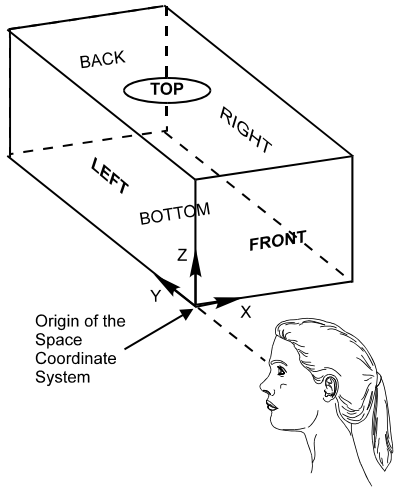


Figure 6 SHAPE = BOX Faces in the Space Coordinate System

For SHAPE = POLYGON spaces, walls can be easily located using the polygon vertex: LOCATION = SPACE-V1, LOCATION = SPACE-V2; LOCATION = TOP and BOTTOM can also be used with SHAPE = POLYGON

Locating Building Shades and Walls

Locating building shades and walls is similar to locating spaces, except that building shades are located in the building coordinate system and walls are located in the space coordinate system. In both cases, X, Y, Z, AZIMUTH, and TILT must be input. TILT is the angle between the vertical (the building, floor or space z-axis) and the surface outward normal. A vertical surface has TILT = 90. A horizontal surface with outward normal pointing straight up (a roof or ceiling, for example) has TILT = 0, and with outward normal pointing straight down (a floor, for example) has TILT = 180. A sloped roof might have TILT = 35 (see Figure 7).

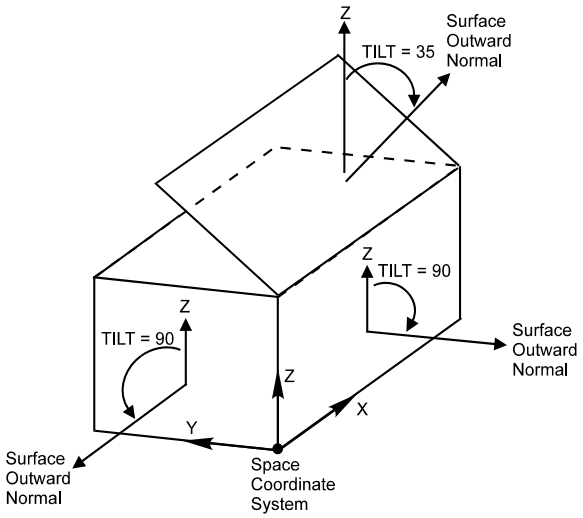


Figure 7 Examples of Tilt for Exterior Walls

For vertical surfaces, AZIMUTH is the clockwise angle between the y-axis (space y-axis for walls or building y-axis for shades) and the surface outward normal (see Figure 8). X, Y, and Z are the coordinates (relative to the space coordinate system for walls or relative to the building coordinate system for shades) of the lower left-hand corner of the surface as viewed from outside (i.e., in a direction opposite to the surface outward normal).

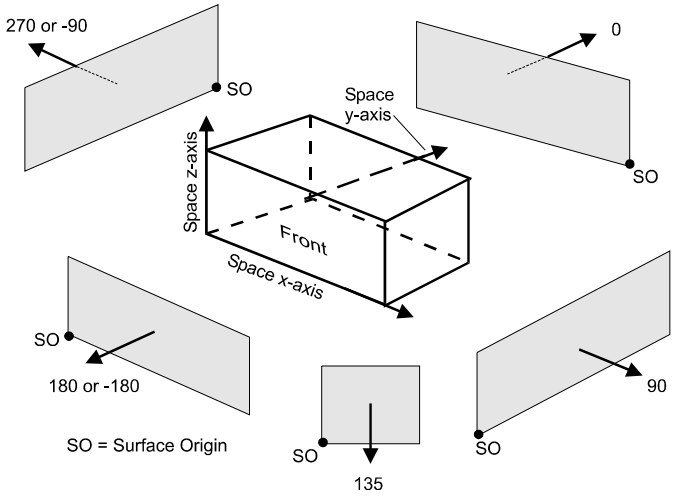


Figure 8 Examples of Wall Azimuth Values

Horizontal or non-vertical surfaces are trickier. The TILT is determined as before. Choose the origin of the surface and specify values for X, Y, and Z. Next, mentally rotate the surface to a vertical position such that the origin does not move (see Figure 9). Specify AZIMUTH, which is again the clockwise angle between the relevant y-axis and the outward normal of the surface in its new position. The values of HEIGHT and WIDTH are the height and width of the surface in its new, or vertical, position (see "Polygons" for walls entered as polygons). Of course, the program will not see the surface in the vertical position because the TILT, which was specified before mentally

rotating the surface to vertical, was already established at its true value. See "Building Shades" for examples of locating BUILDING-SHADEs and FIXED-SHADEs.

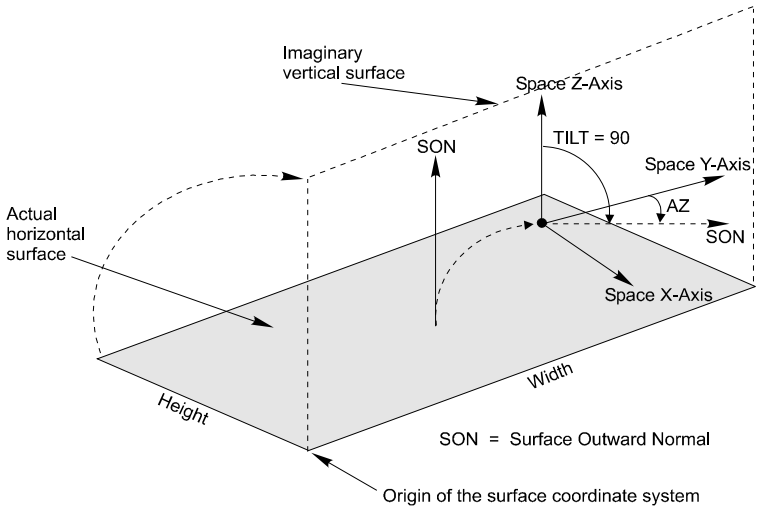


Figure 9 Method of describing a Horizontal Surface

Caution: The incorrect specification of AZIMUTH can result in overlapping walls, walls that are inside-out, outward protruding walls, bay windows instead of recessed windows, north facing instead of south facing windows, incorrect shading, and other problems. In complicated buildings these effects may not be readily noticeable in the simulation output. Incorrectly specifying X, Y and Z for any surface can give a totally incorrect calculation of exterior shading.

Locating Windows and Doors

To locate an opening (a window or door) in the coordinate system of its parent wall, you specify X and Y in the WINDOW or DOOR command. The outside surface of the opening is regarded as lying in the outside surface of the wall (unless SETBACK is specified, in which case the opening is recessed by that amount). Values for X and Y locate the lower left-hand corner of the opening, as viewed from outside, relative to the lower left-hand corner of the wall (see Figure 10).

Locating Fins and Overhangs

See "Building Shades" for examples of locating fins and overhangs relative to windows or doors.

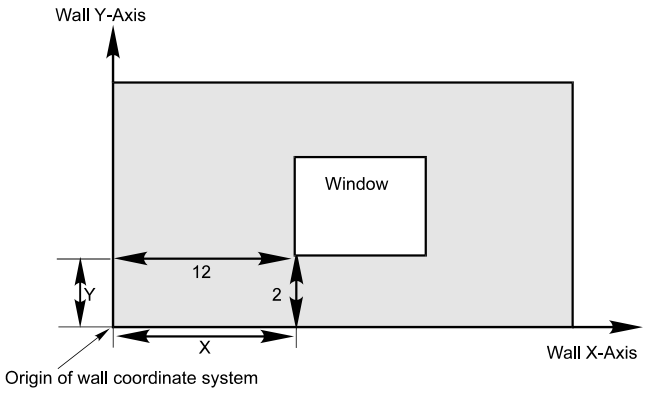


Figure 10 Locating a window on a wall

Polygons

Exterior walls, interior walls and underground walls can be described as polygons of from 3 to 40 sides by using the POLYGON command and the POLYGON keyword. In this case, X, Y, Z, AZIMUTH, and TILT can be used to position and orient the polygon in the space coordinate system. This is illustrated in the following example and in Figure 11.

Example - Polygonal Exterior Wall

```
TRIANG3 = POLYGON
   V1                = ( 0, 0)
   V2                = (20, 0)
   V3                = (10,20) ..

EW-1 = EXTERIOR-WALL
   POLYGON           = TRIANG3
   X                 = 8
   Y                 = 3.5
   Z                 = 6
   AZIMUTH           = 180
   TILT              = 90
   .....
```

Here:

- X,Y and Z are the coordinates of the origin of the polygon's local coordinate system in the space coordinate system.

- TILT is the angle between the z-axis of the space coordinate system and the polygon outward normal, which is a vector, perpendicular to the plane of the polygon, that points toward you as you face the polygon. (The polygon outward normal is also defined as the cross product between the vector from vertex 1 to vertex 2 and the vector from vertex 2 to vertex 3. It is also the cross product of the x and y axes of the polygon's local coordinate system.) A vertical polygon has TILT = 90. A horizontal, upward-facing polygon (e.g., a horizontal roof) has TILT=0. A horizontal, downward-facing polygon (e.g., an exposed floor) has TILT=180.

- AZIMUTH is the angle, measured clockwise, between the x-axis of the space coordinate system and the projection of the polygon outward normal onto the SPACE x-y plane.

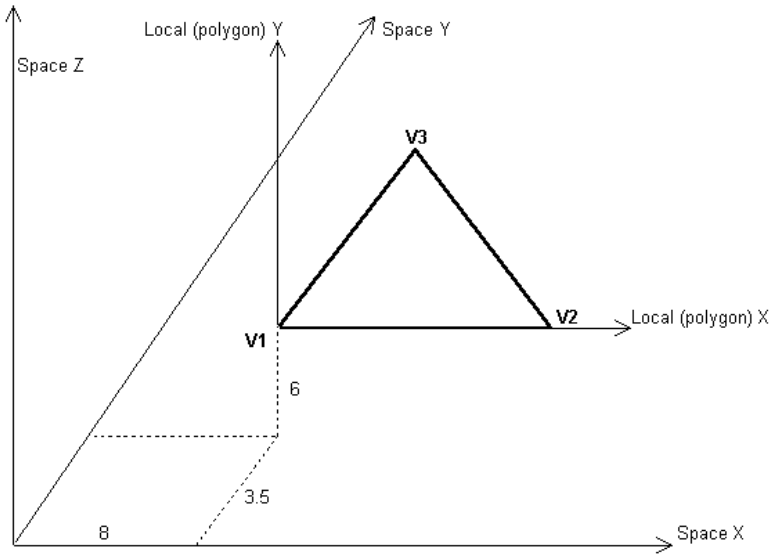


Figure 11 Triangular wall with TILT = 90 and AZIMUTH = 180, located at X = 8, Y =3.5 and Z = 6 in the space coordinate system

Figure 12 shows how to locate a window (or a door) on a polygonal wall. You position the window in the polygon's local coordinate system. In Figure 12a we have chosen the origin of the local coordinate system to be at the first vertex of the polygon (i.e., V1= (0,0)). The lower left hand corner of the window is at X=8, Y=4. Example input in this case is:

Example - Window on a Polygonal Wall

```
TRIANG4 =  POLYGON
   V1                = ( 0, 0)
   V2                = (24, 0)
   V3                = (12,16) ..

EW2 = EXTERIOR-WALL
   POLYGON = TRIANG4
   .....

WIN1 = WINDOW
   X                 = 8
   Y                 = 4
   HEIGHT            = 4
   WIDTH             = 7
   .....
```

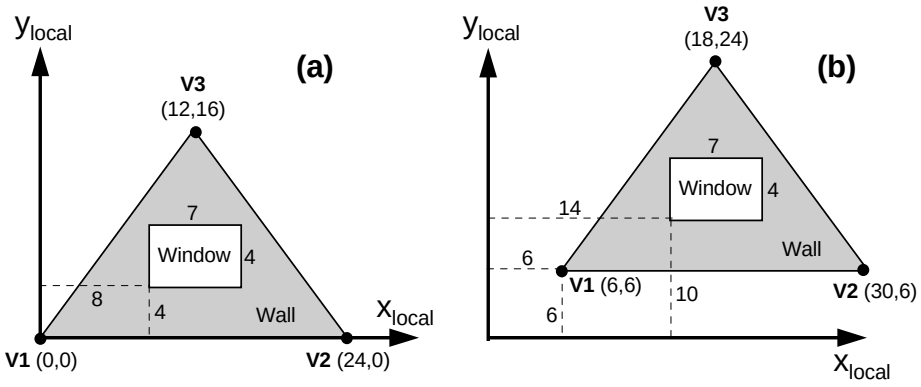


Figure 12 Locating a window in a polygonal wall. The X and Y values of the lower left corner of the window are given in the polygons local coordinate system. In (a), the first vertex of the polygon coincides with the origin of the local coordinate system. In (b), the first vertex of the polygon is displaced from the origin of the local coordinate system.

In Figure 12b, V1= (6,6), so that the polygon is displaced in its local coordinate system. The lower left hand corner of the window is at X=14, Y=10. Example input in this case is:

Example - Window in a Displaced Polygon

```
TRIANG4 = POLYGON
   V1                = ( 6, 6)
   V2                = (30, 6)
   V3                = (18,24) ..

EW2 = EXTERIOR-WALL
   POLYGON           = TRIANG4
   .....

WIN1 = WINDOW
   X                 = 14
   Y                 = 10
   HEIGHT            = 4
   WIDTH             = 7
   .....
```

To avoid confusion, we recommend that you always choose V1 = (0,0), as in Figure 12a. This means that when you locate the window it is with respect to this first polygon vertex.

BUILDING-SHADE, FIXED-SHADE, AND WINDOW SETBACK

Obstructions like trees and neighboring buildings can block solar radiation from reaching the exterior surfaces of a building. This reduces the heat gained by the building via conduction of solar radiation absorbed by walls and roofs and via transmission and absorption of solar radiation by windows.

Four types of obstruction can be modeled:

- *detached shades*, such as trees, that are separate from the building;
- *attached shades*, such as overhangs, that are connected to building;
- *setback shades*, which are associated with windows and doors that are recessed from an exterior wall surface; and
- *self shades*, which are walls of the building, such as in the wing of an L-shaped building, that shade other parts of the same building.

All of these shade types can block both direct radiation (from the sun itself) and diffuse radiation (from the sky and ground). The amount of blockage varies hourly and depends on three main factors:

- the sun position,
- the size and orientation of the shade,
- the transmittance of the shade, and
- the size and orientation of the surface being shaded.

To reduce calculation time, the program calculates hourly shading factors for *direct* solar radiation using hourly sun positions on the first day of each month and uses these factors for the rest of the month. The shading factors for *diffuse* sky and ground solar radiation are calculated once at the beginning of the run and used for the entire run period; for this calculation the radiance of the sky and ground is assumed to be uniform. The shading factors for diffuse radiation are also used in the calculation of the shading of long-wave (IR) radiation from sky and ground.

Detached Shades

Two types of detached shades can be modeled. Building Shades, entered with the BUILDING-SHADE command, move with the building when it is rotated or translated. Fixed Shades, entered with the FIXED-SHADE command, stay fixed with respect to the surface of the earth and do not move when the building is rotated or translated. An example of a Fixed Shade is a hill or an existing nearby building. An example of a Building Shade is a tree that will be located in front of the building no matter what its orientation turns out to be. You can use either Building Shade or Fixed Shade if the building orientation will not change, such as in retrofit analysis of an existing building or in new building analysis when you do not expect to vary the building's azimuth.

Building Shades

You use the BUILDING-SHADE command to specify the size, position, orientation and opacity of a Building Shade. An example is shown in Figure 13. Building Shades are assumed to be rectangular, with dimensions given by HEIGHT and WIDTH. The position of the shade relative to the building is given by X, Y and Z, which are the

coordinates (in the building coordinate system) of the lower left corner of the shade when the shade's outward normal is pointing towards you. The orientation of the shade is given by AZIMUTH and TILT. AZIMUTH is the angle between the Y-axis of the building coordinate system and the projection of the shade's outward normal onto the horizontal plane. TILT is the angle between the shade's outward normal and vertical.

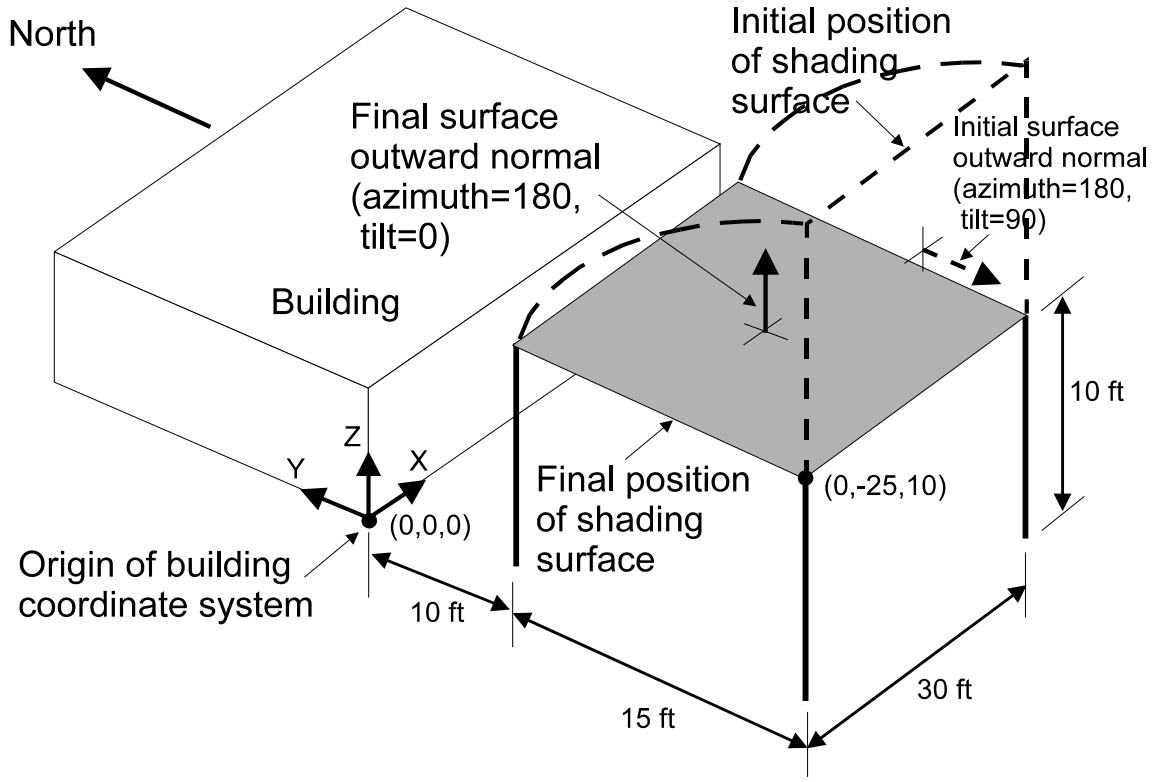


Figure 13 Example of a building shade

It is often difficult to envision the shade's azimuth, especially when the shade is horizontal, as in Figure 13. To help choose the azimuth you can do the following:

- Choose the origin of the shading surface.
- Mentally rotate the surface to a vertical position (without moving the lower left hand corner).
- Determine the azimuth as the angle between the building Y-axis and the shade's outward normal.
- Tilt the shade back to its desired position.

The input corresponding to Figure 13 is shown in the following example.

Example - Building Shades

A building is shaded by the horizontal roof of a carport. The shading roof is located parallel to and 10 feet south of the building with its west edge aligned with the building's west side. The shade is 30 by 15 (long dimension parallel to the building) and 10 feet above the ground.

```
CARPORT = BUILDING-SHADE
   HEIGHT            = 15
   WIDTH             = 30
   TRANSMITTANCE     = 0  $ the default $
   X                 = 0
   Y                 = -25
   Z                 = 10
   AZIMUTH           = 180
   TILT              = 0 ..
```

Building Shades that are not opaque, such as awnings and trees, can be modeled by assigning a non-zero solar transmittance value using the TRANSMITTANCE keyword (the default is 0). If SHADE-SCHEDULE in not entered (see below), the program uses the same transmittance for direct and diffuse solar radiation and assumes the transmittance does not depend on the angle of incidence of solar radiation on the shade.

You can use the BUILDING-SHADE keyword SHADE-SCHEDULE to model shade transmittance that varies with time. If SHADE-SCHEDULE is entered the program uses the SHADE-SCHEDULE value for direct solar radiation and the TRANSMITTANCE value for diffuse solar radiation. This means that the shade transmittance can vary with time for direct solar radiation but is constant for diffuse solar radiation. Here is an example SHADE-SCHEDULE for deciduous trees; the TRANSMITTANCE value in this case, which is used for the diffuse solar radiation, has been chosen to be the average of the summer and winter tree transmittance. If summer shading were more important, it could have be set to the summer tree transmittance.

Example - Shade Schedule for Deciduous Tree

A tree that shades a building has a solar transmittance of 0.9 in the winter and 0.1 in the summer. The tree is modeled as a 20x40 rectangle.

```
TREE-SCH-1 = SCHEDULE
   TYPE              = MULTIPLIER
   THRU MAY 15         (ALL) (1,24) 0.9  $no leaves$
   THRU NOV 1          (ALL) (1,24) 0.1  $with leaves$
   THRU DEC 31         (ALL) (1,24) 0.9  $no leaves$ ..

TREE-1 = BUILDING-SHADE
   HEIGHT            = 30
   WIDTH             = 20
   SHADE-SCHEDULE    = TREE-SCH-1          $ Direct solar
   TRANSMITTANCE     = 0.5                 $ annual diffuse
   X                 = 20
   Y                 = 40
   Z                 = 10
   AZIMUTH           = 180
   TILT              = 90 ..
```

Note that in the previous example the tree is approximated by a rectangle. For more precision, you can decompose non-rectangular Building Shades and Fixed Shades into two or more rectangles to account for the actual shape.

Shading surfaces block but do not reflect solar radiation. However, for the daylighting calculation you can specify that Building Shades and Fixed Shades reflect visible radiation by entering SHADE-VIS-REFL and SHADE-GND-REFL.

Fixed Shades

You use the FIXED-SHADE command to specify the size, position, orientation and opacity of a Fixed Shade. The input for Fixed Shade is the same as that for Building Shade except for X, Y and Z, which are replaced by X-REF, Y-REF and Z-REF, which are the coordinates of the lower left hand corner of the Fixed Shade in the Reference Coordinate System (RCS), as shown in Figure 14.

The RCS is a coordinate system that is fixed with respect to the surface of the earth. Its Y-axis points North and its X-axis points East. The FIXED-SHADE keyword, AZIMUTH, gives the angle between the Y-axis of the RCS and the outward normal of the fixed shade.

The RCS is used only when Fixed Shades are present. In this case it is also necessary to position the building in the RCS in order to obtain the proper geometrical relationship between the Fixed Shade and the building. This is done by assigning values to the X-REF and Y-REF keywords of the BUILD-PARAMETERS command. As shown in Figure 14, X-REF and Y-REF are the coordinates of the origin of the Building Coordinate System in the RCS. The BUILD-PARAMETERS keyword AZIMUTH is the angle between North and the Y-axis of the Building Coordinate System. Since the Y-axis of the RCS points North, AZIMUTH is also the angle between the Y-axis of the RCS and Y-axis of the Building Coordinate System.

Figure 14 also shows the building being translated and rotated (dashed lines) in the RCS by changing the building's X-REF, Y-REF, and AZIMUTH values. This moves the building but not the fixed shade.

TRANSMITTANCE and SHADE-SCHEDULE can be used for Fixed Shades; they have the same effect as for Building Shades.

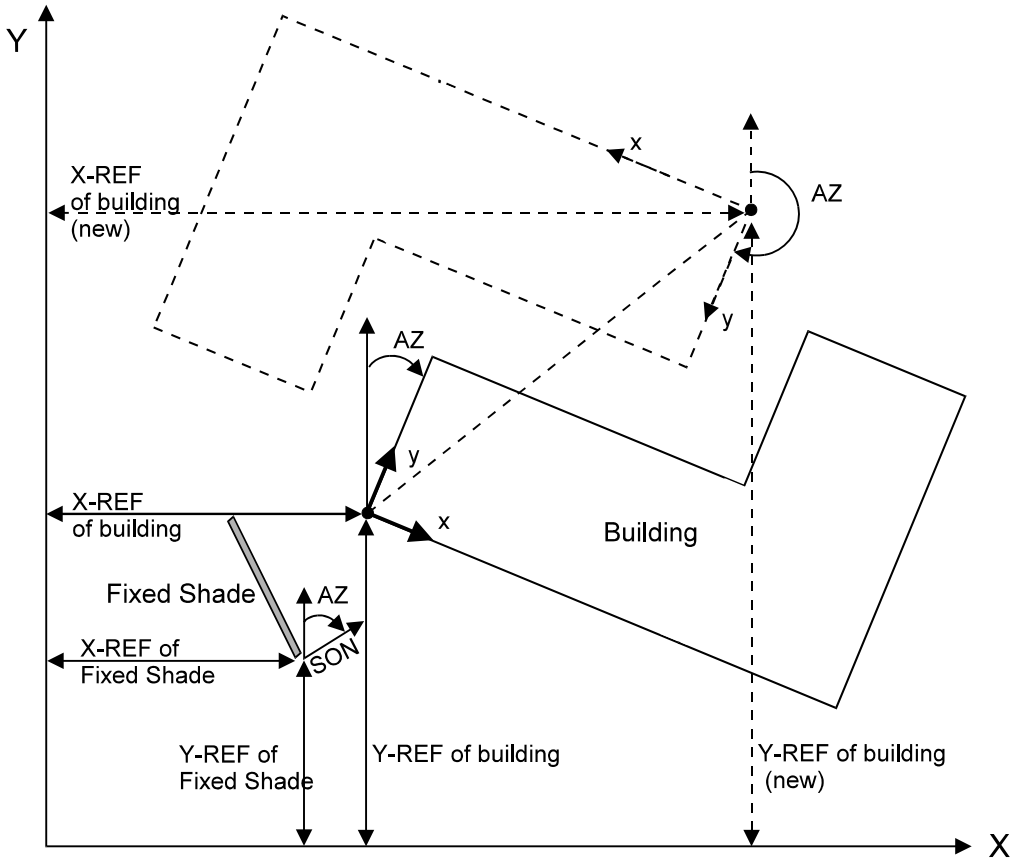


Figure 14 A fixed shade and building shown positioned in the Reference Coordinate System (RCS). When the building is moved (dashed line) the Fixed Shade remains stationary.

Attached Shades

Attached shades are overhangs and fins specified under the WINDOW and DOOR commands. They are positioned relative to a window or door and are attached to the wall in which the window or door is located. Overhangs and fins shade the window or door they are specified under as well as the wall itself and its other windows and doors. When a window or door is moved, its associated attached shades also move.

Figure 15 shows how attached shades are positioned. They are easier to position than detached shades since they are defined relative to a window or door rather than to the building as a whole. In this figure keyword abbreviations OH-A, etc. are used. Following is the correspondence between the abbreviations and full keyword names:

L-F-A: LEFT-FIN-A	R-F-A: RIGHT-FIN-A	OH-A: OVERHANG-A
L-F-B: LEFT-FIN-B	R-F-B: RIGHT-FIN-B	OH-B: OVERHANG-B
L-F-H: LEFT-FIN-H	R-F-H: RIGHT-FIN-H	OH-W: OVERHANG-W
L-F-D: LEFT-FIN-D		OH-D: OVERHANG-D
		OH-ANG: OVERHANG-ANGLE

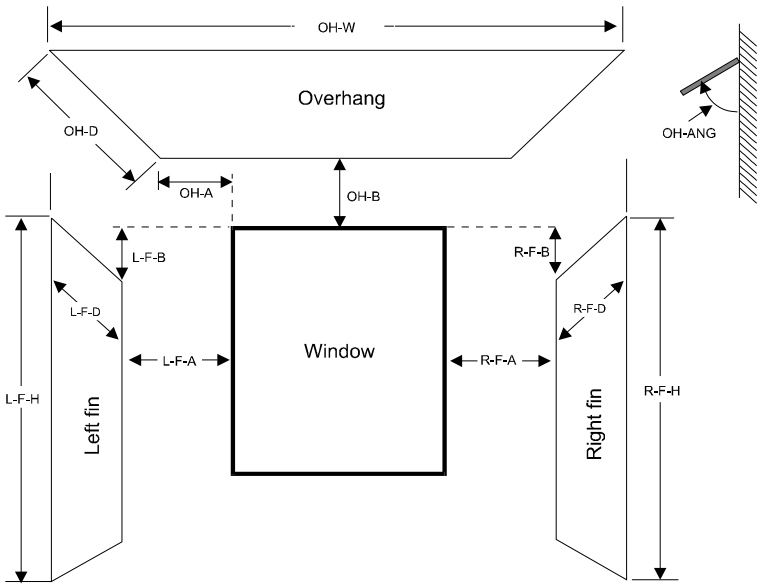


Figure 15 The positioning of attached shades (overhangs and fins) with respect to a window. The values in this figure are all positive and are measured with respect to the glazed part of the window (not with respect to the frame, if present). If the value for L-F-B is negative the left fin will originate at a point above the top edge of the window. Similarly for R-F-B.

Attached shades are assumed to be opaque. Non-opaque shades should be modeled as Building Shades or Fixed Shades. If the parent window or door of attached shades is referred to in another WINDOW or DOOR command with the LIKE keyword, the attached shades are copied.

Setback Shades

The program automatically creates setback shades when SETBACK is used with in the WINDOW or DOOR command to indicate that the window or door is recessed by an amount equal to the SETBACK value from the outside surface of an exterior wall. Three setback shades are created: one at the top of window and one on either side of window. The setback shades are assumed to be opaque.

Self Shades

Self shades correspond to exterior walls, such as those in an L-shaped building, that can shade other walls of the building. To get the shading effect from such walls you specify SHADING-SURFACE = YES in the EXTERIOR-WALL or ROOF command. The default for SHADING-SURFACE is NO, so that if not specified the self shading will not occur. To reduce shading calculation time you should only use SHADING-SURFACE = YES for walls that can actually shade other walls.

EXTERIOR-WALL AND ROOF

Entering a CONSTRUCTION instruction with TYPE = LAYERS (and a value for LAYERS) is done to specify a dynamic, or "delayed", type of construction. In this case, the calculation of heat transfer (by conduction and radiation) considers time and thermal mass. As such, computational time and cost can be greater than when specifying a CONSTRUCTION instruction with TYPE = U-VALUE and a value for U-VALUE but the results are more accurate.

The program treats exterior walls, exterior floors, and roofs in an identical manner. The only difference is the physical orientation of the surface. In the following discussion, all three surfaces will be referred to generically as "exterior wall".

Although there is only one instruction for specifying an exterior wall or roof (see "EXTERIOR-WALL or ROOF Command" in the *DOE-2.2 Dictionary*), there are many possible methods of specifying the physical makeup of an exterior wall (see Figure 16).

In order of increasing complexity, the common ways to specify an EXTERIOR-WALL are:

1. Use TYPE = U-VALUE and enter a value for U-VALUE in the CONSTRUCTION instruction (specify nothing for the LAYERS keyword). Then refer to the CONSTRUCTION instruction, by its U-name, in the CONSTRUCTION keyword of an EXTERIOR-WALL instruction. This will yield a steady-state, or "quick", exterior wall (see Figure 17).

2. Use TYPE = LAYERS and specify a code-word from the "Construction Library" in *DOE-2.2 DOE-2.2 Libraries & Reports*, for the LAYERS keyword in a CONSTRUCTION instruction (see Figure 18). This will yield a dynamic, or "delayed", exterior wall. The MATERIAL and LAYERS instructions are not used with this method. If you cannot find a suitable construction in the Library, use one of the following methods.

3. Specify a list of code-words from the "Materials Library" in *DOE-2.2 Libraries & Reports,* for the MATERIAL keyword in a LAYERS instruction (see Figure 18). Then, with TYPE = LAYERS, refer to the LAYERS instruction, by its U-name, in the LAYERS keyword of a CONSTRUCTION instruction. Finally, refer to the CONSTRUCTION instruction, by its U-name, in the CONSTRUCTION keyword of an EXTERIOR-WALL instruction. This will yield a dynamic, or "delayed", exterior wall.

4. Specify each individual material (by THICKNESS, CONDUCTIVITY, DENSITY, and SPECIFIC-HEAT for TYPE = PROPERTIES or by RESISTANCE for TYPE = RESISTANCE) in a MATERIAL instruction (see Figure 19). Then, using TYPE = LAYERS, refer to all the MATERIAL instructions, by a list of their U-names, in the MATERIAL keyword of a LAYERS instruction. Then, refer to the LAYERS instruction, by its U-name, in the LAYERS keyword of a CONSTRUCTION instruction. Finally, refer to the CONSTRUCTION instruction, by its U-name, in the CONSTRUCTION keyword of an EXTERIOR-WALL instruction. This will yield a dynamic, or "delayed", exterior wall. This method allows you to specify unusual constructions.

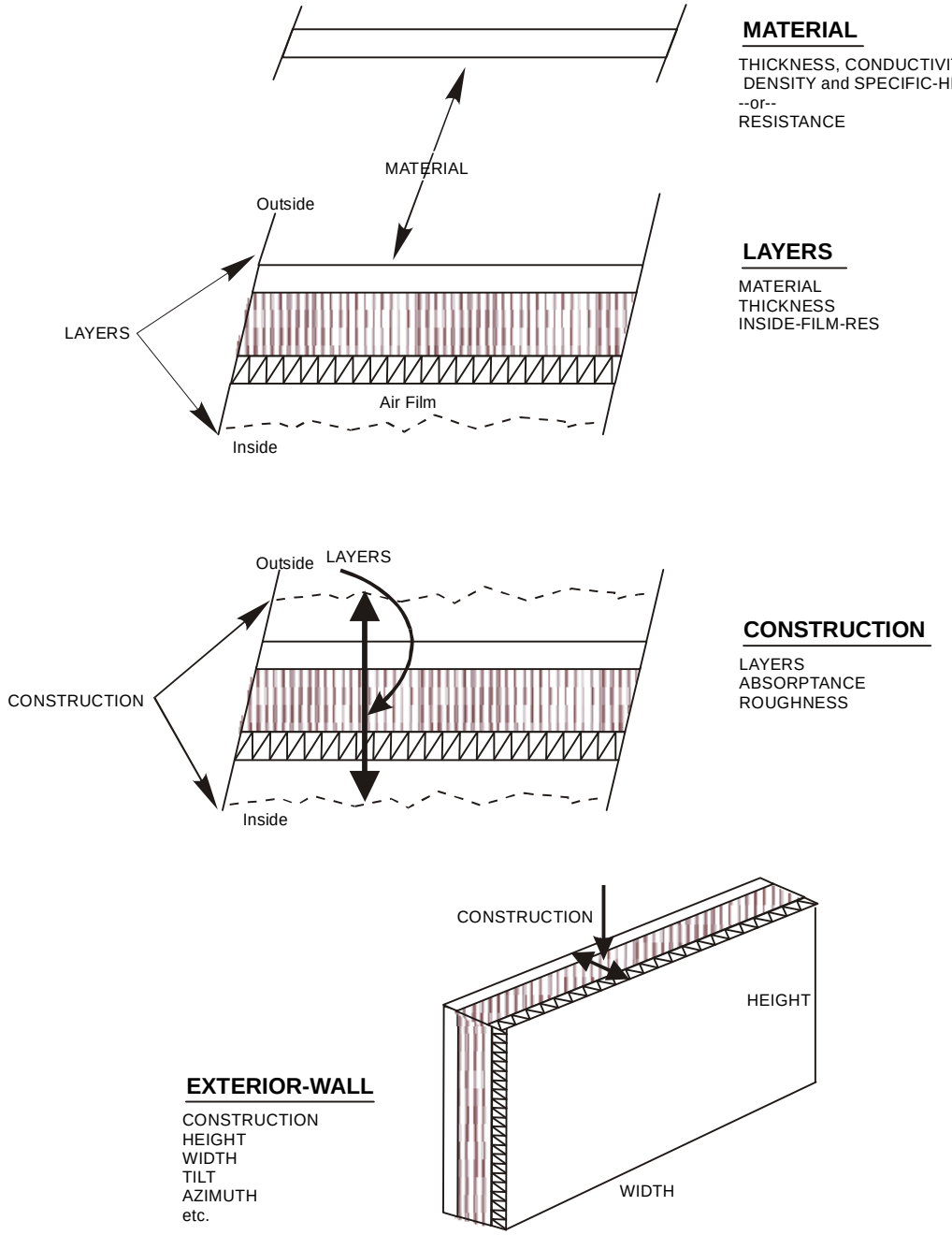


Figure 16 Physical makeup of a "delayed" EXTERIOR-WALL

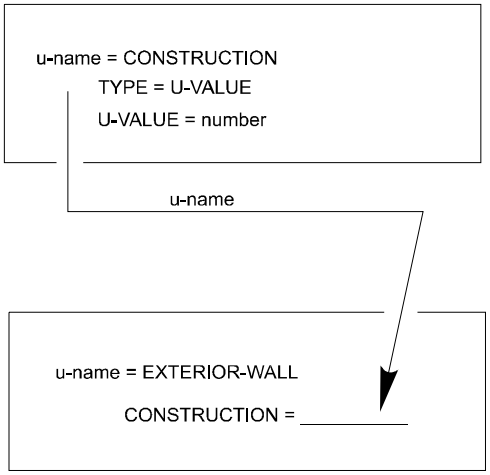


Figure 17 Specifying a "quick" EXTERIOR-WALL

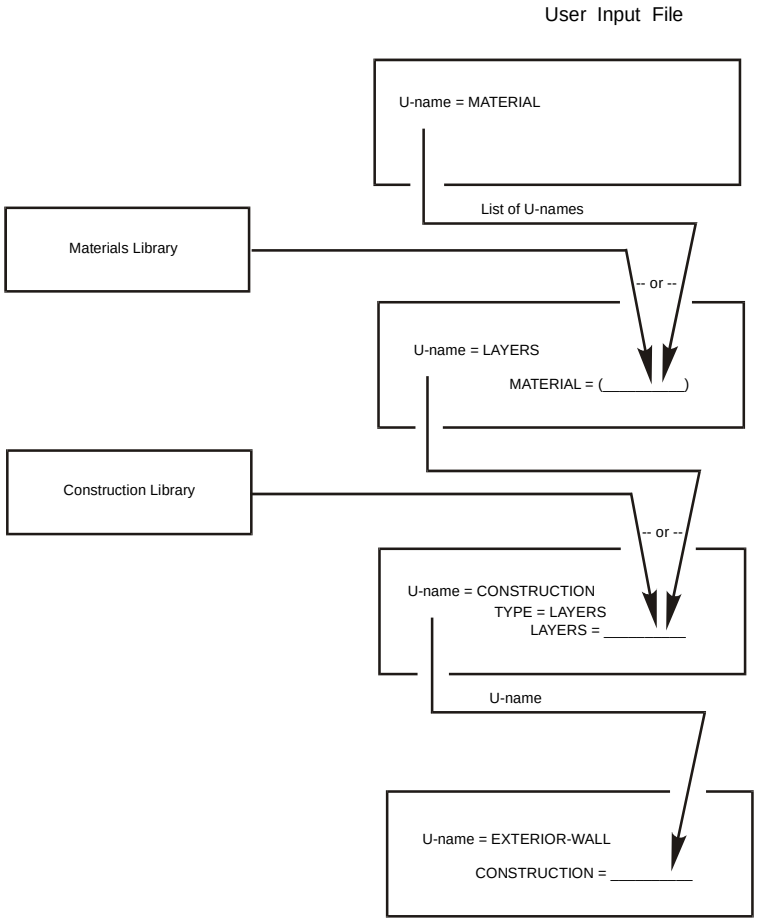


Figure 18 Specifying a "delayed" EXTERIOR-WALL (with a library)

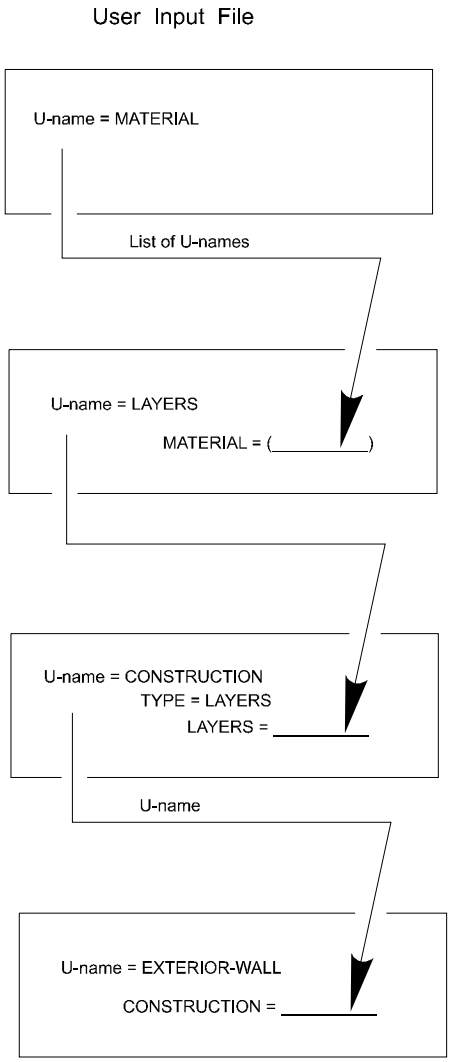


Figure 19 Specifying a "delayed" EXTERIOR-WALL (without a library)

UNDERGROUND-WALL AND UNDERGROUND-FLOOR

Underground surfaces in are walls or floors that are in contact with the ground. An example is a slab-on-grade or a basement wall. Underground surfaces are entered using the UNDERGROUND-WALL command, or the equivalent command, UNDERGROUND-FLOOR. Check the description of these commands in the *DOE-2.2 Dictionary* for information on the keywords for these surfaces.

Heat Transfer

Care needs to be taken in describing the construction of an underground surface in order to get a correct calculation of the heat transfer through the surface and a correct accounting for the thermal mass of the surface, which is important in the weighting factor calculation for the space. The LOADS module calculates the heat transfer through the underground surface as

Q = [PERIM-CONDUCT]*[PERIM-EXPOSED]*(Tg - Ti)

where

PERIM-CONDUCT	is the conductance of the surface per unit length of perimeter (Btu/hr-F-ft or W/m-K),
PERIM-EXPOSED	is the length of the perimeter portion of the surface exposed to outside air (ft or m),
Tg	is the ground temperature (F or C), and
Ti	is the LOADS calculation temperature for the parent space, given by the SPACE:TEMPERATURE keyword (F or C) .

PERIM-CONDUCT is used because the heat transfer occurs mainly through the surface's exposed perimeter region (since this region has relatively short heat flow paths to the outside air) rather than uniformly over the whole area of the surface.

The following procedure for defining an underground surface will properly account for its thermal mass and will give a much better calculation of heat transfer than the previously-used approach in which an effective U-value was defined using the U-EFFECTIVE keyword[3]. The procedure assumes that the monthly ground temperature is the average outside air temperature damped and delayed by three months, which is how the ground temperatures on the weather file are calculated. To force the program to use the weather file values, do not enter ground temperatures using the GROUND-T keyword in the SITE-PARAMETERS command.

Procedure for defining the underground surface construction

1. Choose a value of the perimeter conduction factor, PERIM-CONDUCT, from Table 9, Table 10 or Table 11 for the configuration that best matches the type of surface (slab floor, basement wall, crawl-space wall), foundation depth, amount/location of foundation insulation, and, for slab floors, whether the floor is bare or carpeted.

2. Determine PERIM-EXPOSED, the length (ft or m) of the surface's perimeter that is exposed to the outside air.Figure 21 and Figure 22 show values of PERIM-EXPOSED for example foundation

[3] In the custom weighting factor calculation the U-EFFECTIVE approach overestimated the fraction of space heat gains from solar, lights, people, etc. that is conducted back out of the space through the underground surface.

configurations. The heat transfer through an underground surface with no exposed perimeter, such as a basement floor or slab-on-grade in an interior zone, is assumed to be zero; in this case PERIM-EXPOSED should be zero. In a future version of the program this heat transfer, although small in many cases, will be calculated.

3. Define a construction, consisting of the following (Figure 20):

   - The underground wall or floor, including carpeting, if present, and inside film resistance
   - A 1-ft (0.3-m) layer of soil

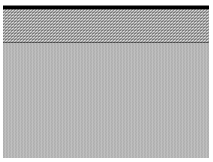


Figure 20 The layer of a soil represents the thermal mass of the ground in contact with the underground surface (a 1-ft [0.3-m] layer is sufficient to account for most of the thermal mass effect)

Example: 50' x 100' slab-on-grade.

The slab consists of uncarpeted, 4-in (10-cm) heavy-weight concrete. This is "Conc HW 140lb 4in (CC03)" in the library. The foundation depth is 4 ft (1.22 m) with R-10 (1.76 m2-K/W) exterior insulation. This gives PERIM-CONDUCT = 0.50 Btu/hr-F-ft (0.86 W/m-K) from Table 9

The slab's exposed perimeter is PERIM-EXPOSED = (2x50) + (2x100) = 300 ft.

We will choose INSIDE-FILM-RES = 0.77 hr-ft2-F/Btu (0.14 m2-K/W). This is the average of the air film resistance for heat flow up¾ 0.61 hr-ft2-F/Btu (0.11 m2-K/W)¾and heat flow down¾ 0.92 hr-ft2-F/Btu (0.16 m2-K/W). For vertical surfaces, such as basement walls, you can use INSIDE-FILM-RES = 0.68 hr-ft2-F/Btu (0.12 m2-K/W).

The layer of soil underneath the slab is "Soil 12in" from the library.

```
LAY-SLAB-1 = LAYERS
   MATERIAL           = ("Soil 12in",
                         "Conc HW 140lb 4in (CC03)")
   INSIDE-FILM-RES    = 0.77 ..

CON-SLAB-1 = CONSTRUCTION
   TYPE               = LAYERS
   LAYERS             = LAY-SLAB-1 ..
```

```
SLAB-1 = UNDERGROUND-FLOOR
   HEIGHT             = 50
   WIDTH              = 100
   TILT               = 180
   PERIM-EXPOSED      = 300
   PERIM-CONDUCT      = 0.50
   CONSTRUCTION       = CON-SLAB-1 ..
```

Notes

1. For basements (Table 10) and crawl spaces (Table 11) the section between ground level and the top of the underground wall is not included in the PERIM-CONDUCT value; therefore, this section should be entered as a separate exterior wall. Also, for shallow basements the wall section between the top of the underground wall and main level of the building should be entered as a separate exterior wall.

2. The floor of a crawl space (Table 11) should be entered as an UNDERGROUND-FLOOR consisting of a 1-ft (0.3-m) layer of soil. Because the exposed perimeter of the floor in this case is zero, PERIM-EXPOSED = 0. Because PERIM-CONDUCT is a required keyword, you need to enter a value for it; a value of zero will do. The input would look like:

```
LAY-CRAWL-1 = LAYERS
   MATERIAL           = ("Soil 12in")
   INSIDE-FILM-RES    = 0.77 ..

CON-CRAWL-1 = CONSTRUCTION
   LAYERS             = LAY-CRAWL-1 ..

CRAWL-1 = UNDERGROUND-FLOOR
   HEIGHT             = 50
   WIDTH              = 100
   TILT               = 180
   PERIM-EXPOSED      = 0
   PERIM-CONDUCT      = 0
   CONSTRUCTION       = CON-CRAWL-1 ..
```

Thermal Mass

Underground surfaces are usually concrete and, in combination with the adjacent soil, generally have high thermal mass. Because of its heat storage capacity this mass attenuates loads due to heat gains (from lights, solar, people, etc.) and causes a time delay between when the heat gain occurs and when it appears as a load on the HVAC system. In general, the higher the heat capacity and the more closely coupled the mass is to the room air, the larger this delay and attenuation will be.

The program will account for thermal mass only if (1) the underground surface is entered with a layers-type construction, following the procedure described in the previous section; and (2) custom weighting factors are calculated for the space, i.e., FLOOR-WEIGHT = 0 (the default) in the SPACE command.

Furniture

Underground floors, particularly slab on grade, are partially covered by furniture. You can get a better calculation of the custom weighting factors for the space containing an underground floor if you specify the type and coverage of the furniture using the SPACE keywords FURNITURE-TYPE, FURN-WEIGHT and FURN-FRACTION. Specifying furniture is particularly important for solar weighting factors because the presence of furniture reduces

the amount of incoming solar radiation that is absorbed by the floor (i.e., some of it is absorbed instead by the furniture and appears more quickly as a load).

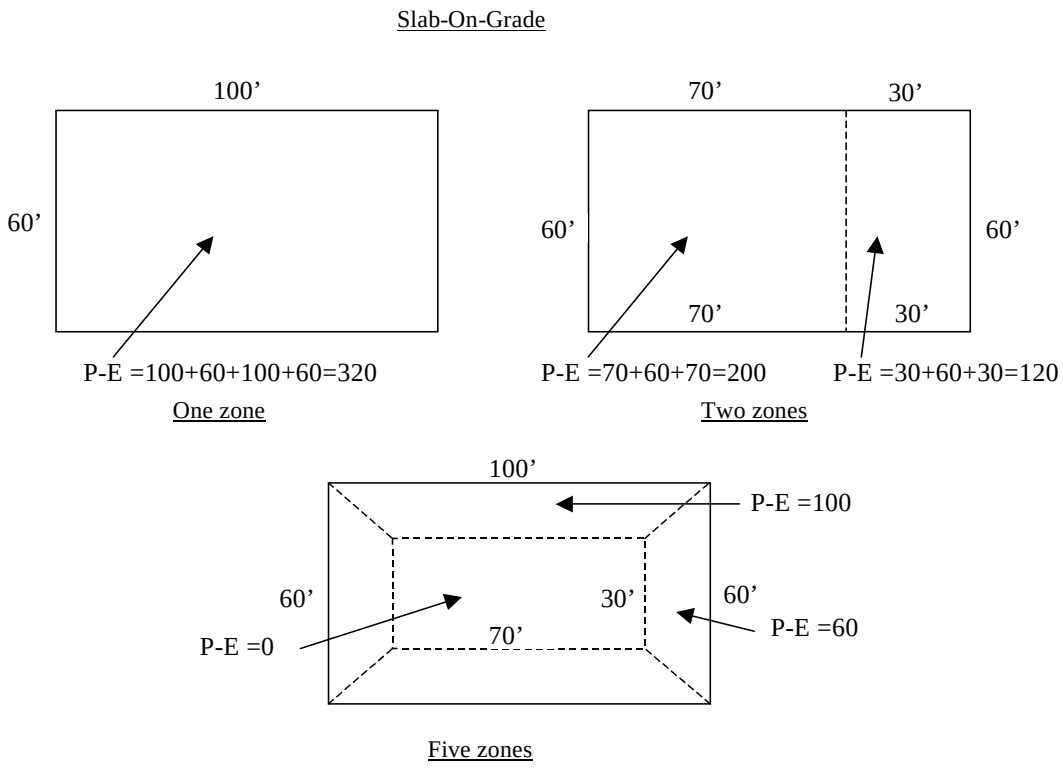


Figure 21 Exposed perimeter calculation for slab-on-grade examples. P-E corresponds to the PERIM-EXPOSED keyword.

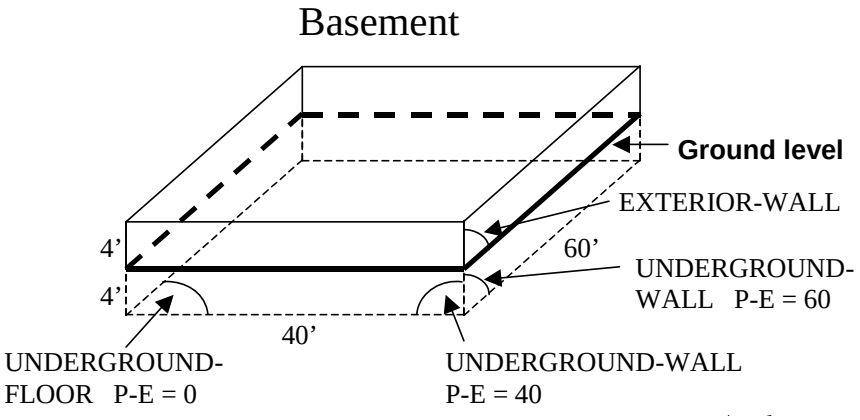


Figure 22 Exposed perimeter calculation for basement. P-E corresponds to the PERIM-EXPOSED keyword.

Table 9 Perimeter Conduction Factors for Slab-On-Grade[4]

[4] Source: Y.J.Huang, L.S.Shen, J.C.Bull and L.F.Goldberg, “Whole-House Simulation of Foundation Heat Flows Using the DOE-2.1C Program,” ASHRAE Trans. 94 (2), 1988, updated by Y.J.Huang, private communication

Slab-On-Grade			
Foundation Depth	Insulation Configuration (see Figure 23)	PERIM-CONDUCT Btu/hr-F-ft (W/m-K)	
		Uncarpetted	Carpetted
2 ft	Uninsulated	1.10 (1.90)	0.77 (1.33)
	R-5 exterior	0.73 (1.26)	0.54 (0.93)
	R-10 exterior	0.65 (1.12)	0.49 (0.85)
	R-5 interior; R-5 gap	0.75 (1.30)	0.57 (0.98)
	R-10 interior	0.89 (1.54)	0.46 (0.79)
	R-10 interior; R-5 gap	0.70 (1.21)	0.53 (0.92)
	R-10 interior; R-10 gap	0.68 (1.17)	0.52 (0.90)
	R-5 2-ft perimeter; R-5 gap	0.78 (1.35)	0.60 (1.04)
	R-10 2-ft perimeter; R-5 gap	0.73 (1.26)	0.57 (0.98)
	R-10 4-ft perimeter	0.79 (1.36)	0.59 (1.02)
	R-10 15-ft perimeter, R-5 gap	0.39 (0.67)	0.34 (0.59)
	R-5 16-in exterior, R-5 2-ft horizontal	0.65 (1.12)	0.48 (0.83)
	R-5 16-in exterior, R-5 4-ft horizontal	0.58 (1.00)	0.43 (0.74)
	R-10 16-in exterior, R-5 2-ft horizontal	0.56 (0.97)	0.41 (0.71)
	R-10 16-in exterior, R-5 4-ft horizontal	0.47 (0.81)	0.35 (0.60)
4 ft	Uninsulated	1.10 (1.90)	0.77 (1.33)
	R-5 exterior	0.61 (1.05)	0.46 (0.79)
	R-10 exterior	0.50 (0.86)	0.37 (0.64)
	R-15 exterior	0.44 (0.76)	0.33 (0.57)
	R-20 exterior	0.40 (0.69)	0.30 (0.52)
	R-5 interior; R-5 gap	0.63 (1.09)	0.48 (0.83)
	R-10 interior; R-5 gap	0.54 (0.93)	0.42 (0.73)
	R-15 interior; R-5 gap	0.50 (0.86)	0.38 (0.66)
	R-20 interior; R-5 gap	0.47 (0.81)	0.36 (0.62)
	R-5 4-ft perimeter; R-5 gap	0.68 (1.17)	0.54 (0.93)
	R-10 4-ft perimeter; R-5 gap	0.61 (1.05)	0.49 (0.85)
	R-10 4-ft perimeter	0.79 (1.36)	0.59 (1.02)
	R-10 15-ft perimeter, R-5 gap	0.39 (0.67)	0.34 (0.59)
	R-5 16-in exterior, R-5 2-ft horizontal	0.65 (1.12)	0.48 (0.83)
	R-5 16-in exterior, R-5 4-ft horizontal	0.58 (1.00)	0.43 (0.74)
	R-10 16-in exterior, R-5 2-ft horizontal	0.56 (0.97)	0.41 (0.71)
	R-10 16-in exterior, R-5 4-ft horizontal	0.47 (0.81)	0.35 (0.60)

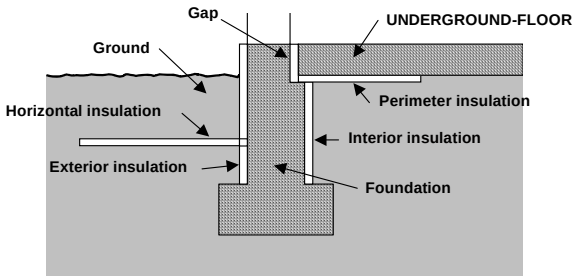


Figure 23 Slab-on-Grade

Table 10 Perimeter Conduction Factors for Basement Walls[5]

Basement Walls

[5] Source: Y.J.Huang, L.S.Shen, J.C.Bull and L.F.Goldberg, "Whole-House Simulation of Foundation Heat Flows Using the DOE-2.1C Program," *ASHRAE Trans.* 94 (2), 1988, updated by Y.J. Huang, private communication.

Underground Wall Height	Construction (see Figure 24)	PERIM-CONDUCT Btu/hr-F-ft (W/m-K)
8 ft (deep basement)	R-0 (uninsulated), concrete	1.94 (3.35)
	4-ft R-5 exterior, concrete	1.28 (2.21)
	8-ft R-5 exterior, concrete	0.99 (1.71)
	4-ft R-10 exterior, concrete	1.15 (1.99)
	8-ft R-10 exterior, concrete	0.75 (1.30)
	8-ft R-15 exterior, concrete	0.63 (1.09)
	8-ft R-20 exterior, concrete	0.56(0.97)
	8-ft R-10 interior, concrete	0.78 (1.35)
	R-0, wood frame	1.30 (2.25)
	R-11, wood frame	0.88 (1.52)
	R-19, wood frame	0.79 (1.37)
	R-30, wood frame	0.66 (1.14)
4 ft (shallow basement)	R-0 (uninsulated), concrete	1.61 (2.78)
	R-5 exterior, concrete	0.89 (1.54)
	R-10 exterior, concrete	0.73 (1.26)
	R-15 exterior, concrete	0.66 (1.14)
	R-20 exterior, concrete	0.95 (1.64)
	R-10 interior, concrete	0.79 (1.37)
	R-0, wood frame	1.10 (1.90)
	R-11, wood frame	0.80 (1.38)
	R-19, wood frame	0.74 (1.28)

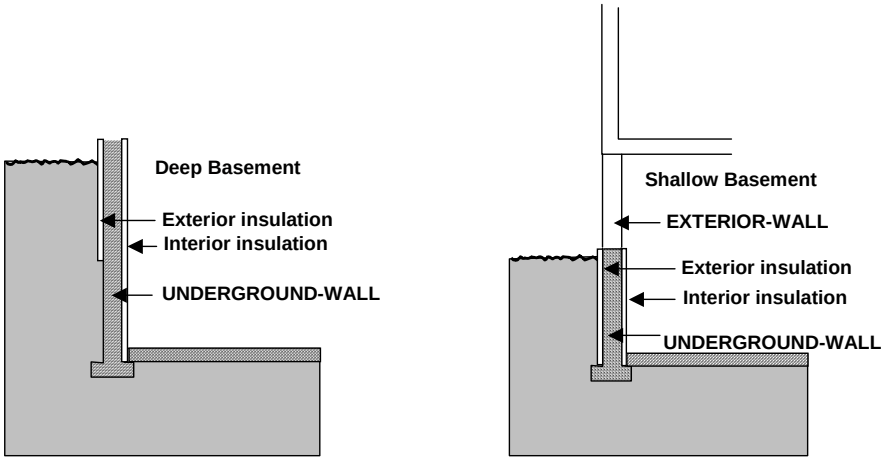


Figure 24 Basement with walls

Table 11 Perimeter Conduction Factors for Crawl Space Walls[6]

Crawl Space Walls		
Wall Height	Construction (see Figure 25)	PERIM-CONDUCT Btu/hr-F-ft (W/m-K)
2 ft	R-0 (uninsulated), concrete	1.29 (2.23)

[6] Source: Y.J.Huang, L.S.Shen, J.C.Bull and L.F.Goldberg, "Whole-House Simulation of Foundation Heat Flows Using the DOE-2.1C Program," *ASHRAE Trans.* 94 (2), 1988, updated by Y.J. Huang, private communication.

Crawl Space Walls		
Wall Height	**Construction** (see Figure 25)	**PERIM-CONDUCT Btu/hr-F-ft (W/m-K)**
	R-5 exterior, concrete	0.93 (1.61)
	R-10 exterior, concrete	0.87 (1.95)
	R-5 interior, concrete	0.97 (1.50)
	R-10 interior, concrete	0.91 (1.57)
	R-5 interior; R-5 4-ft perimeter, concrete	0.73 (1.26)
	R-10 interior; R-10 4-ft perimeter, concrete	0.68 (1.18)
	R-0, wood frame	1.00 (1.73)
	R-11, wood frame	0.88 (1.52)
	R-19, wood frame	0.86 (1.49)
4 ft	R-0 (uninsulated), concrete	1.28 (2.21)
	R-5 exterior, concrete	0.71 (1.23)
	R-10 exterior, concrete	0.59 (1.02)
	R-15 exterior, concrete	0.54 (0.93)
	R-20 exterior, concrete	0.50 (0.86)
	R-5 interior; R-5 4-ft perimeter, concrete	0.64 (1.11)
	R-10 interior; R-10 4-ft perimeter, concrete	0.58 (1.00)
	R-0, wood frame	0.83 (1.44)
	R-11, wood frame	0.59 (1.02)
	R-19, wood frame	0.55 (0.95)

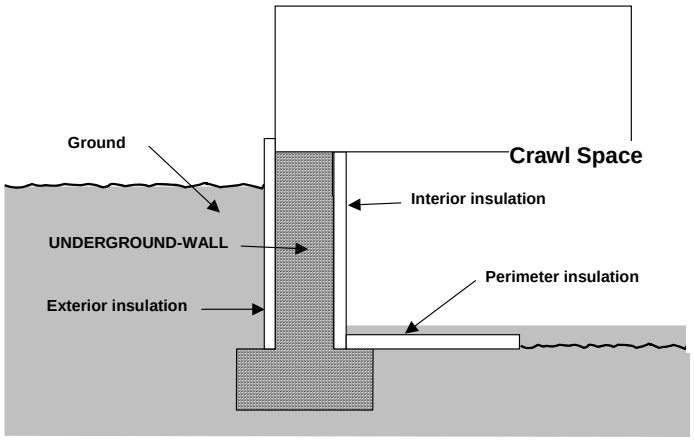


Figure 25 Basement with crawl space walls

INTERIOR-WALL

Interior surfaces in are walls, floors or ceilings that separate spaces or lie within a space. The load in a space is affected by interior surfaces in two ways: (1) conduction can occur across the surfaces from one space to another, and (2) the surfaces can have thermal mass and so can store and release heat.

Conduction

The program determines the heat transfer by conduction across an interior surface from its area and U-value and the temperature difference across the surface. This temperature difference is taken to be the previous-hour difference in the air temperatures of the spaces adjacent to the surface. This heat transfer can be significant for high U-value surfaces between a conditioned and unconditioned space, where the temperature difference is often large. An example is an interior ceiling between an attic or plenum and the conditioned room below it. Except for interior surfaces between a sunspace and a non-sunspace, the program assumes the conduction across the surface is quick (has no time delay).

Thermal Mass Effects

The heat storage capacity of an interior surface is calculated if you give it a delayed-type construction. In this way the time-delaying effect on space loads due to storage and release of heat in the thermal mass of the surface is accounted for. An example where this can be important is a concrete floor or wall between spaces.

Daylighting and Electric Lighting

Because interior surfaces have reflectance they effect the natural illuminance in daylit spaces. For the same reason they effect the electric lighting illuminance when electric lighting is specified with LTG-SPEC-METHOD = ILLUMINANCE.

Interior Surface Types

You can specify four types of interior surfaces using INT-WALL-TYPE:

1. Standard (INT-WALL-TYPE = STANDARD) this is the default. It designates an interior surface separating two spaces that is capable of conducting heat between the spaces.

2. Adiabatic (INT-WALL-TYPE = ADIABATIC) designates an interior surface that does not conduct heat between spaces but can store heat. This type should be used to separate spaces that are considered to be identical and are therefore defined with MULTIPLIER or FLOOR-MULTIPLIER in the SPACE command. An example is the wall, ceiling or floor that separates identical spaces that are side-by-side on one floor of a building or above one another in a multistory building. This type of wall should have a delayed construction.

3. Internal (INT-WALL-TYPE = INTERNAL) designates an interior surface that lies completely inside a space. An example is a water wall used to store solar energy in a space. Another example is a wall between two rooms that are modeled as a single space. Because both sides of such a wall can exchange heat radiatively and convectively with the space, it is recommended that you enter a single wall with area equal to twice the area of the actual wall. This type of wall should have a delayed construction.

4. Air (INT-WALL-TYPE = AIR) designates an interior surface with no mass across which convection can take place. It can be used to model openings between spaces. This type of wall should have a quick construction.

Windows in Interior Surfaces

An interior surface between a sunspace and a non-sunspace can have one or more windows that can conduct heat and through which solar radiation can pass. See "Sunspaces."

SPACE ELECTRIC LIGHTING

Electric lighting systems are important for three main reasons: (1) they consume electricity and are major contributors to overall electrical use in most buildings; (2) they produce heat, which can add to the cooling load or subtract from the heating load, and (3) they produce illumination. You can specify two types of electric lighting: overhead lighting and task lighting. Many spaces will have a combination of these two types.

Overhead Lighting

Overhead electric lighting in a space can be specified in three different ways

1. By giving the installed lighting power or installed lighting power density, and associated lighting schedule (LTG-SPEC-METHOD = POWER-DEFINITION in the SPACE command). In this case the lighting electrical consumption is calculated each hour by multiplying the installed lighting power by the lighting schedule value. For more details on this method, see the description of the LIGHTING-KW, LIGHTING-W/AREA and LIGHTING-TYPE keywords in the SPACE command.

2. By specifying lamp type, luminaire type and number of luminaires, and associated lighting schedule (LTG-SPEC-METHOD = LUMINAIRE-COUNT). In this case the program calculates the lighting power for the luminaire/lamp assembly and then multiplies this by the lighting schedule value. The library contains a variety of luminaires that you can choose from (see Table 1). The library also contains a range of lamp sizes and types (see Table 2). For more details on this method see the description of the LIGHTING-SYSTEM, LUMINAIRE-TYPE and LAMP-TYPE commands and the overhead lighting related keywords in the SPACE command.

3. By specifying lamp type, luminaire type and desired electric lighting illuminance when the lights are fully on (LTG-SPEC-METHOD = ILLUMINANCE). In this case the program first calculates the number of luminaires needed to meet the illuminance setpoint; then it determines the lighting power for each luminaire and multiplies this by the lighting schedule value. For more details on this method see the description of the LIGHTING-SYSTEM, LUMINAIRE-TYPE and LAMP-TYPE commands and the overhead lighting related keywords in the SPACE command.

For all three methods the program also determines the heat gain from the lights, which is the same as the electrical power. The load corresponding to this gain is determined using lighting weighting factors (see "Weighting Factors"). The lighting weighting factors depend on the split of radiative and convective heat from the lights. This split depends on the lighting type, with incandescent lamps having a somewhat higher radiative fraction and a somewhat lower convective fraction than other types. (However, little of the radiation produced by incandescent lamps is in the visible range, which makes them very inefficient as a source of illumination.)

Each space can have up to five lighting subsystems, each with its own lighting schedule.

The program also allows you to specify where the heat from lights goes, which can affect how the HVAC system "sees" the lighting load. You can specify that all of the heat from lights goes into the space, or that some of it goes into the return air and/or plenum (see description of the SPACE keywords LIGHT-TO-SPACE, LIGHT-TO-OTHER and LIGHT-TO-RETURN).

Space-level and building-level electricity consumption for overhead lighting is calculated and reported hourly, monthly and yearly. For attaching overhead lighting electricity consumption to a meter, see description of the LIGHT-ELEC-METER keyword in the SYSTEMS command.

Overhead Lighting Control

Lighting control is done via the lighting schedule (see LIGHTING-SCHEDULE keyword in the SPACE command). However, if a space has daylighting specified, the lights can be controlled by switching them on or off or by dimming them in response to the daylight illuminance level (see "Daylighting").

Task Lighting

Overhead lighting is usually uniformly distributed over space, whereas task lighting is concentrated where work takes place. A space space can have both overhead lighting and task lighting. Task lighting is specified with the SPACE keywords TASK-LIGHTING-KW, TASK-LT-W/AREA and TASK-LIGHT-SCH. Only one task lighting system is allowed per space. In calculating weighting factors and heat gain, the program assumes that task lighting is fluorescent and that all of the heat from task lighting goes into the space.

SPACE DAYLIGHTING

The daylighting calculation* allows you to determine what effect the use of daylighting to dim electric lighting has on energy use, peak loads, and energy cost. The calculation is done in the LOADS program; it has three main stages:

1. The program calculates daylight factors (interior daylight illuminance divided by exterior horizontal illuminance) for later use in the hourly calculation. You specify the coordinates of one or two reference points in a space. The program then integrates over the area of each window to obtain (a) the contribution of direct light from the window to the illuminance at the reference points, and (b) the contribution of light from sky and ground which enters the window and reflects from the walls, floor, and ceiling before reaching the reference points. Taken into account are such factors as window size and orientation, glass transmittance, inside surface reflectance of the space, sun-control devices such as blinds and overhangs, and the luminance distribution of the sky. Since this distribution depends on the position of the sun and cloudiness of the sky, the daylight factors are calculated for standard clear and overcast sky conditions for a series of 20 different solar altitude and azimuth values covering the annual range of sun positions. Analogous factors for discomfort glare are also calculated and stored.

2. An hourly daylight illuminance and glare calculation is performed. The illuminance contribution from each window is found by interpolating the stored daylight factors using the current-hour sun-position and cloud cover, then multiplying by the current-hour exterior horizontal illuminance obtained from measured horizontal solar radiation, if present on the weather file, or from a calculation. If the glare-control option has been specified, the program will automatically close window blinds or drapes in order to decrease glare below a pre-defined comfort level. (A similar option is available to use window shading devices to automatically control solar gain.) Adding the illuminance contributions from all the windows in a space then gives the total illuminance at each reference point.

3. Stepped and continuously dimming control systems are simulated to determine the electrical lighting energy needed to make up the difference, if any, between the daylighting level and the required illuminance. Finally, the zone lighting electrical requirements are passed to the thermal loads calculation.

Guidelines for Daylighting Modeling

Following are some guidelines for preparing input to model the effects of daylighting. Before studying these guidelines, however, you should read the description of each daylighting keyword in the *DOE-2.2 Dictionary*.

Table 12 Daylighting Commands and Keywords

Daylighting Commands	Daylighting Keywords	
in BUILDING-LOCATION	ATM-MOISTURE	ATM-TURBIDITY
in GLASS-TYPE	VIS-TRANS	
in BUILDING-SHADE and FIXED-SHADE	SHADE-VIS-REFL SHADE-GND-REFL	
in SPACE	DAYLIGHTING DAYLIGHT-REP-SCH LIGHT-CTRL-PROB LIGHT-CTRL-STEPS LIGHT-CTRL-TYPE1 LIGHT-CTRL-TYPE2 LIGHT-REF-POINT1 LIGHT-REF-POINT2	LIGHT-SET-POINT1 LIGHT-SET-POINT2 MAX-GLARE MIN-POWER-FRAC MIN-LIGHT-FRAC VIEW-AZIMUTH ZONE-FRACTION1 ZONE FRACTION2
in WINDOW	CONDUCT-TMIN-SCH[a] GLARE-CTRL-PROB MAX-SOLAR-SCH[a] OPEN-SHADE-SCH[a]	SUN-CTRL-PROB[a] VIS-TRANS-SCH WIN-SHADE-TYPE
in WINDOW DOOR ROOF EXTERIOR-WALL UNDERGROUND-WALL UNDERGROUND-FLOOR and INTERIOR-WALL	INSIDE-VIS-REFL	
[a] can also be used without daylighting		

As is the case when custom weighting factors are being used, all of the bounding surfaces of a space should be input, even INTERIOR WALLs across which negligible heat transfer takes place.

Use of Multipliers on Windows and Exterior Walls

If an exterior wall in a daylit space has a number of identical windows, the windows should be entered separately rather than using MULTIPLIER (which would give an incorrect illuminance calculation since individual windows would not be positioned correctly on the wall). Similarly, for a daylit space, MULTIPLIER should not be used on any exterior wall that contains a window.

Weather Files

We recommend that weather files with measured solar radiation (such as TMY2 files) be used for daylighting simulation because the solar information, coupled with a luminous efficacy calculation ((irradiance)*(luminous efficacy)=illuminance), gives a relatively good determination of the exterior daylight availability hour by hour as the sky conditions change. You can do daylighting simulation with non-solar weather files (such as TRY files), but the calculation will not be as accurate.

Thermal Zoning

To correctly calculate both direct and inter-reflected illuminance, one should try to model thermal zones consisting of several rooms separated by interior walls as a representative room with a multiplier. An example of this is shown inFigure 26. ROOM-1 is the representative room, with MULTIPLIER = 4. INTERIOR-WALLs IW-1 and IW-2 should have INT-WALL-TYPE = ADIABATIC. INTERIOR-WALL-IW-3 could be INT-WALL-TYPE = STANDARD or ADIABATIC. The floor and ceiling of ROOM 1 would probably be input as interior walls with INT-WALL-TYPE = ADIABATIC.

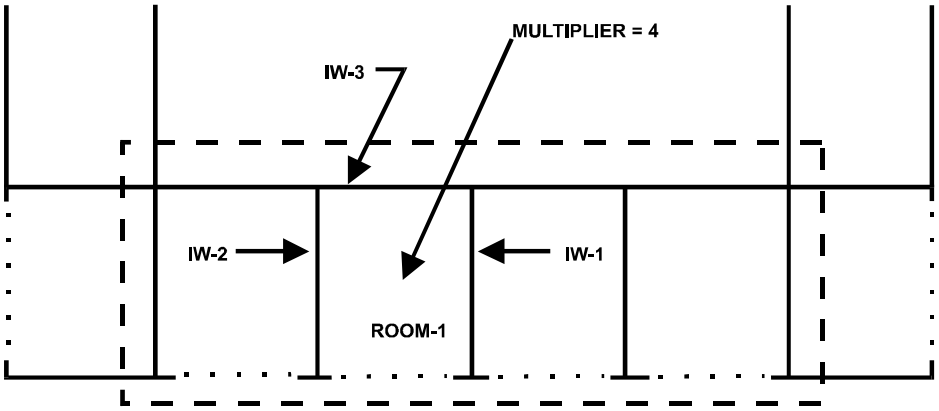


Figure 26 For daylighting purposes the thermal zone indicated by the dashed boundary line should be modeled as a typical room with a MULTIPLIER of 4

Sometimes a representative room cannot be found. Figure 27 shows a section of a building with four rooms having different daylight characteristics because of floor area, orientation, and window size. In this case, the analyst must choose between two alternatives: (1) simplify input by lumping the rooms into a single thermal zone, neglect partitions, and thereby get a possibly questionable daylighting result; or (2) describe each room as a separate thermal zone, input the partitions, and obtain an accurate daylighting calculation.

Surface Orientation

In the calculation of inter-reflected illuminance, the daylighting program uses surface tilt to distinguish between floors, walls, and ceilings. It is therefore important that the TILT values of all the bounding surfaces of a space be correctly specified. This applies not only to EXTERIOR-WALLs, but also to INTERIOR-WALLs, and UNDERGROUND-FLOORs and UNDERGROUND-WALLs.

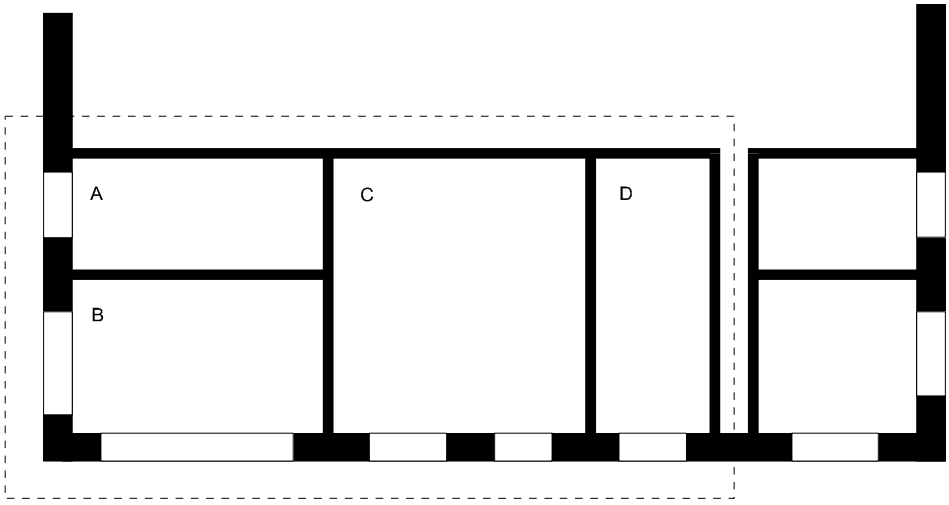


Figure 27 Rooms A, B, C, and D have different daylighting characteristics. If lumped into a single zone, input is simplified, but daylighting calculation will be inaccurate.

Multiple Lighting Zones

The daylighting program allows a thermal zone to be divided into two independently-controlled lighting zones. An example is shown in Figure 28(a), where a relatively deep thermal zone has two lighting zones of equal area.

It is also possible to daylight only part of a thermal zone. Figure 28(b) shows an example in which room A, with 40% of the zone's floor area, is daylit, whereas B, C, and D, having no windows, are not daylit. Note that a reference point and zone fraction are specified only for the daylit room.

Translucent Glazing

Skylights with diffusing glass, translucent fabric roofs, etc., can be modeled as clear glazing with a diffusing shade. For example, the input for a skylight with specularly-reflective, diffusely-transmitting glass, having a visible transmittance at normal incidence of 0.14 and a shading coefficient of 0.20, might be as follows:

```
GT-1 = GLASS-TYPE
   VIS-TRANS          = 1.0
   SHADING-COEF       = 1.0   ..

VT-MULT-1 = SCHEDULE  THRU DEC 31 (ALL) (1,24) (.14)   ..

SC-MULT-1 = SCHEDULE  THRU DEC 31 (ALL) (1,24) (.20)   ..

WINDOW
   GLASS-TYPE         = GT 1
   VIS-TRANS-SCH      = VT-MULT-1
   SHADING-SCHEDULE   = SC-MULT-1
   WIN-SHADE-TYPE     = FIXED-INTERIOR
   etc.  ..
```

If the outside surface of the glazing material reflects diffusely, rather than specularly, WIN-SHADE-TYPE = FIXED EXTERIOR is recommended.

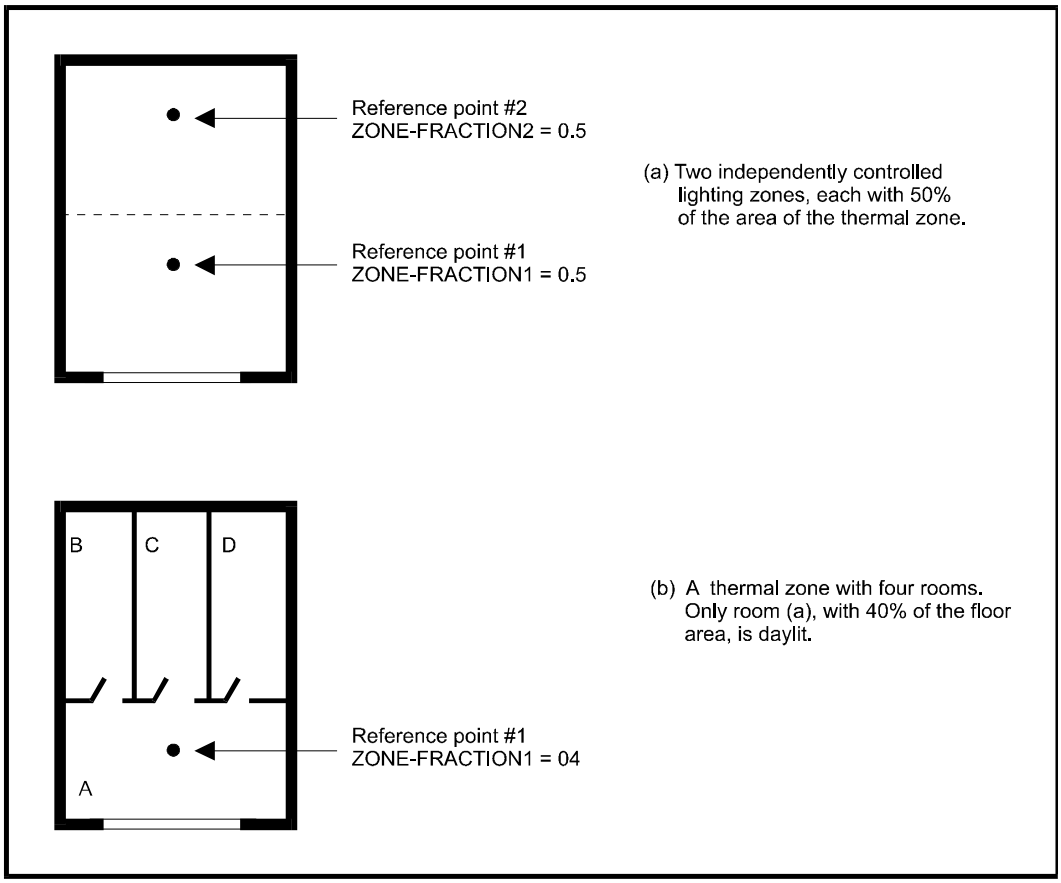


Figure 28 Examples of multiple lighting zones in a single thermal zone.

Fins, Overhangs, and Other Shading Surfaces

The daylighting program accounts for the presence of overhangs and other shading surfaces which affect the amount of solar radiation and visible light that strikes the windows . There are five categories of shading surfaces:

1. Shades defined by BUILDING-SHADE
2. Shades defined by FIXED-SHADE
3. Shades defined by EXTERIOR-WALLs with SHADING-SURFACE = YES (self shades)
4. Shades associated with window SETBACK
5. Overhangs and fins generated by the WINDOW keywords OVERHANG-A, LEFT-FIN-A, RIGHT-FIN-A, etc. (overhangs and fins)

Shade categories 1, 2, and 3 are called *global* shades, as they can affect all exterior surfaces, windows, and doors on a building. Shade categories 4 and 5 are called *local* shades, as they affect only the surface containing the shade.

For daylighting, the program assumes local shades are opaque and black, i.e., they neither transmit nor reflect incident light. A horizontal overhang , for example, is modeled as blocking part of the diffuse light from the sky and the direct light from the sun, and reflecting none of the light from the ground.

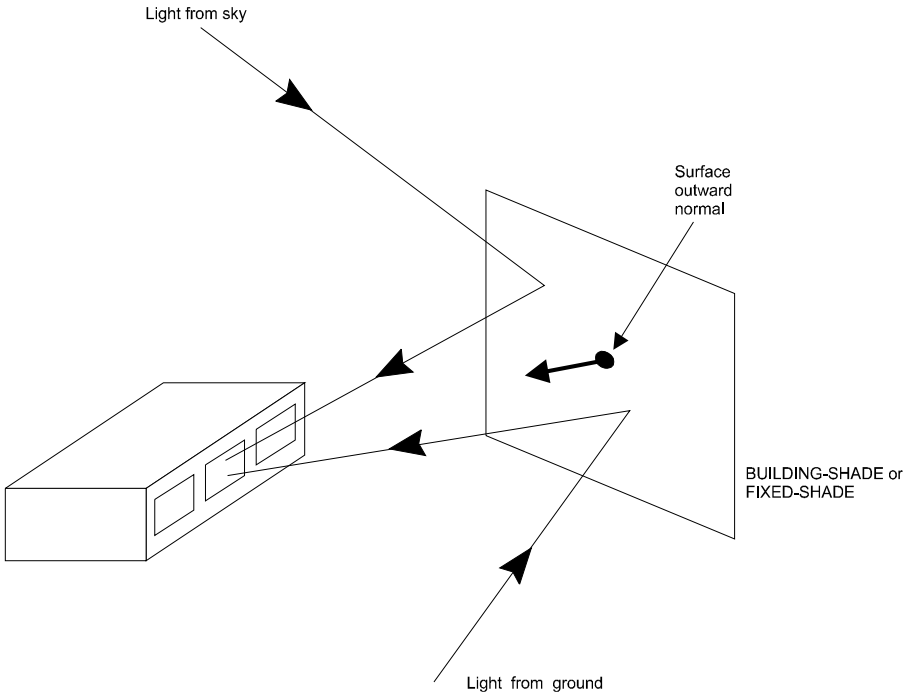


Figure 29 Shading surface oriented so that building sees luminous side of shade (the other side of the shade is assumed to be non-reflective)

Global shades are also assumed to be opaque, but, unlike local shades, they are assumed to have luminance due to the light from the sky and ground that they reflect. (However, the building itself is assumed to have no effect on this luminance. For this reason, light shelves cannot be accurately modeled.) Only one side of the shade is taken to be luminous. For BUILDING SHADE and FIXED SHADE, this is the side from which the surface outward normal points. For self shades , it is the outside of the wall. To receive reflected light from BUILDING SHADEs and FIXED SHADEs (which may represent neighboring buildings, trees, etc.,) the shade azimuth should be chosen so that the surface outward normal points toward the building, as shown in Figure 29. The visible reflectance of BUILDING SHADEs and FIXED SHADEs is given by the SHADE VIS REFL keyword and the ground reflectance by the SHADE GND REFL keyword. The visible reflectance of self shades is calculated from the absorptance of the exterior wall which generates the self shade.

In general, it is recommended that building projections be described as "fins and overhangs", category (5).

Skylights

Skylights with Light Wells

Skylights often have a rectangular light well (Figure 30) which is deep enough to cause substantial attenuation of the light which is transmitted into the room below. This attenuation can be approximately accounted for by multiplying Tvis, the visible transmittance of the skylight glazing material, by We, the light well efficiency factor [7] given in Figure 31. We is determined by the well wall reflectance and by the well index, which is related to the dimensions of the well.

[7] *IES Lighting Handbook*, 1981 Reference Volume, Illuminating Engineering Society of North America, p.9-84 ff.

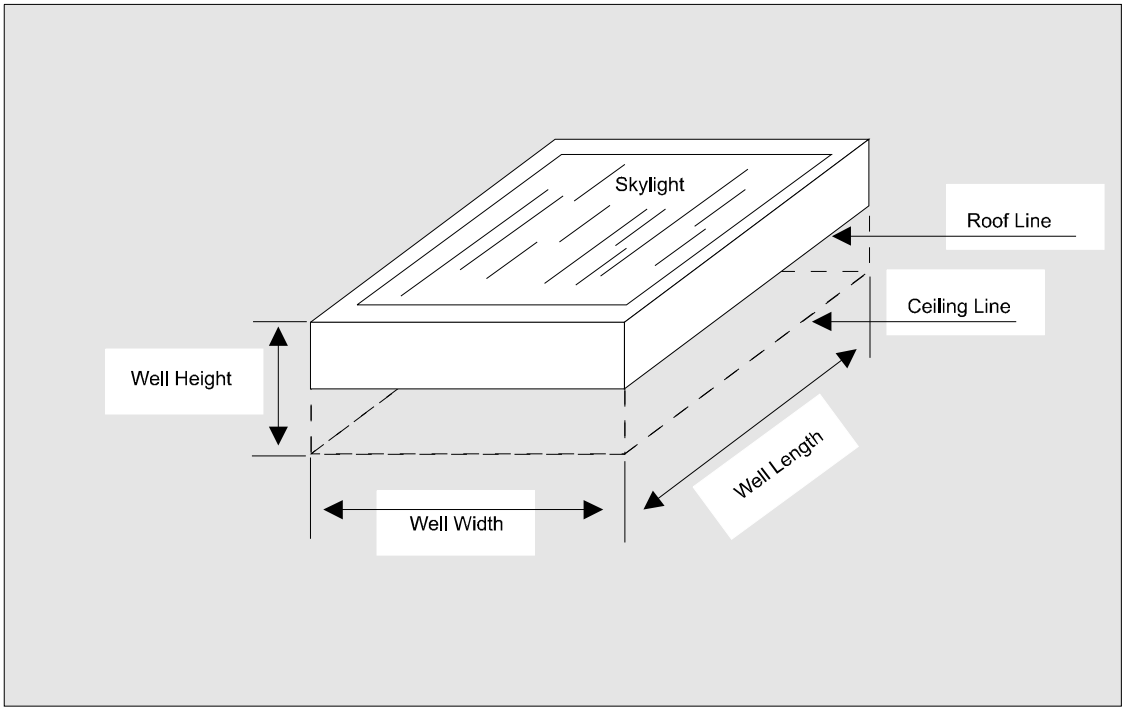


Figure 30 Skylight with light well

For example, if

well height = 3 ft
well width = 4 ft
well length = 6 ft,

then...

$$\text{well index} = \frac{(\text{well height}) \times (\text{well width} + \text{well length})}{2 \times (\text{well length}) \times (\text{well width})} = 0.63$$

If the well wall reflectance is 80%, Figure 31 gives We = 0.74. If Tvis is 90%, then the effective skylight transmittance that would be input is

VIS-TRANS = Tvis x We = 0.90 x 0.74 = 0.67

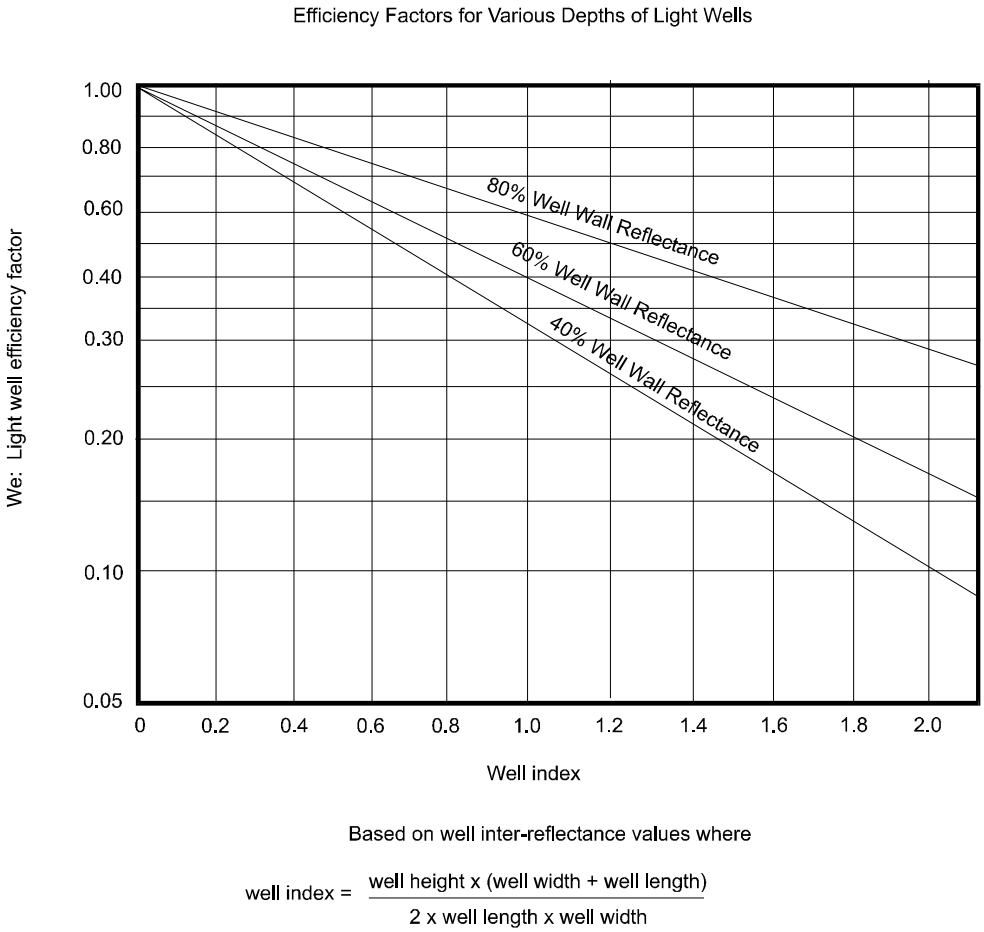


Figure 31 Light well efficiency factor vs. well index. Reprinted with permission from the *IES Lighting Handbook, 1981* Reference Volume, Fig. 9-75.

Domed Skylights

The visible transmittance of the acrylic material commonly used in domed skylights is generally given for the flat-sheet material before it is formed. The forming process produces a dome with a thickness that decreases towards the center. To account for the effect of this thickness variation and for the shape of the dome, the following equation [8] can be used to determine an effective transmittance:

$$Teff = 1.25\ Tvis\ (1.18 - 0.416\ Tvis)$$

where Tvis, the acrylic sheet's unformed visible transmittance at normal incidence, can be obtained from skylight manufacturer's data. (If the skylight has a light well, the above value of Teff should also be multiplied by the well efficiency factor, We, as described in the previous section.)

Example:

A skylight consists of a double dome. The outer dome is transparent with an unformed visible transmittance of 90%. The inner dome is translucent with an unformed transmittance of 40%. The effective transmittance of the outer dome is

[8] *IES Lighting Handbook*, 1981 Reference Volume, Illuminating Engineering Society of North America, p.9-84 ff.

1.25 x 0.9 (1.18 - 0.416 x 0.9) = 0.91

The effective transmittance of the inner dome is

1.25 x 0.4 (1.18 - 0.416 x 0.9) = 0.51

The effective transmittance of both layers (neglecting inter-reflection between the domes) is then 0.91 x 0.51 = 0.46. If the well efficiency factor is 0.67, as in the example in the previous section, the value entered for the net effective transmittance would be

VIS-TRANS = 0.46 x 0.67 = 0.31

Note that, since the inner dome in this example is translucent, a diffusing shade should be assigned with a visible transmittance of 1.0 (see Translucent Glazing).

Multiple Skylights

Rooms often have several individual skylights distributed over the ceiling. If you model this by entering one skylight and using MULTIPLIER, you will get a warning message saying the daylight calculation will be wrong. The problem is that the interior illuminance from a skylight depends on its position relative to the daylighting reference point. If you use a multiplier, all of the skylights pile up in one location rather than being spread out. This would generally greatly overestimate the daylight illuminance at the reference point. The solution is to enter the skylights individually, which would significantly increase the input effort as well as the calculation time, or to use the following equivalent skylight approach.

The idea is to separate the illuminance and thermal calculations in such a way to combine the multiple skylights into (1) a large skylight that gives nearly the same daylight illuminance as the separate skylights but without heat transfer, and (2) another large skylight that gives heat transfer but no daylighting. Here is an example with nine skylights, with sample values shown in square brackets. The original skylight configuration looks like:

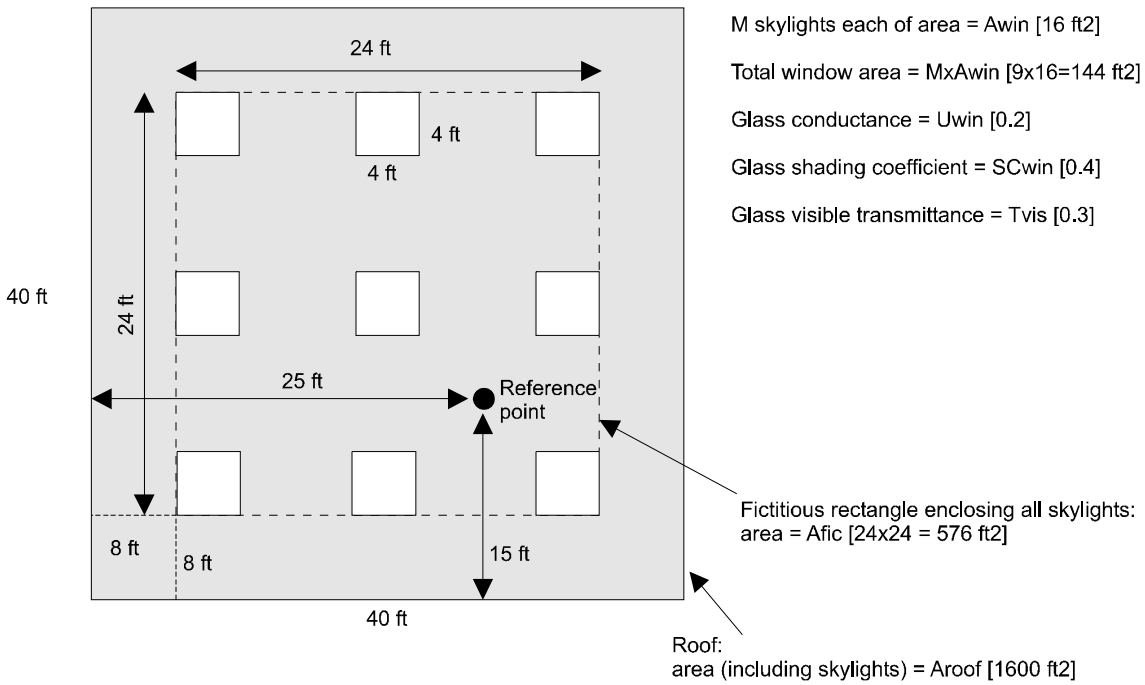


Figure 32 Original Configuration - Room with daylighting and heat transfer

Window Management

There are several ways of controlling the operation of window shading devices. The shading-coefficient and conductance of a window can be modified each hour to account for the presence of a shading device by specifying a SHADING-SCHEDULE and CONDUCT-SCHEDULE for the window. We call this preset schedule control. In addition, there are options to control shading devices when solar gain, outside temperature, or daylight glare exceed user-specified threshold values. These are called threshold controls .

Various control options and their input requirements are summarized in Table 13. In the table, note that there are additional input requirements for windows in spaces for which a daylighting calculation has been requested by specifying DAYLIGHTING = YES in SPACE.

Table 13 Window Shading Device Control Options

Windows in Non Daylit Spaces (DAYLIGHTING = NO)		
Control Type	**Input Required**	**Effect**
Preset schedule	SHADING-SCHEDULE[a]	Shading coefficient of glazingis multiplied hourly by SHADING-SCHEDULE value.
Solar gain control	MAX-SOLAR-SCH SHADING-SCHEDULE[a] (SUN-CTRL-PROB andOPEN-SHADE-SCH optional)[d]	Shade is fully closed if solar intensity on window exceeds MAX-SOLAR-SCH value.
Heat loss control with movable insulation	CONDUCT-TMIN SCH CONDUCT-SCHEDULE SHADING-SCHEDULE(OPEN-SHADE-SCH optional)[d]	Insulation is moved into place if outside drybulb temperature falls below CONDUCT-TMIN-SCH value.
Windows in Daylit Spaces (DAYLIGHTING = YES)		
Control Type	**Input Required**	**Effect**
Preset schedule	VIS-TRANS-SCH SHADING-SCHEDULE[a]	Glass visible transmittance and shading coefficient are multiplied hourly by VIS-TRANS-SCH and SHADING-SCHEDULE, respectively.
Solargaincontrol	MAX-SOLAR-SCH VIS-TRANS-SCH SHADING-SCHEDULE[a] WIN-SHADE-TYPE (SUN-CTRL-PROB andOPEN-SHADE-SCH optional)[d]	Shade is fully closed if solar intensity on window exceeds MAX-SOLAR-SCH value
Heat loss control with movable insulation	CONDUCT-TMIN-SCH CONDUCT-SCHEDULE VIS-TRANS-SCH[b] SHADING-SCHEDULE WIN-SHADE-TYPE [c] (OPEN-SHADE-SCH optional)[d]	Insulation is moved into place if outside drybulb temperature falls below CONDUCT-TMIN-SCH value
Glare control	MAX-GLARE (SPACE command) VIS-TRANS-SCH SHADING-SCHEDULE[a] WIN-SHADE-TYPE[c] (OPEN-SHADE-SCH optional)[d]	Shade is fully closed if daylight glare at either lighting reference point exceeds MAX-GLARE value

[a] Also input CONDUCT-SCHEDULE if shading device significantly affects window conductance.
[b] Since VIS-TRANS-SCH can be assigned, the insulation need not be opaque.
[c] Must be either MOVABLE-INTERIOR (the default) or MOVABLE-EXTERIOR.
[d] See the *DOE-2.2 Dictionary* for a description of OPEN-SHADE-SCH

Notes:

1. For threshold controls, the preset schedule values given by SHADING-SCHEDULE, CONDUCT-SCHEDULE, or VIS-TRANS-SCH are still used, but only when the shading device is closed , i.e., only when a threshold condition is exceeded. When the shading device is open, the schedule values are automatically replaced with a value of 1.0.

2. If two or more threshold controls are specified for the same window (e.g., if MAX-SOLAR-SCH and MAX-GLARE are both input), the shading device will be deployed if either threshold condition is met

3. The program cannot model windows with more than one operable shading device. However, windows with one fixed and one operable shading device can be handled by describing the operable shade with SHADING-SCHEDULE, VIS-TRANS-SCH, etc., and describing the properties of the window-plus-fixed-shade combination in the GLASS-TYPE keywords SHADING-COEF, GLASS-CONDUCTANCE, and VIS-TRANS.

4. CONDUCT-SCHEDULE will have no effect on a window unless a corresponding SHADING-SCHEDULE is also given.

5. WIN-SHADE-TYPE is required only for windows in daylit spaces.

This is modeled by entering the two roof sections shown in Figure 33 and Figure 34:

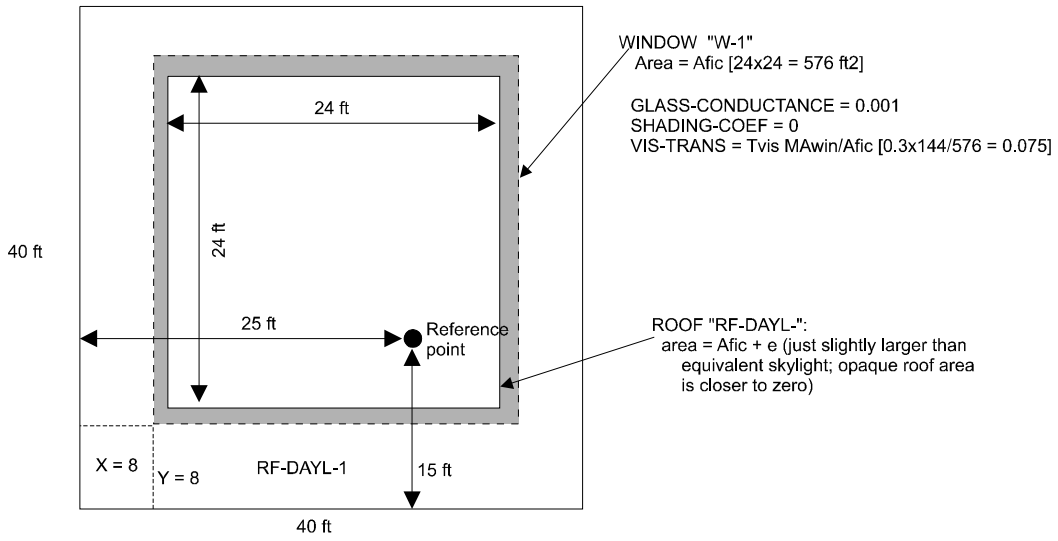


Figure 33 Equivalent roof with daylighting but no heat transfer

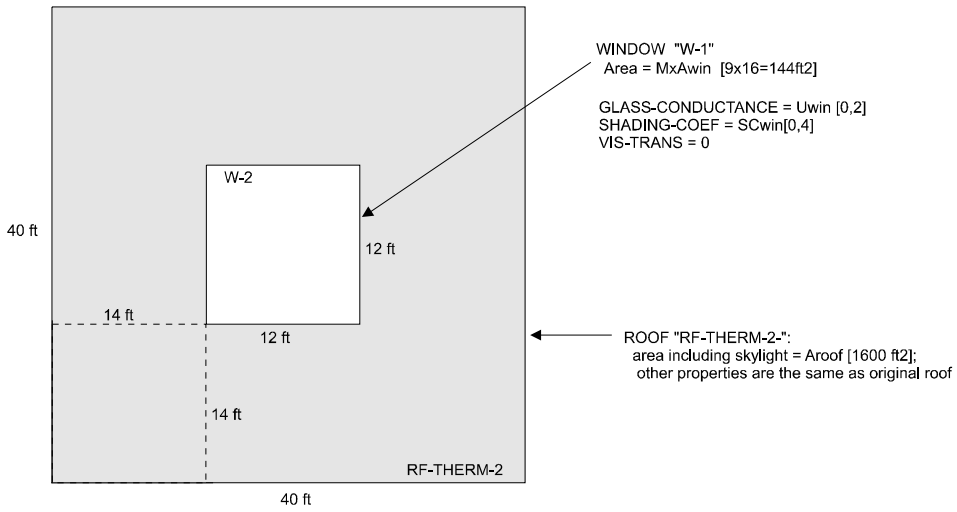


Figure 34 Equivalent roof with daylighting and heat transfer

The input for these two roof sections might look like this:

```
G-1 = GLASS-TYPE
   TYPE               = SHADING-COEF
   SHADING-COEF       = 0.0 $No solar gain$
   GLASS-CONDUCTANCE  = 0.001 $No conduction$
   VIS-TRANS          = 0.075 $Scaled$  ..

G-2 = GLASS-TYPE
   TYPE               = SHADING-COEF
   SHADING-COEF       = 0.4 $Actual value$
   GLASS-CONDUCTANCE  = 0.2 $Actual value$
   VIS-TRANS          = 0.0 $ No daylighting$  ..

S-1 = SPACE
   DAYLIGHTING        = YES
   LIGHT-REF-POINT1   = (25,15,2.5)
   ...

RF-DAYL-1 = ROOF
   H                  = 24.01
   W                  = 24.01 $ Daylighting roof sect. $
   TILT               = 0
   CONSTRUCTION       = CONST 1
   X                  = 8
   Y                  = 8
   Z                  = 10    ..

W-1 = WINDOW
   H                  = 24
   W                  = 24
   X                  = 0
   Y                  = 0
   GLASS-TYPE         = G-1
   INSIDE-VIS-REFL    = 0    ..
```

```
RF-THERM 2 = ROOF
   H                  = 40
   W                  = 40  $ Heat transfer roof sect. $
   TILT               = 0
   CONSTRUCTION       = CONST 1
   X                  = 0
   Y                  = 0
   Z                  = 10    ..

W-2 = WINDOW
   H                  = 12
   W                  = 12
   X                  = 14
   Y                  = 14
   GLASS-TYPE         = G 2   ..
```

Notes:

1. Be sure to adjust the visible transmittance of W-1 as shown in GLASS-TYPE = G-1.

2. The roof section in (B) should be positioned the same as the dashed bounding rectangle in (A). The position of the roof section in (C) is not important unless the roof is shaded. In this case, (C) can overlap (B).

Limitations of the Daylighting Calculation

The built-in daylight illuminance calculation works best when most of the illuminance at a reference point is due to light that reaches the reference point directly from the windows (i.e., without reflection from room surfaces). As a result, the calculation cannot reliably simulate the following configurations:

- interior or exterior light shelves;
- light scoops;
- skylights with deep wells;
- roof monitors;
- rooms with internal obstructions (partitions, etc.) that block light from the windows;
- reference point near the back of side-lit rooms that are very deep, i.e., whose depth is more than three times floor-to-ceiling height;
- the light reaching the reference point comes from windows in another space (for example, an atrium providing daylight to adjacent spaces

In these cases, the recommended procedure is to determine daylight illuminance factors from physical scale model measurements under real or artificial skies, or from a detailed illuminance calculation, such as SUPERLITE[9] . You can then read these factors into the program using the Input Function feature and override the built-in daylight factor calculation. The following example illustrates the procedure. An actual application of this process is described

[9] Contact the Windows and Daylighting Group (phone: 510-486-6845, fax 486-4089) at Lawrence Berkeley National Laboratory for information on SUPERLITE.

in "Modeling Complex Daylighting with DOE-2.1C" by M. Steven Baker, *DOE-2 User News*, Vol. 11, No. 1, Spring 1990[10] .

Example of Using Measured Daylight Factors

This function calculates daylight levels in a space using coefficients obtained by the user from physical scale model measurements of the ratio of interior to exterior illuminance. In the function, the coefficients are multiplied by the hourly total exterior illuminance from sun and sky to give the interior daylight illuminance. The measured coefficients for solar altitudes of 0, 10, 30, 50, and 70 degrees are entered using TABLE.

Notes:

1. This function assumes there are no movable shading devices on the windows that would alter the interior illuminance depending on whether the shades were open or closed.

2. The function does not re-calculate glare, so that the glare levels reported by the program should be ignored.

3. This function is illustrative only; the coefficients in an actual case could also depend on other factors, such as solar azimuth, cloud cover, etc.

```
SPACE1 = SPACE
   DAYLIGHTING        = YES
   ..... other daylighting-related keywords
   DAYL-ILLUM-FN      = (*NONE*,*MODEL-DATA-FN*)

(Other input)

END  ..

FUNCTION-NAME = MODEL-DATA-FN
   RDNCC   = RDNCC            $ DIRECT NORMAL RADIATION, BTUH/SF $
   BSCC    = BSCC             $ SKY DIFFUSE RADIATION $
                              $ ON HORIZONTAL, BTUH/SF $
   RAYCOS3 = RAYCOS3          $ SINE OF SOLAR ALTITUDE $
   PHSUND  = PHSUND           $ SOLAR ALTITUDE IN DEGREES $
   ILLUM   = DAYLIGHT-ILLUM1. $ DAYLIGHT ILLUMINANCE AT REF-PT-1 $
   $ FOOTCANDLES (REF-PT-2 NOT USED) $

ASSIGN
   TAB1    = TABLE (0,0)(10,.005)(30,.007)(50,.0085)
                                (70,.01)(90,.01) ..

CALCULATE ..

   C Get exterior horizontal illuminance from direct sun in fc
   C (Lumens/SF). Assumes that the luminous efficacy of direct
   C solar radiation = 100 Lumens/Watt = 29.3 Lumens/Btuh.

IDIRH = RDNCC * RAYCOS3 * 29.3
```

[10] Contact the Simulation Research Group (phone: 510-486-5711, fax 486-4089) at Lawrence Berkeley National Laboratory for back issues of the *User News*.

```
   C Get exterior horizontal illuminance from sky. Assumes that
   C the luminous efficacy of diffuse solar radiation = 125
   C Lumens/Watt = 36.6 Lumens/Btuh.

IDIFH = BSCC * 36.6

   C Get total exterior illuminance

ITOTH = IDIRH + IDIFH

   C Get interior daylight illuminance for current solar altitude

ILLUM = PWL (TAB1,PHSUND) * ITOTH

END

END-FUNCTION ..
```

Daylighting input examples

Example (1) A space has a single lighting zone with 2.4 watts/ft2 of installed electric lighting power. The photocell of a 5-step lighting control responds to the lighting level at x = 10, y = 20, z = 2.5 ft. The illuminance set point is 60 footcandles. The SPACE daylighting input would then be

```
$ --- ONE LIGHTING ZONE WITH STEPPED SYSTEM --- $

SPACE-1 = SPACE
   DAYLIGHTING       = YES
   LIGHTING-W/SQFT   = 2.4
   LIGHT-REF-POINT1  = (10,20,2.5)
   ZONE-FRACTION1    = 1.0 (the default)
   LIGHT-SET-POINT1  = 60
   LIGHT-CTRL-TYPE1  = STEPPED
   LIGHT-CTRL-STEPS  = 5
```

Example (2) An office space with 2 watts/ft2 of installed electric lighting power has three lighting zones. The first lighting zone, with 40% of the floor area, has a continuously dimmable control system with a setpoint of 60 footcandles and a minimum light output of 10 footcandles at 30% input power. The lighting reference point is at x = 10, y = 10, z = 2.5ft. The second lighting zone, with 50% of the floor area, has a 4-step control system with a setpoint of 60 footcandles. The lighting reference point is at x = 10, y = 25, z = 2.5ft. A third lighting zone with 10% of the floor area is not daylit.

The SPACE daylighting input would then be:

```
$ --- THREE LIGHTING ZONES --- $
```

```
SPACE-2 = SPACE
   DAYLIGHTING        = YES
   LIGHT-CTRL-STEPS   = 4
   LIGHT-CTRL-TYPE1   = CONTINUOUS
   LIGHT-CTRL-TYPE2   = STEPPED
   LIGHT-REF-POINT1   = (10,10,2.5)
   LIGHT-REF-POINT2   = (10,25,2.5)
   LIGHT-SET-POINT1   = 60
   LIGHT-SET-POINT2   = 60
   LIGHTING-W/SQFT    = 2
   MIN-LIGHT-FRAC     = 0.167
   MIN-POWER-FRAC     = 0.3
   ZONE-FRACTION1     = 0.4
   ZONE-FRACTION2     = 0.5
```

Note that no entry is required for the third, non-daylit lighting zone.

Example (3) A space has a task-ambient lighting system. Task lighting is provided by electric lights with an installed power of 0.5 watts/ft2. Ambient lighting with a setpoint of 10 footcandles is provided by daylight plus installed electric lighting at 0.4 watts/ft2 controlled by a 3-step control system. The ambient lighting reference point is at x = 15, y = 20, z = 2.5 ft.

The SPACE daylighting input would then be:

```
$ --- TASK AMBIENT  SYSTEM --- $

SPACE-3 = SPACE
   DAYLIGHTING        = YES
   LIGHT-CTRL-TYPE1   = STEPPED
   LIGHT-REF-POINT1   = (15,20,2.5)
   LIGHT-SET-POINT1   = 10
   LIGHTING-W/AREA    = 0.4
   TASK-LT-W/AREA     = 0.5
   ZONE-FRACTION1     = 1.0 (the default)
     .
```

WINDOW

Specifying the Solar, Optical and Thermal Properties

There are three ways of specifying the solar, optical and thermal properties of a window:: Window Shading Coefficient Method - you enter the window's shading coefficient, visible transmittance and conductance; Window Library Method - you select the window from the library; Window Layers Method -you enter the window layer-by-layer.. These three methods are described in detail below.

For all methods you can specify a window frame (see "Window Frames"). For window considerations related to daylighting, see "Daylighting."

Sun Control

There are several ways of controlling the operation of window shading devices. See "Switchable Glazing," below, and "Window Management" in the "Daylighting" topic.

Creating Custom Windows

You can create custom windows by

1. Using WIN-SPEC-METHOD = LAYERS-INPUT in the WINDOW command and entering the window layer by layer with the WINDOW-LAYERS keyword. This approach is described below under "Window Layers Method."

2. Running the WINDOW 5 computer program. To do this, you enter layer-by-layer glass and gap characteristics in WINDOW 5 and assign your own glass type code value to the window. Running WINDOW 4 produces an output file that you can add to the User Library. You can then set GLASS-TYPE-CODE in the GLASS-TYPE command equal to your assigned glass type code in the User Library.

Using Shading Devices with Windows from the Library

Shading devices (window coverings) can be modeled for glazing from the library by using the WINDOW keywords SHADING-SCHEDULE, CONDUCT-SCHEDULE, VIS-TRANS-SCH, etc.

Example - Window Shade Using SHADING-SCHEDULE

A window with argon-filled, low-E double glazing (GLASS-TYPE-CODE = 2635) has light-colored drapes deployed in the summer that reduce the shading coefficient of the glass by 40% and have negligible effect on the conductance of the glass. The input would look like:

```
SH-SCH-1 = SCHEDULE
   TYPE              = MULTIPLIER
   THRU MAY 31       (ALL)(1,24)(1.0)
   THRU OCT 31       (ALL)(1,24)(0.6)
   THRU DEC 31       (ALL)(1,24)(1.0) ..

GT-1 = GLASS-TYPE
   GLASS-TYPE-CODE = 2635 ..
```

```
WIN-1 = WINDOW
   HEIGHT            = 5
   WIDTH             = 10
   GLASS-TYPE        = GT-1
   SHADING-SCHEDULE  = SH-SCH-1 ..
```

In this example, the multiplier, 0.6, is the ratio of the shading coefficient of the glass with drapes present (a number that can usually be obtained from the glass manufacturer's data sheets) to the shading coefficient of the bare glass (which can be obtained from the glass manufacturer's data sheets or from the Window Library).

Printing Window Library Entries

A printout of the contents of the selected entry can be obtained as part of the input echo by entering DIAGNOSTIC COMMENTS in the line just before the GLASS-TYPE instruction. This printout can be used to verify that the selected entry is what you really want. Detailed print can be turned off by entering DIAGNOSTIC WARNINGS or DIAGNOSTIC ERRORS after the GLASS-TYPE instruction.

Specifying the Solar, Optical and Thermal Properties

There are three ways of specifying the solar, optical and thermal properties of a window:

1. Window Shading Coefficient Method - you enter the window's shading coefficient, visible transmittance and conductance.

2. Window Library Method - you select the window from the library.

3. Window Layers Method -you enter the window layer-by-layer.

The pro's and con's of these methods are as follows:

Window Entry Methods	Pro	Con
Shading coefficient	Convenient for conceptual design; fast calculation time	Inaccurate transmission/absorption angular dependence for multipane or coated glazing
Library	Accurate angular dependence	May not match actual glazing; longer calculation time
Layers	Accurate angular dependence; can match actual glazing; can add blinds	Longer calculation time

Window Shading Coefficient Method

In this method you give the window's shading coefficient (SHADING-COEF keyword) and thermal conductance (GLASS-CONDUCTANCE keyword) in the GLASS-TYPE command, as shown in the following example.

Example - Window shading coefficient method

```
GT-1 = GLASS-TYPE
   TYPE              = SHADING-COEF
   SHADING-COEF      = 0.7
   GLASS-CONDUCTANCE = 0.4 ..
```

```
WIN-1 = WINDOW
   X                 = 12
   Y                 = 3
   HEIGHT            = 4
   WIDTH             = 5
   GLASS-TYPE        = GT-1 ..
```

If you are doing a daylighting calculation you can also enter VIS-TRANS, the visible transmittance of the glazing at normal incidence.

In the shading coefficient method the solar heat gain through the window is first calculated for standard, 1/8-in thick clear glass, then multiplied by the shading coefficient. Because this method uses the angular dependence of solar transmission and absorption for standard glass rather than the actual angular dependence, it can lead to errors, which are generally small (less than 10%) for monthly integrated solar gains but can be large (up to 30% or so) for peak solar gains for multipane or coated glazing. Another limitation of the shading coefficient method is that the conductance does not depend on the temperature difference across the window.

Window Library Method

The Window Library has about 200 window entries covering commonly-available glazings as well as experimental electrochromic glazings. The choices are shown in the "Window Library" in *DOE-2.2 Libraries & Reports*. Included are single-, double-, triple- and quadruple-pane glazings with different tints, coatings, gas fills, glass thicknesses, and gap widths.

The window library was created using WINDOW 4, a computer program that does a very detailed calculation of conduction and solar heat gain through windows. When windows from the library are selected, the window heat transfer calculation will be very close to that in WINDOW 4. This means that the correct angular dependence of solar transmission and absorption is determined, the conductance has the proper dependence on temperature difference across the window, and 2-D conduction effects at the edge of the window glazing are accounted for.

You select a window from the library by specifying GLASS-TYPE-CODE in the GLASS-TYPE command to be one of the values indicated by G-T-C in the "Window Library" in *DOE-2.2 Libraries & Reports*. Because conductance data, number of panes, and solar-optical properties come from the library, you do not have to specify the GLASS-TYPE keywords GLASS-CONDUCTANCE, PANES or VIS-TRANS. In the following example we have selected from the library low-E clear double-pane glazing with 6mm glass and 12mm gap filled with argon (G-T-C number 2635 in the Window Library).

Example - Window library method

```
GT-2 = GLASS-TYPE
   TYPE              = GLASS-TYPE-CODE
   GLASS-TYPE-CODE   = 2635 ..

WIN-1 = WINDOW
   X                 = 12
   Y                 = 3
   HEIGHT            = 4
   WIDTH             = 5
   GLASS-TYPE        = GT-2 ..
```

Window Layers Method

A window can be described layer by layer using the WINDOW-LAYERS keyword in the WINDOW command. This keyword takes as its value a list of U-names of WINDOW-LAYER commands. For example,

```
GLASS-1 = WINDOW-LAYER ....

GAP-1 = WINDOW-LAYER ....

BLIND-1 = WINDOW-LAYER ....

WIN-1 = WINDOW
   WIN-SPEC-METHOD    = LAYERS-INPUT
   WINDOW-LAYERS      = (GLASS-1, GAP-1, BLIND-1)
   etc.  ..
```

describes a window consisting of a layer of glass named GLASS-1, a gap named GAP-1 and a blind named BLIND-1. The WINDOW-LAYERS keyword can only be used if WIN-SPEC-METHOD = LAYERS-INPUT.

For details on using this method, see the descriptions of the WINDOW-LAYER command and the WINDOW-LAYERS keyword of the WINDOW command in the *DOE-2.2 Dictionary*. The main advantage of this method is that the solar and daylighting properties of window blinds can be modeled much more accurately than is possible with the shading coefficient or library methods. In particular, the layers approach properly accounts for the transmittance of blinds as a function of slat angle and for the interreflections between blind and glass layers.

Window Conduction Calculation

Improved Glass Conduction Calculation

The conduction calculation for glazings from the library will be a few percent more accurate if you specify the GLASS-TYPE keyword CONVERGENCE-TOL (in °C for both metric and English runs). This invokes a time-consuming iterative calculation that converges when, for each glass layer, the temperature difference between successive iterations is less than CONVERGENCE-TOL. Because of the increase in calculation time, CONVERGENCE-TOL should only be used for research applications. If CONVERGENCE-TOL is not specified, the glazing U-value is based on glass layer temperatures that are equally spaced between the outside and inside air temperature.

Edge-of-Glass Effects

Because of two-dimensional heat conduction effects in multipane windows, the U-value of the edge-of-glass region (a 2.5-in [6.35-mm] wide border strip at the boundary of the glazing) differs from the U-value in the center-of-the-glass region (the central part of the glazing). The edge-of-glass U-value depends on the center-of- glass U-value and the type of spacer used to separate the panes. For windows selected from the library or defined using the WINDOW-LAYERS keyword, the spacer type is specified with the WINDOW keyword SPACER-TYPE, which takes values ALUMINUM, STEEL, BUTYL/METAL, and INSULATED. SPACER-TYPE is applicable only to multipane windows.

Frames

Window frames can be explicitly defined but we recommend that you do so only if the frame area is more than 10% or so of the glazed area of a window (which is generally the case only in residential applications).

To define a window frame you enter the width, conductance and absorptance of the frame in the WINDOW command. For details on describing frames, see the description of the FRAME-WIDTH, FRAME-CONDUCT and FRAME-ABS keywords under "WINDOW Command" in the *DOE-2.2 Dictionary*.

The program finds the overall window conduction by adding frame, edge-of-glass, and center-of-glass contributions. Thus, all three of these contributions are included in each of the following report quantities:

1. "Window Conduction" in summary reports LS-B, LS-C, LS-E, and LS-F;

2. "Window U-value" and "Window Area" in verification report LV-D;

3. WINDOW hourly report variable #1, "Window U-value."

Additional points to remember about frames are:

1. A window MULTIPLIER also multiplies the frame.

2. Window fins and overhangs shade the frame as well as the glazing.

3. Shading devices, like blinds and drapes, affect only the glazed part of the window; they do not affect the heat conduction through the frame.

Example - Window with Frame

The glazed part of a window is 3 ft wide and 4 ft high. The glazing is double-pane low-E with 6-mm glass thickness and argon gas fill (GLASS-TYPE-CODE = 2635). The wood frame is 3 in (0.25 ft) wide on all sides and has an absorptivity of 0.8 and a conductance (excluding outside air film) of 0.434 Btuh/ft2. The spacer separating the glass panes is aluminum.

```
$ -- Window with frame -- $

GT-1 = GLASS-TYPE
   TYPE               = GLASS-TYPE-CODE
   GLASS-TYPE-CODE    = 2635 ..

WIN-1 = WINDOW
   GLASS-TYPE         = GT-1
   HEIGHT             = 4.0
   WIDTH              = 3.0
   SPACER-TYPE        = ALUMINUM
   FRAME-WIDTH        = 0.25
   FRAME-ABS          = 0.8
   FRAME-CONDUCTANCE  = 0.434 ..
```

Switchable Glazing

"Switchable glazing" is glazing with solar-optical properties, such as transmittance, that change according to environmental conditions. An example is electrochromic glass that can be switched from a bleached state of high transmittance to a colored state of lower transmittance by changing the applied voltage in response to a control variable such as outside temperature or solar radiation. Switchable glazing has the potential for a higher level of solar gain control than is possible with conventional glazing having fixed solar-optical properties. To model switchable glazing you enter the glass type for the unswitched state, the glass type for the fully switched state, the control variable, the switching set points, and a schedule that tells when switching is allowed. Figure 35 shows the control action that the program uses for all control options except SWITCH-CONTROL = DAYLIGHT-LEVEL.

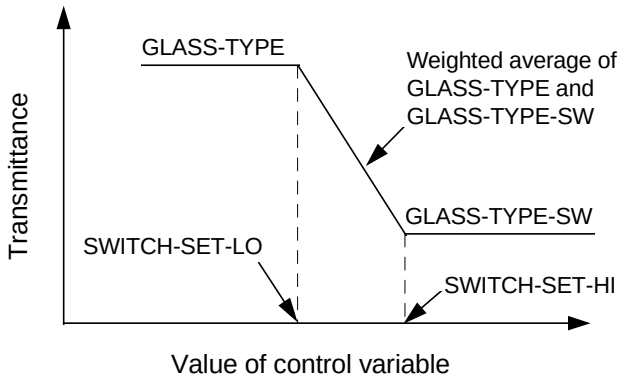


Figure 35 Control action for switchable glazing. Glass properties, such as solar and visible transmittance, depend on the value of a user-specified control variable.

If the value of the control variable is less than SWITCH-SET-LO, the glass is in the unswitched state, with solar-optical properties given by GLASS-TYPE. If the control variable is greater than SWITCH-SET-HI, the glass is in the fully switched state, with solar-optical properties given by GLASS-TYPE-SW. If the control variable is between SWITCH-SET-LO and SWITCH-SET-HI, the glass is in a partially switched state, with solar-optical properties given by a weighted average of GLASS-TYPE and GLASS-TYPE-SW. For example, if TA and TB are the direct solar transmittances for GLASS-TYPE and GLASS-TYPE-SW, respectively, and V is the value of the control variable in a particular hour, then the resultant transmittance is

T = TA (1-S) + TB S

where S, the "switching factor", is given by:

S = 0.0, if V < SWITCH-SET-LO

S = (V - [SWITCH-SET-LO]) / ([SWITCH-SET-HI] - [SWITCH-SET-LO]),

if SWITCH-SET-LO < V < SWITCH-SET-HI

S = 1.0, if V > SWITCH-SET-HI

Thus, S varies from 0.0 for the unswitched state to 1.0 for the fully-switched state. If the low and high switching points are equal (i.e., SWITCH-SET-LO = SWITCH-SET-HIGH), the glass changes from the unswitched state to the fully-switched state with no intermediate, partially-switched states. In this case S has only two values, 0.0 or 1.0. Hourly values of S for each window are printed by hourly report VARIABLE-TYPE = U-name of WINDOW, Variable-List Number 18.

The GLASS-TYPE keyword accepts the U-name of the glass type for the unswitched (clear) state and GLASS-TYPE-SW accepts the U-name of the glass type for the fully switched (colored) state. Both of these glass types types have GLASS-TYPE-CODE > 1000 and must be chosen from the library (see "Window Library Method"). An error will result if the number of glass layers is different for GLASS-TYPE and GLASS-TYPE-SW.

Example - Switchable Glazing

Switching is controlled by incident solar radiation. During the summer, the outer pane of insulating glass switches from clear to fully tinted over a range of 20 to 100 Btuh/ft2 of incident solar radiation.

```
CLEAR-IG-1 = GLASS-TYPE
   TYPE              = GLASS-TYPE-CODE
   GLASS-TYPE-CODE   = 2003 .. $ SC=0.81 $

TINTED-IG-1 = GLASS-TYPE
   TYPE              = GLASS-TYPE-CODE
   GLASS-TYPE-CODE   = 2203 .. $ SC=0.81 $

SUMMER-1 =  SCHEDULE
   TYPE              = ON/OFF
   THRU MAY 31       (ALL)(1,24)(0) $ no switching $
   THRU SEP 30       (ALL)(1,24)(1) $ switching ok $
   THRU DEC 31       (ALL)(1,24)(0) $ no switching $ ..

WIN-1 = WINDOW
   GLASS-TYPE        = CLEAR-IG-1
   GLASS-TYPE-SW     = TINTED-IG-1
   SWITCH-CONTROL    = TOT-SOL-INC
   SWITCH-SET-LO     = 20
   SWITCH-SET-HI     = 100
   SWITCH-SCH        = SUMMER-1
      .....
```

Notes:

1. If there is more than one window in a space, some may have switching control and others not. For example, skylights might be controlled and view windows not. Also, multiple windows in a space can have different control types.

2. Switching control is applicable only to exterior windows (windows in EXTERIOR-WALLs). It does not work for interior windows.

3. Switching control is in effect only during sun-up hours. It does not work at night. It should not be used to switch between window U-values; use the WINDOW keyword CONDUCT-TMIN-SCH instead.

4. Shading devices such as blinds and drapes (as specified with WINDOW keywords SHADING-SCHEDULE, VIS-TRANS-SCH, etc.) can be used in conjunction with switching control of the glazing. In this case, the program decides what state the glazing should be switched to, ignoring the possible presence of shading devices, and then adjusts the solar intensity through the switched glazing for the presence of the shading device.

For more details on modeling switchable glazing see the description of the WINDOW keywords GLASS-TYPE-SW and SWITCH-CONTROL in the *DOE-2.2 Dictionary*..

Electrochromic Switchable Glazing

There are two classes of electrochromics:"absorbing" and "reflecting". For absorbing electrochromics, the near-IR absorptance increases in the colored state. For reflecting electrochromics, the near-IR reflectance increases in the colored state. Reflecting electrochromics have a somewhat lower shading coefficient for a given visible transmittance and so may perform better in daylighting applications in cooling-dominated buildings. The electrochromic entries are as follows:

Table 14 Electrochromic Switchable Glazing Entries

GLASS-TYPE-CODE Range	Description
1800-1801	Single-pane absorbing electrochromic
1802-1803	Single-pane reflecting electrochromic
2800-2805	Double-pane absorbing electrochromic
2820-2825	Double-pane reflecting electrochromic
2840-2845	Double-pane absorbing electrochromic, low-E
2860-2865	Double-pane reflecting electrochromic, low-E

For the single-pane cases, the electrochromic layer is sandwiched between two 3mm clear glass layers. For the double-pane cases, the electrochromic layer is on surface 2, i.e. on the gap side of the outer pane. For the double-pane low-E cases, the low-E coating is on surface 3, i.e. on the gap side of the inner pane. For the double-pane cases you can choose gap widths of 6.3mm (air fill) or 12.7mm (air or argon fill).

Figure 36 shows the properties of the electrochromic switchable glazings that are in the library. A detailed list of the electrochromic entries can be found in the "Window Library" in *DOE-2.2 Libraries & Reports*. The electrochromics in the library are in pairs, such as (1800,1801), (1802,1803), etc.; the first member of the pair is the unswitched, bleached state and the second member is the fully-switched, colored state. For electrochromic switchable glazing simulation, GLASS-TYPE and GLASS-TYPE-SW must correspond to one of these pairs; for example, the following would be an acceptable input using the 1800 and 1801 pair:

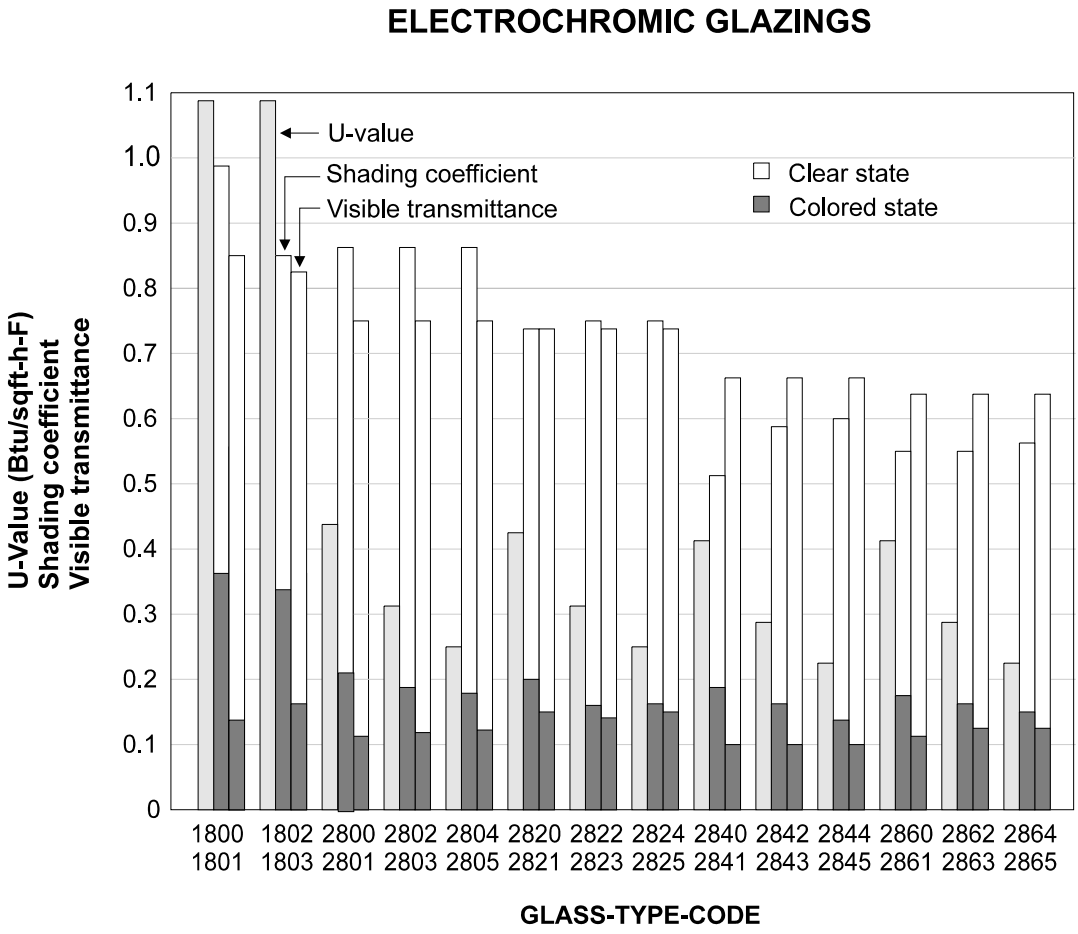


Figure 36 Electrochromic Glazings in the Window Library. Shown are U-value (ASHRAE winter conditions), shading coefficient (ASHRAE summer conditions) and visible transmittance at normal incidence. For each GLASS-TYPE-CODE pair (such as 1800 and 1801), the shading coefficient and visible transmittance for the clear state are given by the unshaded bars, and for the colored state, by shaded bars. Only one U-value is shown for each pair since the U-value changes very little between clear and colored states.

Example - Electrochromic Glazing

```
EC-1 = GLASS-TYPE
   TYPE              = GLASS-TYPE-CODE
   GLASS-TYPE-CODE   = 1800 .. $ Absorbing electrochromic,
                               $ bleached

EC-2 = GLASS-TYPE
   TYPE              = GLASS-TYPE-CODE
   GLASS-TYPE-CODE   = 1801 .. $ Absorbing electrochromic,
                               $ colored
```

```
WI-1 = WINDOW
   GLASS-TYPE          = EC-1
   GLASS-TYPE-SW       = EC-2
   SWITCH-CONTROL      = TOT-SOL-INC
   SWITCH-SET-LO       = 20
   SWITCH-SET-HI       = 100
     .....
```

You will get an error message if the GLASS-TYPE and GLASS-TYPE-SW values for a window are not a legal pair. For example using 1803 instead of 1801 in the above example would give an error message because 1800 and 1803 are not a legal electrochromic pair.

Electrochromics are still in the experimental stage. The electrochromic glazings in the library are generic; they are representative of products that were under development as of December 1992. The electrochromic entries were generated with WINDOW 4 by D. Hopkins and E. Finlayson of the LBNL Windows and Daylighting Group using spectral transmittance and reflectance data. These data were compiled by M. Rubin of LBNL from measurements on actual electrochromics from LBNL and other research organizations.

Example: Window Shading Device Assignment

Window glazing has a visible transmittance of 0.83. Operable drapes have a visible transmittance multiplier of 0.35, a shading coefficient multiplier of 0.25, and a conductance multiplier of 0.85. The drapes will be closed when incident solar intensity exceeds 30 Btu/ft2-hr.

```
SC-SCH-1 = SCHEDULE
   TYPE                = MULTIPLIER
   THRU DEC 31         (ALL)(1,24)(0.25) ..

TVIS-SCH-1 = SCHEDULE
   TYPE                = MULTIPLIER
   THRU DEC 31         (ALL)(1,24)(0.35) ..

COND-MULT-1 = SCHEDULE
   TYPE                = MULTIPLIER
   THRU DEC 31         (ALL)(1,24)(0.85) ..

SOL-SCH-1 = SCHEDULE
   TYPE                = MULTIPLIER
   THRU DEC 31         (ALL)(1,24)(30) ..

GT-1 = GLASS-TYPE
   TYPE                = SHADING-COEF
   VIS-TRANS           = 0.83
   SHADING-COEF        = 0.75
   . . . .

SP-1 = SPACE
   DAYLIGHTING         = YES
   . . . .
```

```
WIN-1 = WINDOW
   GLASS-TYPE         = GT-1
   WIN-SHADE-TYPE     = MOVABLE-INTERIOR
   VIS-TRANS-SCH      = TVIS-SCH-1
   MAX-SOLAR-SCH      = SOL-SCH-1
   SHADING-SCH        = SC-SCH-1
   CONDUCT-SCH        = COND-MULT-1
   . . . .
```

SPACE AS A SUNSPACES

In sunspaces a significant fraction of the heat generated by solar gain in the space is transferred to adjacent spaces. Examples of sunspaces are sun rooms attached to houses and atriums with skylights in commercial buildings. Figure 37 shows the different forms of this heat transfer, such as forced convection and solar through interior windows. These forms are:

1. direct and diffuse solar gain through interior glazing;

2. forced or natural convection through vents or an open doorway;

3. quick or delayed conduction through an interior wall, taking into account solar radiation absorbed on the sunspace side of the wall; and

4. conduction through interior glazing.

Sunspaces in can be mechanically heated and cooled, or they may be unconditioned. The program also simulates venting of the sunspace with outside air to prevent overheating and, for residential applications, the use of a sunspace to preheat outside ventilation air.

You can control the airflow from the sunspace with a schedule or on the basis of a threshold temperature difference between the sunspace and the adjacent space.

You can simulate additional features for sunspaces, including sun control with movable window shades, movable insulation on exterior windows, shading by fins and overhangs, sloped glazing, and the effects of thermal mass.

The model is intended primarily for residential and small commercial building applications. Because the program calculates only a single, average air temperature in a space, *this simulation cannot be expected to give accurate results for multi-story atriums unless there is sufficient air mixing to eliminate temperature stratification.*

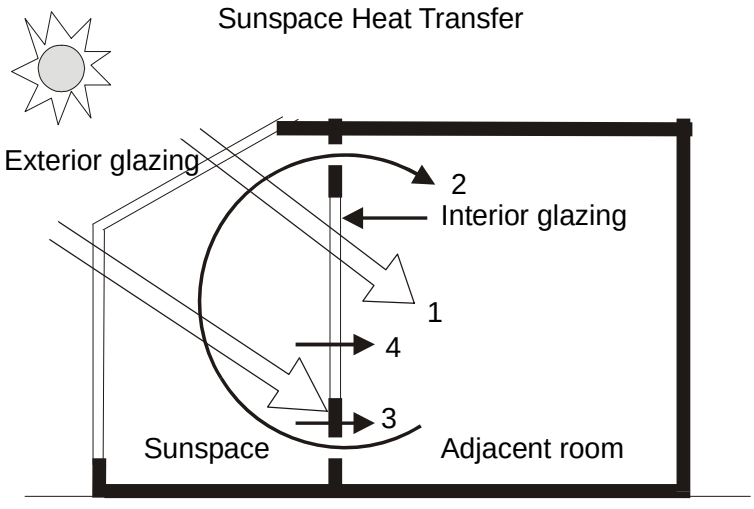


Figure 37 Forms of heat transfer calculated by the sunspace model.

The program's sunspace modeling capabilities include:

1. Solar radiation absorbed by sunspace interior walls is determined hourly and used in conduction calculations.

2. Sunspace interior walls can be positioned with X, Y, Z, AZIMUTH and TILT.

3. Conduction across sunspace interior walls can be quick or delayed

4. Solar gain through sunspace interior windows and conduction across interior windows is calculated. Interior windows can have movable insulation and shading. They can be shaded by BUILDING-SHADEs inside or outside the sunspace.

5. Sunspace INTERIOR-WALLs can have openings through which fan-forced or natural convective heat transfer between sunspaces and adjacent spaces can occur. For natural convection, non-linear DT dependence is accounted for. Convection can be controlled thermostatically or via schedule.

6. A sunspace can be vented with outside air to prevent overheating. Venting is independent of temperature of other zones. It works with any system type except PIU.

7. A sunspace can be modeled as a plenum in order to heat return air.

Sunspace Elements

Interior Windows

Interior windows can be specified by following an INTERIOR-WALL command by one or more WINDOW commands. Interior walls can have windows. The keywords for interior windows are the same as those for exterior windows, with some exceptions:

1. The following keywords are unused:

FRAME-WIDTH	SETBACK
GLARE-CTRL-PROB	SHADING-DIVISION
GLASS-TYPE-SW	SWITCH-CONTROL
GND-FORM-FACTOR	SWITCH-SCH
INF-COEF	SWITCH-SET-HI
LEFT-FIN-A, etc.	SWITCH-SET-LO
OVERHANG-A, etc.	VIS-TRANS-SCH
RIGHT-FIN-A, etc.	WIN-SHADE-TYPE

2. SKY-FORM-FACTOR multiplies the total diffuse radiation incident on an interior window. If the interior window has a setback (relative to the sunspace) or there are obstructions inside the sunspace that shade the interior window, a value of SKY-FORM-FACTOR less than 1.0 should be specified (the default value is 1.0).

Shading devices on interior windows, like Venetian blinds, drapes, or pull-down shades, can be simulated via the keywords SHADING-SCHEDULE and MAX-SOLAR-SCH. Movable insulation on interior windows can be modeled using keywords CONDUCT-SCHEDULE and CONDUCT-TMIN-SCH.

For an accurate calculation of the solar radiation transmitted by a sunspace interior window, it is important to specify the X and Y coordinates of the window. These coordinates are measured with respect to the lower-left hand corner of the INTERIOR-WALL as viewed in the NEXT-TO space (see "INTERIOR-WALL Command" in the *DOE-2.2 Dictionary*). The position of exterior windows should also be carefully specified. The program will only recognize interior windows in an interior wall between a sunspace and a non-sunspace.

Sliding glass doors can be modeled as interior windows. If the interior wall containing the glass door has AIR-FLOW-TYPE = FREE-DOORWAY (see WALL-PARAMETERS, below), the door will be assumed to be open and convection through the opening will be calculated if T(sunspace) - T(adjacent space) > AIR-FLOW-CTRL-DT.

Additional control of the opening and closing of the door can be obtained by using SS-FLOW-SCH (see description of ZONE keywords, below).

An unglazed opening in a sunspace interior wall can be input as a window with GLASS-TYPE-CODE = 0. The program will calculate the solar radiation passing through the opening by using a transmittance of 1.0 for all angles of incidence. WALL-PARAMETERS data, described below, would be entered for the INTERIOR-WALL to specify the convective air flow through the opening.

Interior Doors

Unlike exterior walls, interior walls in cannot have doors. However, an opaque interior door with a conductance significantly different from the sunspace interior wall containing it can be input as a separate interior wall. Alternatively, the door can simply be ignored if the conduction across it is small compared to the overall conduction across the wall. The program will calculate convection through a fully or partially open door if AIR-FLOW-TYPE = FREE-DOORWAY and appropriate values of DOORWAY-H and DOORWAY-W are specified (see WALL-PARAMETERS, below).

Use Glass Type not Shading Coefficient for Sunspaces

You should use GLASS-TYPE-CODE rather than SHADING-COEF for sunspace exterior windows. This allows the program to accurately calculate the hourly direct and diffuse radiation transmitted by the glazing. This is not possible with SHADING-COEF except for standard 1/8" clear glass.

Use Custom Weighting Factors for Sunspaces

Custom Weighting Factors (CWFs) should be used for sunspaces for several reasons:

1. For high conductance spaces, the precalculated (ASHRAE) weighting factors overestimate heating and cooling loads. The overestimate can be as high as 25-30% for heavily glazed spaces.

2. CWFs account for loss of solar gain due to reflection of sunlight back out of exterior windows.

3. CWFs give a more accurate calculation of the generally large temperature swings in a solar-driven space.

4. CWFs will automatically be calculated for any space with FLOOR-WEIGHT = 0 (the default value). Otherwise, the program will use ASHRAE weighting factors.

Positioning Surfaces

For an accurate calculation of solar radiation falling on the interior walls of a sunspace, the bounding surfaces of the sunspace need to be geometrically positioned. This applies to the exterior walls and roofs and their associated windows, and the interior walls and their associated windows. We recommend that a sunspace interior wall be defined in the sunspace rather than in the adjacent room. Otherwise, the adjacent room must be properly located with respect to the sunspace. If this is not done, the interior walls and windows will be mispositioned relative to the sunspace exterior windows, and the projection of solar radiation from the windows onto the interior walls will be incorrect. This will give a wrong calculation of the solar radiation transferred from sunspace to room. Even in this case, there will be no fictitious overall solar gain or loss since the solar that stays in the sunspace plus that transferred to adjacent rooms is constrained by the program to equal that entering the sunspace. There will, however, be an error message if the transferred solar exceeds the entering solar, which would give a net negative solar gain in the

sunspace. This may occur if interior walls or windows on them overlap, if a multiplier is used on an interior window, or if a multiplier is used on rooms adjacent to a sunspace.

Massive Interior Walls

Sunspace interior walls are often fairly heavy, leading to a significant time delay in the heat transfer across them by conduction. Such walls should be described by response factors, i.e., with a delayed-type construction.

The order of defining layers in a delayed interior wall is from "outside" to "inside", where "outside" is the side of the wall in the NEXT-TO space, and "inside" is the side in the space in which the wall was defined. If, as recommended, the interior wall is defined in the sunspace, then the outside of the wall is the side in the adjacent room.

Delayed conduction through interior walls is calculated only for sunspace interior walls. For other interior walls the hourly conduction is quick.

Delayed conduction through an interior between two non-sunspaces can be obtained simply by assigning SUNSPACE = YES to one of the spaces, even though the space is not actually a sunspace. In this case, If the solar flux on the "sunspace" side of the wall is small, it is recommended that INSIDE-SOL-ABS = (0,0) be input for the wall in order to zero out absorption of solar radiation. Otherwise, all interior and exterior walls and windows in the "sunspace" should be geometrically positioned as described above in "Positioning of Sunspace Surfaces".

Moisture from Plants and Trees

Atriums often have plants and trees. Moisture transpiring from leaves and evaporating from soil can produce a significant latent load. You can model this load using the source keywords in SPACE as follows:

```
SOURCE-TYPE       = PROCESS
SOURCE-LATENT     = 1.0
SOURCE-SENSIBLE   = 0.0
SOURCE-POWER      = [latent load in Btu/hr or W]
SOURCE-SCHEDULE   = U-name of schedule
```

Baffles and Louvers

Baffles and louvers on sunspace exterior windows, which block and/or diffuse incoming beam radiation, can be modeled as blinds using the WINDOW-LAYERS keyword in the WINDOW command. See "Window Layers Method" in the Window topic. The blinds can be interior, exterior or between pane. They can be controlled in different ways to manage solar gain.

Air Flow Control

It is usually necessary to control the airflow between a sunspace and adjacent rooms. This is done for one or more of the following reasons:

1. to avoid overheating the room with warmer air from the sunspace;
2. to prevent circulating cold air from the sunspace during the heating season;
3. to run the sunspace fan only when the temperature differerence between sunspace and room air is large enough to give effective heat transfer at design airflow.

The following control mechanisms are available; they can be used singly or in combination.

Differential Control

Using AIR-FLOW-CTRL-DT in WALL-PARAMETERS gives differential temperature control for flow across a sunspace INTERIOR-WALL. Air flow across the wall occurs only if

T(sunspace) - T(room) > AIR-FLOW-CTRL-DT

AIR-FLOW-CTRL-DT is generally chosen to be a few degrees.

For AIR-FLOW-TYPE = FREE-RECIRC, differential control assumes that the vents have back-flow dampers that prevent reverse circulation.

For AIR-FLOW-TYPE = FREE-DOORWAY differential control assumes occupants open and close the intervening doorway as relative temperature conditions change.

For AIR-FLOW-TYPE = FORCED-RECIRC differential control assumes that the fan is controlled by a thermostat that senses the sunspace-to-room temperature difference.

Time-Clock Control

For seasonal or day-night control, you specify SS-FLOW-SCH in the Systems ZONE command for the sunspace. The airflow from a sunspace is multiplied by the hourly SS-FLOW-SCH value. If the schedule value is 0 the flow is cut off completely. This is shown in the following example where there is no flow at night or from June to September.

```
FLOWSCH-1 = SCHEDULE
   TYPE               = MULTIPLIER
   THRU MAY 31        (ALL) ( 1, 8) (0)
                            ( 9,17) (1)
                            (18,24)(0)
   THRU SEP 30        (ALL) ( 1,24) (0)
   THRU DEC 31        (ALL) ( 1, 8) (0)
                            ( 9,17) (1)
                            (18,24) (0) ..

SUNSP-1 =  ZONE
   SS-FLOW-SCH        = FLOWSCH-1
   .....
```

Room Thermostat Flow Control

SS-FLOW-T-SCH allows control of airflow from a sunspace to an adjacent room based on room air temperature. If T(room) is higher than the SS-FLOW-T-SCH value, airflow from the sunspace is turned off. If T(room) is lower than the SS-FLOW-T-SCH value (and the SS-FLOW-SCH value is non-zero) airflow occurs if

T(sunspace) - T(room) > AIR-FLOW-CTRL-DT

Generally, SS-FLOW-T-SCH values should be between the room heating and cooling setpoints. If SS-FLOW-T-SCH is not specified, the program will use a default value of 74F (23.3C) for all hours.

Sun Control

Sun control is generally needed to reduce solar heat gain in a sunspace during the summer. This can be accomplished in one or more ways: (1) with external projections such as overhangs, (2) by making some or all of the sunspace exterior glazing reflective or heat absorbing, (3) by using window coverings, or (4) by using switchable

glazing (see "Electrochromic Switchable Glazings" in the Window topic). The window coverings may be fixed or movable as determined by SHADING-SCHEDULE. They can also be deployed whenever a solar threshold value, as specified by MAX-SOLAR-SCH, is exceeded.

The degree of shading that a sunspace requires depends, of course, on the extent to which it is used as an occupied space.

Example - Exterior Window Covering

Window coverings with a shading coefficient multiplier of 0.3 and 20% solar transmittance are used from June through October whenever incident solar radiation exceeds 50 Btu/ft2-hr:

```
SOLTRANS-1 = SCHEDULE
   TYPE               = MULTIPLIER
   THRU DEC 31        (ALL) (1,24) (0.2)  ..

SHMULT-1 = SCHEDULE
   TYPE               = MULTIPLIER
   THRU DEC 31        (ALL) (1,24) (0.3)  ..

SOL-THRESH-1 = SCHEDULE
   TYPE               = RADIATION
   THRU MAY 31        (ALL) (1,24) (1000)
   THRU OCT 30        (ALL) (1,24) (50)
   THRU DEC 31        (ALL) (1,24) (1000) ..

SUNSPWIN = WINDOW
   SOL-TRANS-SCH      = SOLTRANS-1
   SHADING-SCHEDULE   = SHMULT-1
   MAX-SOLAR-SCH      = SOL-THRESH-1
        .....
```

Sun control may also be desirable for interior windows in a sunspace to prevent excessive solar gain into the adjoining room. SHADING-SCHEDULE and MAX-SOLAR-SCH can be used for interior windows in the same way as they are used for exterior windows. Another way of shading interior windows is to locate one or more BUILDING-SHADEs inside the sunspace. The program cannot model switchable interior glazing.

Reducing Heat Loss from Exterior Glazing

Sunspaces typically have large glazed areas. The high U-value of bare, single glazing—which is about 1.0 Btu/ft2-hr-F (5.7 W/m2)—leads to significant conductive heat loss to the outside in the winter except in very mild climates. Some ways of reducing this heat loss are:

1. Use high-performance glass with a low U-value (see "Window Library" in *DOE-2.2 Libraries & Reports*).

2. Use movable insulation by inputting a CONDUCT-SCHEDULE that decreases the overall window conductance at night, or specify a CONDUCT-TMIN-SCH that moves insulation into place when the outside temperature is low.

3. Use translucent insulating panels in place of some or all of the clear glazing by inputting values for SHADING-COEF and GLASS-CONDUCTANCE obtained from manufacturer's data.

Example - Movable Insulation

R-5 insulating panels cover a single-glazed sunspace exterior window November through April whenever the outside air temperature falls below 40F (4.4C). The shading coefficient multiplier of the insulation is 0.1. The solar transmittance is 2%.

```
CONDMULT-1 = SCHEDULE
   TYPE               = MULTIPLIER
   THRU  DEC 31       (ALL) (1,24) (.12) ..

SHMULT-1 = SCHEDULE
   TYPE               = MULTIPLIER
   THRU  DEC 31       (ALL) (1,24) (.1)  ..

TMIN-1 = SCHEDULE
   TYPE               = TEMPERATURE
   THRU APR 30        (ALL) (1,24) (40)
   THRU OCT 31        (ALL) (1,24) (0)
   THRU DEC 31        (ALL) (1,24) (40) ..

SOLTRANS-1 = SCHEDULE
   TYPE               = MULTIPLIER
   THRU DEC 31        (ALL) (1,24) (.02) ..

SUNSPWIN = WINDOW
   CONDUCT-SCHEDULE   = CONDMULT-1
   SHADING-SCHEDULE   = SHMULT-1
   CONDUCT-TMIN-SCH   = TMIN-1
   SOL-TRAN-SCH       = SOLTRANS-1
       .....
```

In this example the value for the conductance multiplier (0.12) is the ratio of the window conductance (excluding outside air film) with and without insulation; in English units, this ratio is:

$(5+.68)^{-1}/(.68)^{-1} = 0.176/1.47 = 0.12$

The above heat loss measures can also be modeled for interior glazing. In this case, the program expects outside air temperatures for CONDUCT-TMIN-SCH (as for exterior windows), not sunspace air temperatures.

Solar Radiation Absorbed by Interior Walls

The program calculates conduction through a sunspace interior wall by doing a heat balance on both surfaces. (This is done in the Systems program since the actual air temperatures on both sides of the wall have to be known.) The hourly solar radiation absorbed by the sunspace side of the wall is included in the heat balance. Part of this absorbed solar radiation is conducted into the adjacent room.

The amount of solar radiation absorbed depends on the incident flux and the absorptance of the wall. The following section describes how the incident flux is determined. The absorptance is input via the keyword INSIDE-SOL-ABS for INTERIOR-WALL. Typical solar absorptance values are listed in a table in the description of the CONSTRUCTION command. If not specified, the solar absorptance defaults to 0.5 for walls, 0.8 for floors, and 0.3 for ceilings.

In the conduction calculation, the direct and diffuse solar radiation absorbed by an interior wall is assumed to be uniformly distributed over its surface. If part of the wall gets significantly more radiation, you can improve the

conduction calculation by dividing the wall into two or more sections. The sections would then be input as separate interior walls of the same AZIMUTH and TILT, but with X, Y, Z, HEIGHT, and WIDTH chosen to give correct geometrical positioning.

Interior Solar Radiation

Part of the solar radiation entering a sunspace can be transferred directly to adjacent rooms through interior glazing, or indirectly via solar radiation absorbed by the opaque part of interior walls.

- Beam Radiation - To find the beam solar radiation falling on an inside surface, the program projects the image of each sunspace exterior window onto the surface. Summing the contribution from all the exterior windows then gives the net beam radiation incident on the surface. If the surface is an interior window, the transmission and absorption properties of the glazing are used to find the solar gain through the window into the adjacent space. If the surface is opaque, part of the absorbed radiation is conducted to the neighboring space.

- Diffuse Radiation - The diffuse solar radiation striking sunspace interior walls is also calculated. This radiation has three sources: (1) diffuse radiation from exterior windows; (2) diffuse radiation coming from beam radiation reflected from interior surfaces; and (3) diffuse radiation coming from diffuse radiation from exterior windows that reflects from interior surfaces. The diffuse irradiance inside a sunspace is assumed to be uniform.

If a shading device is present on a sunspace exterior window in a given hour, it is assumed that the radiation transmitted by the shade is totally diffuse; i.e., there is no transmitted beam component. The transmittance of the shade is assumed to be the same for direct and diffuse incident radiation and is given by SOL-TRANS-SCH. Solar transmittances for various window treatments can be obtained from the "Window Library" in *DOE-2.2 Libraries & Reports*.

The solar energy absorbed by sunspace interior walls is deducted from the sunspace load. The solar energy transmitted through sunspace interior glazing is also deducted from the sunspace load and credited to the load on adjacent rooms.

Up to 20-25% of the solar radiation entering the sunspace can be reflected back out the exterior windows. The exact percentage depends on inside surface reflectances, glazing fraction, and glass shading coefficient This loss is included in the Custom Weighting Factors for solar gain; therefore, it is accounted for in the

weighted solar load for the space. The loss is not accounted for in the ASHRAE weighting factors, so that Custom Weighting Factors, obtained by specifying FLOOR-WEIGHT = 0 (the default), should be used for sunspaces.

The program does not account for the loss of radiation entering one exterior window and leaving another exterior window without an intermediate reflection. This radiation is included in the solar gain.

Automatic Sizing of Systems Serving Conditioned Sunspaces

We recommend that the automatic HVAC system sizing feature not be used for a system serving a conditioned sunspace. This is because the peak hourly loads used by Systems for sizing do not include the contribution of solar radiation absorbed by the sunspace interior walls. (The effect of this contribution is not accounted for until the Systems hourly calculation; see "Solar Radiation Absorbed by Interior Walls," above). As a result, in the automatic sizing calculation, the Loads cooling peak for sunspaces is underestimated and the heating peak is overestimated.

For a similar reason, auto-sizing of a system serving a space adjacent to a sunspace should be avoided if there is likely to be significant conduction of heat to that space across the adjoining interior wall.

Table 15 Window Treatment and Performance

Window Treatment (1,2,3, or 4) and Fabrication/Finish/Color	Solar Characteristics[2] Transmittance %	Reflectance %	Absorptance %
Lined Drapery			
Satin/NFF3/Goldenrod--Lining: Plain/Opaque/White	15	66	19
Satin/NFF/Dk. Brown--Lining: Plain/Opaque/White	02	57	41
Satin/NFF/White--Lining: Plain/Opaque/White	18	68	14
Mali/NFF/Beige with brown accent (Lining: Plain/Translucent/Beige)	34	47	19
Unlined Satin Drapery			
Brocade/Acrylic Foam back/Beige	08	70	21
Brocade/Acrylic Foam back/Beige	10	67	24
Modified Satin/Acrylic Foam back/Beige	17	73	10
Modified Satin/Acrylic Foam back/Green	09	75	16
Modified Satin/NFF/Variegated Brown	30	51	19
Unlined Casement Drapery			
Mali/NFF/Beige	54	41	05
Mali/NFF/Variegated Brown	29	54	16
Mali/NFF/Beige	56	37	07
Mali/NFF/Beige	36	42	23
Shirred Curtains			
Plain (Ninon)/NFF/Beige	65	27	08
Plain (Ninon)/NFF/White	66	29	05
Leno (Marquisette)/NFF/White	86	14	00
Pleated Curtains			
Plain (Ninon)/NFF/Beige	27	37	37
Venetian Blinds (slats closed)			
2" steel slats/NFF/White	04	55	41
1" aluminum slats/NFF/White	02	57	41
Vertical Blinds (slats closed)			
3.5" film PVC/NFF/White	01	70	28
3.5" Plain weave/NFF/White	31	58	11
Translucent Roller Shade			
Open plain weave/vinyl-coated Fiberglas/White	48	43	09
Plain weave/vinyl-coated cotton/White	19	65	16
Opaque Roller Shade			
Plain weave/vinyl-coated cotton embossed/White	00	66	34
Plain weave/vinyl-coated layer/White	00	74	26
Plain weave/laminated embossed/White	00	75	26
Film/vinyl-coated embossed/White	15	67	18
Roll-up Shade			
Modified plain weave/vinyl tube yarn/NFF/Beige	33	53	14
Drapery Liner			
Plain weave/Acrylic coated/White	18	66	16
Plain weave/Acrylic coated/White	17	70	13
Wooden Shutter (louvers closed)			
Wood/NFF/Beige	00	63	37
Wooden Shutter Frame with Shirred Fabric			
Wood/NFF/Beige--Fabric: Ninon/NFF/White(Width: 3 x frame opening)	62	35	04
Wood/NFF/Beige--Fabric: Ninon/NFF/White(Width: 6 x frame opening)	32	51	17

1. From Solar Optical Properties of Accepted Interior Window Treatments, Eleanor Woodson, Samina Kahn, Patricia Horridge, and Richard W. Tock, ASHRAE Journal, p.40, August 1983.
2. Due to roundoff, sum of transmittance, reflectance, and absorptance may not be 100%.
3 NFF - No Functional Finish relevant to Heat Flux

Use of Multipliers

To obtain an accurate interior solar radiation calculation, we recommend that a multiplier not be used for sunspace exterior windows, interior windows or exteior walls. Also, we recommend that a multiplier not be used on a space adjacent to a sunspace.

The dangers of using multipliers are illustrated in Figure 38 and Figure 39. If the two identical exterior windows in Figure 38 are entered as a single window W-1 with MULTIPLIER = 2, no direct radiation will be calculated to fall on interior wall IW-2, whereas the radiation on IW-3 will be over-estimated by a factor of 2. The two windows should be input separately. In Figure 39 beam radiation strikes the interior window between sunspace and B, but not the one between sunspace and A. If the "identical" spaces A and B are input as A with MULTIPLIER = 2, there will be zero beam radiation transmitted to these spaces from the sunspace.

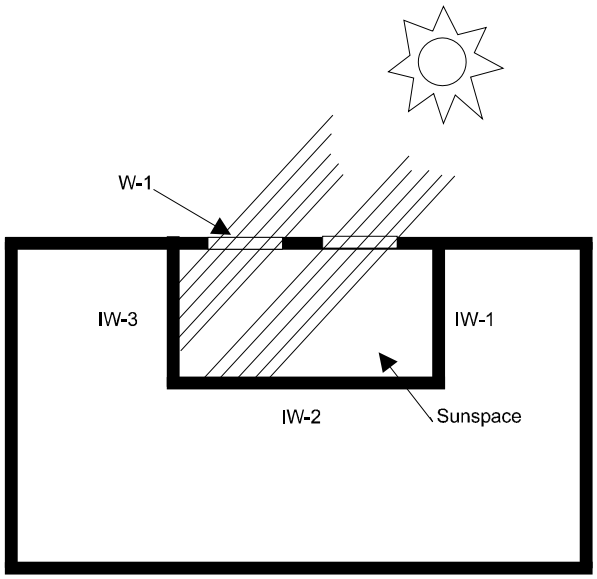


Figure 38 If the two exterior windows are input as a single window W-1 with multiplier = 2, the program will get zero beam radiation striking interior wall IW-2 and twice the actual amount striking IW-3.

If the radiation inside the sunspace is predominately *diffuse*, which would be the case if beam radiation were blocked by overhangs or window shades, the various multipliers discussed *can* be used with little loss of accuracy.

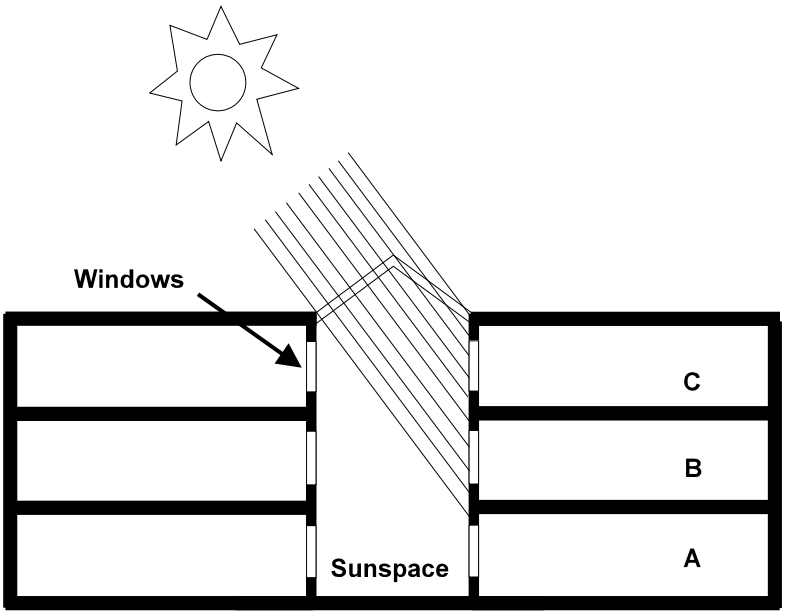


Figure 39 If spaces A and B are input as a single space A with MULTIPLIER = 2, the beam radiation transmitted through the interior windows in these spaces will be calculated to be zero.

Translucent Glazing

Translucent exterior glazing in a sunspace can be modeled with GLASS-TYPE-CODE = 1 and with SHADING-SCHEDULE values equal to the shading coefficient of the glazing. (A SHADING-SCHEDULE is used here to give a window that is diffusely transmitting.) A SOL-TRANS-SCH should also be specified, with a constant value equal to T/0.878, where T is the solar transmittance of the glazing at normal incidence. (The clear reference glazing used in the program has a solar transmittance of 0.878)

Example - Translucent Glazing

A sunspace has single-pane translucent exterior glazing with a shading coefficient of 0.71 and solar transmittance of 0.82:

```
GT-1 = GLASS-TYPE
   TYPE               = GLASS-TYPE-CODE
   GLASS-TYPE-CODE    = 1  ..

SHSCH-1 = SCHEDULE
   TYPE               = MULTIPLIER
   THRU DEC 31        (ALL) (1,24) (0.71)

SOLTRSCH-1 = SCHEDULE
   TYPE               = MULTIPLIER
   THRU DEC 31        (ALL) (1,24) 0.93)  ..
                                  $ .93 = 0.82/0.878 $
```

```
SUNSPWIN-1 = WINDOW
   GLASS-TYPE         = GT-1
   SHADING-SCHEDULE   = SHSCH-1
   SOL-TRANS-SCH      = SOLTRSCH-1
   .....
```

Atrium as Return-Air Plenum

In some commercial buildings some or all of the return air from conditioned zones is passed to a central atrium, from which it is passed back to the central air handling system or exhausted. The atrium thus behaves like a return air plenum. This arrangement can be modeled by assigning ZONE:TYPE = PLENUM to the atrium zone and including the atrium U-name in the PLENUM-NAMES list for the system.

If only part of the system return air goes to the atrium, two plenum zones can be defined, one of them being the atrium and the other being a real or dummy plenum. In a system with two plenums, the return air is split between the plenums in proportion to their floor areas, as given by the AREA keyword in the SPACE commands. Thus, if a fraction f of return air goes to the atrium, the atrium's area divided by the area of the second plenum should be f/(1-f).

If some of the return air is exhausted directly from the atrium, EXHAUST-FLOW can be specified for the atrium zone (EXHAUST-FLOW will also work for plenums that are not sunspaces).

The program accounts for the various forms of sensible and latent heat gain or loss, such as solar gain, infiltration, and moisture from people, for ZONE:TYPE = PLENUM just as it does for ZONE:TYPE = CONDITIONED. There are two important restrictions, however. The atrium as plenum cannot be mechanically cooled (although it can be vented) and it can be heated only with baseboards.

Example - Atrium as Return-Air Plenum

Two-thirds of the return air from five identical conditioned zones goes to a 10,000-ft2 (930-m2) atrium. The remaining one-third goes directly back to the air handling system.

In Loads:

```
CONDZONES = SPACE
   MULTIPLIER         = 5
   ZONE-TYPE          = CONDITIONED
   ....

ATRIUM = SPACE
   AREA               = 10000
   SUNSPACE           = YES
   ZONE-TYPE          = PLENUM
   .....

DUMPLEN = SPACE
   AREA               = 5000
   ZONE-TYPE          = PLENUM
   .....
   ..
```

In HVAC:

```
CondZone = ZONE
   TYPE              = CONDITIONED
   SPACE             = CONDZONES  .....

AtriumZone = ZONE
   TYPE              = PLENUM
   SPACE             = ATRIUM  .....

DumPlenZone = ZONE
   TYPE              = PLENUM
   SPACE             = DUMPLEN  .....
```

Heating, Cooling and Venting of Residential Sunspaces

Sunspaces in SYSTEM:TYPE = RESYS are not heated by the central system. In this case, a sunspace can be heated only with thermostatic baseboards (BASEBOARD-CTRL = THERMOSTATIC). Unlike baseboard heating of the other zones in this system, baseboard heating of a sunspace is independent of the heating requirements of the control zone.

Sunspaces in SYSTEM:TYPE = RESYS and RESVVT are not cooled by the central system. They can, however, be vented with outside air, as explained in the descriptions for the ZONE keywords SS-VENT-SCH, SS-VENT-T-SCH, etc. The venting of a sunspace in these systems is independent of the natural ventilation of the other zones as determined by NATURAL-VENT-SCH, etc., in SYSTEM.

Hourly Reports Variables and Error Messages

Hourly Report Variables for Sunspace Analysis

The following hourly report variables are useful for sunspace analysis (see "HOURLY-REPORT" in *DOE-2.2 Libraries & Reports*).

In `Loads`, VARIABLE-TYPE = U-name of WINDOW:

- Exterior and interior window variables.

In `Systems`, VARIABLE-TYPE = U-name of ZONE:

- #56 For zone adjacent to a sunspace: total interior window heat gain due to solar radiation from sunspace.
- #57 For zone adjacent to a sunspace: total interior window load due to solar radiation from sunspace.
- #58 For sunspace or zone adjacent to a sunspace: heat gain by conduction through interior windows, calculated with the air temperature of the zone fixed at the Loads calculation temperature and actual previous-hour temperatures for adjacent zones.
- #59 For sunspace or zone adjacent to a sunspace: solar radiation absorbed on the sunspace side (opaque part) of interior walls.

- #60 For sunspace or zone adjacent to a sunspace: heat gain by conduction through opaque part of interior walls, calculated with the air temperature of the zone fixed at the Loads calculation temperature and actual previous-hour temperatures for adjacent zones.

Error, Caution, and Warning Messages

In the following, sunspace means a SPACE with SUNSPACE = YES and non-sunspace means a SPACE with SUNSPACE = NO (the default).

Error Message (1)	INTERIOR-WALL <U-name>, WHICH IS BETWEEN A SUNSPACE AND A NON-SUNSPACE, HAS AREA SPECIFIED RATHER THAN HEIGHT AND WIDTH. HEIGHT AND WIDTH ARE REQUIRED FOR CALCULATION OF SOLAR RADIATION ABSORBED ON THE SUNSPACE SIDE OF THIS WALL.
Meaning:	Self-explanatory.
User-Action:	Specify height and width for this wall.

Error Message (2)	EXTERIOR-WALL <U-name>, IN SUNSPACE <U-name>, HAS A MULTIPLIER OF <value>. THE MULTIPLIER ON AN EXTERIOR-WALL (WITH WINDOWS) IN A SUNSPACE SHOULD BE 1.0.
Meaning:	A sunspace has an exterior wall with a multiplier different from 1.0. Since this wall has one or more windows, the use of a multiplier will give an inaccurate calculation of the interior solar radiation distribution from these windows.
User-Action:	Do not use a multiplier on sunspace exterior walls.

Error Message (3)	SPACE <U-name> HAS <value> SUNSPACE COMMON WALLS WITH CONVECTIVE HEAT TRANSFER (AIR-FLOW-TYPE = FORCED-RECIRC, FORCED-OA-PREHT, FREE-RECIRC, OR FREE-DOORWAY). AT MOST ONE COMMON WALL WITH CONVECTIVE TRANSFER IS ALLOWED IN A SPACE.
Meaning:	A space cannot have more than one interior wall across which convective flow is specified using the AIR-FLOW-TYPE keyword in the WALL-PARAMETERS command.
User-Action:	Reduce number of interior walls with convection to one.

Warning Message (1)	WINDOW <U-name> ON INTERIOR WALL <U-name> HAS X=0, Y=0 AND THEREFORE HAS PROBABLY NOT BEEN CORRECTLY POSITIONED. THIS MAY CAUSE AN INACCURATE SOLAR RADIATION TRANSMISSION CALCULATION.
Meaning:	You have probably forgotten to geometrically position the interior window.
User-Action:	Specify X, Y, HEIGHT and WIDTH for the WINDOW. See Fig. 2.2 and "Positioning of Sunspace Surfaces."

Warning Message (2)	<U-name> IS AN INTERIOR-WALL BETWEEN SUNSPACE <U-name> AND SPACE <U-name>. SINCE THE INTERIOR-WALL WAS DEFINED IN <U-name> IT IS IMPORTANT THAT THIS SPACE BE CORRECTLY POSITIONED WITH RESPECT TO THE SUNSPACE TO OBTAIN AN ACCURATE CALCULATION OF SOLAR RADIATION INCIDENT ON THE WALL FROM EXTERIOR WINDOWS IN THE SUNSPACE.
Meaning:	A sunspace interior wall was defined in the adjacent space rather than in the sunspace.
User-Action:	Be sure that the space in which the interior wall was defined is geometrically positioned with respect to the sunspace. Alternatively, define the interior wall in the sunspace.

Warning Message (3)	SPACE <U-name>, WHICH IS NEXT TO SUNSPACE <U-name> HAS MULTIPLIER <value>. A MULTIPLIER DIFFERENT FROM 1.0 MAY CAUSE AN INACCURATE CALCULATION OF HEAT TRANSFER FROM THE SUNSPACE.
Meaning:	The use of a multiplier on a space adjacent to a sunspace multiplies the common

	interior wall. This may give an incorrect calculation of the total solar radiation absorbed by the wall and transmitted by windows in the wall.
User-Action:	See "Use of Multipliers."

Warning Message (4)	WINDOW <U-name> IN INTERIOR-WALL <U-name> HAS MULTIPLIER <value>. A MULTIPLIER DIFFERENT FROM 1.0 MAY CAUSE AN INACCURATE SOLAR RADIATION TRANSMISSION CALCULATION.
Meaning:	The location of sunspace interior glazing is important in the calculation of the amount of solar radiation striking the glazing.
User-Action:	Do not use a multiplier. Input windows separately.

Caution Message (1)	SUNSPACE INTERIOR WALL <U-name> HAS X=0, Y=0 AND THEREFORE MAY NOT BE CORRECTLY POSITIONED. THIS MAY CAUSE AN INACCURATE CALCULATION OF SOLAR RADIATION ABSORBED BY THE WALL.
Meaning:	You have probably forgotten to geometrically position a sunspace interior wall.
User-Action:	Specify X, Y, Z, AZIMUTH, TILT, HEIGHT, and WIDTH

Caution Message (2)	WINDOW <U-name> IN SUNSPACE EXTERIOR-WALL <U-name> HAS MULTIPLIER <value>. A MULTIPLIER DIFFERENT FROM 1.0 MAY CAUSE AN INACCURATE CALCULATION OF THE AMOUNT OF SOLAR RADIATION FROM THIS WINDOW WHICH STRIKES THE INTERIOR WALLS OF THE SUNSPACE.
Meaning:	The geometrical position of a sunspace exterior WINDOW is important in the interior solar radiation calculation.
User-Action:	Do not use multiplier; input windows separately.

Caution Message (3)	WINDOW <U-name> IS IN INTERIOR WALL <U-name> WITH TYPE= AIR, ADIABATIC, or INTERNAL. THIS WINDOW WILL BE IGNORED.
Meaning:	The program calculates heat transfer through interior windows only if they are in an interior wall with INT-WALL-TYPE = STANDARD (the default) and this interior wall is between a sunspace and a non-sunspace. In all other cases interior walls are considered as being without windows.
User-Action:	Remove window from wall, or change INT-WALL-TYPE to STANDARD if heat transfer calculation across the wall is desired.

Caution Message (4)	WINDOW <U-name> IS IN INTERIOR-WALL <U-name> BETWEEN TWO SUNSPACES. THIS WINDOW WILL BE IGNORED.
Meaning:	The program calculates heat transfer through interior windows only if they are in an interior wall with INT-WALL-TYPE = STANDARD (the default) and this INTERIOR-WALL is between a sunspace and a non-sunspace. In all other cases, INTERIOR-WALLs are considered as being without WINDOWS.
User-Action:	Check whether the spaces on either side of this wall should both be sunspaces. If not, assign SUNSPACE = NO to one of them. Otherwise, remove WINDOW from wall.

Caution Message (5)	WINDOW <U-name> IS IN INTERIOR-WALL <U-name> BETWEEN TWO NON-SUNSPACES. THIS WINDOW WILL BE IGNORED. (HEAT TRANSFER WILL BE CALCULATED ONLY FOR WINDOWS

	IN A STANDARD-TYPE INTERIOR WALL BETWEEN A SUNSPACE AND A NON-SUNSPACE.)
Meaning:	The program calculates heat transfer through interior windows only if they are in an INTERIOR-WALL with INT-WALL-TYPE = STANDARD (the default) and this INTERIOR-WALL is between a sunspace and a non-sunspace. In all other cases, INTERIOR-WALLs are considered as being without WINDOWS.
User-Action:	Check whether the spaces separated by this wall should both be non-sunspaces. If not, assign SUNSPACE = YES to one of them. Otherwise, remove WINDOW from wall.

Section

4

Air-side System Types

Introduction

The HVAC sub-program models the performance of the building's heating, cooling, and ventilation systems. These systems are divided into air systems, which consist of fans, ducts, dampers, coils, etc., that provide conditioned air and ventilation to the building's spaces, and primary systems, which consist of boilers, chillers, cooling towers, storage tanks, generators, etc., that provide hot water, chilled water, and electricity to the distribution systems. Air systems are also called "secondary systems," "air distribution systems," "distribution systems," "fan systems," and often just "systems." The primary system is also called the "plant." In larger buildings the primary and secondary systems are connected by circulation loops that bring hot or cold water from the boilers and chillers to the heating and cooling coils. In DOE-2.1E and previous versions, these loops were not explicitly modeled.

The program requires a fair amount of understanding of how HVAC systems operate. A general description of types of systems is given here. Once you have understood the structure of the LOADS input, there should be little difficulty in learning to assemble an HVAC input. The major problem most users have is that the program offers a high degree of flexibility and a large choice of options for HVAC input. In some programs you can simply assign the name of the desired system and the program will pull from its file all of the necessary input. To a degree this can be done by relying on default values and pre-stored control methods. However, this is not the recommended procedure and is an option to be used only until the you feel comfortable with explicitly specifying the many HVAC commands and keywords.

SYSTEM TYPE = SZRH

Single-zone with Subzone Reheat

This is the most commonly applied unit in the industry. It is usually supplied with an outside/return air damper economizer mixing box, air filters, heating coil placed ahead of a cooling coil, and a supply air fan. If the unit is very large (>20,000 FLOW) there may be a return air fan. The coils are usually water coils; however, the heating coil may be electric resistance or furnace. These units are seldom rooftop units since water coils are subject to freezing. Subzone reheat is also seldom used but is an option.

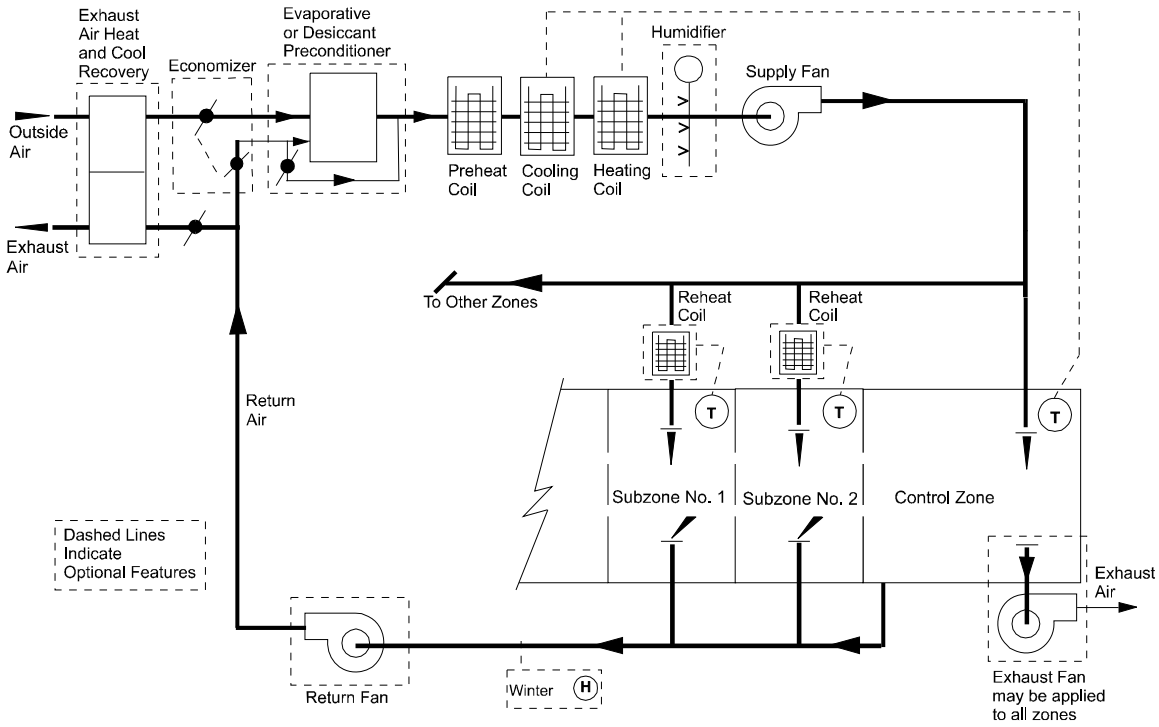


Figure 40 Single-zone Fan System with Optional Subzone Reheat

Other names and/or applications for SZRH

- Variable Temperature-Constant-volume units
- School Unit Ventilators (heating and cooling)
- School Auditorium Units
- School Gymnasium Units (cooling is often locked out using COOLING-SCHEDULE with hourly values set to zero)
- Conference and Meeting Room Units where humidity control is unimportant
- Make-Up Air Units to supply hotel/motel guest rooms with ventilation air

- Heating and Ventilating Units. Again, cooling is locked out but outside cooling is active. These units may also be direct-fired (gas) with products of combustion going into the supply air. If so, set FURNACE-HIR = 1.0

Standard Temperature Controls

If you read SZRH as Single Control Zone with Subzone Reheat the unit controls are easy to understand. In your input, CONTROL-ZONE = U-NAME designates the control zone and all other zones ride with the heating and cooling demands of this zone. Subzones can have reheat coils to prevent overcooling but if subzones are under cooled, the room temperatures must suffer. This is very much like a residential furnace and AC unit with the thermostat (control zone) in the front hallway.

The description of the control of a SZRH unit, starting with the unit on full heating, is as follows:

- On a rise in space temperature of the control zone, heating valves modulate closed.
- On a further rise the outside air economizer dampers modulate open, provided the system has an economizer and outside air can provide cooling.
- On a further rise, the cooling coil valves modulate open. .

Input template for a standard SZRH unit with water coils

Under SYSTEM with suggested values

```
U-name = SYSTEM
   TYPE              = SZRH
   MAX-SUPPLY-T      = 105
   MIN-SUPPLY-T      = 55
   OA-CONTROL        = TEMP
   ECONO-LIMIT-T     = 70
   SUPPLY-STATIC     = 3                 inches total static
   SUPPLY-EFF        = 0.65              overall fan eff
   RETURN-STATIC     = 0.75              if a return fan
   RETURN-EFF        = 0.65              if a return fan
   FAN-SCHEDULE      = U-NAME            fan run times
   NIGHT-CYCLE-CTRL  = CYCLE-ON-ANY      if fans cycle on to
                                         hold night setpoint
   REHEAT-DELTA-T    = 50                if subzone reheat
   CONTROL-ZONE      = U-NAME            of primary zone
   HEAT-SOURCE       = HOT-WATER         if a boiler
   HW-LOOP           = U-NAME            of HW loop
   CHW-LOOP          = U-NAME            of CHW loop
```

Under ZONE with suggested values

```
U-NAME = ZONE                                  different than SPACE
   TYPE              = CONDITIONED
   DESIGN-HEAT-T     = 68
   DESIGN-COOL-T     = 75
   HEAT-TEMP-SCH     = U-NAME                  thermostat heat sch
   COOL-TEMP-SCH     = U-NAME                  thermostat cool sch
   OA-FLOW/PER       = 15
   SPACE             = U-NAME                  of corresponding
                                               SPACE in LOADS module
```

Changes or additions when using SZRH for other applications

School Classroom Unit Ventilators

usually have code requirements for outside ventilation air and minimum air changes. AIR-CHANGES/HR = 6 and OA-FLOW/PER = 15 are common values. A group of six classrooms on a single floor having similar usage schedules and located on a single exposure may be represented as one classroom with a MULTIPLIER = 6. If there is a second floor of six classrooms like those on the first floor, input them separately so as not to miss the roof load. In other words, don't use the keyword FLOOR-MULTIPLIER. The program has a UNIT-VENTILATOR system type but it only has a heating capability.

School Auditorium Units

Are usually freestanding without supply ducts but they do have sound attenuators. Adjust the supply static accordingly.

School Gymnasium and/or Cafeteria Units

may also be freestanding but without sound deadening. Adjust the supply static accordingly.

Conference and Meeting Rooms

may have humidifiers. If so, use MIN-HUMIDITY = ~25 and input correct type of HUMIDIFIER. The SZRH unit is not a good choice for MAX-HUMIDITY control. See PSZ and RHFS units to satisfy this requirement.

Make-up Air Units to Hotel/Motel Guest Rooms

can be simulated. Input an SZRH unit with neutral supply air temperatures and apply the keyword OA-FROM-SYSTEM = to the system (like fan coils) that receives the ventilation air. For motels with indoor swimming pools the Add-On Desiccant Unit may be used to simulate a better job of moisture control.

Heating and Ventilating Units with Active Outside Air Cooling

may be simulated using SZRH. Do not confuse SZRH with HVSYS, which is a central ventilation system for schools sold primarily by Columbus Heating and Ventilation in the eastern and middle United States during the early 1900s to 1950s (prior to the accepted practice of air conditioning schools).

Covered in detail by separate Topics:

- Night Ventilation (SYSTEM command)
- Add On Evaporative Cooling (SYSTEM command)
- Add On (Integrated) Desiccant Cooling (SYSTEM command)

- Service Hot Water heat Pump (SYSTEM and CIRCULATION-LOOP commands)
- Electric and Fuel Meters (SYSTEM, ZONE, ELEC-METER and FUEL-METER commands)
- Building Resources (CIRCULATION-LOOP command)
- Optimum Fan Start
- Heat and Cool Recovery of Relief Air
- Baseboard or Fin Tube Radiation

SYSTEM TYPE = PSZ

Packaged Single-zone with Subzone Reheat

This system and its companion (SZRH) make up more than half of all air conditioning units used by the industry. It is usually supplied with an outside/return air damped economizer mixing box, air filters, a gas or oil furnace, a DX cooling coil, a supply fan, refrigeration compressor and air cooled condenser. If the unit is very large (~30,000 FLOW) there may be a return fan. The cooling coil is direct expansion and the heating may be electric resistance, steam, or furnished by an air to air heat pump. These units are often rooftop units as there is little danger of freezing anything. Heat recovery between relief air (must use return fan) and outside air is another popular option as is heat recovery of condenser heat for reheat thus providing maximum humidity control. Subzone reheat is also seldom used but is an option. The reason that the PSZ can be used for humidity control of the control zone is that the location of the condenser coil is downstream of the cooling coil; this is not true of SZRH.

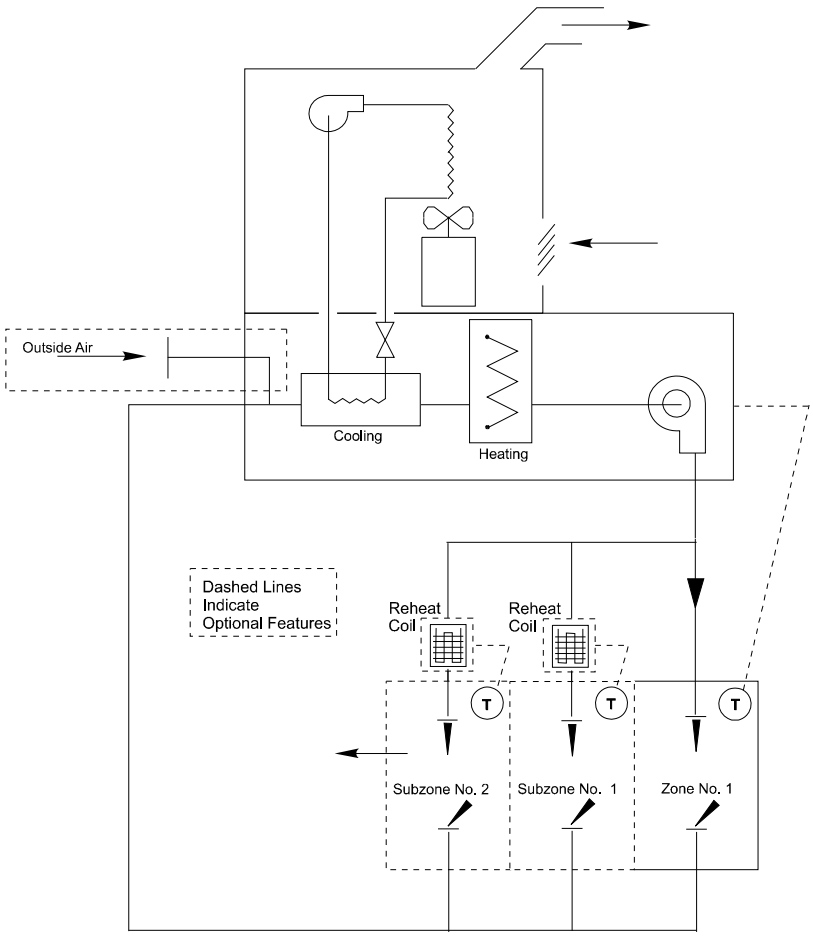


Figure 41 Packaged Single-zone Air Conditioner with Heating and Subzone Reheating Options

Other names and/or applications for PSZ

- Split system variable temperature constant-volume units

- Rooftop variable temperature-constant-volume units
- Self-Contained School Unit Ventilators
- Self-Contained School Auditorium Units
- Self-Contained School Gymnasium Units when cooling is provided
- Computer, Conference, and Meeting Room Units when humidity control is important
- Make-Up Air Units to supply; hotel/motel guest rooms with ventilation air
- Swimming pool units for humidity control
- Heat Recovery from Refrigerated Casework in Food Stores (see topic Refrigerated Casework)
- Heat Recovery from Ice Rink Refrigeration Units

Standard Temperature Controls

If you read PSZ as Packaged Single Control Zone with Subzone Reheat the unit controls are easy to understand. In your input the CONTROL-ZONE = U-NAME designates the control zone and all other zones ride with the heating/cooling demands of the control zone. Subzones can have reheat coils (often electric if the main unit is a furnace) to prevent overcooling but if subzones are undercooled, the room temperatures must suffer. The description of the control of a PSZ unit, starting with the unit on full heating, is as follows:

- On a rise in space temperature of the control zone, the furnace burners are stepped down to off.
- On a further rise the outside air economizer dampers modulate open, provided the system has an economizer and outside air can provide cooling.
- On a further rise the refrigeration compressors are stepped upward to increase the DX cooling output.

The program treats stepped control as fully modulated control and makes no attempt to differentiate between a modulated burner multiple furnace sections and for cooling the use of multiple compressors versus a single compressor furnished with unloading cylinders.

Input template for a standard PSZ with furnace

Under SYSTEM with suggested values

```
U-name = SYSTEM
   TYPE              = PSZ
   MAX-SUPPLY-T      = 105
   MIN-SUPPLY-T      = 55
   OA-CONTROL        = TEMP
   ECONO-LIMIT-T     = 70
   SUPPLY-STATIC     = 3                  inches total static
   SUPPLY-EFF        = 0.65               overall fan eff
   RETURN-STATIC     = 0.75               if a return fan
   RETURN-EFF        = 0.65               if a return fan
   FAN-SCHEDULE      = U-NAME             fan run times
   NIGHT-CYCLE-CTRL  = CYCLE-ON-ANY       if fans cycle on to
                                          hold night setpoint
   REHEAT-DELTA-T    = 50                 if subzone reheat
   CONTROL-ZONE      = U-NAME             of primary zone
   HEAT-SOURCE       = FURNACE            note that HEAT-PUMP
                                          is not a legitimate
                                          choice when humidity
                                          control is required
   ZONE-HEAT-SOURCE  = ELECTRIC
   ..
```

Under ZONE with suggested values

```
U-NAME = ZONE                           different than SPACE
   TYPE              = CONDITIONED
   DESIGN-HEAT-T     = 68
   DESIGN-COOL-T     = 75
   HEAT-TEMP-SCH     = U-NAME           thermostat heat sch
   COOL-TEMP-SCH     = U-NAME           thermostat cool sch
   OA-FLOW/PER       = 15
   SPACE             = U-NAME           of corresponding
   ..                                   SPACE in LOADS module
```

Changes or additions when using PSZ for other applications

School Classroom Unit Ventilators

usually have code requirements for outside ventilation air and minimum air changes. AIR-CHANGES/HR = 6 and OA-FLOW/PER = 25 are common values. A group of six classrooms on a single floor having similar usage schedules and located on a single exposure may be represented as one classroom with a MULTIPLIER = 6. If there is a second floor of six classrooms like those on the first floor, input them separately so as not to miss the roof load. In other words, don't use the keyword FLOOR-MULTIPLIER. The program has a UNIT-VENTILATOR system type but it only has a heating capability.

School Auditorium Units

usually are freestanding without supply ducts but they do have sound attenuators. Adjust the supply static accordingly. To control humidity, input MAX-HUMIDITY = 50, MAX-CONDENSER-RCVRY = 0.6 and REHEAT-DELTA-T = 5.

School Gymnasium and/or Cafeteria Units

may also be freestanding but without sound deadening. Adjust the supply static accordingly.

Computer, Conference, and Meeting Rooms

may have humidifiers. If so, use MIN-HUMIDITY = ~25 and input correct type of HUMIDIFIER. To control humidity, input MAX-HUMIDITY = 50, MAX-CONDENSER-RCVRY = 0.6 and REHEAT-DELTA-T = 50.

Make-up Air Units to Hotel/Motel Guest Rooms

can be simulated. Input a PSZ unit with neutral supply air temperatures and apply the keyword OA-FROM-SYSTEM = to the system (like fan coils) that receives the ventilation air. For motels with indoor swimming pools the Add-On Desiccant Unit may be used to simulate a better job of moisture control; however, most use PSZ with condenser reheat.

Covered in detail by separate topics:

- Refrigerated Casework
- Defrost Controls for Heat Pumps
- Air and Water-side Economizers
- Air and Water Cooled Condensers
- Night Ventilation (SYSTEM command)
- Add-On Evaporative Cooling (SYSTEM command)
- Add-On (Integrated) Desiccant Cooling (SYSTEM command)
- Service Hot Water Heat Pump (SYSTEM and CIRCULATION-LOOP commands)
- Electric and Fuel Meters (SYSTEM, ZONE, ELEC-METER and FUEL-METER commands)
- Building Resources (CIRCULATION-LOOP command)
- Optimum Fan Start
- Heat Recovery of Relief Air
- Natural Ventilation
- Baseboard or Fin Tube Radiation
- Primary/Secondary Pumping

SYSTEM TYPE = SZCI

Single Zone Ceiling Induction

This system had its origin during the 1960s during the design period of very high lighting levels (4. to 5. w/sqft). Some buildings were built without a heating system and relied on the lights, which could be left on at night, to heat the building. This is hard to believe in this age, but electric rates of one cent per kilowatt hour for all-electric buildings was an attractive incentive! The concept of the system was to pull air back through the lights and raise the return plenum temperature. The ceiling VAV induction boxes reduced air flow using a damper which, as it closed, took the shape of a nozzle which induced warm plenum air, mixed with supply air, and supplied it back into the conditioned space. The minimum closure (MIN-FLOW-RATIO) of the nozzle was 0.5 and not adjustable. The system is usually supplied with an outside/return air damper economizer mixing box, air filters, chilled water cooling coil, and supply fan. A return fan is often required to stabilize the outside air intake volume. In very cold climates, an electric preheat coil is used to prevent freeze-up of the chilled water coil. Some of the induction boxes have electric reheat coils but the added resistance of it and any duct work and diffusers tend to reduce the ability to induce air. When humidity control is required an electric main heating coil is furnished.

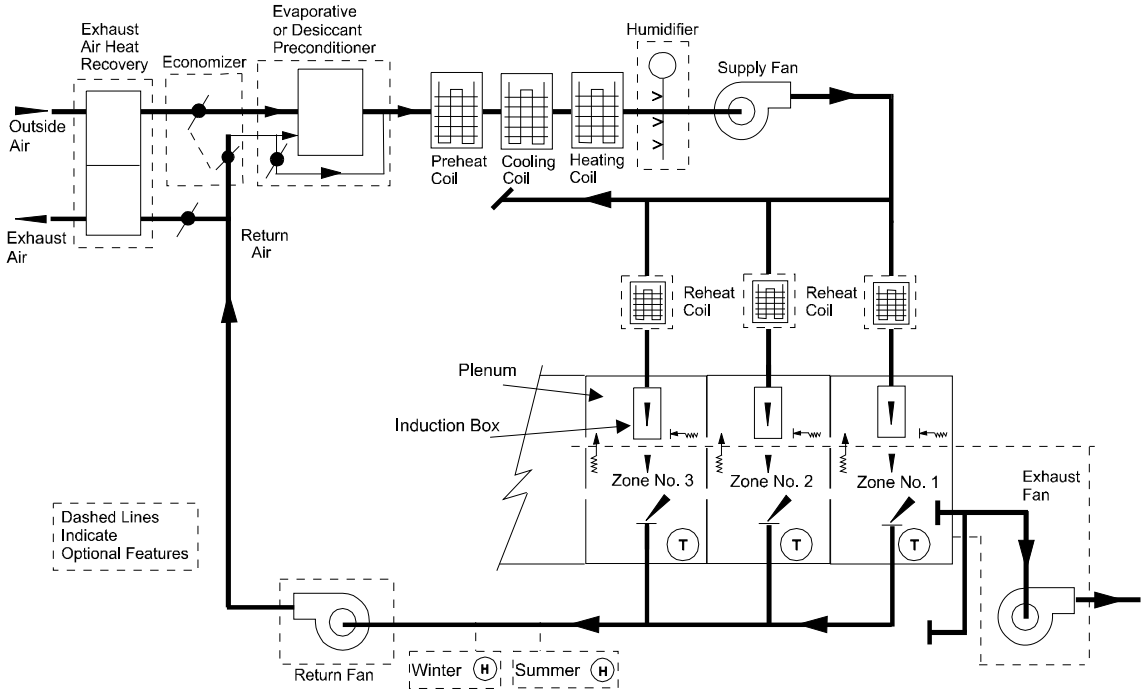


Figure 42 Ceiling Induction System

Other names and/or applications for SZCI

- Heat-of-Light system
- Most all of these systems were installed in office buildings, as they were the buildings with high lighting levels and all electric.

- A retrofit for these systems is to replace the induction boxes with powered induction boxes which allow the MIN-FLOW-RATIO to reduce to 0.0 if desired. At the same time, the problems of downstream resistance of ductwork and reheat coils is overcome.

- A warm-up cycle similar to that described for VAVS can remove the necessity of leaving lights on to heat the building. The electric main heating coil and preheat coil can be replaced with water coils supplied by a gas fired boiler.

Standard Temperature Controls

Normally, the supply air temperature is held CONSTANT as this insures the moisture content leaving the cooling coil meets design specifications.

- Starting with the unit operating during cold weather, on a small rise in supply air temperature sensed by the supply air controller, the economizer modulates open to outside air, provided the system has an economizer and outside air can provide cooling..

- On a further rise, the cooling coil modulates to full open and the space thermostats modulate the induction nozzles which induce ceiling air and return it to the space.

Input template for a standard SZCI unit

Under SYSTEM with suggested values

```
U-name = SYSTEM
   TYPE              = SZCI
   MAX-SUPPLY-T      = 105
   MIN-SUPPLY-T      = 55
   COOL-CONTROL      = CONSTANT
   HEAT-SET-T        = 90                  to enable a main
                                           heating coil
   OA-CONTROL        = TEMP
   ECONO-LIMIT-T     = 70
   SUPPLY-STATIC     = 5                   inches total static
   SUPPLY-EFF        = 0.65                overall fan eff
   RETURN-STATIC     = 0.75                if a return fan
   RETURN-EFF        = 0.65                if a return fan
   FAN-SCHEDULE      = U-NAME              fan run times
   NIGHT-CYCLE-CTRL  = CYCLE-ON-ANY        if fans cycle on to
                                           hold night setpoint
   REHEAT-DELTA-T    = 50                  if subzone reheat
   HEAT-SOURCE       = HOT-WATER           if a boiler
   HW-LOOP           = U-NAME              of HW loop
   CHW-LOOP          = U-NAME              of CHW loop
   ..
```

Under ZONE with suggested values

```
U-NAME = ZONE                                  different than SPACE
   TYPE              = CONDITIONED
   DESIGN-HEAT-T     = 68
   DESIGN-COOL-T     = 75
   HEAT-TEMP-SCH     = U-NAME                  thermostat heat sch
   COOL-TEMP-SCH     = U-NAME                  thermostat cool sch
   OA-FLOW/PER       = 15
   SPACE             = U-NAME                  of corresponding
   ..                                          SPACE in LOADS module
```

Changes or additions when using SZCI for other applications

- As an option to COOL-CONTROL = CONSTANT change to WARMEST or RESET.
- MAX-HUMIDITY can be partially controlled by lowering the supply air temperature when WARMEST or RESET is used.
- MIN-HUMIDITY can be controlled by the addition of moisture to supply air.
- Powered Induction boxes can replace the Ceiling Induction boxes.

Covered in detail by separate Topics are the following:

- Night Ventilation (SYSTEM command)
- Add On Evaporative Cooling (SYSTEM command)
- Add On (Integrated) Desiccant Cooling (SYSTEM command)
- Service Hot Water heat Pump (SYSTEM and CIRCULATION-LOOP commands)
- Electric and Fuel Meters (SYSTEM, ZONE, ELEC-METER and FUEL-METER commands)
- Building Resources (CIRCULATION-LOOP command)
- Optimum Fan Start
- Heat Recovery of Relief Air

SYSTEM TYPE = VAVS

Variable Air Volume System

This system concept was not accepted by the industry for many years, with the first systems installed in the 1970s. The delay was threefold: the noise made by volume reduction dampers, the buildup of air pressure resulting from some dampers closing and others remaining open, and resistance by code officials who felt the system was deficient and caused low recirculation and low ventilation rates especially in schools and other public buildings. When these problems got sorted out the VAV concept really took off and soon became the most popular system, as it continues to be today. There are many variations, notably PVAVS, SZCI, PIU, CBVAV, DDVAV, DFDDVAV, PVVT, and RESVVT.

The VAVS is usually supplied with outside/return damper air economizer damper mixing box, air filters, a preheat coil when necessary to protect the chilled water coil from freezing, a reheat coil for humidity control, a humidifier and a variable flow supply air fan (using motor speed, inlet vanes, or discharge vanes to control flow), and normally a return fan to stabilize the negative pressure in the mixing box. The air distribution system was often constructed with high velocity ducts and, therefore, the supply fan total system pressure requirements sometimes approached 10". More modern designs typically have supply pressures in the range of 4-6" TSP.

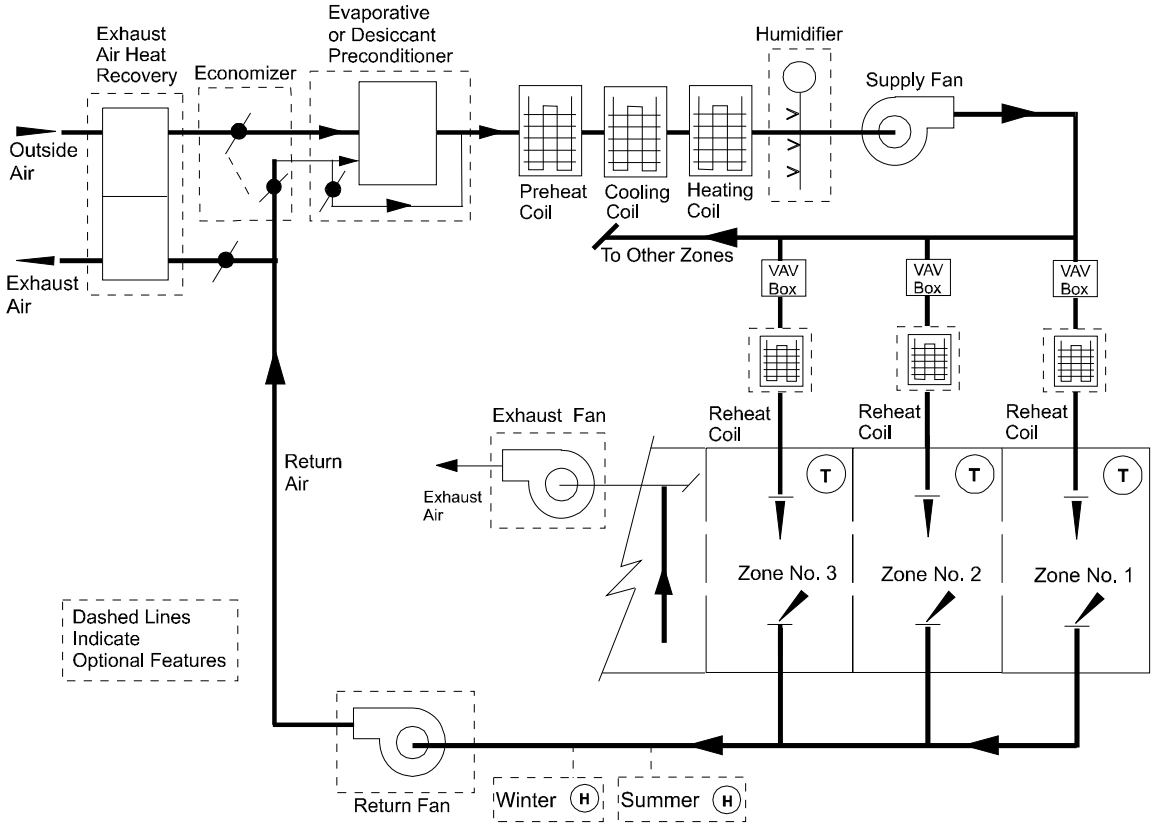


Figure 43 Variable-air volume system

Other names (none) and/or control applications for VAVS

- To simulate a laboratory system that operates at 100 percent outside air during daytime but drops to 50 percent flow rate at night to accommodate fume hoods, set MIN-FLOW-RATIO =1.0 during the day and input a MIN-FLOW-SCH = U-NAME of a SCHEDULE with hourly values of 0.5 at night and weekends and (-999) during daytime hours.

- Set MIN-HUMIDITY = 40 for laboratories to add moisture.

- Set MIN-HUMIDITY = 60 for public buildings but only if the supply air temperature is allowed to be reset. The VAVS system is not a good choice for buildings where high humidity must be controlled.

- Set COOL-CONTROL = WARMEST to reset the supply air temperature upward as loads permit and reduce both cooling energy and reheat energy. HEAT-SET-T is also required to enable the main heating coil.

- Set RECOVERY-EFF = 0.60 to simulate an exchange of sensible heat between outside air and relief air. Specify RETURN-FAN-LOC = RELIEF to simulate an exhaust fan.

Standard Temperature Controls

Normally, the supply air temperature is held CONSTANT as this insures that moisture removal leaving the cooling coil meets design specifications.

- Starting with the unit operating during cold weather, on a small rise in supply air temperature sensed by the supply air controllers the economizer modulates open, provided the system has an economizer and outside air can provide cooling.

- On a further rise the cooling coil modulates to full open (and the economizer closes once outdoor conditions are unfavorable). The preheat coil operates independently of the supply air controller to prevent freeze-up of the cooling coil.

- The main handling unit reheat coil holds the supply temperature if the air leaving the cooling coil is colder than desired due to a MAX-HUMIDITY requirement.

- MIN-HUMIDITY is controlled by adding moisture to the supply air which can occur during winter when outside moisture loads are very low.

- For each zone, as the zone temperature drops below the cooling setpoint, the VAV dampers modulate closed until they are at their MIN-FLOW-RATIO. As the zone temperature approaches the heating setpoint, the reheat coil, if available, will begin to modulate open to prevent the zone temperature from dropping any further. For REVERSE-ACTION thermostats, the VAV damper will also modulate back open during heating. In addition to increasing heating capacity, this control sequence can help to reduce the thermal stratification that could otherwise result when a small quantity of heated air is introduced at the ceiling level.

Input template for a standard VAVS unit

Under SYSTEM with suggested values

```
U-name = SYSTEM
   TYPE              = VAVS
   MAX-SUPPLY-T      = 105
   MIN-SUPPLY-T      = 55
   COOL-CONTROL      = CONSTANT
   HEAT-SET-T        = 90
   OA-CONTROL        = TEMP
   ECONO-LIMIT-T     = 70
   SUPPLY-STATIC     = 6                  inches total static
   SUPPLY-EFF        = 0.65               overall fan eff
   RETURN-STATIC     = 0.75               if a return fan
   RETURN-EFF        = 0.65               if a return fan
   FAN-SCHEDULE      = U-NAME             fan run times
   FAN-CONTROL       = INLET
   NIGHT-CYCLE-CTRL  = CYCLE-ON-ANY       if fans cycle on to
                                          hold night setpoint
   MIN-FLOW-RATIO    =0.30
   REHEAT-DELTA-T    = 50                 if subzone reheat
   HEAT-SOURCE       = HOT-WATER          if a boiler
   HW-LOOP           = U-NAME             of HW loop
   CHW-LOOP          = U-NAME             of CHW loop
   ..
```

Under ZONE with suggested values

```
U-NAME = ZONE                           different than SPACE
   TYPE              = CONDITIONED
   DESIGN-HEAT-T     = 68
   DESIGN-COOL-T     = 75
   HEAT-TEMP-SCH     = U-NAME           thermostat heat sch
   COOL-TEMP-SCH     = U-NAME           thermostat cool sch
   THERMOSTAT-TYPE   = REVERSE-ACTION   re-opens during heat
   OA-FLOW/PER       = 15
   SPACE             = U-NAME           of corresponding
   ..                                   SPACE in LOADS module
```

Covered in detail by separate Topics:

- Night Ventilation (SYSTEM command)
- Add On Evaporative Cooling (SYSTEM command)
- Add On (Integrated) Desiccant Cooling (SYSTEM command)
- Service Hot Water Heat Pump (SYSTEM and CIRCULATION-LOOP commands)
- Electric and Fuel Meters (Metering topic; SYSTEM, ZONE, ELEC-METER and FUEL-METER commands)
- Building Resources (CIRCULATION-LOOP command)
- Optimum Fan Start (SYSTEM command)
- Heat Recovery from Relief Air

- Baseboard or Fin Tube Radiation (SYSTEM command)

SYSTEM TYPE = PIU

Powered Induction Unit

This system is an outgrowth of the SZCI system and overcomes the problem associated with both SZCI and VAVS. The concept of the system is to pull air back through the lights using a fan box to raise the return plenum temperature. In a building the favored sources of warm return air are the core zones which are modeled as standard VAV (SVAV); at least one core zone must be specified for the simulation, even if it is simply a corridor. The PIU boxes may be specified as series or parallel (see figures below) and both types reduce the primary air from the main air-handling unit but use a PIU fan to induce warm ceiling plenum air which, when mixed together, provides a near-constant recirculated flow in the conditioned space. The MIN-FLOW-RATIO may be set at any desired point. The system is usually supplied with an outside/return air damper economizer mixing box, air filters, preheated coil in cold climates, chilled water cooling coil, a main heating coil, and supply fan. A return fan is often required to pull the return air back and to stabilize the outside air intake volume. The powered induction boxes are furnished with either electric or hot water coils and the fan in the box can affect the resistance of supply ductwork and air diffusers.

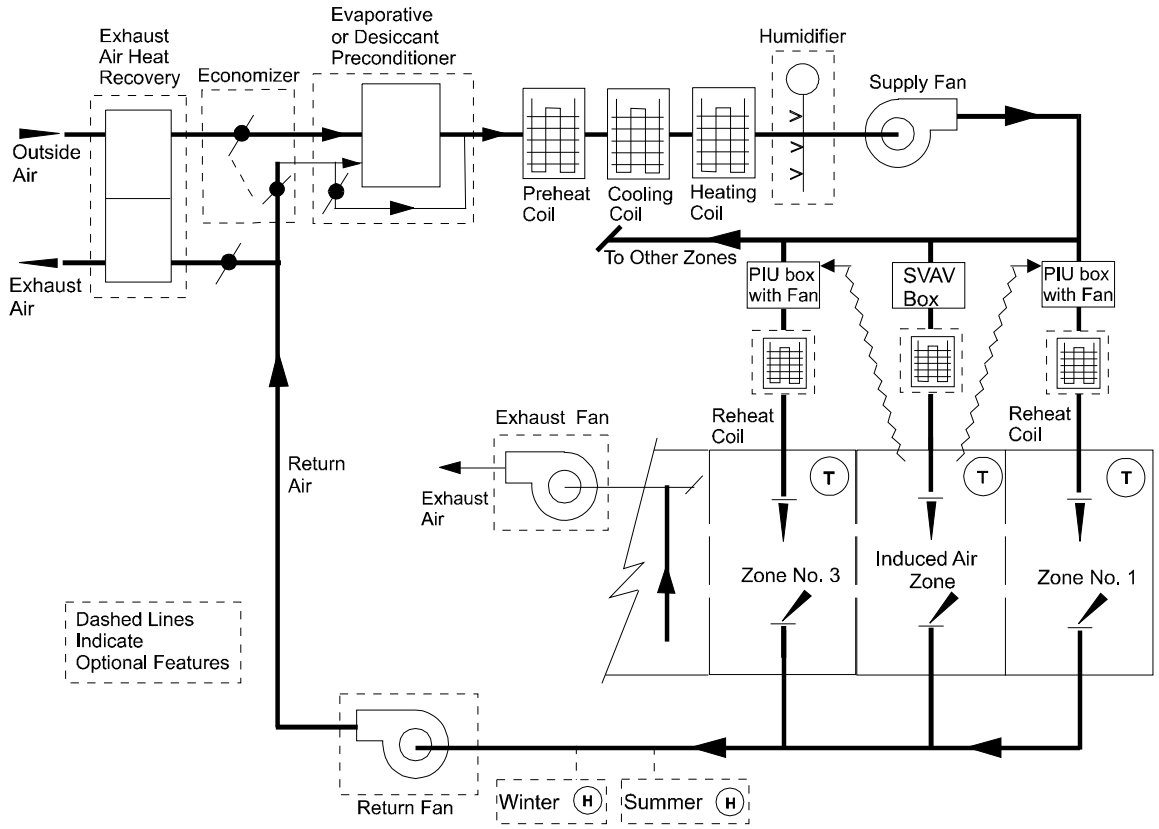


Figure 44 Powered induction unit

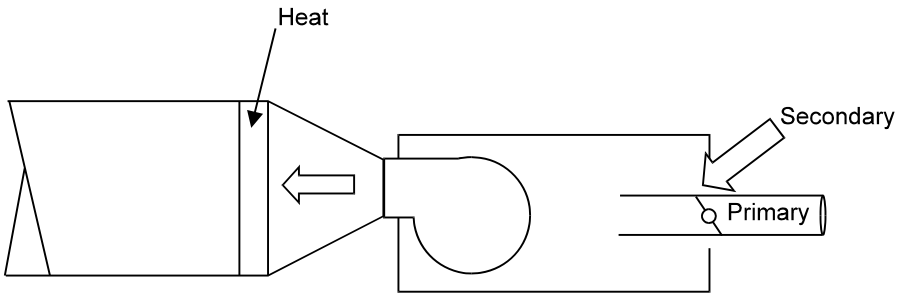


Figure 45 Series PIU

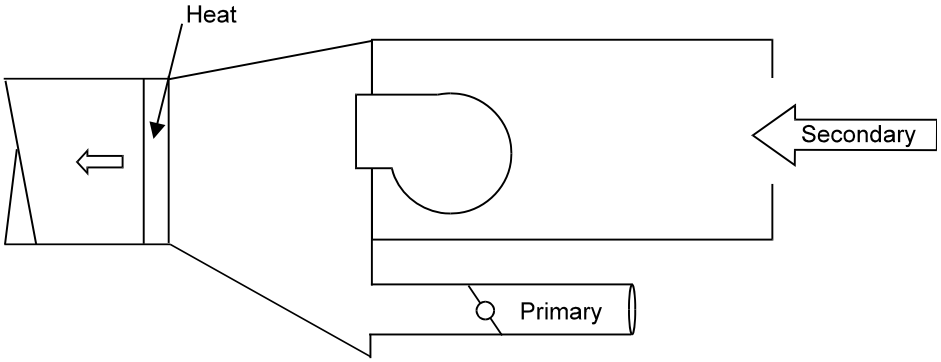


Figure 46 Parallel PIU

Other names and/or applications for PIU

- Fan Powered Boxes
- A useful simulation when necessary to simulate Clean Room and Surgery systems requiring 25 or more recirculated air changes.
- A mixture of Series and Parallel boxes is allowed.
- Core zones must be specified as having standard VAV (SVAV) boxes.
- Zones with PIU boxes use the keyword INDUCED-AIR-FROM = U-NAME of the core zone which is assumed to always require cooling.

Standard Temperature Controls

Normally, the supply air temperature is held CONSTANT as this insures the moisture content leaving the cooling coil meets design specifications.

- Starting with the unit operating during cold weather, on a small rise in supply air temperature sensed by the supply air controller, the economizer modulates open to outside air, provided the system has an economizer and outside air can provide cooling.
- On a further rise, the cooling coil modulates to maintain the supply air setpoint.

- The zone thermostat(s) modulate the PIU box dampers which draw ceiling air in and return it (mixed with primary air) to the conditioned space.

- As an option to cycling the main air fans to hold night setback temperatures, the PIU boxes (which use much less fan energy) may be used. To simulate this set NIGHT-CYCLE-CTRL = ZONE-FANS-ONLY.

Input template for a standard PIU unit

```
U-name = SYSTEM
   TYPE              = PIU
   MAX-SUPPLY-T      = 105
   MIN-SUPPLY-T      = 55
   COOL-CONTROL      = CONSTANT
   OA-CONTROL        = TEMP
   ECONO-LIMIT-T     = 70
   SUPPLY-STATIC     = 5                  inches total static
   SUPPLY-EFF        = 0.65               overall fan eff
   RETURN-STATIC     = 0.75               if a return fan
   RETURN-EFF        = 0.65               if a return fan
   FAN-SCHEDULE      = U-NAME             fan run times
   NIGHT-CYCLE-CTRL  = ZONE-FANS          if zone fans cycle on
                                          to hold night setpt
   HEAT-SOURCE       = HOT-WATER          if a boiler
   HW-LOOP           = U-NAME             of HW loop
   CHW-LOOP          = U-NAME             of CHW loop
   ..
```

Under ZONE with suggested values

```
U-NAME = ZONE                           different than SPACE
   TYPE              = CONDITIONED
   DESIGN-HEAT-T     = 68
   DESIGN-COOL-T     = 75
   HEAT-TEMP-SCH     = U-NAME             thermostat heat sch
   COOL-TEMP-SCH     = U-NAME             thermostat cool sch
   OA-FLOW/PER       = 15
   SPACE             = U-NAME             of corresponding
                                          SPACE in LOADS module
   TERMINAL-TYPE     = SVAVS, SERIES-PIU, PARALLEL-PIU
   INDUCED-AIR-ZONE  = U-NAME             of SVAV core zone
   REHEAT-DELTA-T    = 50                 applies to PIU boxes
   MIN-FLOW-RATIO    = 0.3
   ZONE-FAN-FLOW     = user-supplied value
   ZONE-FAN-T-SCH    =U-NAME of SCHEDULE  for parallel PIUs
   ..
```

Changes or additions when using PIU for other applications

- As an option to COOL-CONTROL = CONSTANT change to WARMEST or RESET.

- MAX-HUMIDITY can be controlled by lowering the supply air temperature when WARMEST or RESET is used.

- MIN-HUMIDITY can be controlled by the addition of moisture to supply air.

- Powered Induction boxes may be used to model Clean Rooms with the primary system to cool the air and a secondary system to provide the large recirculated air change requirements. This may require overriding the default limits of ZONE-FAN-KW of the standard PIU box. Set LIST = NO-LIMITS prior to this input and then back to LIMITS following the input. Also set MIN-FLOW-RATIO = 1.0.

Covered in detail by separate Topics:

- Night Ventilation (SYSTEM command)

- Add On Evaporative Cooling (SYSTEM command)

- Add On (Integrated) Desiccant Cooling (SYSTEM command)

- Service Hot Water Heat Pump (SYSTEM and CIRCULATION-LOOP commands)

- Electric and Fuel Meters (Metering topic; SYSTEM, ZONE, ELEC-METER and FUEL-METER commands)

- Building Resources (CIRCULATION-LOOP command)

- Optimum Fan Start (SYSTEM command)

- Heat Recovery from Relief Air

- Baseboard or Fin Tube Radiation (SYSTEM command)

SYSTEM TYPE = CBVAV

Ceiling Bypass Variable Air Volume

The concept for this system was in response to a problem of the standard VAV system. This problem was due to a pressure build-up behind open boxes (resulting in flow rates much greater than design) as VAV boxes closed and others remained open. In the CBVAV system the volume was reduced to the zones but the excess was simply bypassed into a return plenum. The fan volume thus remained at constant flow and the flow rate at the cooling coil also remained constant, removing maximum moisture. The system is usually supplied with an outside/return air damper economizer mixing box, air filters, a preheat coil in very cold climates, a cooling coil, a main heating coil, and a supply fan. A return fan was seldom used unless the distribution system was very long. The balance of return air and outside air is relatively easy when the air volume is constant. The VAV bypass boxes are usually fitted with reheat coils for individual zone control. When humidity control is required a main heating coil is furnished. Note that fan energy saving is non-existent with the CBVAV but, fortunately, the total static pressure of the system is usually less than standard VAV. As this system requires simultaneous heating and cooling it is not acceptable in today's buildings but many systems were installed and some retrofits are possible.

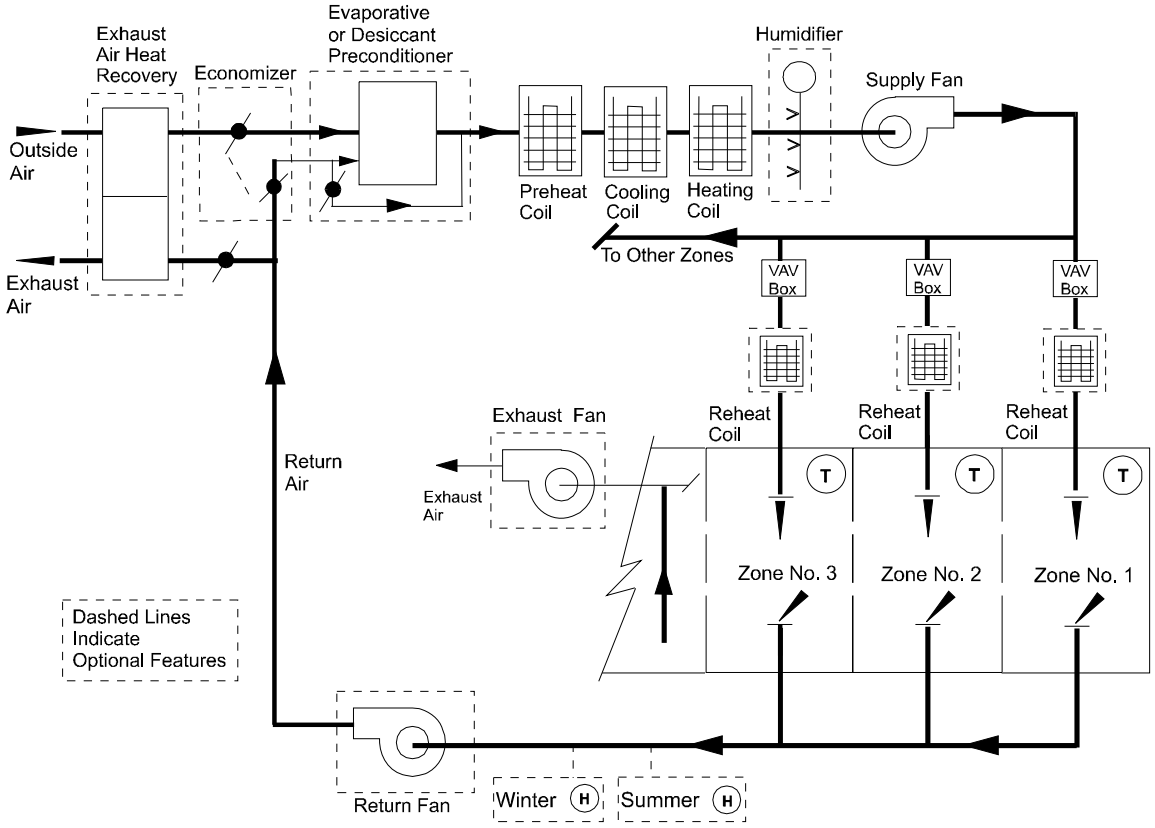


Figure 47 Ceiling-bypass VAV

Other names and/or applications for CBVAV

- The CBVAV was installed in office buildings of moderate size (less than 100,000 square feet).

- The CBVAV system was also popular with modular ceiling designs. By installing a second plenum below the return plenum, and using acoustic tiles with air delivery holes, the ceiling was less cluttered with only lighting fixtures showing and these could be moved to accommodate partition changes. Another design used supply/return air lighting fixtures which also provided a less cluttered ceiling design.

- A possible retrofit is to set COOL-CONTROL = WARMEST and MAX-HUMIDITY = 60. This reduces both the cooling loads and heating reheat loads.

Standard Temperature Controls

Normally, the supply air temperature is held CONSTANT as this insures the moisture content leaving the cooling coil meets design specifications.

- Starting with the unit operating during cold weather, on a small rise in supply air temperature sensed by the supply air controller, the economizer modulates open to outside air, provided the system has an economizer and outside air can provide cooling.

- On a further rise the cooling coil modulates to full open (and the economizer closes once outdoor conditions are unfavorable). The preheat coil operates independently of the supply air controller to prevent freeze-up of the cooling coil.

- The main handling unit reheat coil holds the supply temperature if the air leaving the cooling coil is colder than desired due to a MAX-HUMIDITY requirement.

- MIN-HUMIDITY is controlled by adding moisture to the supply air which can occur during winter when outside moisture loads are very low.

- And individual space thermostats modulate the air flow to the zone while increasing bypass air to the return plenum.

Input template for a standard CBVAV unit

Under SYSTEM with suggested values

```
U-name = SYSTEM
   TYPE              = CBVAV
   MAX-SUPPLY-T      = 105
   MIN-SUPPLY-T      = 55
   COOL-CONTROL      = CONSTANT          or WARMEST
   MAX-HUMIDITY      = 60                which overrides
                                         WARMEST to force
                                         sufficient moisture
                                         removal
   MIN-HUMIDITY      = 25                to inject moisture
                                         into the supply air
                                         stream
   HEAT-SET-T        = 90
   OA-CONTROL        = TEMP
   ECONO-LIMIT-T     = 70
   SUPPLY-STATIC     = 5                 inches total static
   SUPPLY-EFF        = 0.65              overall fan eff
   RETURN-STATIC     = 0.75              if a return fan
   RETURN-EFF        = 0.65              if a return fan
   FAN-SCHEDULE      = U-NAME            fan run times
   FAN-CONTROL       = INLET
   NIGHT-CYCLE-CTRL  = CYCLE-ON-ANY      if fans cycle on to
                                         hold night setpoint
   MIN-FLOW-RATIO    =0.30
   REHEAT-DELTA-T    = 50                if subzone reheat
   HEAT-SOURCE       = HOT-WATER         if a boiler
   HW-LOOP           = U-NAME            of HW loop
   CHW-LOOP          = U-NAME            of CHW loop
   ..
```

Under ZONE with suggested values

```
U-NAME = ZONE                           different than SPACE
   TYPE              = CONDITIONED
   DESIGN-HEAT-T     = 68
   DESIGN-COOL-T     = 75
   HEAT-TEMP-SCH     = U-NAME            thermostat heat sch
   COOL-TEMP-SCH     = U-NAME            thermostat cool sch
   OA-FLOW/PER       = 15
   SPACE             = U-NAME            of corresponding
   ..                                    SPACE in LOADS module
```

Covered in detail by separate Topics

- Night Ventilation (SYSTEM command)
- Add On Evaporative Cooling (SYSTEM command)
- Add On (Integrated) Desiccant Cooling (SYSTEM command)
- Service Hot Water Heat Pump (SYSTEM and CIRCULATION-LOOP commands)
- Electric and Fuel Meters (Metering topic; SYSTEM, ZONE, ELEC-METER and FUEL-METER commands)

- Building Resources (CIRCULATION-LOOP command)
- Optimum Fan Start (SYSTEM command)
- Heat Recovery from Relief Air
- Baseboard or Fin Tube Radiation (SYSTEM command)

SYSTEM TYPE = RHFS

Reheat Fan System

Years ago the RHFS system was accepted as the system that provided the best comfort conditions in a building. However, when the energy crisis hit in the mid-1970's, RHFS was the first retrofit target. RHFS was applied as the only choice of system that could meet the design specifications for Federal buildings (±2F heating and ±5F cooling in all spaces). It should be obvious that the designers were not negligent prior to the energy crisis, as they knew RHFS was energy intensive, but it was the only system that could meet design criteria for all high profile buildings. RHFS is usually supplied with outside/return air economizer damper mixing box, air filters, a preheat coil when necessary to protect the chilled water coil from freezing, a reheat coil for humidity control, a humidifier, and a supply fan. If the unit were very large, a return fan was also supplied. The chilled water coil was drained in the winter to prevent freezing in very cold climates. Terminal reheat coils located in the duct work just ahead of space air diffusers were usually hot water coils, but ELECTRIC is an option.

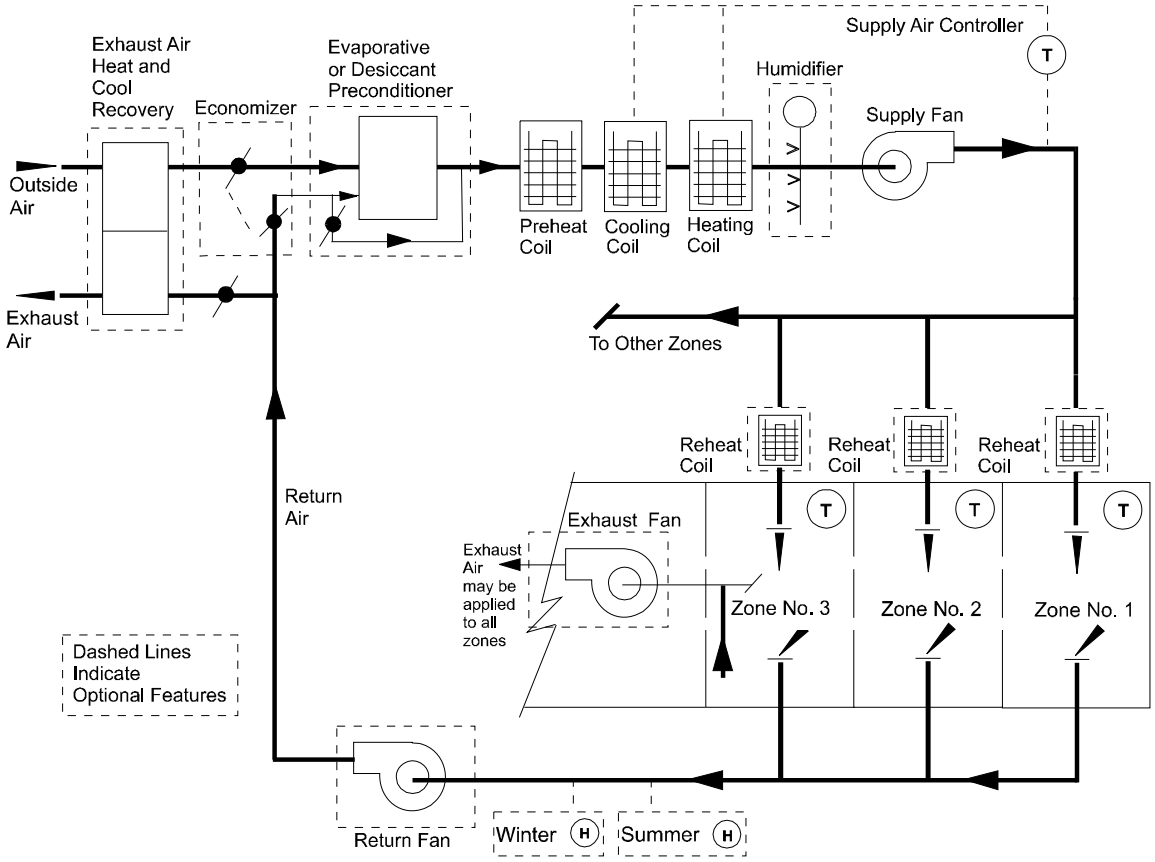


Figure 48 Reheat fan system

Other names and/or applications for RHFS are

- Hospitals (surgery, intensive care, recovery, x-ray), theaters and auditoriums
- Concert halls (to protect musical instruments from high humidity)

- Office buildings and laboratories that require humidity controls
- Clean rooms
- Swimming pools to control humidity when pools are enclosed

Standard Temperature Controls

Normally, the supply air temperature is held CONSTANT as this insures that moisture removal leaving the cooling coil meets design specifications.

- Starting with the unit operating during cold weather, on a small rise in supply air temperature sensed by the supply air controllers the economizer modulates open, provided the system has an economizer and outside air can provide cooling
- On a further rise the cooling coil modulates to full open (and the economizer closes once outdoor air conditions are unfavorable) and the reheat coils modulate to hold space thermostat setpoints.
- The preheat coil operates independently of the supply air controller to prevent freeze-up of the cooling coil.
- The main handling unit reheat coil holds the supply temperature if the air leaving the cooling coil is colder than desired due to a MAX-HUMIDITY requirement.
- MIN-HUMIDITY is controlled by adding moisture to the supply air low humidity during winter when outside moisture loads are very low.
- A thermostat in each zone modulates a reheat coil to maintain zone temperature

Input template for a standard RHFS unit

```
U-name = SYSTEM
   TYPE              = RHFS
   MAX-SUPPLY-T      = 105
   MIN-SUPPLY-T      = 50
   COOL-CONTROL      = CONSTANT
   OA-CONTROL        = TEMP
   ECONO-LIMIT-T     = 70
   HEAT-SET-T        = 75                  which enables the
                                           main reheat coil
                                           and sets a limit
                                           supply air temp
   REHEAT-DELTA-T    = 50
   MAX-HUMIDITY      = 50
   MIN-HUMIDITY      = 30
   SUPPLY-STATIC     = 4.                  inches total static
   SUPPLY-EFF        = 0.65                overall fan eff
   RETURN-STATIC     = 1.                  if a return fan
   RETURN-EFF        = 0.65                if a return fan
   FAN-SCHEDULE      = U-NAME              fan run times
   NIGHT-CYCLE-CTRL  = ZONE-FANS           if zone fans cycle on
                                           to hold night setpt
   HEAT-SOURCE       = HOT-WATER           if a boiler
   HW-LOOP           = U-NAME              of HW loop
   CHW-LOOP          = U-NAME              of CHW loop
   ..
```

Under ZONE with suggested values

```
U-NAME = ZONE                            different than SPACE
   TYPE              = CONDITIONED
   DESIGN-HEAT-T     = 68
   DESIGN-COOL-T     = 75
   HEAT-TEMP-SCH     = U-NAME              thermostat heat sch
   COOL-TEMP-SCH     = U-NAME              thermostat cool sch
   OA-FLOW/PER       = 15
   SPACE             = U-NAME              of corresponding
   ..                                      SPACE in LOADS module
```

Changes or additions when using RHFS for other applications

- As an option to COOL-CONTROL = CONSTANT change to WARMEST. This is a popular retrofit to existing systems.

- Another popular retrofit to existing systems is to reduce the supply air flow to all spaces by using a signal like WARMEST for SPEED control of the supply fan motor. Use MIN-FLOW-RATIO = 0.6 to reduce the air flow signaling the SPEED controller to slow the fan motor. By limiting the reduction in air flow to 0.6, the supply air balance to spaces is usually acceptable even though there are no VAV boxes. This same retrofit has been used successfully on multizone systems (MZS) but less so on PMZS because it aggravates the control of stepped DX equipment.

- To simulate control of outside ventilation air by a CO2 sensor control, use a MIN-OUTSIDE-SCH that resembles the one used for occupancy in the LOADS sub-program.

Covered in detail by separate Topics:

- Night Ventilation (SYSTEM command)
- Add On Evaporative Cooling (SYSTEM command)
- Add On (Integrated) Desiccant Cooling (SYSTEM command)
- Service Hot Water Heat Pump (SYSTEM and CIRCULATION-LOOP commands)
- Electric and Fuel Meters (Metering topic; SYSTEM, ZONE, ELEC-METER and FUEL-METER commands)
- Building Resources (CIRCULATION-LOOP command)
- Optimum Fan Start (SYSTEM command)
- Heat Recovery from Relief Air
- Baseboard or Fin Tube Radiation (SYSTEM command)

SYSTEM TYPE = EVAP-COOL

Stand-Alone Evaporative Cooler System

The program models stand alone evaporative cooling units as an add-on to standard HVAC systems with mechanical cooling components. Indirect, indirect-direct, and (for residential system RESYS2) direct evaporative cooling units may be selected. The performance of add-on variable-volume evaporative cooling units is modeled by effectiveness curves which you may replace. Caution: evaporative cooling keywords should not be input if desiccant cooling is specified.

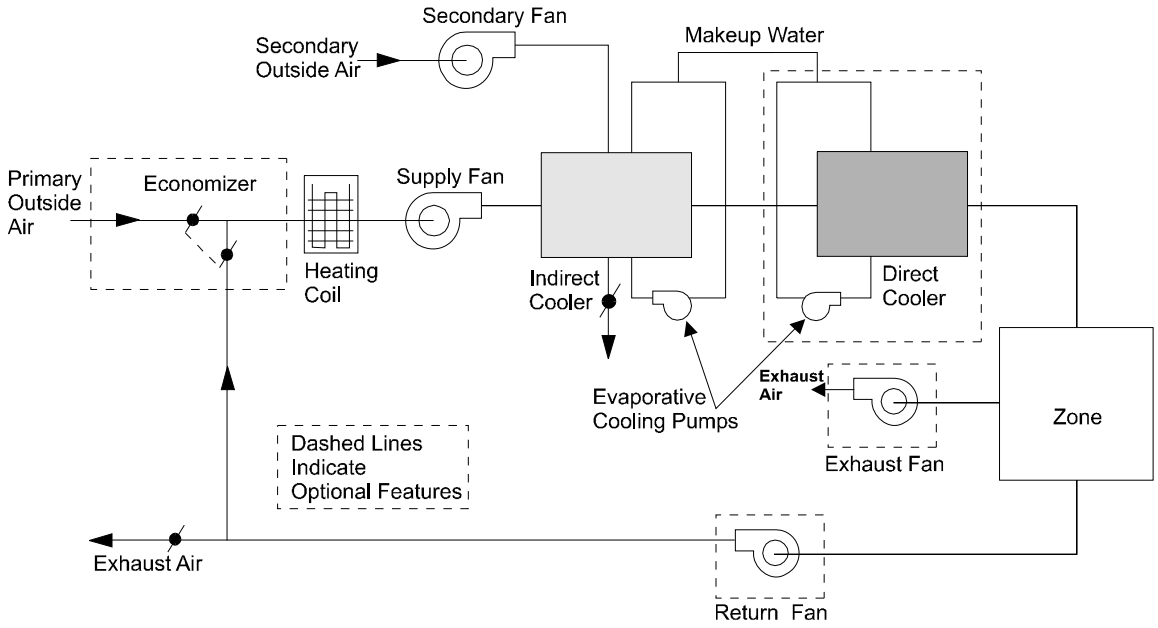


Figure 49 Stand-alone evaporative cooling system

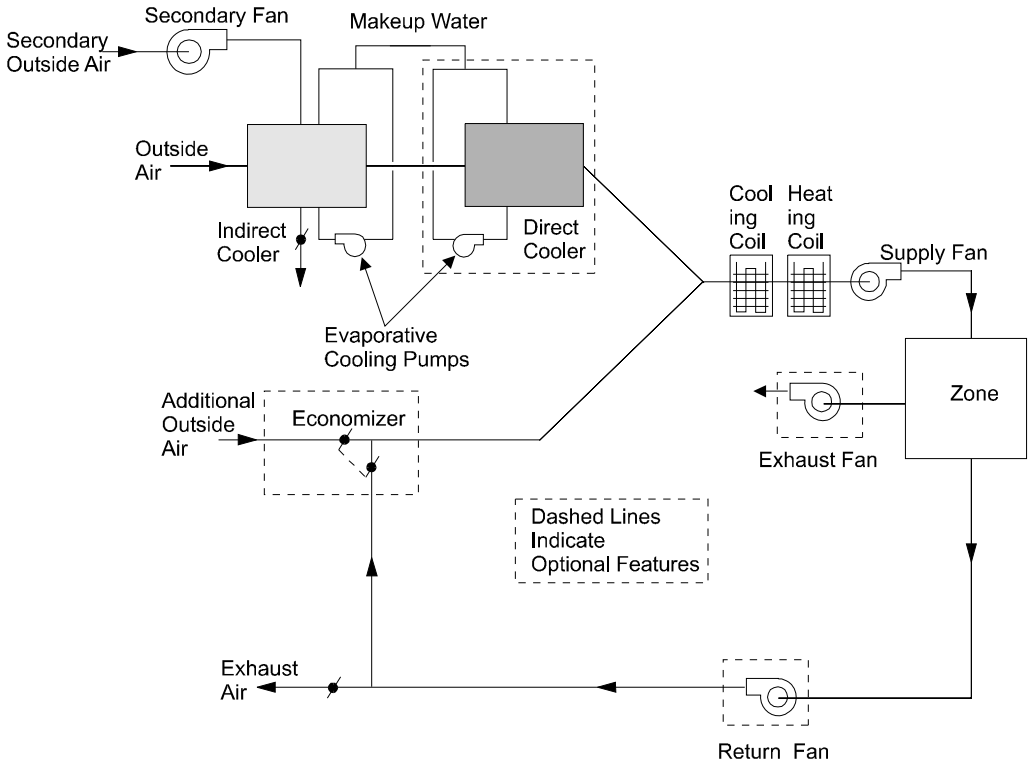


Figure 50 Add-on evaporative cooling unit in which only outside air is passed through the evaporative cooler (EVAP-CL-AIR allowed to default), shown integrated with a conventional HVAC system.

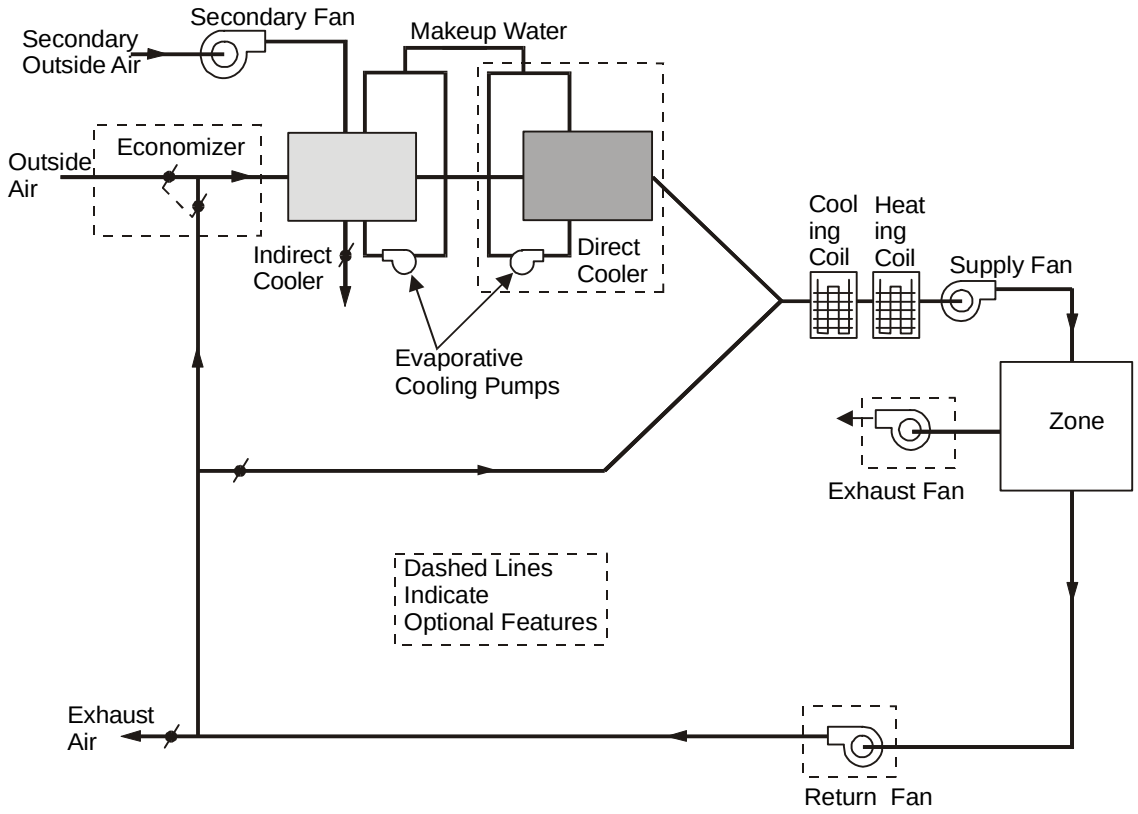


Figure 51 Add-on evaporative cooling unit in which some of the return air passes through the evaporative cooler (EVAP-CL-AIR > MIN-OUTSIDE-AIR), shown integrated with a conventional HVAC system

Input template for a standard EVAP-COOL unit

Under SYSTEM with suggested values

```
U-name = SYSTEM
   TYPE               = EVAP-COOL
   MAX-SUPPLY-T       = 105
   MIN-SUPPLY-T       = 55
   OA-CONTROL         = TEMP
   ECONO-LIMIT-T      = 70
   SUPPLY-STATIC      = 3                  inches total static
   SUPPLY-EFF         = 0.65               overall fan eff
   RETURN-STATIC      = 0.75               if a return fan
   RETURN-EFF         = 0.65               if a return fan
   FAN-SCHEDULE       = U-NAME             fan run times
   NIGHT-CYCLE-CTRL   = CYCLE-ON-ANY       if fans cycle on to
                                           hold night setpoint
   REHEAT-DELTA-T     = 50                 if subzone reheat
   ..
```

Under ZONE with suggested values

```
U-NAME = ZONE                                  different than SPACE
   TYPE              = CONDITIONED
   DESIGN-HEAT-T     = 68
   DESIGN-COOL-T     = 75
   HEAT-TEMP-SCH     = U-NAME                  thermostat heat sch
   COOL-TEMP-SCH     = U-NAME                  thermostat cool sch
   OA-FLOW/PER       = 15
   SPACE             = U-NAME                  of corresponding
   ..                                          SPACE in LOADS module
```

Covered in detail by separate Topics:

- Night Ventilation (SYSTEM command)
- Add On Evaporative Cooling, (SYSTEM command)
- Add On (Integrated) Desiccant Cooling (SYSTEM command)
- Service Hot Water Heat Pump (SYSTEM and CIRCULATION-LOOP commands)
- Electric and Fuel Meters (Metering topic; SYSTEM, ZONE, ELEC-METER and FUEL-METER commands)
- Building Resources (CIRCULATION-LOOP command)
- Optimum Fan Start (SYSTEM command)
- Heat Recovery from Relief Air
- Baseboard or Fin Tube Radiation (SYSTEM command)

SYSTEM TYPE = MZS

Multizone System

This system is usually installed in small commercial buildings and in schools. The unit is prefabricated complete with zone distribution/mixing dampers, the number of which is limited by the physical size of the unit. This unit is usually supplied with an outside/return air damper economizer box, air filters, a supply fan located ahead of hot water heating and chilled water cooling coils in parallel, and finally the hot and cold air mixing dampers, each set to modulate independently to serve the zone's temperature requirement. MZS units seldom require return fans unless heat recovery is being simulated. Since these units were originally conceived as mixing hot air with cold air to arrive at the desired supply air temperature (simultaneous heating/cooling) they are not viewed favorably today. They were used in many buildings between the years 1950 and 1975 and now are often subject to retrofits to limit simultaneous heating/cooling.

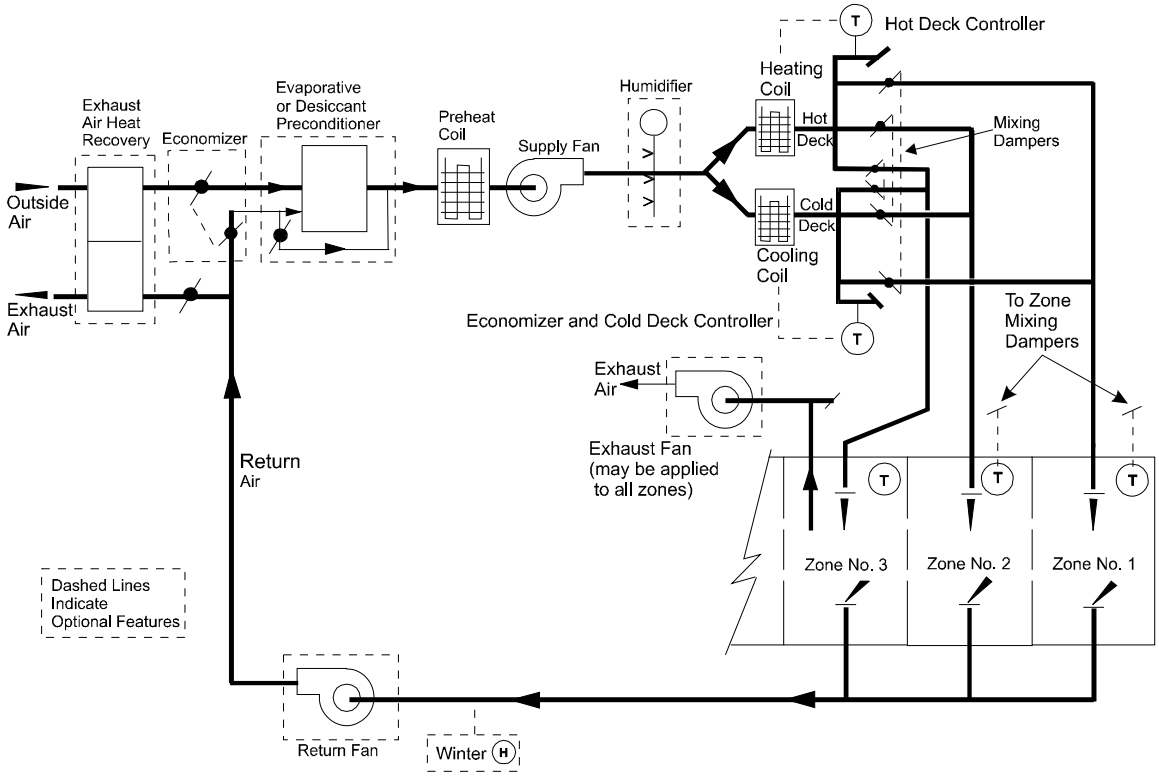


Figure 52 Multizone fan system

Other control applications for MZS

- Connect the MZS to Double Bundle Chillers so that the heating coils are supplied from heat rejected by the chiller's condenser. The heat from the condenser can be considered "free" and thus removes the aspect of simultaneous heating/cooling to reduce energy uses when "constant" control of cold and hot deck is present.

- Use a "warmest" control of the cooling coil and "coldest" control of the heating coil to reduce simultaneous heating and cooling. Reset control based on outside air temperature also reduces simultaneous heating and cooling, although not as effectively.

- Humidity control of space relative humidity is very difficult as the air bypassing the cooling coil lets moisture pass to the zones. It is possible to put both the cold and hot decks under control of a humidity sensor and increase hot deck temperatures which effectively drives more air through the cold deck for increased moisture removal. This, however, is not a control feature of the program.

- Use MAX-HUMIDITY to override the "warmest" control of the cooling coil.

- Use MIN-HUMIDITY to simulate adding moisture to the supply air.

- See the topic Dual Fan Dual-duct as an add-on to MZS even though this is an unlikely system configuration as the units are usually too small to warrant the necessary expenditure for DFDD.

Standard Temperature Controls for MZS

- The original method of controlling the MZS system was to hold the heating and cooling deck temperatures constant.

- The cold deck controller also operated the economizer dampers so that whenever the outside air temperature was below the cold deck temperature, the chilled water coil was not used.

- All space temperatures were then controlled by individual room thermostats that modulated the hot deck damper closed as it opened the cold deck to obtain a satisfactory mixed air temperature.

- For the systems that were DD/VAV, the volume of cold air to the space served was first reduced to the MIN-FLOW-RATIO and only then did mixing of the two air streams begin.

Of critical importance is the use of the zone HEAT-TEMP-SCH and COOL-TEMP-SCH in relation to constant-volume mixing boxes vs. variable-volume:

- For a constant-volume mixing box (which is the default for a PMZS), the HEAT-TEMP-SCH controls both heating and cooling, and the COOL-TEMP-SCH is ignored (but see below). This is because separate control to different heating and cooling setpoints is not possible while maintaining a constant-volume flow. The HEAT-TEMP-SCH setpoint should be set midway between the desired heating and cooling temperatures, and the throttling range broadened to encompass the range. For example, if full heating is desired at 70F and full cooling at 76F, then the setpoint should be 73F and the themostat THROTTLING-RANGE set to 6F. If the thermostat setting is changed seasonally, then the throttling range can be narrowed. For example, a setpoint temperature of 70F and THROTTLING-RANGE of 2F will provide full heating at 69F and full cooling at 71F. The only way to prevent this reset or schedule the cold deck temperature (best), or lock out cooling (not as good, as economizer cooling may still occur).

- As stated above, the COOL-TEMP-SCH is ignored in a constant-volume mixing box, with the following exception. If the cold deck is to be reset using COOL-CONTROL = WARMEST, then the COOL-TEMP-SCH is used by the cold deck controller to determine the required cold deck temperature.

- For a variable-volume mixing box, the HEAT-TEMP-SCH controls the hot deck damper, and the COOL-TEMP-SCH controls the cold deck damper.

Input template for a standard MZS unit with water coils

Under SYSTEM with suggested values

```
U-name = SYSTEM
   TYPE              = MZS
   MAX-SUPPLY-T      = 105
   MIN-SUPPLY-T      = 55
   COOL-CONTROL      = CONSTANT
   HEAT-CONTROL      = CONSTANT
   OA-CONTROL        = TEMP
   ECONO-LIMIT-T     = 60                  many pre-1975 system
                                           lacked economizer
                                           sections now required
                                           by ASHRAE
   MIN-OUTSIDE-SCH   = U-NAME              of SCHEDULE to hold
                                           outside dampers
                                           closed at night
   SUPPLY-STATIC     = 3                   inches total static
   SUPPLY-EFF        = 0.60                overall fan eff
   FAN-SCHEDULE      = U-NAME              fan run times
   NIGHT-CYCLE-CTRL  = CYCLE-ON-ANY        if fans cycle on to
                                           hold night setpoint
   HEAT-SOURCE       = HOT-WATER           if a boiler
   HW-LOOP           = U-NAME              of HW loop
   CHW-LOOP          = U-NAME              of CHW loop
   ..
```

Under ZONE with suggested values

```
U-NAME = ZONE                             different than SPACE
   TYPE              = CONDITIONED
   DESIGN-HEAT-T     = 68
   DESIGN-COOL-T     = 75
   HEAT-TEMP-SCH     = U-NAME              thermostat heat/cool
                                           setpoint schedule
   COOL-TEMP-SCH     = U-NAME              this schedule is not
   `                                       used for CONSTANT
                                           cold deck but is
                                           required for WARMEST
                                           control
   OA-FLOW/PER       = 15
   SPACE             = U-NAME              of corresponding
   ..                                      SPACE in LOADS module
```

Changes or additions when using MZS

- To simulate a three-duct multistage system (hot deck, a bypass deck, and a cold deck), use HEATING-SCHEDULE with hourly values set to 65F. This locks out heating above 65F OAT and leaves the hot deck as a bypass. Similarly, lock out the cooling below 65F (COOLING-SCHEDULE with hourly values of 65F) which leaves the cold deck as a bypass.

Covered in detail by separate Topics:

- Night Ventilation (SYSTEM command)

- Add On Evaporative Cooling (SYSTEM command)
- Add On (Integrated) Desiccant Cooling (SYSTEM command)
- Service Hot Water heat Pump (SYSTEM and CIRCULATION-LOOP commands)
- Electric and Fuel Meters (SYSTEM, ZONE, ELEC-METER, and FUEL-METER commands)
- Building Resources (CIRCULATION-LOOP command)
- Optimum Fan Start
- Heat and Cool Recovery of Relief Air
- Baseboard or Fin Tube Radiation

SYSTEM TYPE = DDS

Dual-duct System

This system had its heyday in the 1960s but fell out of favor during the energy crisis as it is almost as energy intensive as the reheat fan system as it also uses simultaneous heating/cooling. In fact, in the winter its energy use is nearly identical to RHFS. The system is often installed with VAV boxes and this substantially reduces energy use. The unit is usually supplied with outside/return air damper economizer box, air filters, a supply fan located ahead of hot water heating and chilled water coils which are in parallel, and finally separate distribution ducts (hot and cold) to which mixing boxes are connected at each separate zone. These units often require return air fans (especially with VAV) to pull the air back from the building zones as the supply air distribution may be extensive. Preheat coils are used in very cold climates to protect the water coils from freezing.

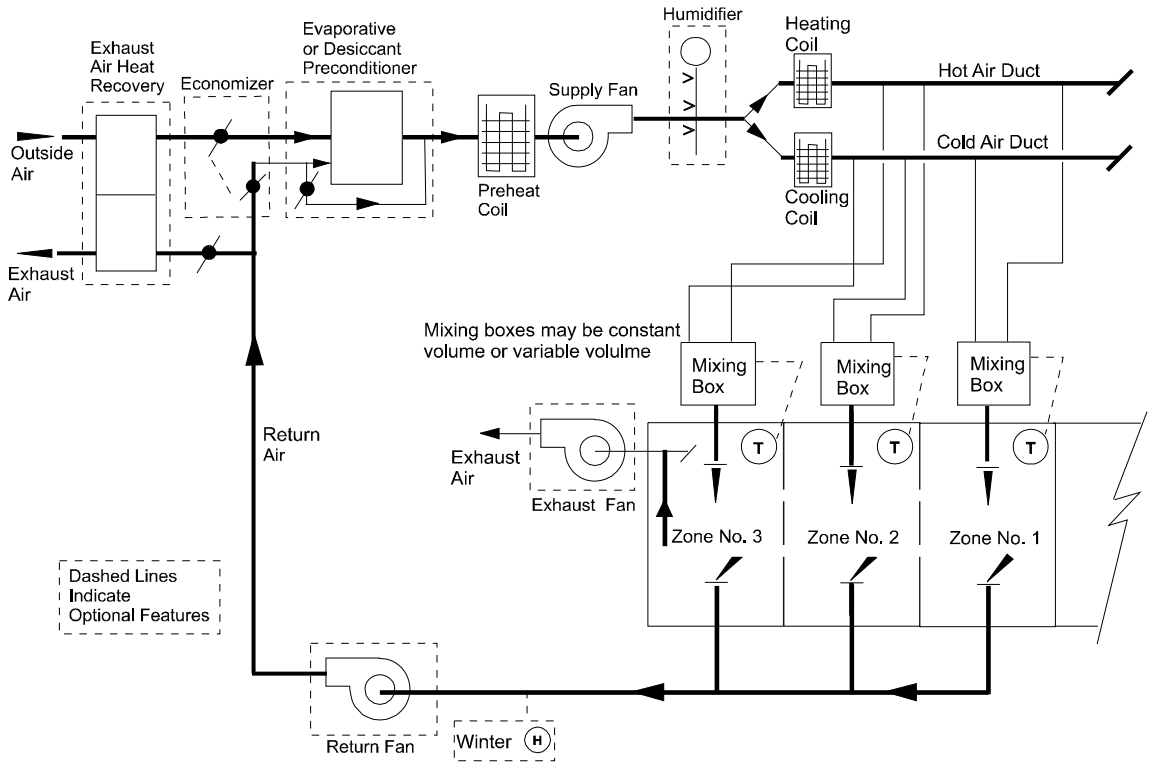


Figure 53 Dual-duct fan system

Other control applications for DDS

- Connect the DDS to Double Bundle Chillers so that the hot coils are supplied from heat rejected by the chiller's condenser. The heat from the condenser can be considered "free" and thus removes the aspect of simultaneous heating/cooling to reduce energy use when "constant" control of cold and hot deck is present.

- Use a "warmest" control of the cooling coil and "coldest" control of the heating coil to reduce simultaneous heating and cooling. Reset control based on outside air temperature also reduces simultaneous heating and cooling, although not as effectively.

- Humidity control of space relative humidity is very difficult as the air bypassing the cooling coil lets moisture pass to the zones. It is possible to put both the cold and hot decks under control of a humidity sensor and increase hot deck temperatures and decrease cold deck temperatures to increase moisture removal. This strategy, however, is not a control feature of the program.

- Use MIN-HUMIDITY to simulate adding moisture to the supply air.

- See the topic Dual Fan Dual-duct as an add-on to DDS.

Standard Temperature Controls for DDS

The original method of controlling the DDS system was to hold the heating and cooling deck temperatures constant. The cold deck controller also operated the economizer dampers so that whenever the outside air temperature was below the cold deck temperature, the chilled water coil was not used. All space temperatures were then controlled by individual room thermostats that modulated the hot deck damper closed as it opened the cold deck to obtain a satisfactory mixed air temperature. For the systems that were DD/VAV, the volume of cold air to the space served was first reduced to the MIN-FLOW-RATIO and only then did mixing of the two air streams begin.

Of critical importance is the use of the zone HEAT-TEMP-SCH and COOL-TEMP-SCH in relation to constant-volume mixing boxes vs. variable-volume:

- For a constant-volume mixing box (which is the default for a PMZS), the HEAT-TEMP-SCH controls both heating and cooling, and the COOL-TEMP-SCH is ignored (but see below). This is because separate control to different heating and cooling setpoints is not possible while maintaining a constant-volume flow. The HEAT-TEMP-SCH setpoint should be set midway between the desired heating and cooling temperatures, and the throttling range broadened to encompass the range. For example, if full heating is desired at 70F and full cooling at 76F, then the setpoint should be 73F and the themostat THROTTLING-RANGE set to 6F. If the thermostat setting is changed seasonally, then the throttling range can be narrowed. For example, a setpoint temperature of 70F and THROTTLING-RANGE of 2F will provide full heating at 69F and full cooling at 71F. The only way to prevent this reset or schedule the cold deck temperature (best), or lock out cooling (not as good, as economizer cooling may still occur).

- As stated above, the COOL-TEMP-SCH is ignored in a constant-volume mixing box, with the following exception. If the cold deck is to be reset using COOL-CONTROL = WARMEST, then the COOL-TEMP-SCH is used by the cold deck controller to determine the required cold deck temperature.

- For a variable-volume mixing box, the HEAT-TEMP-SCH controls the hot deck damper, and the COOL-TEMP-SCH controls the cold deck damper.

Input template for a standard DDS unit with water coils

Under SYSTEM with suggested values

```
U-name = SYSTEM
   TYPE              = DDS
   MAX-SUPPLY-T      = 105
   MIN-SUPPLY-T      = 55
   COOL-CONTROL      = CONSTANT
   HEAT-CONTROL      = CONSTANT
   OA-CONTROL        = TEMP
   ECONO-LIMIT-T     = 60
   SUPPLY-STATIC     = 4.5                inches total static
   SUPPLY-EFF        = 0.65               overall fan eff
   RETURN-STATIC     = 1.0                if a return fan
   RETURN-EFF        = 0.65               if a return fan
   FAN-SCHEDULE      = U-NAME             fan run times
   NIGHT-CYCLE-CTRL  = CYCLE-ON-ANY       if fans cycle on to
                                          hold night setpoint
   HEAT-SOURCE       = HOT-WATER          if a boiler
   HW-LOOP           = U-NAME             of HW loop
   CHW-LOOP          = U-NAME             of CHW loop
   ..
```

Under ZONE with suggested values

```
U-NAME = ZONE                           different than SPACE
   TYPE              = CONDITIONED
   DESIGN-HEAT-T     = 68
   DESIGN-COOL-T     = 75
   HEAT-TEMP-SCH     = U-NAME             thermostat heat sch
   COOL-TEMP-SCH     = U-NAME             thermostat cool sch
                                          when constant-volume
                                          used for deck reset
                                          only
   OA-FLOW/PER       = 15
   SPACE             = U-NAME             of corresponding
   ..                                     SPACE in LOADS module
```

Changes or additions when using DDS

- To simulate DDS/VAV system, set MIN-FLOW-RATIO = 0.5, FAN-CONTROL = INLET or SPEED. You must also specify the zone COOL-TEMP-SCH (this keyword is ignored in constant-volume systems, except when resetting the cold deck using COOL-CONTROL = WARMEST).

- To simulate "warmest" and "coldest" control of cold and hot decks set COOL-CONTROL = WARMEST and HEAT-CONTROL = COLDEST.

- To simulate overriding the "warmest" control for increased moisture removal, set MAX-HUMIDITY = 60, which is allowed by ASHRAE 90.1P.

- To simulate no simultaneous heating and cooling, set both COOLING-SCHEDULE and HEATING-SCHEDULE with hourly values of 65 OAT

Covered in detail by separate Topics:

- Night Ventilation (SYSTEM command)

- Add On Evaporative Cooling (SYSTEM command)
- Add On (Integrated) Desiccant Cooling (SYSTEM command)
- Service Hot Water heat Pump (SYSTEM and CIRCULATION-LOOP commands)
- Electric and Fuel Meters (SYSTEM, ZONE, ELEC-METER, and FUEL-METER commands)
- Building Resources (CIRCULATION-LOOP command)
- Dual Fan Dual-duct
- Optimum Fan Start
- Heat and Cool Recovery of Relief Air
- Baseboard or Fin Tube Radiation

SYSTEM TYPE = PMZS

Packaged Multizone System

This system is usually installed in small commercial buildings and in schools. The unit is prefabricated complete with the zone distribution dampers, the number of which is limited by the physical size of the unit. The unit is usually supplied with an outside/return air damper economizer box, air filters, a supply fan located ahead of a heating coil and a DX cooling which are in parallel, and finally, the hot and cold air mixing dampers, each set modulates independently to serve the zone's temperature requirement. PMZS units seldom require return fans unless heat recovery is being simulated. Again, these units are not in great use due to the aspect of simultaneous heating/cooling.

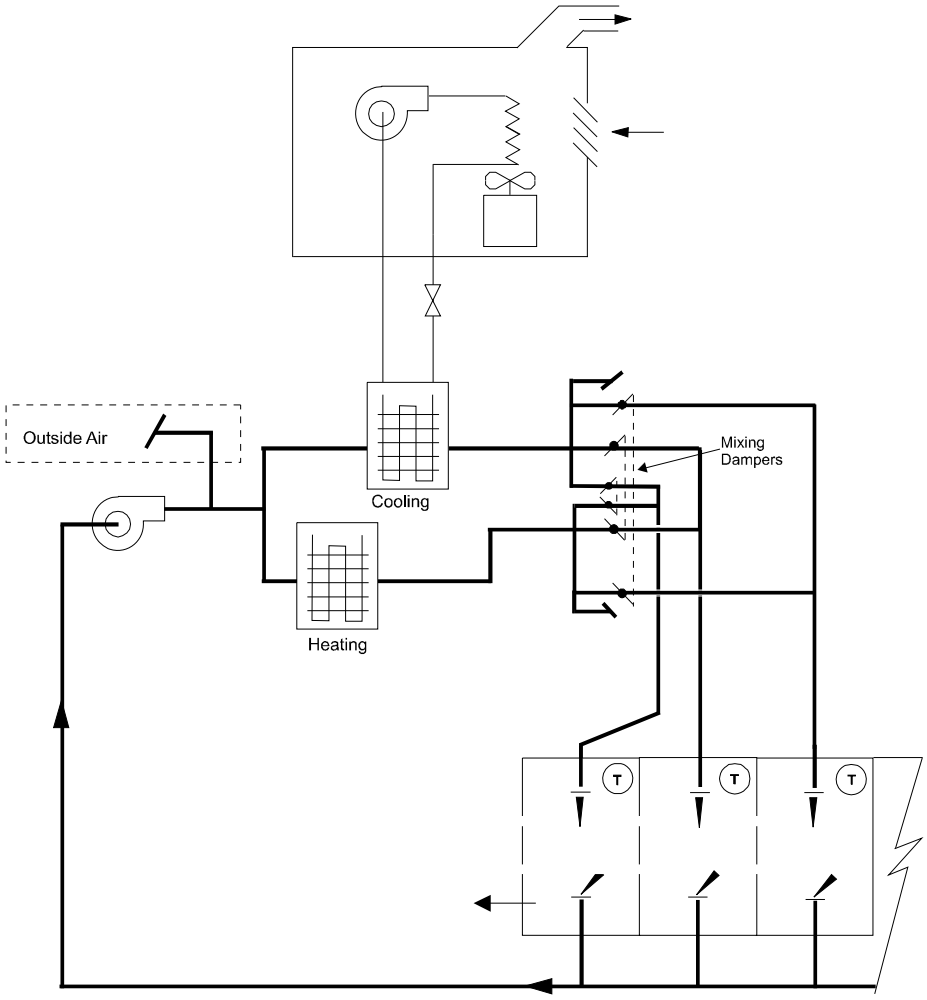


Figure 54 Packaged multizone system

Other names and/or applications for PMZS

- Split system multizone units (i.e., air-handling unit with remote condensing unit)
- Rooftop multizone units
- Use MAX-HUMIDITY to override the "WARMEST" control of the cooling coil

- Use MIN-HUMIDITY to simulate adding moisture to the supply air

- See the topic Dual Fan Dual-duct as an add-on to PMZS even though this is an unlikely system configuration

Standard Temperature Controls

The original concept for these units was to provide constant temperature cold and hot air and mix these two air streams to satisfy the zone temperature. This is much the same as mixing hot and cold water in your shower, but of course here you have paid for the hot water but the cold water is not "chilled" water. If it were, the analogy would be complete for PMZS control. The heating and cooling coils are now often controlled so that they never operate at the same time. For that reason humidity control suffers greatly as air bypasses the cold coil. You can use COOLING-SCHEDULE and HEATING-SCHEDULE to simulate this control by setting hourly values at 65F for both schedules.

Another problem inherent in PMZS is that of operating DX coils and furnaces at low flow rates, since both sources operate with stepped control. The use of economizer outside air dampers further aggravates this problem and increases the moisture content of the mixed air bypassing the cooling coil.

Of critical importance is the use of the zone HEAT-TEMP-SCH and COOL-TEMP-SCH in relation to constant-volume mixing boxes vs. variable-volume:

- For a constant-volume mixing box (which is the default for a PMZS), the HEAT-TEMP-SCH controls both heating and cooling, and the COOL-TEMP-SCH is ignored (but see below). This is because separate control to different heating and cooling setpoints is not possible while maintaining a constant-volume flow. The HEAT-TEMP-SCH setpoint should be set midway between the desired heating and cooling temperatures, and the throttling range broadened to encompass the range. For example, if full heating is desired at 70F and full cooling at 76F, then the setpoint should be 73F and the themostat THROTTLING-RANGE set to 6F. If the thermostat setting is changed seasonally, then the throttling range can be narrowed. For example, a setpoint temperature of 70F and THROTTLING-RANGE of 2F will provide full heating at 69F and full cooling at 71F. The only way to prevent this reset or schedule the cold deck temperature (best), or lock out cooling (not as good, as economizer cooling may still occur).

- As stated above, the COOL-TEMP-SCH is ignored in a constant-volume mixing box, with the following exception. If the cold deck is to be reset using COOL-CONTROL = WARMEST, then the COOL-TEMP-SCH is used by the cold deck controller to determine the required cold deck temperature.

- For a variable-volume mixing box, the HEAT-TEMP-SCH controls the hot deck damper, and the COOL-TEMP-SCH controls the cold deck damper.

Input template for a PMZS rooftop unit

Under SYSTEM with suggested values

```
U-name = SYSTEM
   TYPE              = PMZS
   MAX-SUPPLY-T      = 105
   MIN-SUPPLY-T      = 55
   COOL-CONTROL      = CONSTANT
   HEAT-CONTROL      = CONSTANT
   OA-CONTROL        = TEMP
   ECONO-LIMIT-T     = 60                  many pre-1975 system
                                           lacked economizer
                                           sections now required
                                           by ASHRAE
   COOLING-SCHEDULE  = U-NAME              with 65F hourly value
   HEATING-SCHEDULE  = U-NAME              with 65F hourly value
   MIN-OUTSIDE-SCH   = U-NAME              of SCHEDULE to hold
                                           outside dampers
                                           closed at night
   FAN-SCHEDULE      = U-NAME              fan run times
   NIGHT-CYCLE-CTRL  = CYCLE-ON-ANY        if fans cycle on to
                                           hold night setpoint
   HEAT-SOURCE       = FURNACE             HEAT-PUMP not
                                           legitimate
   ..
```

Under ZONE with suggested values

```
U-NAME = ZONE                              different than SPACE
   TYPE              = CONDITIONED
   DESIGN-HEAT-T     = 68
   DESIGN-COOL-T     = 75
   HEAT-TEMP-SCH     = U-NAME              thermostat heat sch
   COOL-TEMP-SCH     = U-NAME              this schedule is not
   `                                       used for CONSTANT
                                           cold deck but is
                                           required for WARMEST
                                           control
   OA-FLOW/PER       = 15
   SPACE             = U-NAME              of corresponding
   ..                                      SPACE in LOADS module
```

Changes or Additions when using PMZS for other Applications

As an option to FURNACE for the HEAT-SOURCE, use either ELECTRIC or HOT-WATER. Most of the PMZS units are rooftop so freezing is of concern in most all climates.

Covered in detail by separate topics are the following:

- Air and Water-side Economizers
- Air and Water Cooled Condensers
- Night Ventilation (SYSTEM command)
- Add-On Evaporative Cooling (SYSTEM command)

- Add-On (Integrated) Desiccant Cooling (SYSTEM command)
- Service Hot Water Heat Pump (SYSTEM and CIRCULATION-LOOP commands)
- Electric and Fuel Meters (SYSTEM, ZONE, ELEC-METER and FUEL-METER commands)
- Building Resources (CIRCULATION-LOOP command)
- Optimum Fan Start
- Heat Recovery of Relief Air
- Baseboard or Fin Tube Radiation

SYSTEM TYPE = FC

Fan Coil

Fan coil units represent one of the earliest attempts to provide individual room control. They range in sizes of 100 cfm to ~500 cfm for standard units, but often even the larger units are referred to as fan coil units. Small units are located under the windows or above the entryways of hotels, motels, and office buildings. The unit has an optional outside air intake, an air filter, a supply fan, a chilled water coil that may also serve as a heating coil (called a two-pipe system), or the heating coil may be a second coil served by hot water (called a four-pipe system).

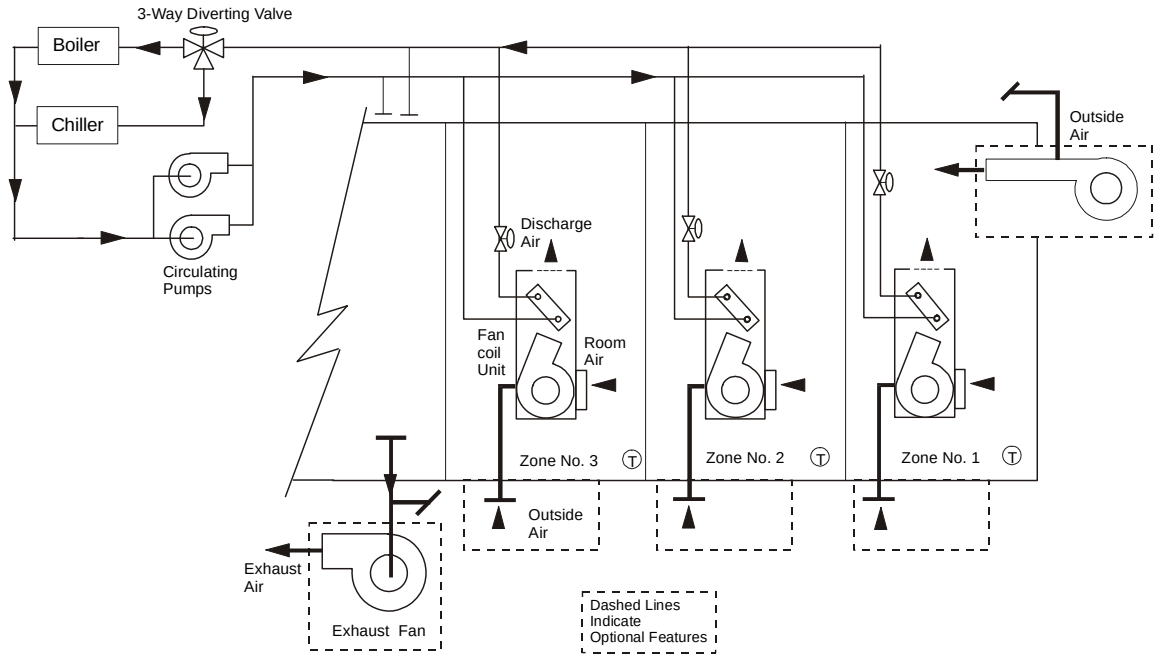


Figure 55 Two-pipe fan coil

Other names and/or applications for FC

- To specify a two-pipe system, attach the fan coil to a CIRCULATION-LOOP of TYPE = PIPE-2. See the topic on Circulation Loops.

- To specify a four-pipe system, attach the fan coil to two CIRCULATION-LOOPs of TYPE = HW and CHW. See the topic on Circulation Loops.

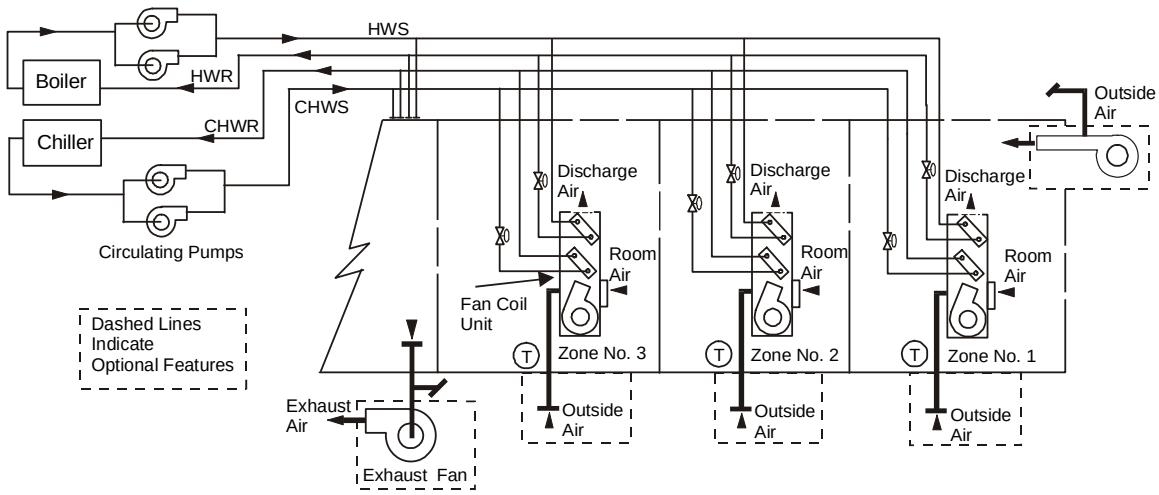


Figure 56 Four-pipe fan coil system

Standard Temperature Controls

The least expensive temperature control for two-pipe fan coils is to cycle the fan off and on at various speed settings with the thermostat located behind a cover on the unit. Another thermostat senses whether the water being delivered is hot or cold and reverses the thermostat action. The water to the coils of the four-pipe system may be modulated using valves that are opened and closed by the room thermostat.

There were many maintenance problems associated with fan coil systems. The outside air intake is a source of cold air that can freeze the coils or, when the fan is not running, spill out onto the floor. For this reason, make-up air units are now used to treat the outside air (removing moisture in summer and adding heat in winter). Use the keyword OA-FROM-SYSTEM = U-name of make-up air system to simulate make-up air to the fan coil units.

The other major problem of a two-pipe fan coil system is the long period of time required to change the circulating loop temperature from heating to cooling. The heat stored in the hot water contained in the loop must be dissipated before the water can be sent to the chiller. Otherwise, the chiller will overload or can even blow its refrigerant charge. Buildings with high solar loads really need to be able to go from heating to cooling quickly; it is for this reason that four-pipe systems now dominate.

The changeover from heating to cooling in a two-fan fan coil system can be either scheduled on the basis of time, scheduled on the basis of outdoor air temperature, or can "snap" over on the basis of a zone temperature. These control modes are determined by the associate circulation loops. Refer to the CIRCULATION-LOOP command for more information on these control modes.

Input template for a standard 2-pipe FC unit

```
U-name = SYSTEM
   TYPE               = FC
   MAX-SUPPLY-T       = 105
   MIN-SUPPLY-T       = 55
   SUPPLY-STATIC      = 3                  inches total static
   SUPPLY-EFF         = 0.65               overall fan eff
   FAN-SCHEDULE       = U-NAME             fan run times
   NIGHT-CYCLE-CTRL   = CYCLE-ON-ANY       if fans cycle on to
                                           hold night setpoint
   HW-LOOP            = "Two-pipe Loop"
   CHW-LOOP           = "Two-pipe Loop"
   ..

U-NAME = ZONE                              different than SPACE
   TYPE               = CONDITIONED
   DESIGN-HEAT-T      = 68
   DESIGN-COOL-T      = 75
   HEAT-TEMP-SCH      = U-NAME             thermostat heat sch
   COOL-TEMP-SCH      = U-NAME             thermostat cool sch
   OA-FLOW/PER        = 15
   SPACE              = U-NAME             of corresponding
                                           SPACE in LOADS module
```

Under CIRCULATION-LOOP command

```
"Two-pipe Loop" = CIRCULATION-LOOP
   TYPE               = PIPE-2
   LOOP-OPERATION     = SNAP
   SNAP-T             = 60.                heating below 60F,
                                           cooling above 60F
   ..
```

Covered in detail by separate Topics:

- Electric and Fuel Meters (SYSTEM, ZONE and ELEC-METER or FUEL-METER commands)
- Building Resources (CIRCULATION-LOOP command)
- Circulation Loops
- Enhanced Moisture Balance
- Baseboard or Fin Tube Radiation

SYSTEM TYPE = IU

Induction Units

By combining the features of fan coil units with a primary air unit (make-up) we have the induction unit system. The induction units can be either two-pipe or four-pipe and are usually located under the windows. Since they only serve the perimeter, a central or core system is also needed, as well as a primary air system. In lieu of using fans to pull air through the zone coils at the room units, a primary air system is required to supply air at a pressure great enough to induce secondary air from the room and through the unit coils. The amount of induced air is constant and is a function of the velocity of the air leaving a set of nozzles supplying the primary air. The rates of induced air to primary air is the induced air ratio. As an example, 1 part primary plus 2 parts secondary gives a ratio of 2.0 for a total supply to the room of 3 parts.

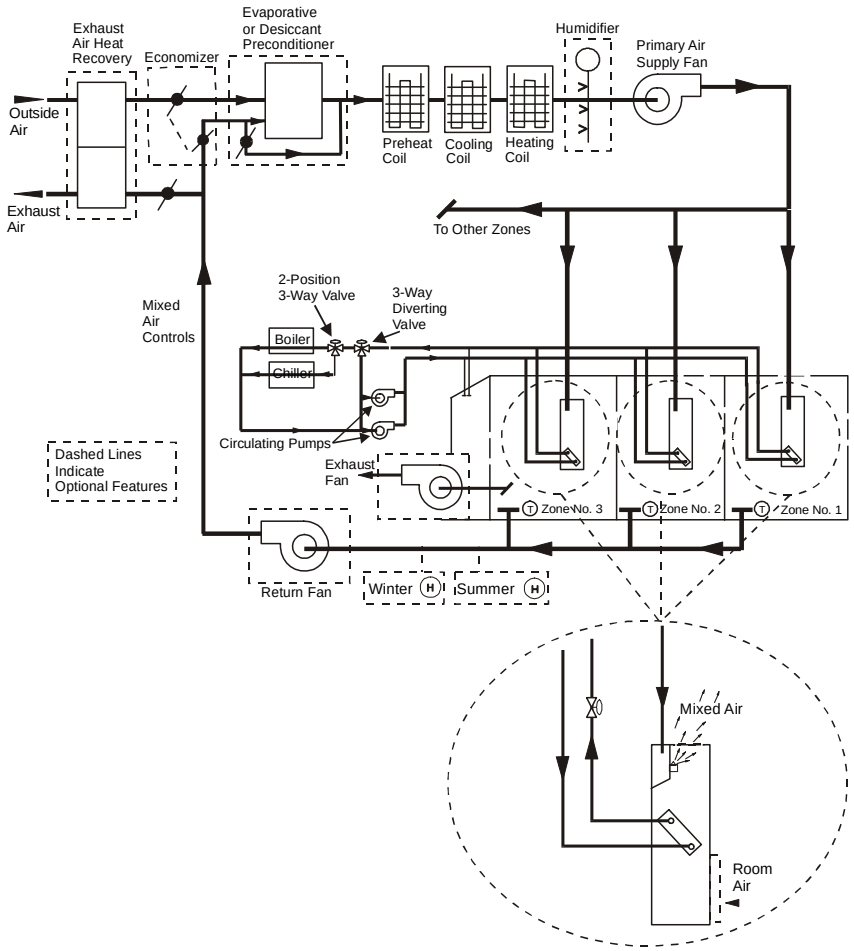


Figure 57 Two-Pipe Induction Unit. The central coils (preheat, cooling and heating) are connected to the same loops as the zonal coils

Ideally, the primary air just matches the ventilation requirement of the total of all the rooms and the primary air is then exhausted. Thus the primary air unit is a 100 percent outside air unit. This is especially true of hotel and hospital

rooms, each with bathroom exhaust. For office buildings where this system was applied for hundreds of high rise buildings, some small portion was often returned and the rest exhausted from toilet rooms.

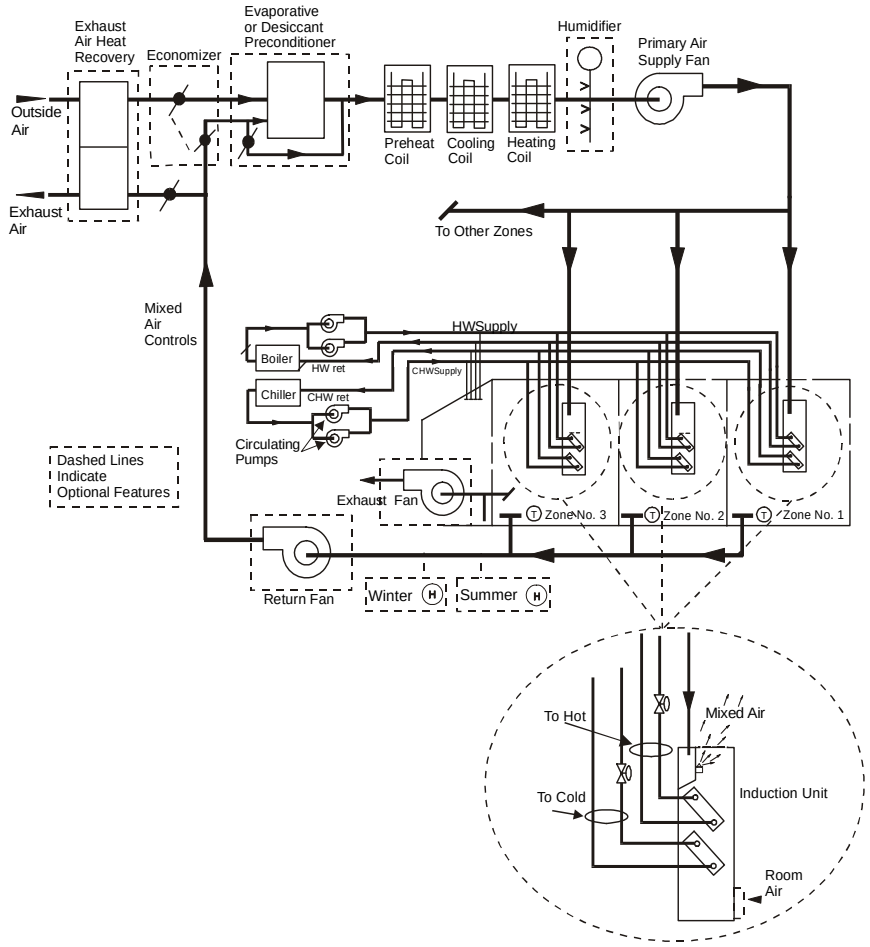


Figure 58 Four-Pipe Induction Unit The central coils (preheat, cooling and heating) are connected to the same loops as the zonal coils

The induction unit is used very little today because it fits the description of requiring simultaneous heating and cooling. This is true of the two-pipe system, but less so for the four-pipe system.

Other names and/or applications for IU

One application that has been simulated a number of times is to think of the unit coils as being ceiling radiant panels. Set the induced air ratio the same as for induction, but set the supply fan total static at 1 to 1.5 inches less than necessary for induction. If the ceiling panels require changeover from heating to cooling, use the two-pipe description; otherwise, use four-pipe for separate heating and cooling panels.

Standard Temperature Controls

Two-Pipe IU

The original concept for this system differed for summer vs winter operation. For summer, the primary air provided a limited source of warm air which was reset as a function of outside air temperature (90F LA at 60F OAT, and 60F

LA at 90F OAT) was typical. The room coil acted more like a re-cool coil than a conventional cooling coil and was controlled by a room thermostat. At a scheduled time (or often overridden by the operator) the system changed over to winter operation. The primary air now provided a limited source of cooling air held at a constant temperature. The room coil acted as a reheat coil and was controlled by a room thermostat. Depending on how temperate the climate was, many systems were operated on the summer mode year around with no changeover.

Four-Pipe IU

To be rid of the problems of changeover (see discussion for fan coil) the four-pipe induction became more acceptable for use. For summer and winter the primary air provided a limited source of neutral air (neither adding nor removing heat in relation to the room setpoint). The induction unit coils operated in sequence to hold the room setpoint. As an option to the description above the primary air in the summer was often supplied with some cooling as it needed to be cooled low enough to have a dew-point lower than the cooling coils in the room.

Input template for a standard 2- and 4-pipe IU units

Input template for a standard 2-pipe IU unit

Under SYSTEM with suggested values

```
U-name = SYSTEM
   TYPE              = IU
   MAX-SUPPLY-T      = 105
   MIN-SUPPLY-T      = 55
   MAX-HUMIDITY      = 50                  to minimize
                                           condensation on room
                                           coils
   COOL-CONTROL      = RESET
   COOL-RESET-SCH    = U-NAME
   HEAT-SET-T        = 90
   SUPPLY-STATIC     = 8                   inches total static
   SUPPLY-EFF        = 0.65                overall fan eff
   RETURN-STATIC     = 0.75                if a return fan
   RETURN-EFF        = 0.65                if a return fan
   FAN-SCHEDULE      = U-NAME              fan run times
   FAN-CONTROL       = INLET
   NIGHT-CYCLE-CTRL  = CYCLE-ON-ANY        if fans cycle on to
                                           hold night setpoint
   INDUCTION-RATIO   = 2.2
   REHEAT-DELTA-T    = 50                  if subzone reheat
   HEAT-SOURCE       = HOT-WATER           if a boiler
   HW-LOOP           = "Two-pipe Loop"
   CHW-LOOP          = "Two-pipe Loop"
   ..
```

Under ZONE with suggested values

```
U-NAME = ZONE                            different than SPACE
   TYPE              = CONDITIONED
   DESIGN-HEAT-T     = 68
   DESIGN-COOL-T     = 75
   HEAT-TEMP-SCH     = U-NAME              thermostat heat sch
   COOL-TEMP-SCH     = U-NAME              thermostat cool sch
   OA-FLOW/PER       = 15
   SPACE             = U-NAME              of corresponding
   ..                                      SPACE in LOADS module
```

Under CIRCULATION-LOOP command

```
"Two-pipe Loop" = CIRCULATION-LOOP
   TYPE              = PIPE-2
   LOOP-OPERATION    = SNAP
   SNAP-T            = 60.
   ..
```

Input template for a standard 4-pipe IU unit

Under SYSTEM command

```
U-name = SYSTEM
   TYPE              = IU
   MAX-SUPPLY-T      = 105
   MIN-SUPPLY-T      = 55
   MAX-HUMIDITY      = 50                  to minimize
                                           condensation on room
                                           coils
   COOL-CONTROL      = CONSTANT            RESET is often used
   COOL-SET-T        = 65                  = RESET
   COOL-RESET-SCH    = U-NAME
   HEAT-SET-T        = 70
   SUPPLY-STATIC     = 8                   inches total static
   SUPPLY-EFF        = 0.65                overall fan eff
   RETURN-STATIC     = 0.75                if a return fan
   RETURN-EFF        = 0.65                if a return fan
   FAN-SCHEDULE      = U-NAME              fan run times
   FAN-CONTROL       = INLET
   NIGHT-CYCLE-CTRL  = CYCLE-ON-ANY        if fans cycle on to
                                           hold night setpoint
   INDUCTION-RATIO   = 2.2
   REHEAT-DELTA-T    = 50                  if subzone reheat
   HEAT-SOURCE       = HOT-WATER           if a boiler
   HW-LOOP           = "HW Loop"           of HW loop
   CHW-LOOP          = "CHW Loop"          of CHW loop
   ..
```

Under ZONE command with suggested values

```
U-NAME = ZONE                            different than SPACE
   TYPE              = CONDITIONED
   DESIGN-HEAT-T     = 68
   DESIGN-COOL-T     = 75
   HEAT-TEMP-SCH     = U-NAME              thermostat heat sch
   COOL-TEMP-SCH     = U-NAME              thermostat cool sch
   OA-FLOW/PER       = 15
   SPACE             = U-NAME              of corresponding
   ..                                       SPACE in LOADS module
```

Under CIRCULATION-LOOP command

```
"HW Loop" = CIRCULATION-LOOP
   TYPE              = HW
   etc.
```

```
"CHW Loop" = CIRCULATION-LOOP
   TYPE              = CHW
   etc.
```

Covered in detail by separate Topics:

- Add On (Integrated) Desiccant Cooling (SYSTEM command)
- Service Hot Water heat Pump (SYSTEM and CIRCULATION-LOOP commands)
- Electric and Fuel Meters (SYSTEM, ZONE, ELEC-METER and FUEL-METER commands)
- Building Resources (CIRCULATION-LOOP command)
- Optimum Fan Start
- Circulation Loops

SYSTEM TYPE = FPH

Floor Panel Heating System

The Floor Panel Heating System provides heating for one or more zones by circulation of heated fluid through a network of pipes embedded in the floor or ceiling. A single pump, rather than the primary-secondary pumping arrangement shown in the schematic, is installed for single-zone systems. Space temperature in each zone is controlled by varying the temperature of the fluid circulating in that zone. Hourly head addition rate is determined for this system exactly as for other types of systems. Pumping energy associated with this system is accounted for by the PUMP attached to the CIRCULATION-LOOP. The ratio of panel heat losses to panel heating output must be estimated by you and entered; see the keyword PANEL-HEAT-LOSS in the ZONE command). The program assumes that this ratio of heat loss to heat output remains constant over the full range of panel output.

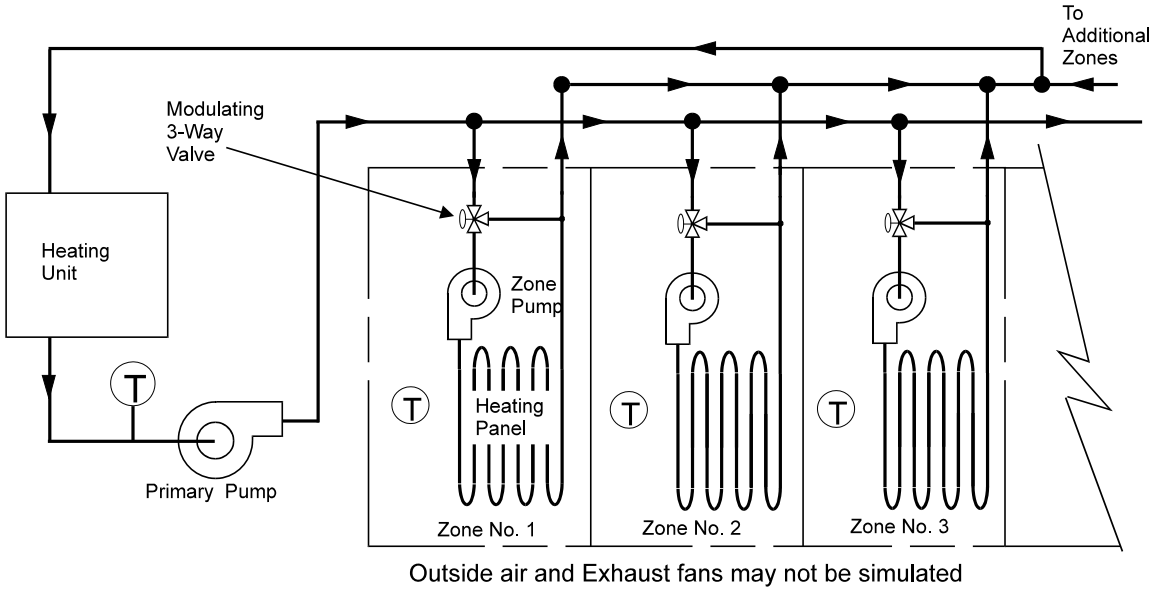


Figure 59 Floor panel heating system

Other names and/or applications for FPH

None. Note that the simulation does not account for the radiant comfort aspects of panel heating; therefore, a lower heating setpoint is appropriate.

Input template for a standard FPH unit

```
U-name = SYSTEM
   TYPE              = FPH
   HEAT-SOURCE       = HOT-WATER          if a boiler
   HW-LOOP           = U-NAME             of HW loop
   ..
```

Under ZONE with suggested values

```
U-NAME = ZONE                              different than SPACE
   TYPE              = CONDITIONED
   DESIGN-HEAT-T     = 68
   DESIGN-COOL-T     = 75                  required but unused
   HEAT-TEMP-SCH     = U-NAME              thermostat heat sch
   SPACE             = U-NAME              of corresponding
   ..                                      SPACE in LOADS module
```

Covered in detail by separate Topics are the following:

- Electric and Fuel Meters (SYSTEM, ZONE, ELEC-METER, and FUEL-METER commands)
- Building Resources (CIRCULATION-LOOP command)

SYSTEM TYPE = PTAC

Packaged Terminal Air Conditioner

PTAC systems are designed primarily for commercial installations to provide heating and cooling for a room or zone. They may be mounted in a console, in a window or through-the-wall (which is a more permanent installation). When they are cooling-only units, we know them as window units and use them for both residential and commercial applications. The unit consists of a refrigerant compressor, air-cooled condenser and fan which rejects heat to the outdoors, an evaporator coil and a fan to circulate the room air through the unit. There is usually a small outside air opening to satisfy minimum ventilation requirements but no economizer. The heating is usually done with an electric resistance coil or a heat pump or both. FURNACE is also an option. For applications like hotel and motel rooms, the heating coil may be a hot water coil. The program also allows you to select the type of supplemental heat used for a heat pump and the type of defrost control for such a unit.

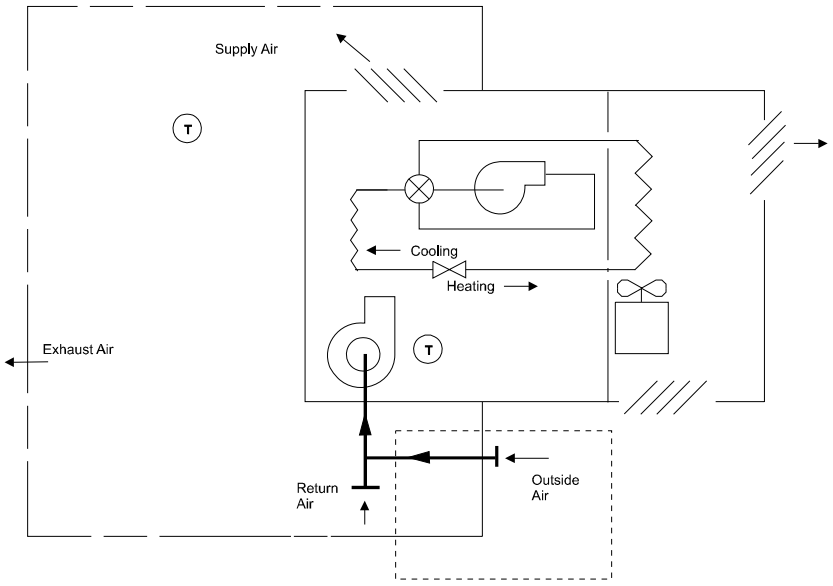


Figure 60 Packaged terminal air conditioner with heat pump

Other names and/or applications for PTAC

- Window units
- Under-window cabinet air conditioners
- Through-the-wall units
- Split-PTAC system
- Applications may be temporary installations or more permentnt ones, but replacement is always a problem for permanent installations because physical sizes vary by manufacturer and with time as new models appear. The units are used in hotel and motel guest rooms, apartments, hospital rooms, nursing homes, office buildings, residences.

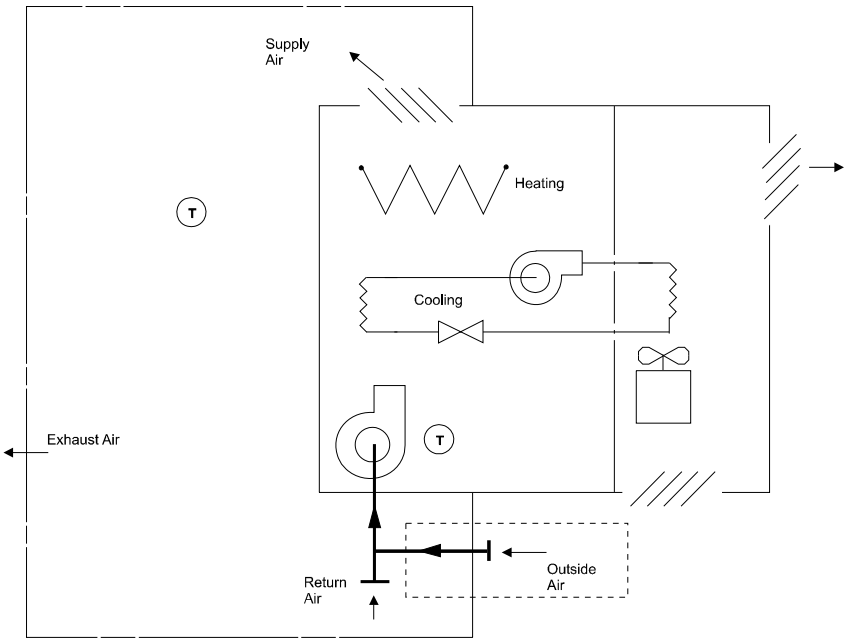


Figure 61 Packaged terminal air conditioner with DX cooling

Standard Temperature Controls

Usually you have the option of running the evaporator fan continuously or letting it cycle with the cooling/heating thermostat. You select whether the unit is to heat or cool, although the program simulates the unit as being able to switch from heat to cool automatically. The outside air damper is always open when the fan is on.

Input template for a standard PTAC unit

```
U-name = SYSTEM
   TYPE               = PTAC
   MAX-SUPPLY-T       = 105
   MIN-SUPPLY-T       = 55
   SUPPLY-STATIC      = 0.5                 inches total static
   SUPPLY-EFF         = 0.55                overall fan eff
   FAN-SCHEDULE       = U-NAME              fan run times
   NIGHT-CYCLE-CTRL   = CYCLE-ON-ANY        if fans cycle on to
                                            hold night setpoint
   HEAT-SOURCE        = HEAT-PUMP
   HP-SUPP-SOURCE     = ELECTRIC
   ..
```

Under ZONE with suggested values

```
U-NAME = ZONE                                  different than SPACE
   TYPE              = CONDITIONED
   DESIGN-HEAT-T     = 68
   DESIGN-COOL-T     = 75
   HEAT-TEMP-SCH     = U-NAME                  thermostat heat sch
   COOL-TEMP-SCH     = U-NAME                  thermostat cool sch
   OA-FLOW/PER       = 10
   SPACE             = U-NAME                  of corresponding
   ..                                          SPACE in LOADS module
```

Covered in detail by separate Topics:

- Night Ventilation (SYSTEM command)
- Add On Evaporative Cooling (SYSTEM command)
- Add On (Integrated) Desiccant Cooling (SYSTEM command)
- Service Hot Water heat Pump (SYSTEM and CIRCULATION-LOOP commands)
- Electric and Fuel Meters (SYSTEM, ZONE, ELEC-METER, and FUEL-METER commands)
- Building Resources (CIRCULATION-LOOP command)
- Optimum Fan Start
- Heat Recovery of Relief Air
- Evaporative Pre-Cooler
- Baseboard or Fin Tube

SYSTEM TYPE = HP

Water Loop Heat Pump System

The water loop heat pump system (HP) (also known as water source heat pump, California heat pump, and incremental heat pump) provides heating and cooling for a number of individually controlled zones by operation of heat pump units located in each space to be conditioned. Each heat pump unit may provide a fixed quantity of outside ventilation air or, if no outside air is specified, recirculated air only. Enhancements have been made to this system to provide a more accurate model of the types of equipment and operating strategies in current use. Heat pumps having different performance characteristics may be grouped in different SYSTEMs. For example, smaller perimeter units without outside air can be placed in one SYSTEM command, and larger core units (possibly with outside air) can be placed in another system command.

A water loop heat pump can also be modeled using a PVVT, PSZ or PVAVS system be selecting a HEAT-SOURCE as HEAT-PUMP combined with specifying that the CONDENSER-TYPE is WATER-COOLED. This alternate method of building a WLHP model allows the use of the many options that are available for those types of systems; many of those options are not available for the HP type of system.

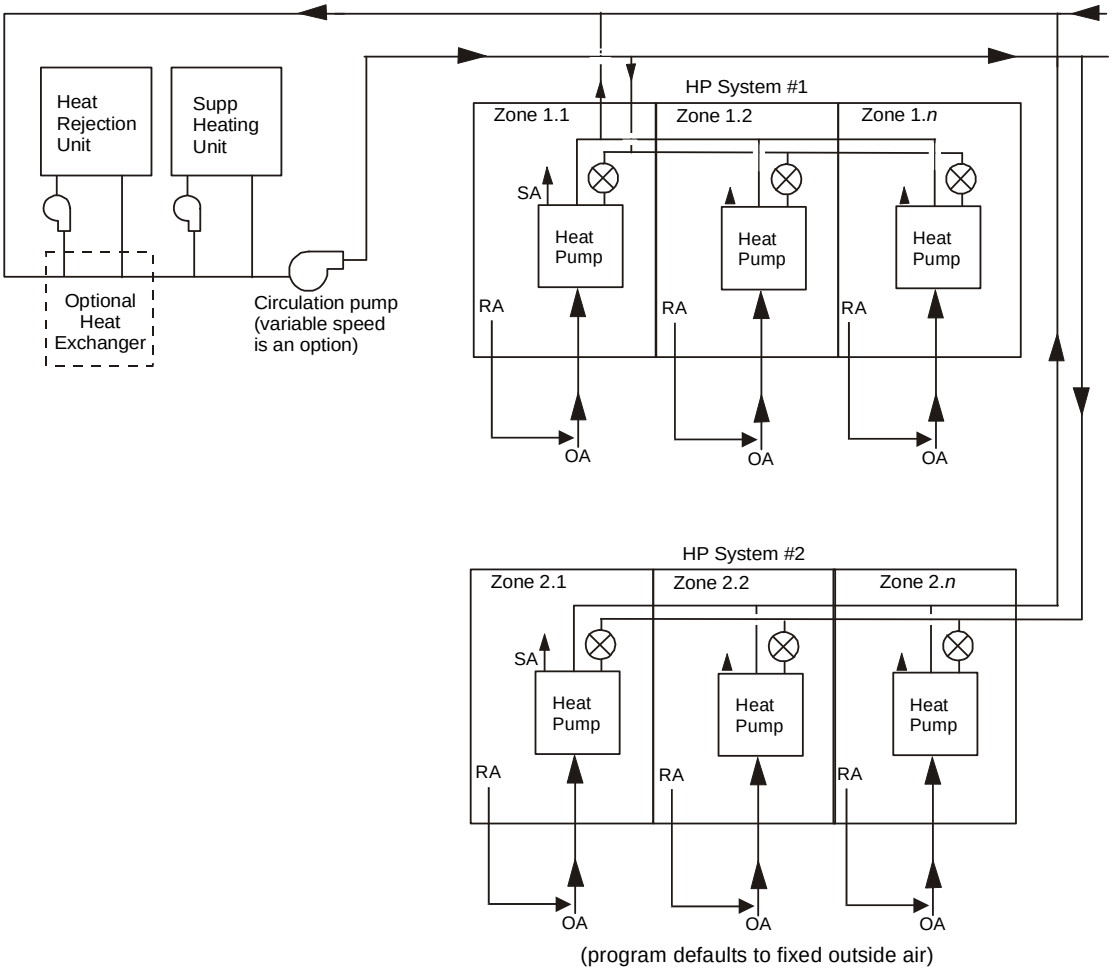


Figure 62 Water-loop heat pump system

The heat pump units draw heat from, or reject heat to, a CIRCULATION-LOOP of TYPE = WLHP. The connection of different units to the same or different circulation loops is totally flexible. Various units in the same

SYSTEM may attach to the same or different CIRCULATION-LOOPs. Entire SYSTEMs of heat pumps may also attach to the same or different CIRCULATION-LOOPs.

Each heat pump consists of a refrigerant compressor, a room air-to-refrigerant exchanger, a fluid-to-refrigerant exchanger connected to the water loop, controls to switch the evaporating and condensing functions from one heat exchanger to the other, a supply air fan, and a dual-setpoint ZONE thermostat. When the heat pump is used in the room heating mode, the room air-to-refrigerant heat exchanger is used to reject heat to the room and simultaneously accept heat from the water loop. In the room cooling mode, this same heat exchanger is used as a refrigerant evaporator, and room heat and compressor heat are rejected to the water loop. Each heat pump provides dehumidification when in the cooling mode but has no dehumidification control, per se. Humidification (adding moisture to the air) cannot be simulated.

If some of the heat pump units are operating in the heating mode while others are operating in the cooling mode, then the loop will act to transfer heat from the rooms that require cooling to the rooms that require heating. Perfect balance of heating and cooling loads is seldom achieved, however, so the circulation loop must incorporate some means of adding additional heat or rejecting surplus heat. This is commonly achieved via a boiler and fluid cooler (closed circuit cooling tower). However, some systems are now using a ground source heat-exchanger either with or without supplemental equipment. See the CIRCULATION-LOOP topic for more information on loop equipment and configurations.

Other names and/or applications for HP

- Water source heat pump
- California heat pump
- Incremental heat pump
- Ground Source Heat Pump uses these same algorithms and can be modeled as described in a separate topic, Circulation Loops. In lieu of a boiler and chiller, the loop utilizes a ground-loop heat-exchanger.

Standard Temperature Controls

Temperature is controlled in each zone by on/off operation of the compressor in the unit. The fan operation defaults to continuous when outside air is specified, although fan cycling with the compressor on-off cycle may be specified by using no outside air. You must input both a heating and cooling setpoint using schedules that are referenced by HEAT-TEMP-SCH and COOL-TEMP-SCH. The heat pump provides cooling when the space temperature is in the COOL-TEMP-SCH throttling range and heating in the HEAT-TEMP-SCH range. It does not operate when the space temperature is between the two setpoints.

A CIRCULATION-LOOP is connected to the water-to-refrigerant heat exchanger in each heat pump. The circulating fluid absorbs heat from those units that are operating in the cooling mode and are rejecting heat, and serves as a heat source to those units that are operating in the heating mode. Because some zone units may be cooling while others are heating, the temperature of the circulating fluid will depend on the relative quantities of each. When cooling demand exceeds heating demand and the fluid temperature increases to the control setpoint (see the CIRCULATION-LOOP keywords COOL-SETPT-CTRL and COOL-SETPT-T which may be allowed to float or be held at a fixed temperature) the fluid cooler rejects heat to the atmosphere. When heating demand exceeds cooling and the fluid temperature decreases to the minimum allowable value (see the CIRCULATION-LOOP keywords HEAT-SETPT-CTRL and HEAT-SETPT-T), then heat is added from a boiler or other heat source. No heat is added or rejected when heating and cooling requirements hold a balance between the high and low temperature limits of the water loop. The volume of water in the water loop determines how fast the loop temperature changes as well as how much heat may be stored in the loop. By specifying a large volume, one may simulate a storage tank.

Input template for an HP unit

```
FANS-ON = SCHEDULE
   TYPE              = ON/OFF
   THRU DEC 31        (WD)  ( 1, 7) (0)
                            ( 8,18) (1)
                            (19,24) (0)
                      (WEH)  (1,24) (0) ..
```

For the central plant

```
CWP-1 = PUMP  ..

WLHP-LOOP = CIRCULATION-LOOP
   TYPE              = WLHP
   LOOP-PUMP         = CWP-1
   PUMP-SCH          = FANS-ON             forces pump to run
                                           with heat pump fans
   HEAT-SETPT-T      = 65.
   COOL-SETPT-T      = 85.  ..

BOILER-1 = BOILER
   TYPE              = HW-BOILER
   HW-LOOP           = WLHP-LOOP  ..

TOWER-1 = HEAT-REJECTION
   TYPE              = FLUID-COOLER
   CW-LOOP           = WLHP-LOOP  ..
```

Under SYSTEM command

```
SYSTEMS-REPORT                           SS-P is generated
   SUMMARY           = (SS-A,SS-H,SS-J) .. automatically

SYSTEM1 = SYSTEM
   TYPE              = HP
   OA-CONTROL        = TEMP
   DRY-BULB-LIMIT    = 68
   FAN-SCHEDULE      = FANS-ON
   CW-LOOP           = WLHP-LOOP
   CW-VALVE          = YES                 if variable flow
   ..
```

Under ZONE with suggested values

```
U-NAME = ZONE                            different than SPACE
   TYPE              = CONDITIONED
   DESIGN-HEAT-T     = 68
   DESIGN-COOL-T     = 75
   HEAT-TEMP-SCH     = U-NAME              thermostat heat sch
   COOL-TEMP-SCH     = U-NAME              thermostat cool sch
   OA-FLOW/PER       = 15
   SPACE             = U-NAME              of corresponding
   ..                                      SPACE in LOADS module
```

Note that the heat pump units, especially in the smaller sizes equipped with direct-drive fans, may not be available in the sizes resulting from automatic sizing by the program; increased accuracy will result if you input the nominal fan sizes that are to be installed. The program checks the zone cooling and heating capacities and then sizes the units to meet the larger of these two requirements

Covered in detail by separate Topics:

- Night Ventilation (SYSTEM command)
- Service Hot Water heat Pump (SYSTEM and CIRCULATION-LOOP commands)
- Electric and Fuel Meters (SYSTEM, ZONE, ELEC-METER and FUEL-METER commands)
- Building Resources (CIRCULATION-LOOP command)
- Optimum Fan Start
- Heat Recovery of Relief Air
- Ground Source Heat Pump (CIRCULATION-LOOP command)

SYSTEM TYPE = HVSYS

Heating Ventilating System

Prior to mechanical cooling, the heating and ventilating system was the most commonly applied system for all public buildings covered by ventilation codes. It was usually supplied with an outside/return air damper, economizer mixing box, air filters, heating coils, and a supply air fan. If the unit was very large (> 20,000 cfm) there may have been a return fan. The coils were usually steam coils; however, small schools were often built with the steam boiler enclosed by a masonry wall and the ventilation air was heated by passing around and over the boiler surfaces. The air was delivered to the building as tempered air (i.e., neutral to the room) and the cast iron radiators in each room were then controlled, sometimes manually, to hold comfort conditions. As modern pneumatic controls were developed, the systems were controlled to provide outside air cooling during the spring and fall seasons. Ventilation codes were satisfied by running the fan in the summer with 100 percent outside air.

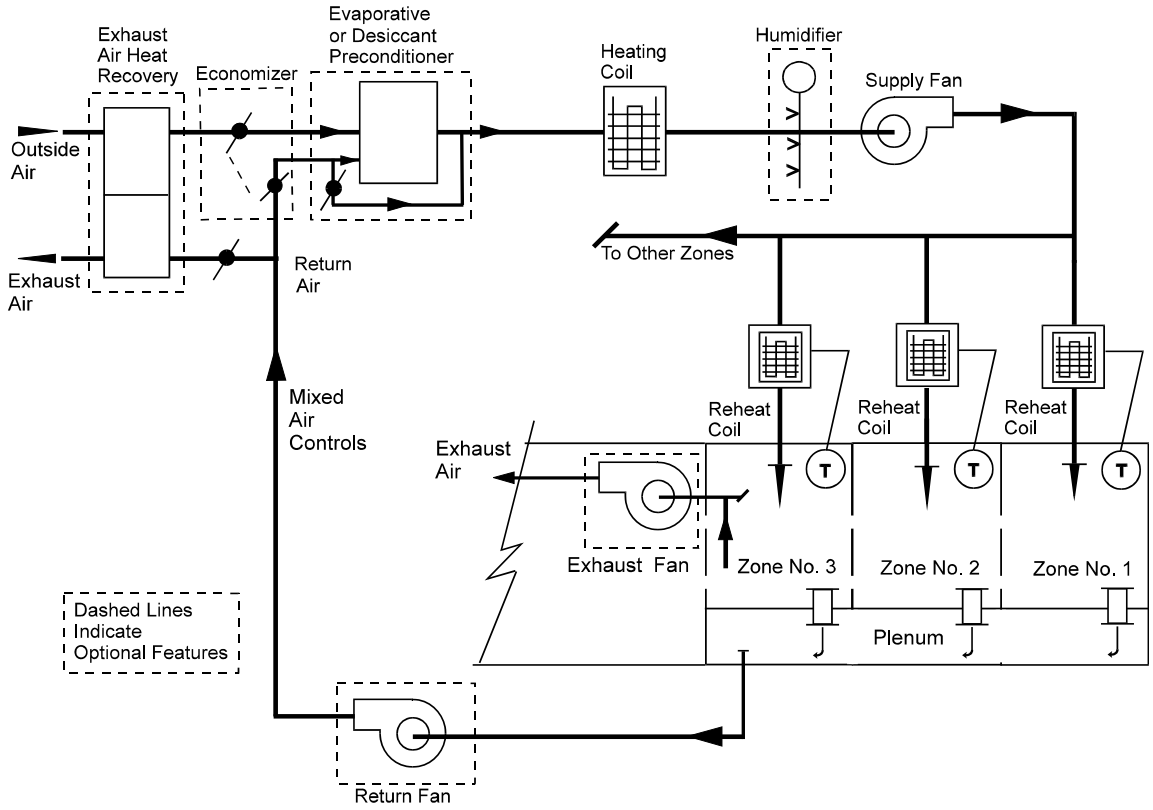


Figure 63 Heating and ventilating system

Other names and/or applications for HVSYS

- The system was designed by one dominant firm and thus often referred to by the manufacturer's name "Columbus Heat and Vent System."

- Applications were public schools, college classroom buildings and dormitories, laboratories, churches, office buildings, and any buildings to which ventilation codes applied.

Standard Temperature Controls

The early systems did not have automatic temperature controls. The janitor banked his coal-fired boiler in the evening and opened the flue draft dampers in the morning. The temperature in each room was controlled by manually adjusting a valve on the radiator inlet. In the 1930s, automatic controls were applied to the outside air mixing dampers and to the radiators. The main supply air temperature was held at warm temperatures to quickly raise the building temperature in the morning but kept at neutral temperatures during the day.

Input template for a standard HVSYS unit with water coils

Under SYSTEM command

```
U-name = SYSTEM
   TYPE              = HVSYS
   MAX-SUPPLY-T      = 105
   MIN-SUPPLY-T      = 55
   OA-CONTROL        = TEMP
   ECONO-LIMIT-T     = 70
   HEAT-SET-T        = 65
   SUPPLY-STATIC     = 3                  inches total static
   SUPPLY-EFF        = 0.65               overall fan eff
   RETURN-STATIC     = 0.75               if a return fan
   RETURN-EFF        = 0.65               if a return fan
   FAN-SCHEDULE      = U-NAME             fan run times
   FAN-CONTROL       = INLET
   NIGHT-CYCLE-CTRL  = CYCLE-ON-ANY       if fans cycle on to
                                          hold night setpoint
   HEAT-SOURCE       = HOT-WATER          if a boiler
   HW-LOOP           = "HW Loop"          of HW loop
   ..
```

Under ZONE with suggested values

```
U-NAME = ZONE                           different than SPACE
   TYPE              = CONDITIONED
   DESIGN-HEAT-T     = 68
   DESIGN-COOL-T     = 75
   HEAT-TEMP-SCH     = U-NAME             thermostat heat sch
                                          also for economizer
                                          if HEAT-CONTROL =
                                          COLDEST
   OA-FLOW/PER       = 15
   SPACE             = U-NAME             of corresponding
                                          SPACE in LOADS module
   BASEBOARD-CONTROL = OUTDOOR-RESET      or THERMOSTATIC
   BASEBOARD-RATING  = [cap of radiator]
```

Changes or additions when using HVSYS for other applications

- The most popular retrofit for this system is to add control valves to existing radiation as most of these systems over-heated the building for lack of adequate control. Better control of supply air temperatures, if heating is by steam coils, is also helpful.

- Use HEAT-CONTROL = COLDEST to reset the supply air temperature to just satisfy the coldest room. This system is almost identical to a standard reheat system but minus the mechanical cooling

coil. RHFS can be used as an alternate approach, locking out the cooling. The supply air temperature is simulated at the MIN-SUPPLY-T which should always be set to a neutral temperature.

Covered in detail by separate Topics are the following:

- Night Ventilation (SYSTEM command)
- Add On Evaporative Cooling (SYSTEM command)
- Add On (Integrated) Desiccant Cooling (SYSTEM command)
- Service Hot Water heat Pump (SYSTEM and CIRCULATION-LOOP commands)
- Electric and Fuel Meters (SYSTEM, ZONE, ELEC-METER, and FUEL-METER commands)
- Building Resources (CIRCULATION-LOOP command)
- Optimum Fan Start
- Recovery of Relief Air
- Baseboard or Fin Tube Radiation

SYSTEM TYPE = PVAVS

Packaged Variable Air Volume System

This system follows the development of VAVS. It is popular today as modular systems for high rise buildings. The PVAVS is usually supplied with outside/return damper air economizer damper mixing box, air filters, a direct expansion (DX) cooling coil, a reheat coil for humidity control, a variable flow supply fan (using motor speed, inlet vanes, or discharge vanes) to control flow without overpressuring the VAV mixing boxes. A return fan is usually supplied although small packaged rooftop units are often supplied without one. The unit has self-contained multiple refrigeration compressors and air cooled condensers. The hermetic compressors are normally supplied with hot gas bypass control to maintain stable operation at low cooling loads, but to also protect them from overheating as the compressor motors are gas cooled.

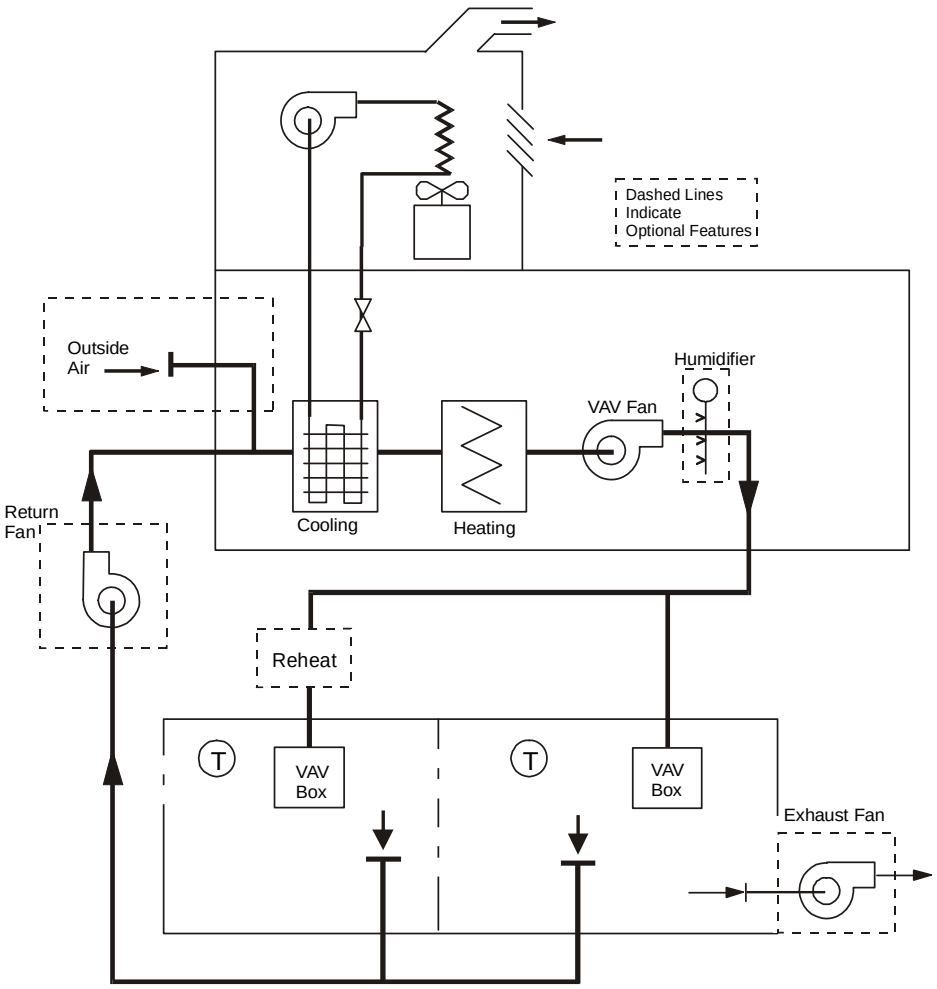


Figure 64 Packaged variable-air volume system

Other names and/or applications for PVAVS

- Rooftop PVAVS
- Self-contained VAVS

- Packaged reheat systems may be simulated by setting the MIN-FLOW-RATIO = 1.0.
- To simulate using the heat rejected by the refrigeration cycle, set MAX-COND-RCVRY = 0.6.
- To simulate a water cooled condenser, set CONDENSER-TYPE = WATER-COOLED and the CW-LOOP = U-name of a circulation loop of TYPE = CW or WLHP.
- To simulate a water-side economizer in conjunction with a water cooled condenser, set WS-ECONO = YES and the WSE-LOOP = U-name of a circulation loop of TYPE = CW.
- To simulate an evaporative precooler to cool the outside air entering an air cooled condenser, set CONDENSER-TYPE = EVAP-PRECOOLED.
- To simulate humidity control, set MAX-HUMIDITY = 60 and MIN-HUMIDITY = 25.
- To reset the supply air, set COOL-CONTROL = WARMEST.
- Another method of reset of supply air is to set COOL-CONTROL = RESET.
- Economizer and compressor operation may be either fully integrated or mutually exclusive depending on the value of ECONO-LOCKOUT.

Standard Temperature Controls

Normally, the supply air temperature is held CONSTANT as this insures that moisture removal leaving the cooling coil meets design specifications.

- Starting with the unit operating during cold weather, on a small rise in supply air temperature sensed by the supply air controllers the economizer modulates open, provided the system has an economizer and outside air can provide cooling.
- On a further rise the cooling coil modulates to full open (and the economizer closes once outdoor conditions are unfavorable). The preheat coil operates independently of the supply air controller to prevent freeze-up of the cooling coil.
- The main handling unit reheat coil holds the supply temperature if the air leaving the cooling coil is colder than desired due to a MAX-HUMIDITY requirement.
- MIN-HUMIDITY is controlled by adding moisture to the supply air which can occur during winter when outside moisture loads are very low.
- For each zone, as the zone temperature drops below the cooling setpoint, the VAV dampers modulate closed until they are at their MIN-FLOW-RATIO. As the zone temperature approaches the heating setpoint, the reheat coil, if available, will begin to modulate open to prevent the zone temperature from dropping any further. For REVERSE-ACTION thermostats, the VAV damper will also modulate back open during heating. In addition to increasing heating capacity, this control sequence can help to reduce the thermal stratification that could otherwise result when a small quantity of heated air is introduced at the ceiling level.

Input template for a standard PVAVS unit

Under SYSTEM with suggested values

```
U-name = SYSTEM
   TYPE              = PVAVS
   MAX-SUPPLY-T      = 105
   MIN-SUPPLY-T      = 55
   COOL-CONTROL      = CONSTANT
   HEAT-SET-T        = 90
   OA-CONTROL        = TEMP
   ECONO-LIMIT-T     = 70
   SUPPLY-STATIC     = 4                  inches total static
   SUPPLY-EFF        = 0.65               overall fan eff
   RETURN-STATIC     = 0.75               if a return fan
   RETURN-EFF        = 0.65               if a return fan
   FAN-SCHEDULE      = U-NAME             fan run times
   FAN-CONTROL       = INLET
   NIGHT-CYCLE-CTRL  = CYCLE-ON-ANY       if fans cycle on to
                                          hold night setpoint
   MIN-FLOW-RATIO    =0.30
   REHEAT-DELTA-T    = 50                 if subzone reheat
   HEAT-SOURCE       = HOT-WATER          if a boiler
   HW-LOOP           = U-NAME             of HW loop
   CHW-LOOP          = U-NAME             of CHW loop
   ..
```

Under ZONE with suggested values

```
U-NAME = ZONE                           different than SPACE
   TYPE              = CONDITIONED
   DESIGN-HEAT-T     = 68
   DESIGN-COOL-T     = 75
   HEAT-TEMP-SCH     = U-NAME             thermostat heat sch
   COOL-TEMP-SCH     = U-NAME             thermostat cool sch
   THERMOSTAT-TYPE   = REVERSE-ACTION     re-opens during heat
   OA-FLOW/PER       = 15
   SPACE             = U-NAME             of corresponding
   ..                                     SPACE in LOADS module
```

Covered in detail by separate Topics are the following:

- Refrigerated Casework
- Defrost Controls for Heat Pumps
- Air and Water-side Economizers
- Air and Water Cooled Condensers
- Night Ventilation (SYSTEM command)
- Add-On Evaporative Cooling (SYSTEM command)

- Add-On (Integrated) Desiccant Cooling (SYSTEM command)
- Service Hot Water Heat Pump (SYSTEM and CIRCULATION-LOOP commands)
- Electric and Fuel Meters (SYSTEM, ZONE, ELEC-METER and FUEL-METER commands)
- Building Resources (CIRCULATION-LOOP command)
- Optimum Fan Start
- Heat Recovery of Relief Air
- Natural Ventilation
- Baseboard or Fin Tube Radiation

SYSTEM TYPE = PVVT

Packaged Variable-volume Variable Temperature

This system is another variation of the VAV system. It is used almost exclusively for small (~5,000 ft2) commercial buildings but it can be used with modular building systems (e.g., one or two PVVT units for each floor of a multistory building) much the same as PVAVS is often used. Since the unit is either supplying cold or warm air which is then varied by zone dampers, the zones connected to each unit should have nearly similar needs. The first named zone controls whether the unit is on cooling or heating. For purposes of simulation, we suggest that similar zones be accumulated and modeled as one zone and assigned to one PVVT unit. The PVVT unit is usually supplied with an airside economizer, air filters, cooling coil, full sized electric heating coil used only for supplemental heat if a heat pump, a speed controlled VAV fan with a variable speed compressor, or a constant speed fan, controlled intermittently, if the compressor is standard electric/constant speed. A return fan is seldom supplied with such small units.

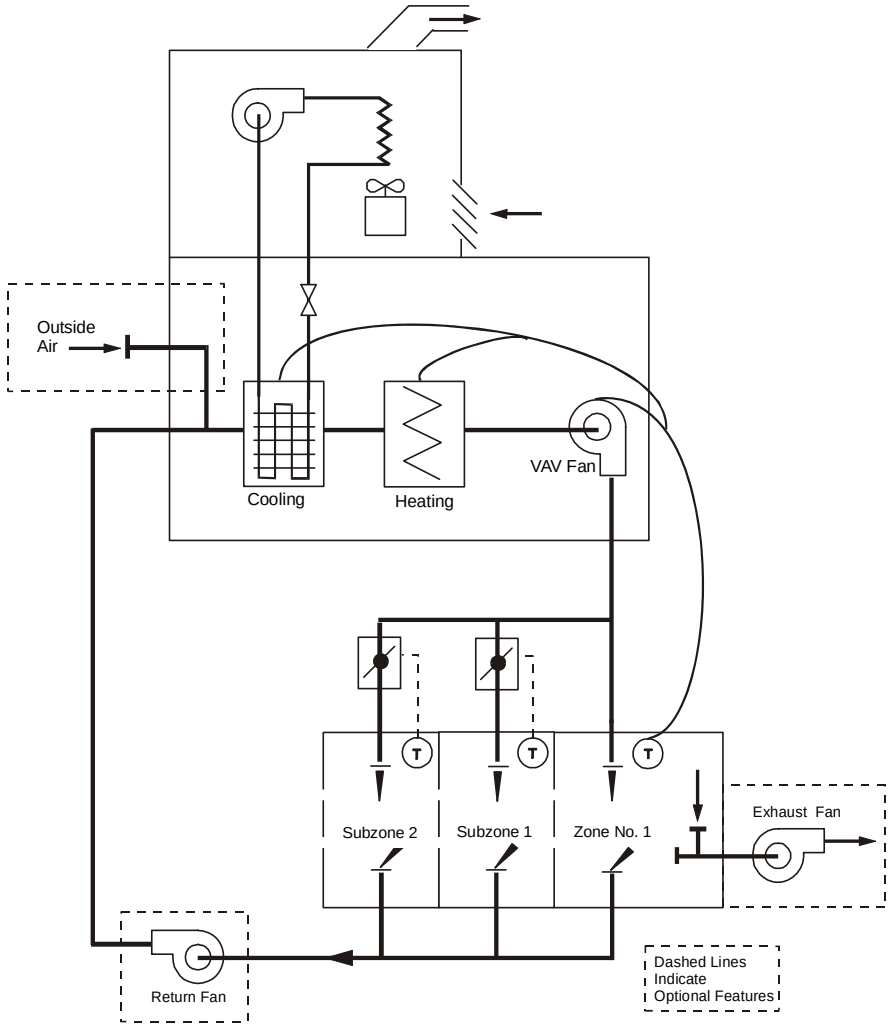


Figure 65 Packaged variable-volume variable-temperature system

Other names and/or applications for PVVT

- No other names
- The PVVT may be modeled with three different refrigeration cycles:
- Conventional compressor with either air cooled or water cooled condenser.
- A variable speed electric heat pump and air cooled condenser
- A gas engine driven heat pump and air cooled condenser
- In addition, the economizer for the unit with a conventional compressor may have either an air-side or water-side economizer. For the heat pump unit only, the air-side economizer is appropriate.

Standard Temperature Controls

- The CONTROL-ZONE controls whether the PVVT unit is heating or cooling. It also controls the supply airflow and temperature.
- In the cooling mode, as the control zone's cooling load drops, the zone airflow is first reduced with the supply air temperature held constant. Once the control zone is at its MIN-FLOW-RATIO, the flow is held constant and the supply air temperature is reset to maintain zone temperature.
- In the heating mode, as the control zone's heating load drops, the zone airflow is first reduced with the supply air temperature held constant. Once the control zone is at its MIN-FLOW-RATIO, the flow is held constant and the supply air temperature is reset to maintain zone temperature.
- The airflow to all other zones is proportionate to the airflow of the control zone. In other words, the thermostats in the non-control zones have no effect on either airflow or supply air temperature. They may, however, modulate a reheat coil or baseboard.
- At the central level, the airhandler modulates the airflow and supply temperature according to the control zone. If available, the unit will utilize an economizer prior to activating cooling, depending on the economizer control.

Input template for a standard PVVT unit with standard compressor with water cooled condenser

Under SYSTEM with suggested values

```
U-name = SYSTEM
   TYPE              = PVVT
   MAX-SUPPLY-T      = 95
   MIN-SUPPLY-T      = 55
   OA-CONTROL        = TEMP
   ECONO-LIMIT-T     = 70
   SUPPLY-STATIC     = 2                  inches total static
   SUPPLY-EFF        = 0.65               overall fan eff
   FAN-SCHEDULE      = U-NAME             fan run times
   NIGHT-CYCLE-CTRL  = CYCLE-ON-ANY       if fans cycle on to
                                          hold night setpoint
   MIN-FLOW-RATIO    =0.50                vary supply flow
                                          above,supply T below
   REHEAT-DELTA-T    = 50                 if subzone reheat
   HEAT-SOURCE       = ELECTRIC
   CONDENSER-TYPE    = WATER-COOLED       by CW or WLHP loop
   CW-LOOP           = U-NAME             of CW or WLHP loop
   CONTROL-ZONE      = U-NAME             of control zone
   ..
```

Under ZONE with suggested values

```
U-NAME = ZONE                           different than SPACE
   TYPE              = CONDITIONED
   DESIGN-HEAT-T     = 68
   DESIGN-COOL-T     = 75
   HEAT-TEMP-SCH     = U-NAME             thermostat heat sch
   COOL-TEMP-SCH     = U-NAME             thermostat cool sch
   OA-FLOW/PER       = 15
   SPACE             = U-NAME             of corresponding
   ..                                     SPACE in LOADS module
```

Covered in detail by separate Topics are the following:

- Night Ventilation (SYSTEM command)
- Service Hot Water heat Pump (SYSTEM and CIRCULATION-LOOP commands)
- Electric and Fuel Meters (SYSTEM, ZONE, ELEC-METER and FUEL-METER commands)
- Building Resources (CIRCULATION-LOOP command)
- Optimum Fan Start
- Heat Recovery of Relief Air (return fan is required)
- Variable Speed Electric Heat Pumps
- Gas Heat Pumps
- Water Cooled Condenser Option
- Evaporative Precooler for Air Cooled DX Units

SYSTEM TYPE = UHT

Unit Heater

The simulation is for a unit heater serving one zone. Therefore, entering multiple zones assigned to one system is simulated as a zonal system with one unit heater in each zone. The unit consists of a fan and heating coil which may be hot water, steam, electric resistance, or a furnace. The unit is not capable of introducing outside air and exhaust fans are not applicable. Space temperature control is accomplished by on-off cycling of the fan

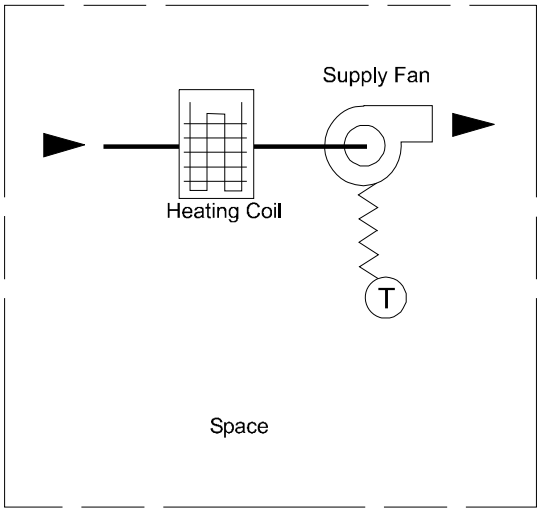


Figure 66 Unit heater. Note that the fan is most likely to be propeller-type, except for cabinet heaters

Other names and/or applications for UHT

- Suspended unit heaters
- Cabinet unit heaters
- If it is necessary to simulate infrared heaters, use UHT; however, note that the program does not attempt to account for the radiant heating aspect of infrared heaters. You can lower the thermostat setpoint and thus give a credit for the comfort aspect of such applications.
- In large industrial applications, a furnace that is direct-fired with the products of combustion vented to the space is sometimes allowed by code. Use a FURNACE-HIR of 1.05 to cover this situation.

Standard Temperature Controls

The zone thermostat for each unit cycles the fan ON-OFF.

Input template for a standard UHT unit

Under SYSTEM command

```
U-name = SYSTEM
   TYPE              = UHT
   MAX-SUPPLY-T      = 105
   SUPPLY-STATIC     = 0.3                inches total static
   SUPPLY-EFF        = 0.55               overall fan eff
   FAN-SCHEDULE      = U-NAME             fan run times
   NIGHT-CYCLE-CTRL  = CYCLE-ON-ANY       if fans cycle on to
                                          hold night setpoint
   HEAT-SOURCE       = HOT-WATER          if a boiler
   HW-LOOP           = "HW Loop"          of HW loop
   ..
```

Under ZONE with suggested values

```
U-NAME = ZONE                             different than SPACE
   TYPE              = CONDITIONED
   DESIGN-HEAT-T     = 68
   HEAT-TEMP-SCH     = U-NAME             thermostat heat sch
   SPACE             = U-NAME             of corresponding
   ..                                     SPACE in LOADS module
```

Covered in detail by separate Topics are the following:

- Electric and Fuel Meters (SYSTEM, ZONE, ELEC-METER and FUEL-METER commands)
- Building Resources (CIRCULATION-LOOP command)

SYSTEM TYPE = UVT

Unit Ventilator

This simulation is for a unit ventilator serving one zone. Therefore, entering multiple zones assigned to one system is simulated as a zonal system with one unit ventilator in each zone. This unit was used almost entirely in school classrooms prior to the accepted practice of air conditioning schools. The transition to air conditioning was made by adding a cooling coil; however, this type of unit ventilator with a cooling coil is not available. (Use SZRH or PSZ serving multiple zones of similar use for this application.) The unit consists of an economizer damper, air filter, fan and heating coil which may be hot-water, steam, electric coil or furnace. Fin tube radiation is often installed on each side of the unit ventilator to blanket the windows with warm air. The radiation is usually controlled using an outdoor reset and is operative nights and weekends to hold minimum night temperatures. Optionally, the radiation may be thermostatically controlled, riding in parallel with the unit heating coil even when the fan is on.

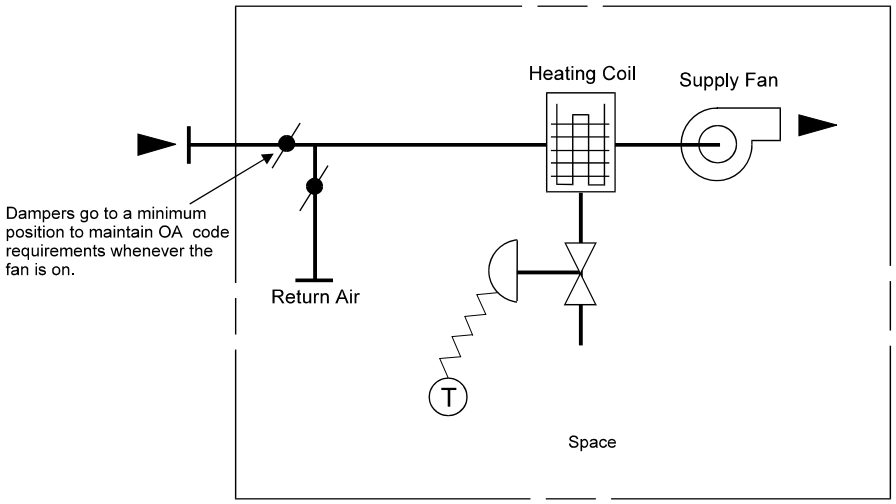


Figure 67 Unit ventilator

Other names and/or applications for UVT

The UVT system is seldom applied for applications other than schools. Also note that the HVSYS system is a central ventilation system that is also a school system.

Standard Temperature Controls

The unit ventilator control is nearly the same as that for large systems. On a rise of temperature in the classroom, the heating coil modulates closed but only to the point that the supply air temperature does not fall below the minimum of approximately 55F. On a further rise of space temperature the economizer modulates to full open with a limit of 80F to 90F, at which point the outside damper returns to its minimum position. The outside air damper is closed, whenever the fan is off, on nights and weekends. In very cold climates the fan cycles on to hold the night setback; however, the fin tube radiation normally is sufficient for this purpose.

Input template for a standard UVT unit

```
U-name = SYSTEM
   TYPE               = UVT
   MAX-SUPPLY-T       = 105
   MIN-SUPPLY-T       = 55
   OA-CONTROL         = TEMP
   ECONO-LIMIT-T      = 80
   SUPPLY-STATIC      = 0.5                inches total static
   SUPPLY-EFF         = 0.65               overall fan eff
   FAN-SCHEDULE       = U-NAME             fan run times
   NIGHT-CYCLE-CTRL   = CYCLE-ON-ANY       if fans cycle on to
                                           hold night setpoint
   HEAT-SOURCE        = HOT-WATER          if a boiler
   HW-LOOP            = U-NAME             of HW loop
   ..
```

Under ZONE with suggested values

```
U-NAME = ZONE                           different than SPACE
   TYPE               = CONDITIONED
   DESIGN-HEAT-T      = 68
   DESIGN-COOL-T      = 75
   HEAT-TEMP-SCH      = U-NAME             thermostat heat sch
   COOL-TEMP-SCH      = U-NAME             thermostat cool sch
   THERMOSTAT-TYPE    = REVERSE-ACTION     re-opens during heat
   OA-FLOW/PER        = 15
   SPACE              = U-NAME             of corresponding
   ..                                      SPACE in LOADS module
```

Covered in detail by separate Topics are the following:

- Electric and Fuel Meters (SYSTEM, ZONE, ELEC-METER and FUEL-METER commands)
- Building Resources (CIRCULATION-LOOP command)
- Baseboard or Fin Tube Radiation

SYSTEM TYPE = RESYS

Residential System

This system type is old and is not well supported. We recommend you use the RESYS2 system instead. If you use this system, you MUST model the residence as one zone; the loads from multiple zones will not be accounted for correctly.

The RESYS system models direct expansion cooling coils with a remote condensing unit (compressor and air cooled condenser). The heating may be an oil- or gas-fired furnace, an electric resistance coil, or a heat pump with any of the above heating units providing supplemental heat. This is the standard unit installed in millions of homes. It is also used for small commercial buildings using multiple units. In RESYS there is no provision for outside ventilation air to meet code requirements; therefore, infiltration of outside air must be used for this purpose. This system also allows for the simulation of opening and closing windows to provide cooling, rather than mechanical cooling. A residential direct evaporative cooler may also be simulated. An evaporative precooler for the air cooled condenser may also be simulated. A gas heat pump may be simulated in lieu of the electric single-speed heat pump that is the standard configuration. Heat can be recovered from the gas heat pump for domestic hot water use.

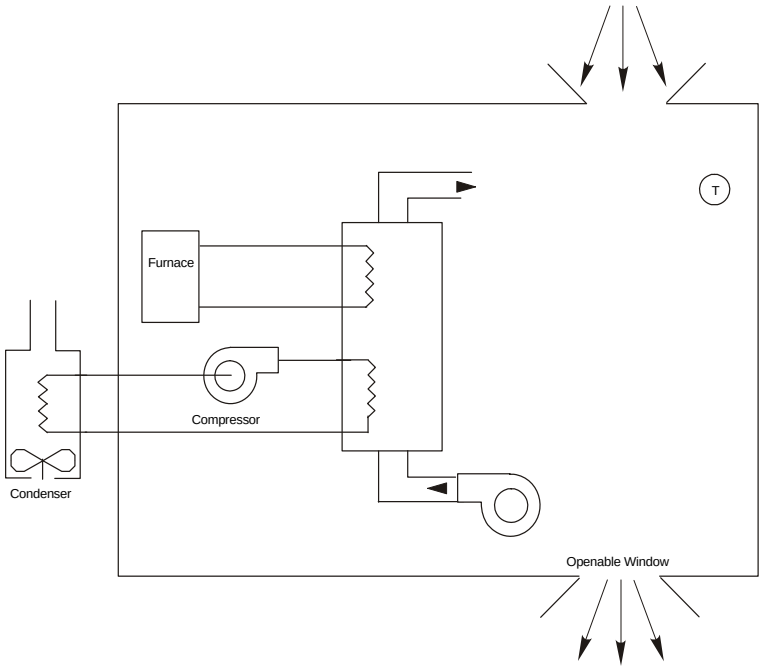


Figure 68 Residential system DX cooling coil

Other names and/or applications for RESYS

These units are often applied to apartment buildings (one for each apartment) and again for small office buildings.

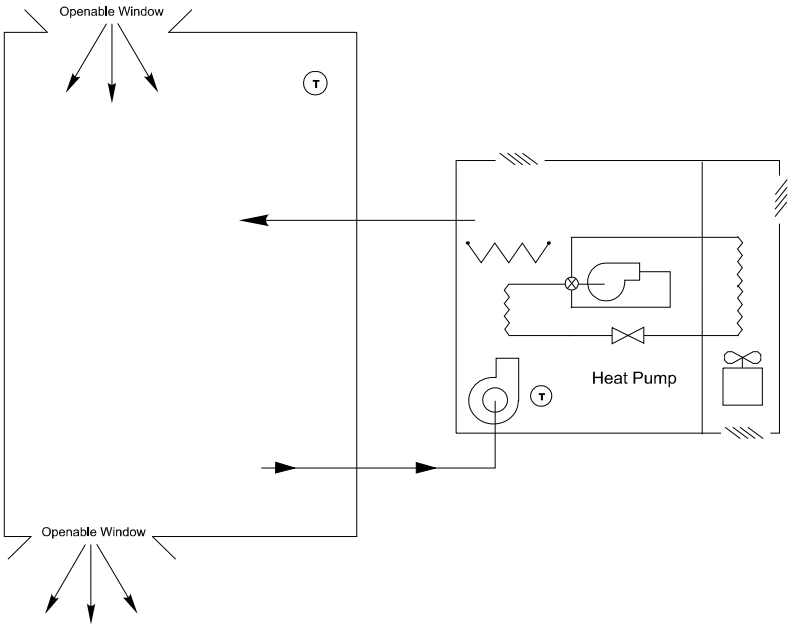


Figure 69 Residential system with heat pump

Standard Temperature Controls

The entire area served by a RESYS unit MUST be modeled as one zone . In cases where the residence is very large and there is considerable diversity in the use of the building, more than one unit can be applied.

On a rise in space temperature, the heating is cycled to hold thermostat setpoints. On a further rise in space temperature, the cooling is cycled using either a direct evaporative cooler or mechanical cooling, but not both. There is no provision for economizer controls.

Input template for a standard RESYS unit with air-to-air heat pump

```
U-name = SYSTEM
   TYPE              = RESYS
   MAX-SUPPLY-T      = 105
   MIN-SUPPLY-T      = 55
   SUPPLY-STATIC     = 1.5                 inches total static
   SUPPLY-EFF        = 0.55                overall fan eff
   FAN-SCHEDULE      = U-NAME              fan run times
   NIGHT-CYCLE-CTRL  = CYCLE-ON-ANY        if fans cycle on to
                                           hold night setpoint
   HEAT-SOURCE       = HEAT-PUMP           if a boiler
   CONTROL-ZONE      = U-NAME
   ..
```

Under ZONE with suggested values

```
U-NAME = ZONE                              different than SPACE
   TYPE              = CONDITIONED
   DESIGN-HEAT-T     = 68
   DESIGN-COOL-T     = 75
   HEAT-TEMP-SCH     = U-NAME              thermostat heat sch
   COOL-TEMP-SCH     = U-NAME              thermostat cool sch
   OA-FLOW/PER       = 15
   SPACE             = U-NAME              of corresponding
   ..                                      SPACE in LOADS module
```

Covered in detail by separate Topics are the following:

- Evaporative Cooling (SYSTEM command)
- Electric and Fuel Meters (SYSTEM, ZONE, ELEC-METER and FUEL-METER commands)
- Building Resources (CIRCULATION-LOOP command)
- Natural Ventilation
- Air Source Heat Pump Enhancements
- Gas Heat Pump
- Residential Evaporative Cooler
- Evaporative Precooler
- Domestic Hot Water Heater

SYSTEM TYPE = RESVVT

Residential Variable-volume, Variable Temperature System

RESVVT is a forced air residential system featuring multiple, individually-ducted and thermostatically-controlled zones. Central heating and cooling are available from a high efficiency variable speed heat pump. Each thermostat controls a motor-driven air damper in the zone duct. These dampers can close completely, giving zero air flow to the zone. Thus some zones can be conditioned, while other zones are permitted to float. The system can also do simultaneous heating and cooling in the sense that it can switch between cooling one group of zones and heating another group of zones within the one hour time step. The electronically commutated motors for the compressor and indoor fan allow the heat pump unit to adjust the FLOW and compressor rpm to meet the total cooling or heating load.

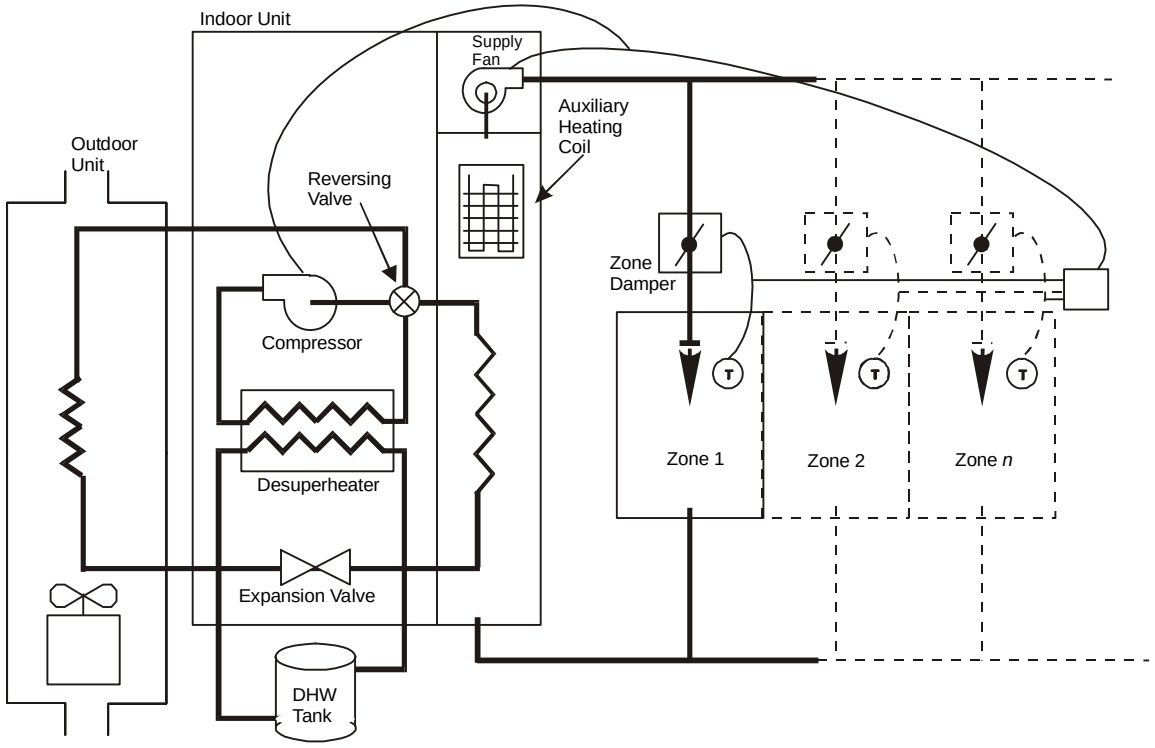


Figure 70 Residential Variable-volume, Variable Temperature System (RESVVT) with variable speed heat pump and desuperheater

The RESVVT system differs from the other residential system model (RESYS) in that RESYS allows only one thermostatically controlled zone. RESYS does not allow the heating or cooling in the non-control zones to be adjusted for different occupancy levels or different desired comfort levels. The ability of the RESVVT system to turn off the conditioning of unoccupied zones through thermostat setup or setback should yield considerable energy savings compared to a RESYS system.

Other names and/or applications for RESVVT

- Some manufacturers offer the RESVVT system so that the heat pump is used to heat domestic hot water. See separate topics on Variable Speed Electric Heat Pumps and Domestic Hot Water Heater and Tank Model.

- The Gas Heat Pump is not applicable to RESVVT system.

Standard Temperature Controls

The RESVVT system most closely resembles a VAV system. It differs primarily in its capability to have the duct air dampers close completely. The RESVVT system is designed to be used in a situation with a high degree of load diversity. For example, in a residence the bedrooms might be unconditioned during the day and early evening, while comfort levels are maintained in the rest of the house. At night the opposite might be true; the bedrooms would be conditioned while the rest of the house could be unconditioned or minimally conditioned. In these circumstances some care should be used in specifying the input. The zone flows cannot be expected to add up to the system flow. Indeed, depending on how the house is zoned and on its expected occupancy patterns, the individual zone flows might each equal the supply flow. You will have to specify by hand the individual zone flows, as well as the supply flow and cooling capacity, since the program cannot size the system correctly to reflect thermostat setups and setbacks that result from diverse occupancy patterns. It is important not to oversize the compressor; if it is oversized it may spend the majority of its operating hours cycling on and off rather than operating in the potentially more efficient variable speed mode .

Input template for a standard RESVVT unit with water coils

(Note: RESVVT has no outside air capability and cannot use natural ventilation. Use infiltration to fulfill this need)

Under SYSTEM with suggested values

```
U-name = SYSTEM
   TYPE              = RESVVT
   MAX-SUPPLY-T      = 105
   MIN-SUPPLY-T      = 55
   CONTROL-ZONE      = U-NAME              of control zone
   HP-SUPP-SOURCE    = ELECTRIC
   COMPRESSOR-TYPE   = HEAT-PUMP
   COOL-CAPACITY     = hand calculated design capacity
   COOL-SH-CAP       = hand calculated design capacity;
                       should be about 75% of COOL-CAPACITY
   SUPPLY-FLOW       = hand calculated peak design flow
                       (i.e., less than sum of ASSIGNED-FLOWs)
   ..
```

Under ZONE with suggested values

```
U-NAME = ZONE                                 different than SPACE
   TYPE              = CONDITIONED
   DESIGN-HEAT-T     = 68
   DESIGN-COOL-T     = 75
   HEAT-TEMP-SCH     = U-NAME              thermostat heat sch
   COOL-TEMP-SCH     = U-NAME              thermostat cool sch
   ASSIGNED-FLOW     = a hand calculated value
                       for each zone is required
    SPACE            = U-NAME              of corresponding
   ..                                       SPACE in LOADS module
```

Changes or additions when using RESVVT for other applications

Residential systems are often installed using multiple units in small commercial buildings. To satisfy minimum outside air energy requirements, we suggest a 100 percent make-up air unit using a packaged single-zone (PSZ)

systems with the supply air feeding to a corridor or dummy core zone. Hold this zone at a constant temperature, relative to and neutral (e.g., 68F) to the zones connected to RESVVT units. All make-up air is then exhausted as relief air.

Covered in detail by separate Topics are the following:

- Night Ventilation (SYSTEM command)
- Service Hot Water heat Pump (SYSTEM and CIRCULATION-LOOP commands)
- Electric and Fuel Meters (SYSTEM, ZONE, ELEC-METER and FUEL-METER commands)
- Building Resources (CIRCULATION-LOOP command)
- Domestic Hot Water Heater and Tank Model
- Variable Speed Electric Heat Pump

Section

5

Central Plant Components

As described in the previous section's introduction, HVAC equipment can be divided into equipment directly involved with conditioning air, (air-side or secondary equipment), and equipment that supports the airside functions (water-side, plant, or primary equipment). The previous section described the modeling of the air-side equipment. This section describes the central plant equipment. Included in this category are:

4. Boilers - gas and electric
5. Chillers - electric, hot-water absorption, gas-absorption, and engine-driven
6. Heat-rejection equipment - cooling towers, fluid coolers, dry coolers, and ground-loop heat-exchangers
7. Domestic water heaters - gas, electric, and heat-pump
8. Circulation loops and pumps - hot-water, chilled-water, condenser water, domestic water, and water-loop heat pumps
9. Electric generators – engine and turbine
10. Utility meters – electric, fuel, steam, and chilled water

CIRCULATION-LOOP

The SYSTEMS and PLANT programs from DOE-2.1E are combined into a single module called HVAC. One reason for this is to improve the connectivity between the loads incurred by the secondary HVAC systems (air handler coils, reheat coils, etc.) and the primary HVAC equipment (boilers, chillers, etc.) For this purpose, the concept of circulation loops has been implemented.

A circulation loop (sometimes called simply "loop") allows equipment with a thermal demand (a heating coil, for example) to be connected to a thermal supplier (a boiler, for example). Different coils may be connected to the same or different loops, and each loop may serve an entire system, multiple systems, or only a portion of a single system. A new command, CIRCULATION-LOOP, allows you to enter information about a loop. New keywords allow you to assign secondary and primary components like coils, pumps, boilers, chillers, etc., to loops.

Loop connections

Since each loop is connected to specific coils, different coils in the same system may be on different loops, and the connections are as flexible as is possible. For example, the central heating coil of a single-zone reheat system (SZRH) may be on one loop, the reheat coils on another loop, and the baseboards on yet another loop. Moreover, not all the reheat coils need be on the same loop; they can be assigned to different loops on a zone-by-zone basis.

The only restriction to connecting a load to a loop is that the load and loop must be of compatible types. Using the previous example, a reheat coil may be connected to any of the hot water loops, but may not be connected to a chilled water loop. The same is true on the supply side. A boiler may be connected to a hot water loop but not to a chilled water loop. Also, each demander or supplier may be connected to only one loop. A coil cannot draw heat from more than one loop, nor can a boiler supply heat to more than one loop (but see the discussion of primary/secondary loops, below).

Loop types

The program incorporates a variety of loop types as indicated by the TYPE keyword in the CIRCULATION-LOOP command. They are:

- Chilled water (TYPE = CHW)
  A chilled water loop supplies chilled water to cooling coils. The cooling coils may be in central cooling equipment, such as in a VAV system, or in zonal equipment, such as a in a fan-coil unit. In addition, a chilled water loop may be assigned to handle a process cooling load, such as a manufacturing load or a computer room unit load. Chilled water loops pass their load onto one or more chillers or to a chilled water utility.

- Hot water (TYPE = HW)
  A hot water loop supplies hot water to heating coils. The coil types include preheat, central heating, reheat, baseboards, supplemental heat for air-to-air heat pumps, and radiant floor panel systems. In addition, a hot water loop may be assigned to handle a process heating load, such as a manufacturing load. Hot water loops pass their load onto one or more boilers or to a hot water/steam utility.

- Two-pipe (TYPE = PIPE-2)
  A two-pipe loop supplies both chilled water and hot water to coils, but never during the same hour. The coils served may be any or all of the coils listed for chilled water and hot water loops. When the loop is in the heating mode, any cooling loads that may co-exist will not be met and the temperature of the zones demanding cooling will float accordingly. The same is true for heating loads once the loop has switched over to cooling. The loop may change over from heating to cooling on the basis of an indoor or outdoor temperature or according to a schedule.

- Condenser water (TYPE = CW)
  This loop accepts the heat rejection loads of chillers, packaged direct expansion (DX) HVAC systems with water-cooled condensers, water-side economizers, and electrical generators. The loop rejects the condenser heat to one or more cooling towers.

- Water-loop heat pump (TYPE = WLHP)
  This type of loop is primarily intended to serve water-loop heat-pump systems (system type HP). Individual heat pump units operate according to their zone thermostats and reject heat to or take heat from the loop to which they are assigned. Depending on the relative number of units operating in the heating or cooling mode, the loop will be thermally unbalanced and the loop temperature will either rise or fall. One or more boilers and one or more cooling towers operate to keep the loop within specified temperature limits. The loop assignment may be by system or zone by zone.

  In addition to heat pump units, any cooling unit with a water-cooled condenser may be assigned to a loop of this type. For example, a DX cooling unit serving a computer room can reject its heat to a WLHP loop so that the computer room heat can be recovered by a WLHP unit. Chillers may also be assigned to this loop.

- Domestic hot water (TYPE = DHW)
  A domestic hot water loop supplies potable heated water to kitchens, bathrooms, etc. No coils can be assigned to this type of loop; however, process loads defined in the SPACE command can.While rare, heating coils can be assigned to this loop, so that heating loads are met by a DW-HEATER. This is sometimes done in apartments or other relatively small residences in mild climates.

Flow and temperature control

Each loop can be constant flow or variable flow, and constant temperature or variable temperature. Furthermore, some of the loads attached to a loop may be constant flow while others may be variable flow. Loop temperatures may be fixed, scheduled, reset on outside air or zone air temperature, or reset according to load.

Pumps

Each loop may have one or more pumps. In variable-flow loops, the pumps can modulate their capacity by riding the pump curve, staging, or through two-speed or variable-speed operation. The pump's head set-point can be fixed or vary with load, and the differential head pressure sensor can be located at various places on the loop.

The pressure loss, or pumping head, of a loop is not specified directly. Instead, the head is defined for each component in the loop, such as coils, piping and chillers. This is so that the program can properly calculate the head as the flow varies in different parts of the loop. In addition, a static head can be specified for components that are at atmospheric pressure, such as a chilled-water storage tank, and that connect to a loop that is pressurized.

Hours of operation

A loop may be activated in either a passive or active fashion. In the passive mode, the loop will run whenever there is non-zero coil load. Alternatively, the loop may operate in a standby mode where it is active whenever there is an HVAC system attached to it that is running with heating/cooling schedules activated. Finally, it may operate in a scheduled mode where the heating/cooling modes of associated air handlingair-handling systems are locked out when the loop is scheduled off.

Thermal losses

Each loop can be modeled with or without thermal losses taken into account. Thermal losses may be to a conditioned zone, to the outdoors, to a tunnel, or to the ground. Some loops may be indoors and others outdoors, and one or more indoor loops may be in the same or different zones. On an hourly basis, the thermal losses from

each loop will vary according to the loop temperature (both supply and return), the temperature of the loop's environment, and the loop flow rate.

Heat recovery

Heat may be recovered from a waste heat source to a loop. For example, the waste heat from a chiller may be attached to a hot water loop and heat will be recovered whenever the return temperature of the loop is less than the maximum allowable condensing temperature of the chiller. In a similar fashion, heat may be recovered from the jacket of an engine-generator by attaching the jacket to a loop.

Primary, secondary and equipment-recirculation loops

A single loop can deliver energy directly from the central plant equipment to the end uses, or two or more loops can be arranged in a primary/secondary fashion, where a primary loop serves one or more secondary loops. A primary loop may be constant flow while one or more of its secondaries is variable flow, and vice versa. In addition to primary/secondary loops, central plant equipment (boilers, chillers, etc.) can have equipment-recirculation loops dedicated to individual pieces of equipment, and connecting that equipment to the primary loop.

The program can simulate any number of primary and/or secondary loops, subject only to the available size of data arrays. Some of the primary loops may have secondary loops and others not. By definition, a secondary loop cannot exist without a primary loop to which it is attached. The number of equipment-recirculation loops is simply a function of the number and type of boilers, chillers, etc., that are present.

The introduction of loops has caused four of the previously existing system types to become consolidated into two new system types. The 2-pipe and 4-pipe fan-coil systems (TPFC and FPFC) are now simply fan-coil systems (FC). The 2-pipe and 4-pipe induction units (TPIU and FPIU) are now simply induction units (IU). To simulate a 2-pipe system, connect the fan-coils to a loop with TYPE = PIPE-2. A 4-pipe system is simulated by connecting the hot-water and chilled-water coils to hot-water and chilled-water loops, respectively. As previously mentioned, coils in different zones may be connected to different loops.

Primary loops

A primary loop is the most basic loop; it is the only mandatory component of a loop system. A primary loop can serve the demands of zero or more coils and zero or more secondary loops. It may also have zonal process loads attached as well as a process load not specific to any zone (such as a manufacturing load). To meet these loads, the primary loop may have zero or more pieces of central plant equipment attached or it may be attached to the appropriate utility energy source.

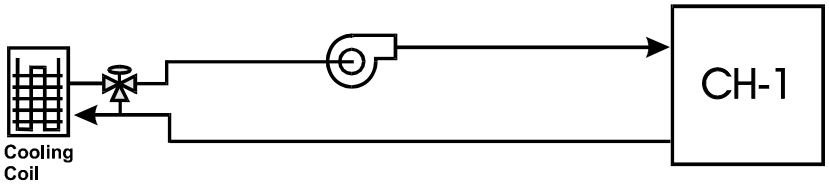


Figure 71 Simple chilled-water primary loop consisting of a chiller, piping, a pump, and a cooling coil

A primary loop usually has a pump and always has a temperature control setpoint.

Secondary loops

A secondary loop is identical to a primary loop except that it is supplied by a primary loop rather than central plant equipment. A secondary loop may have zero or more coils attached and it may have the process loads described above for the primary loops. A secondary loop is attached to a primary loop that handles all of the secondary loop loads.

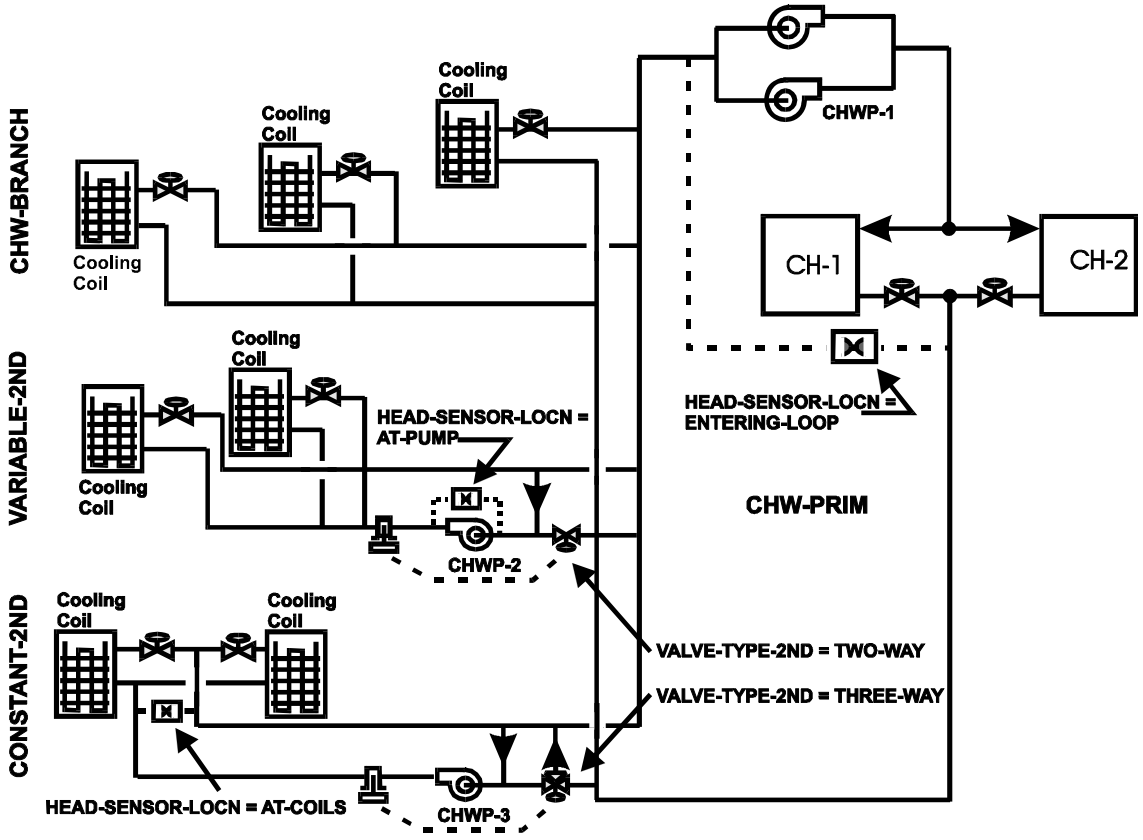


Figure 72 Examples of secondary loops attached to a primary loop

A secondary loop may not be attached to another secondary loop; nor can it be attached directly to any central plant equipment.

A secondary loop may or may not have its own circulation pump. When a secondary pump exists, the secondary loop may have its own operation schedule and temperature setpoint, subject to the availability and temperature of the energy in the primary loop. When a secondary pump does not exist the secondary loop operates in a purely passive fashion: it passes its load, temperature, flow, and head requirements on to the primary loop. This pumpless secondary loop is in actuality an extension of the primary loop; its main value is in allowing different legs of a primary loop to be in different zones so that the program can more accurately model the thermal losses. In addition, the coil and piping head losses can be assigned more accurately.

Equipment-recirculation loops

An equipment-recirculation loop is dedicated to an individual piece of central plant equipment. It is active only when its associated equipment is active. It does not have its own temperature control; instead, it controls to the temperature setpoint of the primary loop. An equipment-recirculation loop is assumed to be small in length and to have no thermal losses.

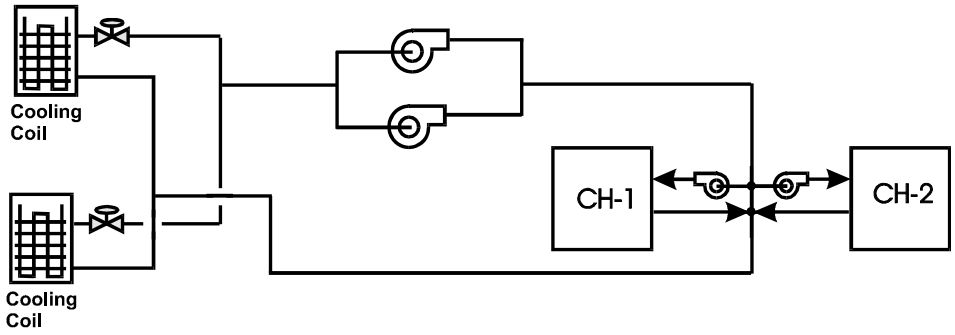


Figure 73 A primary CHW loop having two chillers, each with its own recirculation loop and pump.

An equipment-recirculation loop always has its own pump and can be constant flow or variable flow, independent of the primary loop flow. Some types of plant equipment can have more than one equipment-recirculation loop and pump. For example, an absorption chiller can have four equipment-recirculation pumps: one for the evaporator, one for the hot water supply, one for the condenser, and one for heat recovery.

Examples of Loop Types

The following examples show different circulation loop features and illustrate possible loop configurations. These examples are for chillers and their loops; similar configurations are possible for the other equipment types. Note that only the keywords that illustrate the relevant points are shown; additional keywords may be required for simulation.

Example 1. Basic Loop

This example illustrates the simplest possible loop (Figure 74). It consists of a loop, a single chiller, a single pump, and a single cooling coil. Since there is only one chiller, the pump may be attached directly to the loop via the CIRCULATION-LOOP:LOOP-PUMP keyword, or via the CHILLER:CHW-PUMP keyword with no difference in the results. Note that the coil on this loop has a 3-way valve, which causes the loop to be constant flow. The chiller is air cooled, so that no condenser water loop is necessary.

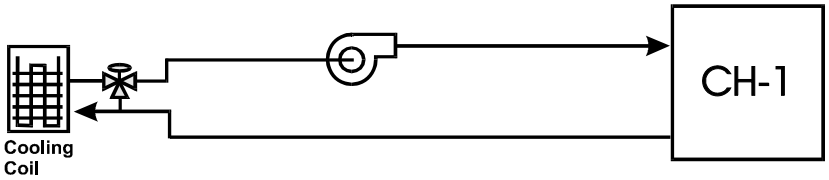


Figure 74 Simple loop

```
CHW-PUMP = PUMP ..

COOL-LOOP = CIRCULATION-LOOP
   TYPE               = CHW
   LOOP-PUMP          = CHW-PUMP ..

CH-1 = CHILLER
   TYPE               = ELEC-OPEN-CENT
   CHW-LOOP           = COOL-LOOP
   CONDENSER-TYPE     = AIR COOLED ..
```

```
SYST-1 = SYSTEM
   TYPE               = VAVS
   CHW-LOOP           = COOL-LOOP
   CHW-VALVE-TYPE     = THREE-WAY ..
```

For all primary loops, the pump is always located on the return leg of the loop, upstream of the chillers or other plant components. For secondary loops, the pump is always located on the supply leg of the loop, as illustrated in other examples. This distinction is not normally important, as the impact of pump heat on loop temperature is normally very small. It may become important when the loop is variable flow and the pump has no means of capacity control other than riding its curve; in this case, very small flows can experience large temperature rises across the pump (in extreme cases, pump heat can cause the loop flow to boil).

Example 2. Variable-Flow Loop

This example expands upon Example (1) by adding two more coil loads. The first coil has a three-way valve, and the second and third coils have two way valves. This causes the loop to be variable flow. The minimum loop flow is equal to the design flow of the coil with the 3-way valve.

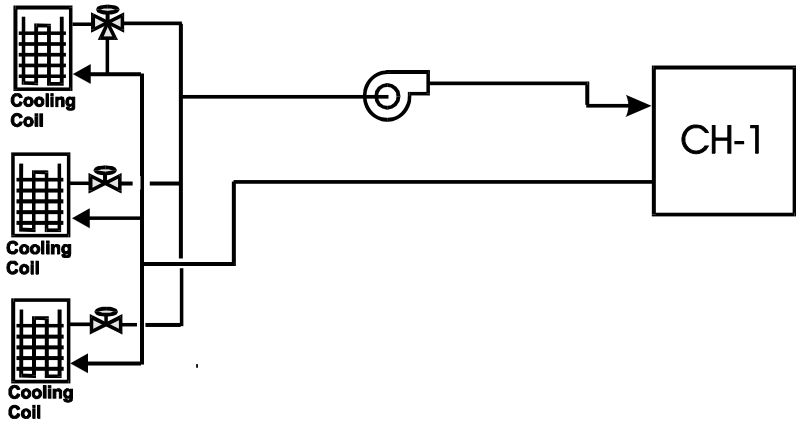


Figure 75 Loop with variable flow

```
CHW-PUMP = PUMP ..

COOL-LOOP = CIRCULATION-LOOP
   TYPE               = CHW
   LOOP-PUMP          = CHW-PUMP ..

CH-1 = CHILLER
   TYPE               = ELEC-OPEN-CENT
   CHW-LOOP           = COOL-LOOP
   CONDENSER-TYPE     = AIR COOLED ..

SYST-1 = SYSTEM
   TYPE               = VAVS
   CHW-LOOP           = COOL-LOOP
   CHW-VALVE-TYPE     = THREE-WAY ..

SYST-2 = SYSTEM
   LIKE               = SYST-1
   CHW-VALVE-TYPE     = TWO-WAY ..
```

```
SYST-3 = SYSTEM
   LIKE                = SYST-1
   CHW-VALVE-TYPE      = TWO-WAY ..
```

Figure 76 illustrates the location of the loop temperature sensor and the pump head pressure sensor. The loop temperature sensor is always located on the supply-side of the chillers or other loop components. The location of the pump head pressure sensor varies according to the HEAD-SENSOR-LOCN.

Note that a head pressure sensor is used only when the pumps have some means of capacity control, and HEAD-SETPT-CTRL = FIXED.

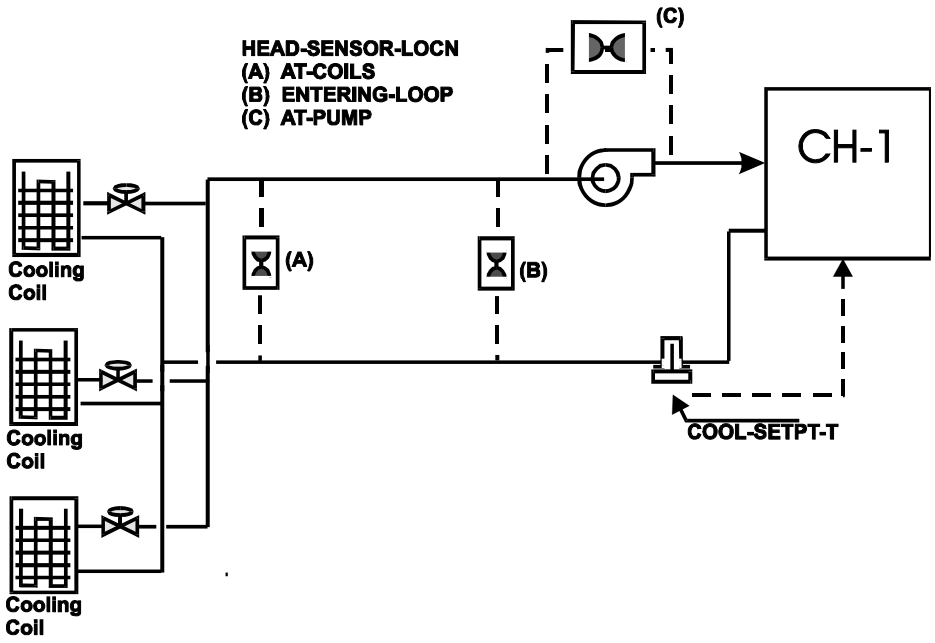


Figure 76 Variable flow configuration showing location of loop temperature sensor and pump head pressure sensor

Example 3. Variable-Flow with Bypass

This example is similar to Example 2 but all of the coils have 2-way valves. If all of the valves are closed or mostly closed, the loop flow may drop below what is acceptable to the chiller. This can be solved by adding a bypass to the loop. Specifying LOOP-MIN-FLOW = 0.5 will cause the bypass valve to open whenever the flow drops below 50% of design, and will modulate so that the flow back to the chiller is never less than 50%. Alternatively, LOOP-RECIRC-FLOW = 0.5 will cause a constant 50% of design flow to bypass, in addition to the coil flows. Obviously, using LOOP-RECIRC-FLOW instead of LOOP-MIN-FLOW will use more pumping energy.

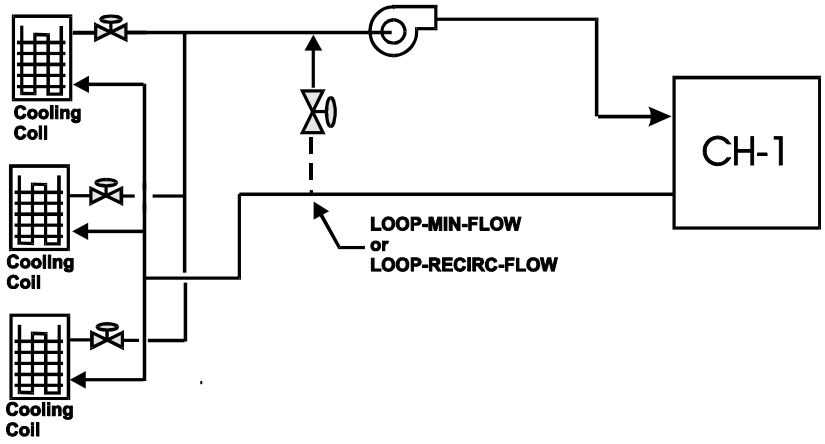


Figure 77 Variable flow with bypass

```
CHW-PUMP = PUMP ..

COOL-LOOP = CIRCULATION-LOOP
   TYPE               = CHW
   LOOP-PUMP          = CHW-PUMP
   LOOP-MIN-FLOW      = 0.5 ..

CH-1 = CHILLER
   TYPE               = ELEC-OPEN-CENT
   CHW-LOOP           = COOL-LOOP
   CONDENSER-TYPE     = AIR COOLED ..

SYST-1 = SYSTEM
   TYPE               = VAVS
   CHW-LOOP           = COOL-LOOP
   CHW-VALVE-TYPE     = TWO-WAY ..

SYST-2 = SYSTEM
   LIKE               = SYST-1  ..

SYST-3 = SYSTEM
   LIKE               = SYST-1  ..
```

Example 4. Variable-Flow Loop, Constant-Flow Chiller

This example illustrates a configuration which is variable flow to the coils but constant flow at the chiller. The chiller has its own recirculation pump that decouples its flow from the loop flow. If the sizing and head of the two pumps are allowed to default, the program will size both pumps equally. However, the head of the chiller pump will include only the chiller head, and the head of the loop pump will include the head of the loop and coils, but will exclude the chiller head. Therefore, while both pumps have the same flow capacity, the loop pump will usually have a greater head and require a larger motor.

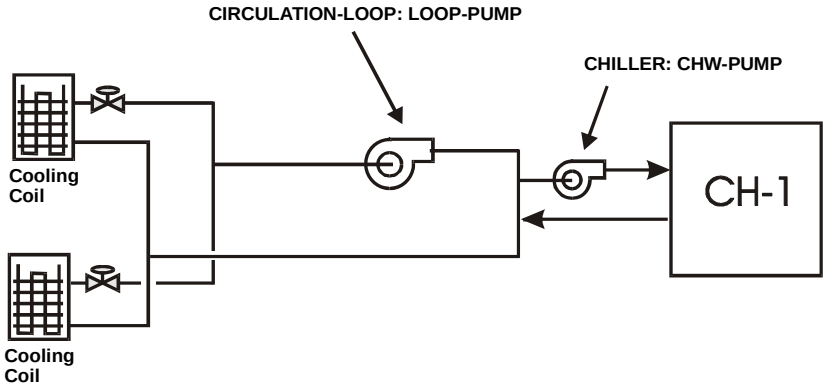


Figure 78 Variable-flow loop, Constant-flow chiller

```
CHW-PUMP = PUMP ..

CHILLER-PUMP = PUMP ..

COOL-LOOP = CIRCULATION-LOOP
   TYPE               = CHW
   LOOP-PUMP          = CHW-PUMP ..

CH-1 = CHILLER
   TYPE               = ELEC-OPEN-CENT
   CHW-LOOP           = COOL-LOOP
   CHW-PUMP           = CHILLER-PUMP
   CHW-FLOW-CTRL      = CONSTANT-FLOW
   CONDENSER-TYPE     = AIR COOLED ..

SYST-1 = SYSTEM
   TYPE               = VAVS
   CHW-LOOP           = COOL-LOOP
   CHW-VALVE-TYPE     = TWO-WAY ..

SYST-2 = SYSTEM
   LIKE               = SYST-1  ..
```

Example 5. Variable-Flow Loop, Constant-Flow Chillers

This example is similar to Example 4, but has two chillers, each with its own recirculation pump. The loop is variable-flow, and the chillers are constant-flow. Note that the current version of the program assumes the chillers are in parallel; series or series/parallel arrangements are not simulated.

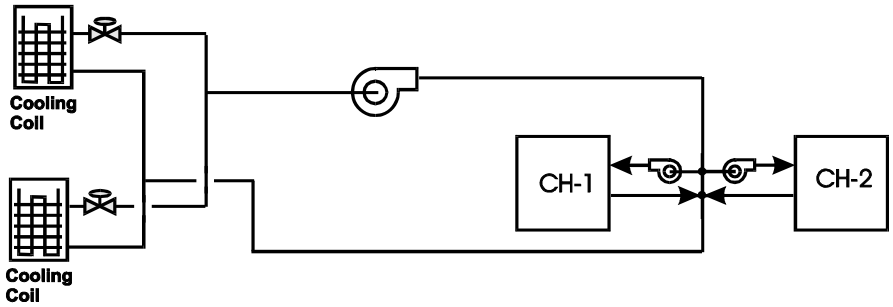


Figure 79 Variable-flow loop, constant-flow chillers

```
CHW-PUMP = PUMP ..

CH-1-PUMP = PUMP ..

CH-2-PUMP = PUMP ..

COOL-LOOP = CIRCULATION-LOOP
   TYPE               = CHW
   LOOP-PUMP          = CHW-PUMP ..

CH-1 = CHILLER
   TYPE               = ELEC-OPEN-CENT
   CHW-LOOP           = COOL-LOOP
   CHW-PUMP           = CH-1-PUMP
   CHW-FLOW-CTRL      = CONSTANT-FLOW
   CONDENSER-TYPE     = AIR COOLED ..

CH-2 = CHILLER
   LIKE               = CH-1
   CHW-PUMP           = CH-2-PUMP  ..

SYST-1 = SYSTEM
   TYPE               = VAVS
   CHW-LOOP           = COOL-LOOP
   CHW-VALVE-TYPE     = TWO-WAY ..

SYST-2 = SYSTEM
   LIKE               = SYST-1 ..
```

Example 6. Variable-Flow Loop with Staged Pumps, Constant-Flow Chillers

This example is similar to Example 5, but the loop has two pumps instead of one. By default, each pump will be sized at 50% of the design loop flow, but at 100% of the required head. The program will stage the pumps as required by the flow.

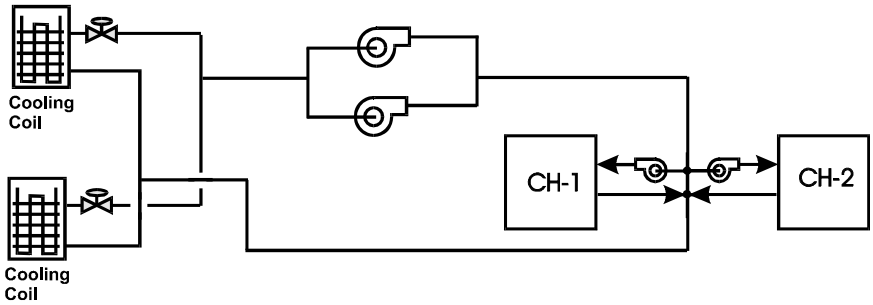


Figure 80 Variable-flow loop, constant-flow chiller, staged loop pumps

```
CHW-PUMP = PUMP
   NUMBER              = 2 ..

CH-1-PUMP = PUMP ..

CH-2-PUMP = PUMP ..

COOL-LOOP = CIRCULATION-LOOP
   TYPE                = CHW
   LOOP-PUMP           = CHW-PUMP ..

CH-1 = CHILLER
   TYPE                = ELEC-OPEN-CENT
   CHW-LOOP            = COOL-LOOP
   CHW-PUMP            = CH-1-PUMP
   CHW-FLOW-CTRL       = CONSTANT-FLOW
   CONDENSER-TYPE      = AIR COOLED ..

CH-2 = CHILLER
   LIKE                = CH-1
   CHW-PUMP            = CH-2-PUMP ..

SYST-1 = SYSTEM
   TYPE                = VAVS
   CHW-LOOP            = COOL-LOOP
   CHW-VALVE-TYPE      = TWO-WAY ..

SYST-2 = SYSTEM
   LIKE                = SYST-1  ..
```

Example 7. Variable-Flow Loop and Variable-Flow Chillers

This example is similar to Example 6 except the chillers do not have recirculation pumps. Flow through each chiller therefore varies as the coil flow, and according to whether one or both chillers are operating. As before, the loop pumps stage with the flow. Note that the program assumes that an inactive chiller has a valve that isolates it from the loop. In other words, the chilled water flow is allocated only to chillers that are actively meeting the load.

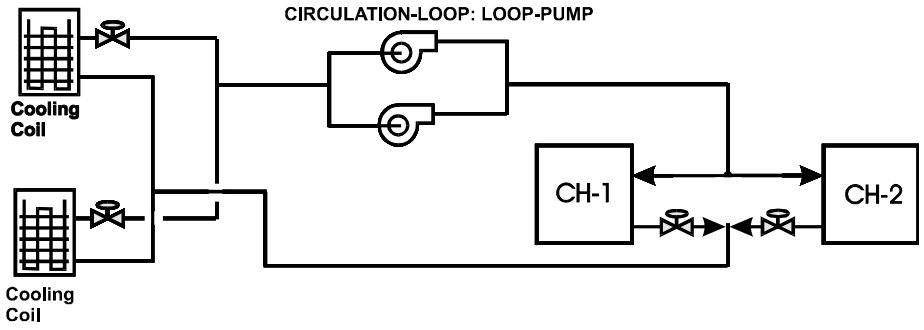


Figure 81 Two chillers, variable flow

In this example, an interesting situation may arise if the coil valves are all 3-way instead of 2-way. In this case, the loop will be constant-flow even at very low loads. Assume that the loop load is 40% of design, and both chillers are equally sized. The program will first attempt to allocate the load to the chillers, and will select a single chiller since one chiller can easily handle the load. However, since the loop is constant flow, one chiller operating would have to handle twice its design flow (at approximately 4 times is design head). A flow rate this high could result in severe tube erosion in the evaporator. In addition,s the excessively high pressure drop through the evaporator would probably not allow the pump to meet the required flow rate.

Each hour, the program first allocates the load to the equipment (by default or through the COOL-EQUIP-CTRL sequence defined by you). It then checks to see if the flow through each equipment component does not exceed the CHILLER:CHW-MAX-FLOW. If the flow exceeds this limit, then the program turns on as many equipment units as required to meet the flow requirements.

In this example, a 40% loop load would be met by both chillers running, each operating as 20% capacity. To prevent the program from reallocating the load based on flow, set the CHW-MAX-FLOW to a higher value (a value of 2.0 or higher in the above example would suffice). Also you should specify the head of the pump, and take into account the high chiller head requirement when operating in this mode.

```
CHW-PUMP = PUMP
   NUMBER              = 2 ..

COOL-LOOP = CIRCULATION-LOOP
   TYPE                = CHW
   LOOP-PUMP           = CHW-PUMP ..

CH-1 = CHILLER
   TYPE                = ELEC-OPEN-CENT
   CHW-LOOP            = COOL-LOOP
   CHW-MAX-FLOW        = 2.0
   CONDENSER-TYPE      = AIR COOLED ..

CH-2 = CHILLER
   LIKE                = CH-1  ..

SYST-1 = SYSTEM
   TYPE                = VAVS
   CHW-LOOP            = COOL-LOOP
   CHW-VALVE-TYPE      = TWO-WAY ..

SYST-2 = SYSTEM
   LIKE                = SYST-1  ..
```

Example 8. Variable-Flow Loop, Variable-Flow Chillers with Dedicated Pumps

This example is similar to Example 7, but the loop does not have its own pumps. Instead, a pump is dedicated to each chiller. Flow through each chiller therefore varies as the coil flow, and according to whether one or both chillers are operating. This arrangement is more efficient than Example (7), as the pumps stage with the chillers.

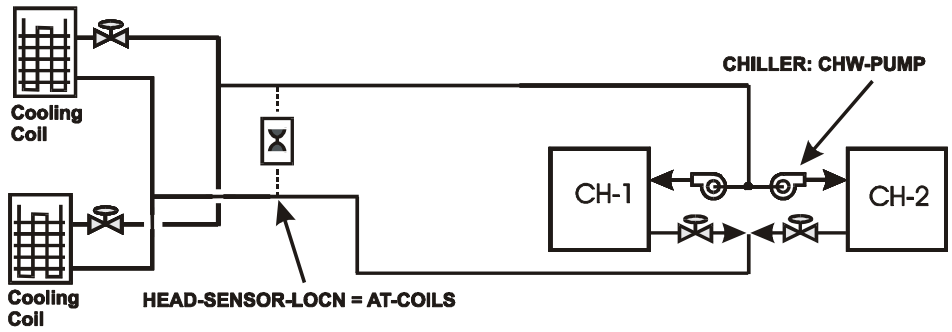


Figure 82 Two chillers, variable flow, dedicated pumps

When the loop is powered by chiller pumps rather than its own pumps, the chiller pumps can have the same capacity control mechanisms as a loop pump. Each chiller can have multiple pumps (staged capacity), or a 2-speed or variable-speed pump. If the CIRCULATION-LOOP:HEAD-SETPT-CTRL = FIXED, then the chiller pumps will control according to the loop's HEAD-SETPT at the HEAD-SENSOR-LOCN. Alternatively, when the HEAD-SETPT-CTRL = VALVE-RESET, the chiller pumps will modulate so that one of the coil valves is fully open.

```
CH-1-PUMP = PUMP
   CAP-CTRL           = VAR-SPEED-PUMP  ..

CH-2-PUMP = PUMP
   CAP-CTRL           = VAR-SPEED-PUMP ..

COOL-LOOP = CIRCULATION-LOOP
   TYPE               = CHW
   HEAD-SETPT-CTRL    = FIXED
   HEAD-SENSOR-LOCN   = AT-COILS
   HEAD-SETPT         = 30  ..

CH-1 = CHILLER
   TYPE               = ELEC-OPEN-CENT
   CHW-LOOP           = COOL-LOOP
   CHW-PUMP           = CH-1-PUMP
   CONDENSER-TYPE     = AIR COOLED ..

CH-2 = CHILLER
   LIKE               = CH-1
   CHW-PUMP           = CH-2-PUMP ..

SYST-1 = SYSTEM
   TYPE               = VAVS
   CHW-LOOP           = COOL-LOOP
   CHW-VALVE-TYPE     = TWO-WAY ..
```

```
SYST-2 = SYSTEM
   LIKE              = SYST-1  ..
```

Example 9. Primary/Secondary Loops

This example illustrates how secondary loops attach to a primary loop. The primary loop is called CHW-PRIMARY. Attached to the primary loop are three secondary loops, as well as a coil (for the sake of brevity, coil attachments are not included in the input for this example, but a coil can attach to either a primary or secondary loop.)

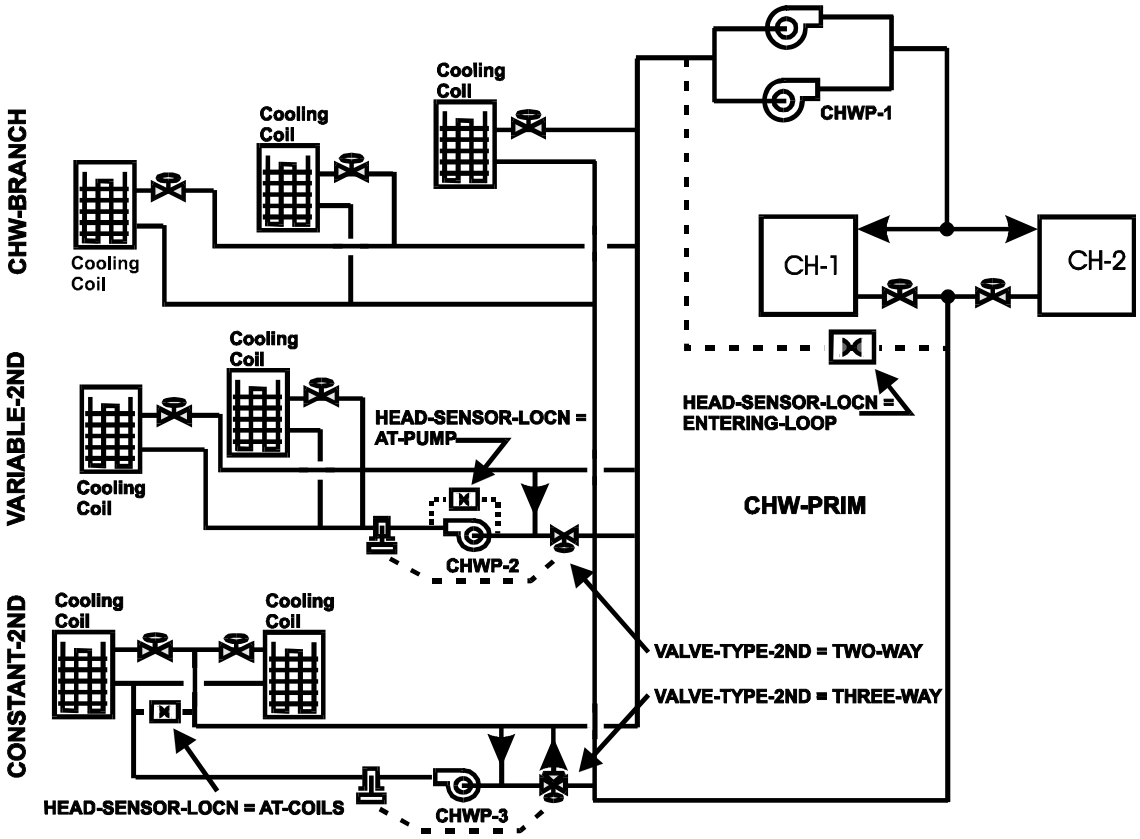


Figure 83 Secondary loops attached to a primary loop

The uppermost secondary loop, CHW-BRANCH, does not have its own pump. In this case, the primary loop pump also powers the secondary loop. In addition, this type of loop cannot have any type of temperature control independent of the primary loop.

The middle secondary loop, VARIABLE-2ND, is a variable-flow loop, both within the loop as well as in its interface to the primary loop. Because this loop has its own pump, it can have its own temperature control setpoint (the default is the primary loop temperature). In this case, the loop will draw water from the primary loop and mix it with water returning from the secondary loop to maintain the secondary loop setpoint. Note that the temperature in the secondary loop is limited by the temperature in the primary loop. In other words, if a primary chilled-water loop is operating at 45F (7.2C), the secondary loop cannot operate at 40F (4.4C).

The bottom secondary loop, CONSTANT-2ND, is variable-flow within the secondary loop (and secondary pump), but constant-flow in its interface to the primary loop. Because it has its own pump, this loop may have its own temperature control setpoint. In secondary loops, the pump head sensor, if used, may be located at either AT-

PUMP or AT-COILS. Specifying ENTERING-LOOP is the same as AT-PUMP. The secondary pumps may have capacity control schemes that are independent of the primary loop.

```
CHWP-1 = PUMP
   NUMBER             = 2  ..

CHWP-2 = PUMP
   CAP-CTRL           = VAR-SPEED-PUMP ..

CHWP-3 = PUMP
   CAP-CTRL           = TWO-SPEED-PUMP ..

CHW-PRIM = CIRCULATION-LOOP
   TYPE               = CHW
   LOOP-PUMP          = CHWP-1
   HEAD-SETPT-CTRL    = FIXED
   HEAD-SENSOR-LOCN   = ENTERING LOOP
   HEAD-SETPT         = 30
   COOL-SETPT-T       = 45 ..

BRANCH = CIRCULATION-LOOP
   TYPE               = CHW
   SUBTYPE            = SECONDARY
   PRIMARY-LOOP       = CHW-PRIM  ..

VARIABLE-2ND = CIRCULATION-LOOP
   TYPE               = CHW
   SUBTYPE            = SECONDARY
   PRIMARY-LOOP       = CHW-PRIM
   LOOP-PUMP          = CHWP-2
   HEAD-SENSOR-LOCN   = AT-PUMP
   VALVE-TYPE-2ND     = TWO-WAY
   COOL-SETPT-T       = 50.  ..

CONSTANT-2ND = CIRCULATION-LOOP
   TYPE               = CHW
   SUBTYPE            = SECONDARY
   PRIMARY-LOOP       = CHW-PRIM
   LOOP-PUMP          = CHWP-3
   HEAD-SENSOR-LOCN   = AT-COILS
   VALVE-TYPE-2ND     = THREE-WAY
   COOL-SETPT-T       = 48 ..

CH-1 = CHILLER
   TYPE               = ELEC-OPEN-CENT
   CHW-LOOP           = CHW-PRIM
   CONDENSER-TYPE     = AIR COOLED ..

CH-2 = CHILLER
   LIKE               = CH-1  ..
```

Configuring Condenser Loops

The next set of examples illustrate how condenser loops can be configured. Note that, when multiple chillers exist in a plant, the chillers may be connected to condenser loops independently of their connections to chilled-water

loops. For example, chillers serving a common chilled-water loop may reject heat to either a common condenser water loop, or to separate condenser water loops. Conversely, chillers serving separate chilled-water loops may reject heat to either a common or separate condenser water loops. In the following examples, only the condenser loop configurations are illustrated; any of the chilled-water configurations illustrated previously may be mated with any of these condenser water configurations.

Example 10. Condenser Loop with Two Cooling Towers Serving Three Chillers

This example shows three chillers coupled to two cooling towers via a single condenser water loop. The loop has two pumps. The flow in the condenser water loop varies according to the number of chillers operating, and the loop pumps and towers will stage accordingly.

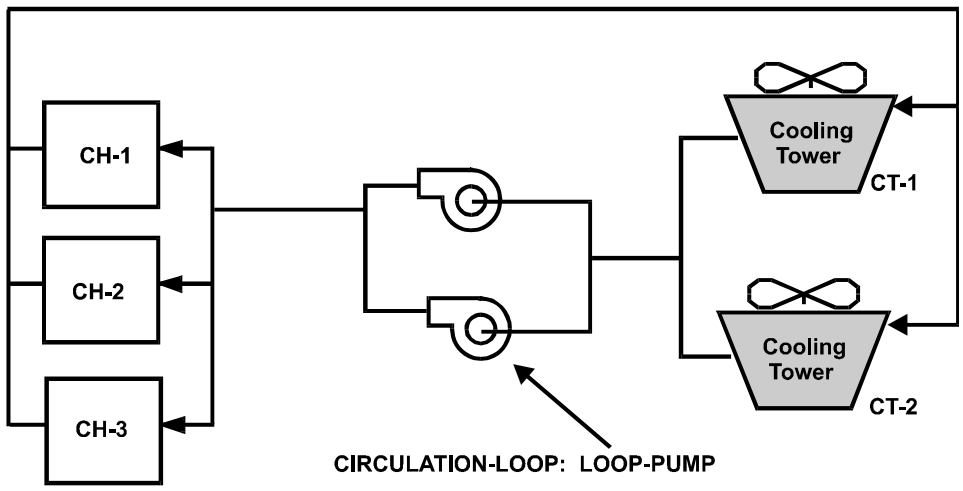


Figure 84 Condenser loop with two cooling towers servinig three chillers

```
CWP-1 = PUMP
   NUMBER             = 2  ..

CONDENSER-LOOP = CIRCULATION-LOOP
   TYPE               = CW
   LOOP-PUMP          = CWP-1  ..

CH-1 = CHILLER
   TYPE               = ELEC-OPEN-CENT
   CHW-LOOP           = CHILLED-LOOP         not shown
   CONDENSER-TYPE     = WATER-COOLED
   CW-LOOP            = CONDENSER-LOOP  ..

CH-2 = CHILLER
   LIKE               = CH-1  ..

CH-3 = CHILLER
   LIKE               = CH-1  ..

TWR-1 = HEAT-REJECTION
   TYPE               = OPEN-TWR
   CW-LOOP            = CONDENSER-LOOP  ..
```

```
TWR-2 = HEAT-REJECTION
   LIKE                = TWR-1  ..
```

Example 11. Condenser Loop with Dedicated Pumps on Chillers

This example is similar to Example 10, except that each chiller has its own dedicated condenser water pump. In this case, each chiller's condenser pump stages with the chiller. Flow in the loop (and thru the towers) varies according to the number of chillers operating.

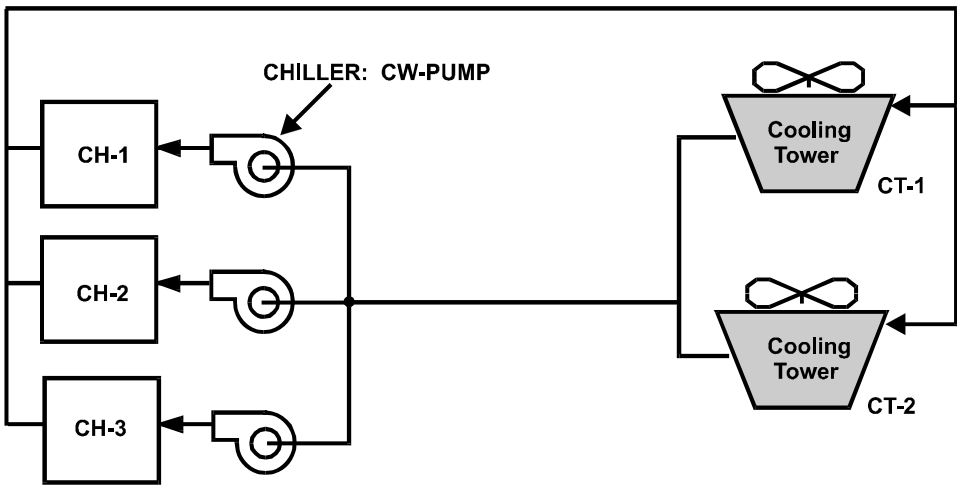


Figure 85 Condenser loop with dedicated pumps on chillers

```
CWP-1 = PUMP  ..

CWP-2 = PUMP  ..

CWP-3 = PUMP  ..

CONDENSER-LOOP = CIRCULATION-LOOP
   TYPE               = CW  ..

CH-1 = CHILLER
   TYPE               = ELEC-OPEN-CENT
   CHW-LOOP           = CHILLED-LOOP        not shown
   CONDENSER-TYPE     = WATER-COOLED
   CW-LOOP            = CONDENSER-LOOP
   CW-PUMP            = CWP-1  ..

CH-2 = CHILLER
   LIKE               = CH-1
   CW-PUMP            = CWP-2  ..

CH-3 = CHILLER
   LIKE               = SYST-1
   CW-PUMP            = CWP-3  ..

TWR-1 = HEAT-REJECTION
   TYPE               = OPEN-TWR
   CW-LOOP            = CONDENSER-LOOP  ..
```

```
TWR-2 = HEAT-REJECTION
   LIKE                = TWR-1  ..
```

Example 12. Condenser Loop with Dedicated Chiller Pumps and Tower Recirculation Pumps

This example shows how pumps can be dedicated to each chiller condenser, as well as to each cooling tower. In this mode the loop flow is decoupled from the tower flow; each pump stages with its respective chiller or tower.

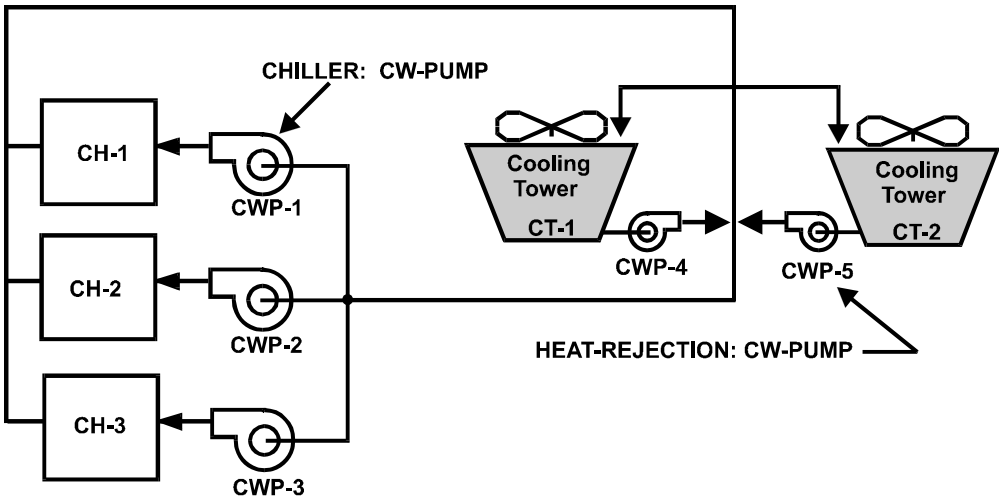


Figure 86 Condenser loop with dedicated chiller pumps and tower recirculation pumps

In this example, the flow through each tower is constant, independent of the condenser loop flow, because CW-FLOW-CTRL = CONSTANT-FLOW. If VARIABLE-FLOW was specified instead, then the flow through each tower would match the loop flow. If both towers are operating, then the flow to each tower would be prorated on the basis of capacity.

```
CWP-1 = PUMP  ..

CWP-2 = PUMP  ..

CWP-3 = PUMP  ..

CWP-4 = PUMP  ..

CWP-5 = PUMP  ..

CONDENSER-LOOP = CIRCULATION-LOOP
   TYPE               = CW  ..

CH-1 = CHILLER
   TYPE               = ELEC-OPEN-CENT
   CHW-LOOP           = CHILLED-LOOP        not shown
   CONDENSER-TYPE     = WATER-COOLED
   CW-LOOP            = CONDENSER-LOOP
   CW-PUMP            = CWP-1  ..
```

```
CH-2 = CHILLER
   LIKE                = CH-1
   CW-PUMP             = CWP-2  ..

CH-3 = CHILLER
   LIKE                = CH-1
   CW-PUMP             = CWP-3  ..

TWR-1 = HEAT-REJECTION
   TYPE                = OPEN-TWR
   CW-LOOP             = CONDENSER-LOOP
   CW-FLOW-CTRL        = CONSTANT-FLOW
   CW-PUMP             = CWP-4  ..

TWR-2 = HEAT-REJECTION
   LIKE                = TWR-1
   CW-PUMP             = CWP-5  ..
```

Example 13. ILLEGAL Condenser Loop Configuration

This example is similar to Example 10, except that each tower has its own dedicated condenser water pump. This configuration is illegal. The reason for this is complex, but has to do with the ability of the program to allow pumps dedicated to chiller condensers to power a CW loop (as shown in example 11). Since it is more common for condenser pumps to stage with chillers than it is for pumps to stage with towers, this configuration was deemed to be illegal. If you attempt this configuration, the program will demand that you attach condenser pumps to the chillers.

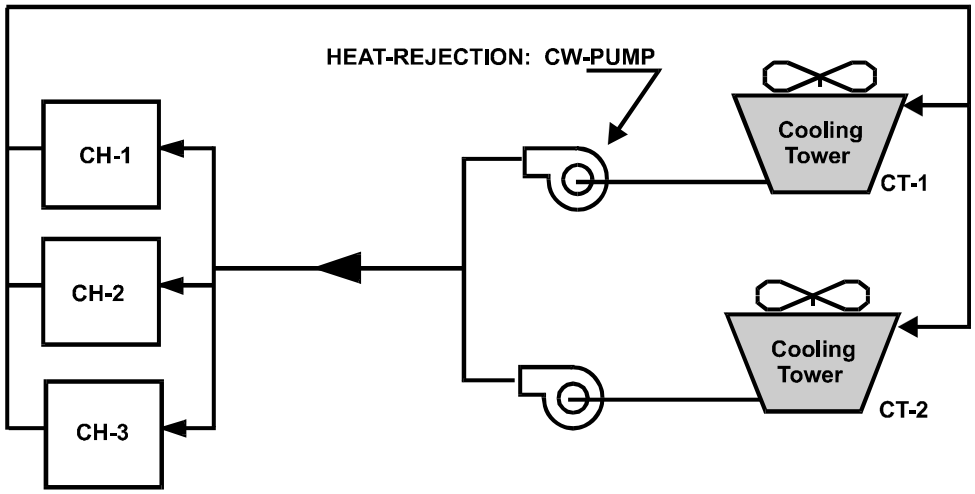


Figure 87 ILLEGAL condenser pump configuration

Example 14. Open Cooling Towers with Heat-Exchangers

This example is similar to Example 10, except that each tower is isolated from the loop via a heat-exchanger. This configuration is more commonly used with a water-loop heat-pump system than with chillers, but is shown here to

illustrate how the loop can be configured. In this case, the loop pumps stage as required to meet the chiller demands. The recirculating pump on each tower stages with its tower.

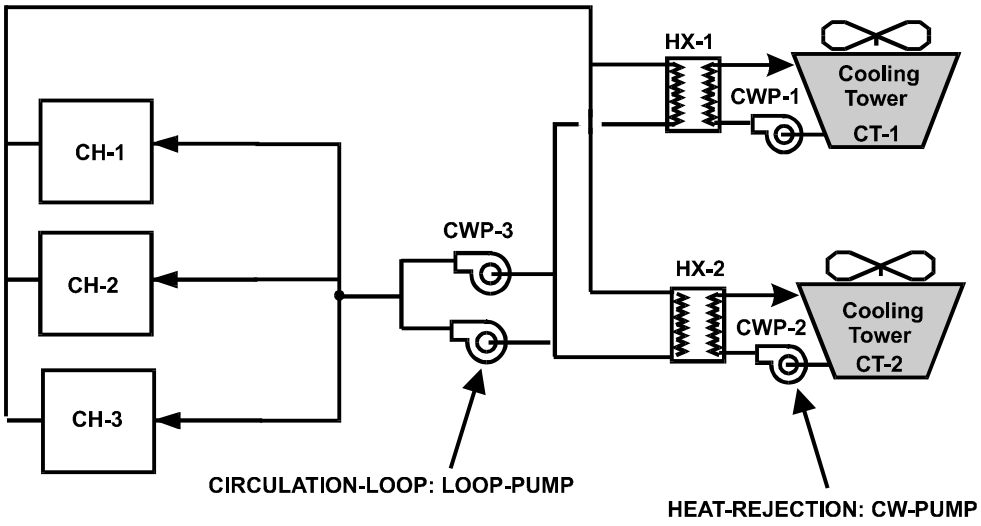


Figure 88 Open cooling towers with heat exchangers

```
CWP-1 = PUMP  ..

CWP-2 = PUMP  ..

CWP-3 = PUMP
   NUMBER             = 2  ..

CONDENSER-LOOP = CIRCULATION-LOOP
   TYPE               = CW
   LOOP-PUMP          = CWP-3  ..

CH-1 = CHILLER
   TYPE               = ELEC-OPEN-CENT
   CHW-LOOP           = CHILLED-LOOP         not shown
   CONDENSER-TYPE     = WATER-COOLED
   CW-LOOP            = CONDENSER-LOOP  ..

CH-2 = CHILLER
   LIKE               = CH-1  ..

CH-3 = CHILLER
   LIKE               = CH-1  ..

HX-1 = HEAT-EXCHANGER
   TYPE               = WATER-TO-WATER  ..

HX-2 = HEAT-EXCHANGER
   LIKE               = HX-1  ..
```

```
TWR-1 = HEAT-REJECTION
   TYPE               = OPEN-TWR&HX
   CW-LOOP            = CONDENSER-LOOP
   CW-HX              = HX-1
   CW-PUMP            = CWP-1  ..

TWR-2 = HEAT-REJECTION
   LIKE               = TWR-2
   CW-HX              = HX-2
   CW-PUMP            = CWP-2  ..
```

Example 15. Open Cooling Towers with Heat-Exchangers and Dedicated Chiller Pumps

This example is similar to Example 14, except that each chiller has its own dedicated condenser pump, and the loop does not have a pump.

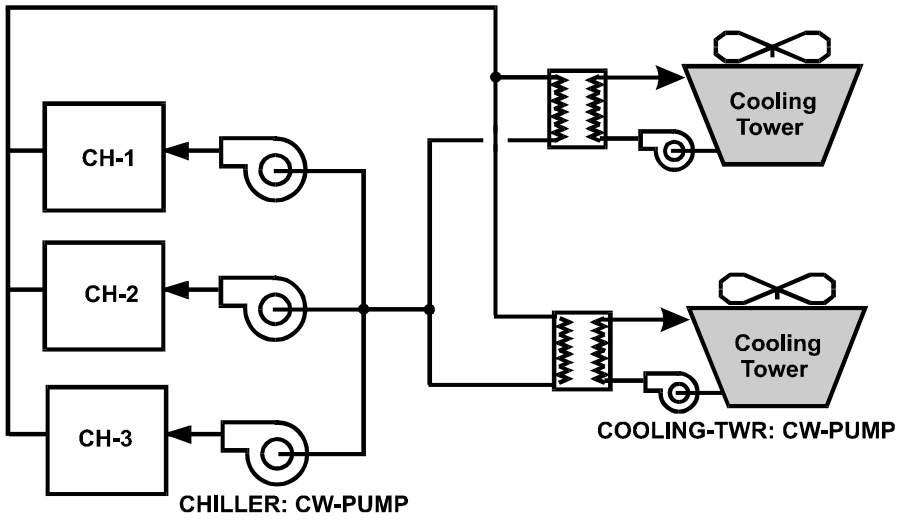


Figure 89 Open cooling towers with heat exchangers and dedicated chiller pumps

```
CWP-1 = PUMP  ..

CWP-2 = PUMP  ..

CWP-3 = PUMP  ..

CWP-4 = PUMP  ..

CWP-5 = PUMP  ..

CONDENSER-LOOP = CIRCULATION-LOOP
   TYPE               = CW  ..
```

```
CH-1 = CHILLER
   TYPE               = ELEC-OPEN-CENT
   CHW-LOOP           = CHILLED-LOOP         not shown
   CONDENSER-TYPE     = WATER-COOLED
   CW-LOOP            = CONDENSER-LOOP
   CW-PUMP            = CWP-1  ..

CH-2 = CHILLER
   LIKE               = CH-1
   CW-PUMP            = CWP-2  ..

CH-3 = CHILLER
   LIKE               = CH-1
   CW-PUMP            = CWP-3  ..

HX-1 = HEAT-EXCHANGER
   TYPE               = WATER-TO-WATER  ..

HX-2 = HEAT-EXCHANGER
   LIKE               = HX-1  ..

TWR-1 = HEAT-REJECTION
   TYPE               = OPEN-TWR&HX
   CW-LOOP            = CONDENSER-LOOP
   CW-HX              = HX-1
   CW-PUMP            = CWP-4  ..

TWR-2 = HEAT-REJECTION
   LIKE               = TWR-1
   CW-HW              = HX-2
   CW-PUMP            = CWP-5  ..
```

Example 16. Multi-Cell Open Tower with Heat-Exchanger

This example illustrates a multi-cell tower coupled to a condenser loop via a heat-exchanger. In this configuration, the tower flow (and tower pump flow) stages with the number of cells operating, where the number of cells operating is determined by the load and CELL-CTRL.

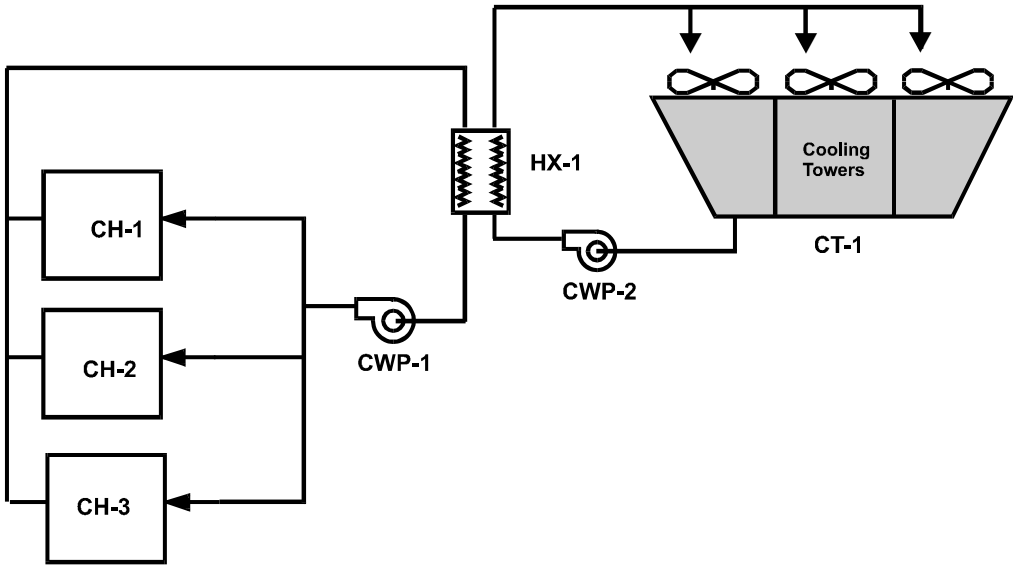


Figure 90 Multi-cell open tower with heat exchanger

```
CWP-1 = PUMP  ..

CWP-2 = PUMP  ..

CONDENSER-LOOP = CIRCULATION-LOOP
   TYPE = CW
   LOOP-PUMP          = CWP-1  ..

CH-1 = CHILLER
   TYPE               = ELEC-OPEN-CENT
   CHW-LOOP           = CHILLED-LOOP        not shown
   CONDENSER-TYPE     = WATER-COOLED
   CW-LOOP            = CONDENSER-LOOP  ..

CH-2 = CHILLER
   LIKE               = CH-1  ..

CH-3 = CHILLER
   LIKE               = CH-1  ..

HX-1 = HEAT-EXCHANGER
   TYPE               = WATER-TO-WATER  ..

TWR-1 = HEAT-REJECTION
   TYPE               = OPEN-TWR&HX
   NUMBER-OF-CELLS    = 3
   CELL-CTRL          = MIN-CELLS
   CW-LOOP            = CONDENSER-LOOP
   CW-HX              = HX-1
   CW-PUMP            = CWP-2  ..
```

Example 17. Separate Condenser Loops

This example illustrates two chillers attached to two separate condenser water loops. As previously stated, the chillers may be attached to either a common or separate chilled-water loops.

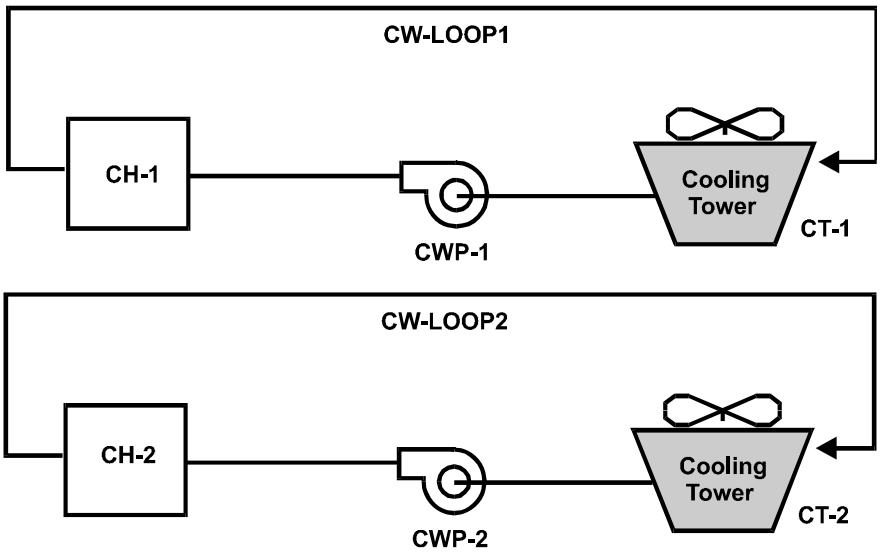


Figure 91 Separate condenser loops

```
CWP-1 = PUMP  ..

CWP-2 = PUMP  ..

CW-LOOP1 = CIRCULATION-LOOP
   TYPE               = CW
   LOOP-PUMP          = CWP-1  ..

CW-LOOP2 = CIRCULATION-LOOP
   LIKE               = CW-LOOP1
   LOOP-PUMP          = CWP-2  ..

CH-1 = CHILLER
   TYPE               = ELEC-OPEN-CENT
   CHW-LOOP           = CHILLED-LOOP        not shown
   CONDENSER-TYPE     = WATER-COOLED
   CW-LOOP            = CW-LOOP1  ..

CH-2 = CHILLER
   LIKE               = CH-1
   CW-LOOP            = CW-LOOP2  ..

TWR-1 = HEAT-REJECTION
   TYPE               = OPEN-TWR
   CW-LOOP            = CONDENSER-LOOP1  ..
```

```
TWR-2 = HEAT-REJECTION
   LIKE               = TWR-1
   CW-LOOP            = CONDENSER-LOOP2  ..
```

LOAD-MANAGEMENT AND EQUIP-CTRL

LOAD-MANAGEMENT is a global command in the sense that it is not restricted to a specific circulation-loop or electric meter (although you can restrict it in that manner). Instead, it is designed to allow you to coordinate multiple EQUIP-CTRL sequences across multiple loads.

You can define more than one LOAD-MANAGEMENT sequence. The program will process each sequence and perform the specified actions according to the specified criteria. If two or more LOAD-MANAGEMENT sequences try to assign conflicting instructions to a given load (loop or meter), then the load will use the instruction assigned the highest priority. In this manner, you can build up relatively complex control structures using relatively simple EQUIP-CTRL and LOAD-MANAGEMENT sequences.

WARNING: Do not use the LOAD-MANAGEMENT and EQUIP-CTRL sequences to force all equipment off (no heating or cooling) for a CIRCULATION-LOOP unless you have also forced off the loop. The program will abort if a loop has a load, but no equipment is available to at least partially satisfy the load. In other words, 100% of a loop load cannot be an overload. (This warning might not apply once the PEAK-SHAVING strategy is implemented.) This warning does not apply to ELEC-GENERATORS attached to an ELEC-METER; any unmet load will simply be passed on to the utility.

Flow-Based Override of Load-Allocation Routines

The EQUIP-CTRL sequences and the default equipment allocation routine select equipment on the basis of the circulation-loop load. However, CIRCULATION-LOOPs are load, temperature, and flow-based. Because of this, it is possible that a mix of equipment can be selected that satisfies the load, but is incompatible with the flow and/or temperature requirements.

For example, assume that four 150-ton chillers supply a chilled-water loop. All coils on the loop have 3-way valves, so the loop has a constant flow of 1,440 gpm. The chillers are arranged in parallel, so that the design flow through each chiller is 360 gpm. The design supply temperature is 44°F, with a 10°F temperature rise. The current cooling load is 150 tons, and the loop supply set-point is 44°F. In theory, one chiller could handle this load; however, the flow and temperature requirements must also be met in one of the following fashions:

- *One chiller can run, with the entire loop flow going through the chiller's evaporator.*
  This solution is not usually possible. For all coils to receive their design flow, the flow through the single chiller would have to be four times higher than design, and the pressure drop through the chiller would be approximately 16 times higher than design. The loop pumps could not possibly meet this head requirement, and if they could, the resulting water velocity could severely erode the tubes in the chiller's evaporator. If this approach is used, the result in most systems will be a drop in loop flow. The reduced flow may starve some of the coils, and/or may cause an increase in the supply temperature of any attached secondary loops. Proper modeling of this effect requires an iterative approach, and is planned for a future program release.

- *One chiller can run, with three-quarters of the loop flow bypassing the chiller.*
  This solution will satisfy the flow requirements, but may not satisfy the load and/or temperature requirements. At a 25% load, the loop return temperature will be 46.5°F. To meet the required 44°F supply set-point, the chiller will have to produce water at 36.5°F. If this is not possible, then the loop temperature will float upward. The increased temperature may starve some of the coils, which may in turn result in higher airflows in VAV systems. Proper modeling of this effect requires an iterative approach, and is planned for a future program release.

- *More than one chiller can run, so that the load, flow, and temperature requirements are all met.*
  This is the approach currently used. The program first allocates the load to the equipment using an EQUIP-CTRL sequence or by default. It then compares the flow capacity of the selected equipment to the required loop flow. If the loop flow is not met, then the program will re-select a mix of equipment that best meets the flow requirement. This approach may override any EQUIP-CTRL sequences assigned to the loop. To prevent this from happening, you can set the equipment's MAX-FLOW to a large value. This will allow a small unit to handle a large flow, but the increased chiller head loss may trigger a warning that the pump cannot meet the head and flow requirements. You can ignore this warning unless you are conducting pump studies.

Examples of EQUIP-CTRL and LOAD-MANAGEMENT Sequences

The following are examples of some EQUIP-CTRL and LOAD-MANAGEMENT sequences. Additional examples are included in the "Electric Generators and Cogeneration" section of this manual.

Example 1: Chiller Staging

Assume a central plant has three chillers: 200 tons, 400 tons, and 600 tons. By default, the program will use the 200 ton chiller whenever the load is in the range of 100-200 tons, the 400-ton chiller for loads in the range of 201-400 tons, etc.

In this example, this is not the optimum control sequence, however, because the 200 and 600-ton machines are newer and much more efficient than the 400-ton machine. The following sequence will optimally load the newer machines and reserve the 400-ton machine for periods of maximum load:

```
CHLR-CTRL = EQUIP-CTRL
   TYPE              = COOLING
   CIRCULATION-LOOP  = COOLING-LOOP

   LOADS-THRU-1      = 2.4                  Millions of Btuh
   CHILLERS-1        = (CHLR-200)

   LOADS-THRU-2      = 7.2
   CHILLERS-2        = (CHLR-600)

   LOADS-THRU-3      = 9.6
   CHILLERS-3        = (CHLR-600,CHLR-200)

   LOADS-THRU-4      = 14.4
   CHILLERS-4        = (CHLR-600,CHLR-200,CHLR-400)
   ..
```

The spacing between load ranges is for clarity only; as with all BDL input, spacing has no effect on the meaning of the input.

Some types of equipment, such as chillers and cooling towers, have a capacity that varies hourly in accordance with ambient temperature and/or other conditions. Therefore, it often makes sense not to have to specify a fixed LOADS-THRU range, but to allow the range to default to the hourly capacity of the equipment listed within the range. If you do not specify the LOADS-THRU keyword, the value will default to the hourly capacity. The following is equivalent to the example above, but takes into account the varying hourly capacities:

```
CHLR-CTRL = EQUIP-CTRL
   TYPE              = COOLING
   CIRCULATION-LOOP  = COOLING-LOOP

   CHILLERS-1        = (CHLR-200)
   CHILLERS-2        = (CHLR-600)
   CHILLERS-3        = (CHLR-600,CHLR-200)
   CHILLERS-4        = (CHLR-600,CHLR-200,CHLR-400)
   ..
```

Also, while the above input works, it uses more keywords than is needed; also for clarity. The following input is equivalent:

```
CHLR-CTRL = EQUIP-CTRL
   TYPE               = COOLING
   CIRCULATION-LOOP   = COOLING-LOOP

   CHILLERS-1         = (CHLR-200)
   CHILLERS-2         = (CHLR-600,CHLR-200,CHLR-400)
   ..
```

This input, while shorter than the previous, is also somewhat overspecified because the default sequencing for the EQUIP-2 list is the order entered. In other words, the CHILLERS-SEQ-2 did not need to be listed in this example.

If sequencing is not specified, equipment is started in the order listed. The following examples are equivalent, and illustrate how sequencing works:

```
   LOADS-THRU-2       = 999.
   CHILLERS-2         = (CHLR-600,CHR-200,CHLR-400)

   LOADS-THRU-2       = 999.
   CHILLERS-2         = (CHLR-600,CHR-200,CHLR-400)
   CHILLERS-SEQ-2     = (       1,      2,       3)

   LOADS-THRU-2       = 999.
   CHILLERS-2         = (CHLR-200,CHLR-400,CHLR-600)
   CHILLERS-SEQ-2     = (       2,       3,       1)
```

In all these examples, CHLR-600 starts first, followed by CHLR-200, followed by CHLR-400. To force CHLR-600 and CHLR-200 to always start and run together, give them the same sequence number:

```
   LOADS-THRU-2       = 999.
   CHILLERS-2         = (CHLR-200,CHLR-400,CHLR-600)
   CHILLERS-SEQ-2     = (       1,       2,       1)
```

Example 2: Boiler and Chiller/Heater

A boiler and chiller/heater are attached to the same heating loop. The boiler should always run first, followed by the chiller heater:

```
BLR-CTRL = EQUIP-CTRL
   TYPE               = HEATING
   CIRCULATION-LOOP   = HEATING-LOOP
   LOADS-THRU-1       = 999.                optional keyword
   BOILERS-1          = ( BOILER-1)
   BOILERS-SEQ-1      = (        1)
   CHLR/HTRS-1        = (CHILLER-1)
   CHLR/HTRS-SEQ-1    = (        2)
   ..
```

Example 3: Coordination of Thermal Storage

Thermal storage is to augment a chiller whenever the cooling loads exceeds the chiller capacity:

```
COOLING-LOOP = CIRCULATION-LOOP
   TYPE               = CHW
   ……
   ..

CHILLER-1 = CHILLER
   TYPE               = ELEC-OPEN-CENT
   CHW-LOOP           = COOLING-LOOP
   ……
   ..

COLD-TANK = THERMAL-STORAGE
   TYPE               = COLD-WATER-TANK
   DCHRG-LOOP         = COOLING-LOOP
   CHRG-LOOP          = COOLING-LOOP
   CHRG-SCH           = TES-CHRG-SCH
   …
   ..

COOL-CTRL = EQUIP-CTRL
   TYPE               = COOLING
   CIRCULATION-LOOP   = COOLING-LOOP

   CHILLERS-1         = (CHILLER-1)
   CHILLERS-SEQ-1     = (        1)
   STORE-SEQ-1        = (        2)
   ..
```

To base load the tank, reverse the sequencing:

```
COOL-CTRL = EQUIP-CTRL
   TYPE               = COOLING
   CIRCULATION-LOOP   = COOLING-LOOP

   CHILLERS-1         = (CHILLER-1)
   CHILLERS-SEQ-1     = (        2)
   STORE-SEQ-1        = (        1)
   ..
```

To use the tank only between the hours of 1 p.m. and 5 p.m., a LOAD-MANAGEMENT sequence must be used:

```
USE-TES-DAY = DAY-SCHEDULE
   TYPE               = FLAG
                        ( 1,13) (1.)
                        (13,17) (2.)
                        (18,24) (1.)  ..

USE-TES-WEEK = WEEK-SCHEDULE
   TYPE               = FLAG
                        (ALL) USE-TES-DAY  ..
```

```
USE-TES-SCH = SCHEDULE
   TYPE              = FLAG
   THRU DEC 31         USE-TES-WEEK  ..

USE-CHILLER = EQUIP-CTRL
   TYPE              = COOLING
   CIRCULATION-LOOP  = COOLING-LOOP

   CHILLERS-1        = (CHILLER-1)
   ..

USE-TES = EQUIP-CTRL
   TYPE              = COOLING
   CIRCULATION-LOOP  = COOLING-LOOP

   CHILLERS-1        = (CHILLER-1)
   CHILLERS-SEQ-1    = (        2)         chiller second
   STORE-SEQ-1       = (        1)         tank first
   ..

COOL-CTRL = LOAD-MANAGEMENT
   TYPE              = SCHEDULED
   EQUIP-CTRL-SCH    = USE-TES-SCH

   CTRL-FLAG-1       = 1.0
   EQUIP-CTRL-1      = (USE-CHILLER)

   CTRL-FLAG-2       = 2.0
   EQUIP-CTRL-2      = (USE-TES)
   CTRL-PRIORITY-2   = 10.
   ..
```

The CTRL-PRIORITY-2 specification is only necessary if there are other LOAD-MANAGEMENT sequences specified that may conflict with this one. In this case, this LOAD-MANAGEMENT will override any LOAD-MANAGEMENT sequence having a lower priority, and in turn will be overriden by any LOAD-MANAGEMENT sequence having a higher priority, where the lowest possible priority is 1.0, and the highest is 100.

ELEC-METER, FUEL-METER, STEAM-METER AND CHW-METER

Building Sub-metering and the ELEC-METER Command

DOE-2.1E allowed you to specify up to five electric meters to which any electrical "end use" class could be connected; these connections where specified by using the M1, M2, M3, M4 and M5 code-words. This capability is now extended by adding the ELEC-METER command that allows you to define up to 25 meters to which electrical equipment can be attached. In addition, the concept of "meter levels" is added; this allows you to define sub-metering of usage and demand.

This concept is implemented within the ELEC-METER command through the use of the TYPE keyword, which takes values SUB-METER, BUILDING and UTILITY. These types define three meter levels with SUB-METER being the lowest, BUILDING being the middle and UTILITY being the highest. The intent is that all meters that are to be connected to a UTILITY-RATE should be of TYPE = UTILITY; this is not an enforced restriction, but if another type of meter is connected to a UTILITY-RATE there will most likely be double counting of energy consumption by the ECONOMICS program. Note, however, that you might want to take advantage of listing BUILDING meters or SUB-METERs in a UTILITY-RATE if sub-meters exist for various tenants in a building, and you wish to have the program calculate the charges for each tenant.

A fourth TYPE of ELEC-METER also exists: ELECTRIC-SALE. This type of meter accepts on-site electric generator output that will be sold to a utility. It is not possible to attach any electric demands (such as lighting) to this meter. Only generators can attach to this type of meter, so that their power can be sold. See "Electric Generators and Cogeneration" in this manual for more information.

Each meter TYPE, including ELECTRIC-SALE, can have associated transformers that can include losses and part-load performance characteristics.

All types of meters, except TYPE= UTILITY, must be attached to an electric meter of higher level. Thus, Each meter of TYPE = SUB-METER must be attached to one and only one meter of TYPE = BUILDING or UTILITY. And each meter of TYPE = BUILDING meter must be attached to one and only one meter of TYPE = UTILITY. In this manner a three-level tree is constructed that contains all meters. The consumptions/demands for meters of TYPE = BUILDING or SUB-METER are summed into their parent electric meter; thus reports for meters of TYPE = UTILITY or BUILDING contain the component consumptions/demands of all lower-level (child) meters. This structure allows the description of a site (with a site substation transformer) with multiple buildings, each with its own BUILDING meter plus transformer and, possibly floor-by-floor SUB-METERs and transformer.

Example input:

```
BANKinB1 = ELEC-METER
   TYPE               = SUB-METER
   TRANSFORMER-SIZE   = 15 ..

FLOOR2inB1 = ELEC-METER
   TYPE               = SUB-METER
   TRANSFORMER-SIZE   = 10 ..

FLOOR3inB1 = ELEC-METER
   TYPE               = SUB-METER
   TRANSFORMER-SIZE   = 10 ..
```

```
STOREinB2 = ELEC-METER
   TYPE               = SUB-METER
   TRANSFORMER-SIZE   = 20 ..

FLOOR2inB2 = ELEC-METER
   TYPE               = SUB-METER
   TRANSFORMER-SIZE   = 10 ..

OFF-BLDG-1 = ELEC-METER
   TYPE               = BUILDING
   TRANSFORMER-SIZE   = 30
   SUB-METERS         = (BANKinB1, FLOOR2inB1, FLOOR3inB1) ..

OFF-BLDG-2 = ELEC-METER
   TYPE               = BUILDING
   TRANSFORMER-SIZE   = 25
   SUB-METERS         = (STOREinB2, FLOOR2inB2) ..

SUBSTATION = ELEC-METER
   TYPE               = UTILITY  $ no transformer - utility owned
   BLDG/SUB-METERS    = (OFF-BLDG-1,OFF-BLDG-2) ..
```

CURVE-FIT FOR EQUIPMENT PERFORMANCE

Equipment operating characteristics are specified in two parts. First, the design ARI [ARI = Air-Conditioning and Refrigeration Institute standards for the design, fabrication, and installation of heating, ventilating, and air conditioning equipment] capacity and consumption are given. Second, the off-design to design functional relationship is specified. For example, the hourly available cooling capacity from a unit is:

CAP_T = available total cooling capacity
= CAP_{ARI} * CAP(WBT,DBT)

where CAP(WBT,DBT) is the modifier function to account for off-design WBT and DBT. In keyword form, the equation is expressed as

CAP_T = COOLING-CAPACITY * COOL-CAP-FT(WBT,DBT),

where WBT and DBT are the wet- and drybulb temperatures of air entering the coil for water coils and entering air wetbulb and outdoor drybulb for direct expansion (DX) units.

For cooling capacity, it is necessary also to know the sensible part of the total capacity. This is, in a similar manner,

$SCAP_T$ = $SCAP_{ARI}$ * SCAP(WBT,DBT)

or in keyword form,

$SCAP_T$ = COOL-SH-CAP * COOL-SH-FT(WBT,DBT).

For all types, $SCAP_T$ is not allowed to exceed CAP_T , and for DX units, $SCAP_T$ is also corrected for entering drybulb temperatures other than at the ARI rating point (the ARI rating point is: wetbulb entering = 67F, drybulb entering = 80F, and drybulb outdoor = 95F).

To calculate the energy input to produce the available output, a similar rated value and modifier function is used.

EIR_T = ratio of electric input to cooling output

EIR_T = EIR_{ARI} * EIR(WBT,DBT) * EIR(PLR),

or in keyword form,

EIR_T = COOLING-EIR * COOL-EIR-FT(WBT,DBT) * COOL-EIR-FPLR(PLR).

The PLR is the ratio of actual load output to full load output at the operating point. Thus, the electrical input to the unit is

QE = CAP_T * EIR_T

or

QE = CAP_{ARI} * CAP(WBT,DBT) * EIR_{ARI} * EIR(WBT,DBT) * EIR(PLR)

Similar relationships are used for heat pumps where the ARI point is changed (outdoor drybulb = 47F and entering drybulb = 70F). For hot water and electric coils, the capacity is assumed to be independent of operating conditions. For gas and oil furnaces, the input is simply a function of the part-load output and the capacity is assumed to be constant.

Chilled Water Coils

These equations apply to all systems utilizing chilled-water coils (SZRH, DDS, MZS, TPIU, FPIU, TPFC, FPFC,CBVAV, VAVS, RHFS, HVSYS)

CAP_T = COOLING-CAPACITY * COOL-CAP-FT(EWB,EDB)

$SCAP_T$ = COOL-SH-CAP * COOL-SH-FT(EWB,EDB) or CAP_T , whichever is smaller,

CBF = COIL-BF * COIL-BF-FT(EWB,EDB) * COIL-BF-FFLOW(FLOWPLR).

DX systems

These equations apply to all systems utilizing DX coils (HP, RESYS, PSZ, PMZS, PVAVS, PTAC)

CAP_T = COOLING-CAPACITY * COOL-CAP-FT(EWB,ODB),

$SCAP_T$ = COOL-SH-CAP * COOL-SH-FT(EWB,ODB) + [1.08 * (SUPPLY-FLOW) * (1 - COIL-BF)* (EDB - 80)] or CAP_T , whichever is smaller,

CBF = COIL-BF * COIL-BF-FT(EWB,ODB) * COIL-BF-FFLOW(FLOWPLR),

EIR_T = COOLING-EIR * COOL-EIR-FT(EWB,ODB) * COOL-EIR-FPLR(PLR),

QE = CAP_T * EIR_T .

Electric Heating

(HEAT-SOURCE = ELECTRIC)

CAP_T = HEATING-CAPACITY

QE = Heating load

Electric heat pump

(HEAT-SOURCE = HEAT-PUMP)

CAP_T = HEATING-CAPACITY * HEAT-CAP-FT(ODB,EDB)

EIR_T = HEATING-EIR * HEAT-EIR-FT(ODB,EDB) * HEAT-EIR-FPLR(PLR)

QE = CAP_T = EIR_T

Gas or oil furnace

(HEAT-SOURCE = GAS-FURNACE or OIL-FURNACE)

CAP_T = HEATING-CAPACITY

HIR_T = FURNACE-HIR * FURNACE-HIR-FPLR(PLR)

QE = CAP_T * HIR_T

ELEC-GENERATOR AND COGENERATION

The electric generator algorithms have been revised from DOE-2.1E to take advantage of the new component-based structure as well as the electric metering and circulation loop features. As before, the program simulates engine-driven generators, gas turbine generators, and steam turbine generators. The generators may run full out, track the electric mode of a meter, track the thermal load of a circulation loop, track the greater or lesser of electric and thermal loads, or be forced off. Electricity generated may be consumed on site, sold in its entirety to a utility, or consumed on site with only the surplus sold to a utility. Cogeneration input examples are given at the end of this section.

Selling generated electricity to a utility

To sell generated electricity to a utility you specify TYPE = ELECTRIC-SALE in the UTILITY-RATE command. This works the same as TYPE = ELECTRIC except that the program automatically makes calculated costs negative so that the numbers represent an income stream rather than an expense. While the list of ELEC-METERS for this TYPE = ELECTRIC-SALE can include electric meters of any type (UTILITY, BUILDING, SUB-METER, ELECTRIC-SALE), the intent is that only ELECTRIC-SALE meters be listed.

Buy/sell arrangements using only one meter are also possible. These are common for small photovoltaic systems, where the periods of surplus production can be credited against the electricity bought during other times. For an example of a buy/sell arrangement using a PV array, refer to the *DOE-2.2 Dictionary,* UTILITY-RATE examples

Steam Turbines

Improvements have been made to the steam turbine model from DOE-2.1E, most notably in its ability to thermal track a heating load, but the model is still relatively limited in its capabilities. The biggest limitation is that the program cannot simulate steam loops. (Likewise, previous program versions could not simulate steam distribution systems.) Steam loops can have large thermal losses and also have auxiliary equipment (de-aerators, etc.) that consume considerable amounts of steam on an annual basis.

The operation of a steam loop feeding a turbine can be roughly approximated by using a hot-water loop having the right characteristics and neglecting thermal losses. Thermal losses cannot be simulated since the very high heat capacity of steam will cause the program to erroneously calculate a supply vs. return temperature differential on the order of 1000F (555K). It is not possible to approximate a steam loop on the downstream (heat recovery) side of the turbine. This is because the program does not have a steam heating coil model and consequently cannot equate a heating load to a steam flow.

Another limitation is in the steam turbine default performance characteristics. Given a supply pressure of 125 psig (8619 mbarg) and an exhaust pressure of 15 psig (1034 mbarg) the maximum possible thermodynamic efficiency of a steam turbine (isentropic expansion) is on the order of 10%. However, the default mechanical efficiency is also 10%, so the net overall efficiency is approximately 1%. The default performance characteristics of the model were derived over 20 years ago and were based on a turbine designed to drive a boiler feedwater pump, with the turbine exhaust directed to either a low-pressure steam demand or the de-aeration system. Its extremely low efficiency makes it completely unsuitable for power generation. *Accordingly, you should not use this model without also specifying the performance characteristics for the actual turbine.*

There is an example input, below, for a plant with a steam turbine, where the turbine is supplied by a "steam loop" and the exhaust heats a hot-water loop.

Some plants use a steam turbine that runs in conjunction with a gas turbine. The waste heat of the gas turbine is used to generate the steam that powers the steam turbine. The best way to simulate this configuration is to combine the performance characteristics of the two turbines into a single component and simulate it as a gas turbine.

Cogeneration Rules

The following rules apply to electric generators that also recover heat.

1. For heat to be recovered from a generator, the generator must be attached to a circulation loop. The default is that no heat will be recovered.

2. Generators serving the same meter do not have to be attached to the same circulation loop. Generators serving the same circulation loop do not have to be attached to the same meter.

3. The jacket and exhaust of an engine-driven generator may be attached to the same or different loops and may thermal track each one separately.

4. There are two heat recovery modes, active and passive. It is important to understand the difference between these two modes of operation:

   - Active heat recovery occurs when a generator tracks a thermal load. When thermal tracking, the generator operates to deliver the required amount of heat and the electrical output is a by-product. In this mode, a generator appears to the circulation loop the same as a boiler and its operation may be coordinated with boilers, etc., using an EQUIP-CTRL sequence. If you do not specify an EQUIP-CTRL sequence the program will preferentially load the generator prior to enabling any other heating equipment, without regard to the size of the heating load. Since a generator may be relatively inefficient at low loads, this may not be the most economical control sequence. You can avoid this by defining an EQUIP-CTRL sequence that uses a boiler for low loads and only enables cogeneration equipment when the load is sufficiently large that the equipment can run efficiently.

   - Passive heat recovery occurs when a generator is tracking an electric load or is running at maximum output. In these modes the generator operates to satisfy its electrical requirement and the recoverable heat is a by-product. The program calculates the heat available for heat recovery and applies it to the circulation loop attached to the generator before attempting to operate any other heating equipment attached to the same loop. In other words, passive recoverable heat is always used first. If this generator is listed in a loop's EQUIP-CTRL sequence but is operating in a passive heat recovery mode, then its recoverable heat will be applied to the heating demand prior to the EQUIP-CTRL sequence and the generator will be ignored within the sequence.

   - Active or passive heat recovery may exist when the generator tracking mode is either TRACK-GREATER or TRACK-LESSER. When the mode is TRACK-GREATER active heat recovery will occur when the thermal load causes the generator to operate at an output higher than the electric load, and passive heat recovery will occur when the electric load is greater than the thermal load. The opposite is true for TRACK-LESSER. Since both of these modes may result in active heat recovery, you should specify EQUIP-CTRL sequences for the generator's circulation loops to coordinate the generators with the other heating equipment.

5. Any given generator may operate in a passive heat recovery mode one hour, and switch to an active mode the next. In addition, some generators on a loop may be operating in a passive mode, while other generators on the same loop may be operating in an active mode. In this case, the heat recoverable from the passive machines will be used first, with the remainder of the load assigned to boilers or active generators via either an EQUIP-CTRL sequence or by default.

6. A generator will never operate below its ELEC-GENERATOR:MIN-RATIO value. When electric tracking the load must be at least as large as the minimum part load ratio; otherwise the unit will be off for the hour. This is important to note for peak shaving strategies. The examples, below, illustrate this

point. When thermal tracking, the generator will operate at the minimum part load ratio or greater. If operation at the minimum part load ratio produces more heat than is required, the generator will continue to operate and the excess heat will be wasted. To prevent this waste, you should construct EQUIP-CTRL sequences that use a boiler for low thermal loads, and switch to generators once the thermal load is high enough for the generators to operate efficiently.

7. Cogeneration heat recovery is energy based rather than being temperature and flow dependent. This means that heat recovery can occur regardless of the loop temperature and flow conditions. For example, jacket heat from an engine generator can provide heat to a loop operating at 300F (149C), which may be unreasonable. Also, the amount of heat recoverable is independent of loop temperature. This assumption may not be reasonable for generators using a heat exchanger to recover exhaust heat, when coupled to a loop with large temperature swings. Also, energy-based heat recovery means that, unlike other types of equipment, no pump can be directly attached to a generator component, nor are pressure losses modeled. It is up to the user to design reasonable models.

8. A steam turbine cannot recover heat to the same loop from which the heat is drawn. If attached in this manner high-entropy, low-pressure steam from the exhaust will recycle back into the turbine inlet, providing a portion of the power and violating the second law of thermodynamics. The program does not check for this condition.

Cogeneration Examples

In the following examples only enough of the input is shown to illustrate the points being made. These examples assume that you already have a working knowledge of the EQUIP-CTRL and LOAD-MANAGEMENT commands. You should refer to those commands for more information on the techniques used in these examples.

Example 1. 24-Hour Buy/Sell with Heat Recovery

A hospital runs its generator at maximum output 24 hours a day, sells all the power generated, buys all the power consumed by lights, etc., and recovers heat to its heating loop.

```
EM1 = ELEC-METER
   TYPE               = UTILITY  ..

Sale = ELEC-METER
   TYPE               = ELECTRIC-SALE
   COGEN-TRACK-MODE   = MAX-OUTPUT  ..

FM1 = FUEL-METER
   TYPE               = NATURAL-GAS  ..

MASTER-METERS
   MSTR-ELEC-METER    = EM1
   MSTR-FUEL-METER    = FM1  ..

Generator-1 = ELEC-GENERATOR
   TYPE               = ENGINE-GENERATOR
   CAPACITY           = 100                  kW
   ELEC-METER         = Sale
   EXH-LOOP           = Heating-Loop
   JAC-LOOP           = Heating-Loop  ..
```

EM1 was not explicitly used anywhere in this example; it is simply the meter that lights, etc. will be attached to by default, as defined in the MASTER-METERS command. The FUEL-METER keyword was not specified in the ELEC-GENERATOR command since the generator will default to the master fuel meter. Note that the exhaust heat and the jacket heat both go to the same loop; they could be attached to separate loops if desired. No SURPLUS-METER is specifed for the generator since there is no surplus; all power will be sold via the "Sale" meter.

Example 2. 12-Hour Surplus Sale with Heat Recovery

A hospital runs its generator at maximum output from 6 am to 6 pm, consumes as much power on site as possible, sells only the surplus power, and recovers heat to the heating loop.

```
Gen-DS = DAY-SCHEDULE-PD
   TYPE               = FLAG
   VALUES             = (0,0,0,0,0,5,5,5,    $ 0=don't run
                         5,5,5,5,5,5,5,5,    $ 5=maximum output
                         5,0,0,0,0,0,0,0)  ..

Gen-WS =  WEEK-SCHEDULE-PD
   TYPE               = FLAG
   DAY-SCHEDULES      = Gen-DS  ..

Gen-Sch = SCHEDULE-PD
   TYPE               = FLAG
   MONTH              = 12
   DAY                = 31
   WEEK-SCHEDULES     = Gen-WS  ..

EM1 = ELEC-METER
   TYPE               = UTILITY
   COGEN-TRACK-SCH    = Gen-Sch  ..

Sale = ELEC-METER
   TYPE = ELECTRIC-SALE  ..

FM1 = FUEL-METER
   TYPE               = NATURAL-GAS  ..

MASTER-METERS
   MSTR-ELEC-METER    = EM1
   MSTR-FUEL-METER    = FM1  ..

Gen-1 = ELEC-GENERATOR
   TYPE               = ENGINE-GENERATOR
   CAPACITY           = 100  $ kW
   ELEC-METER         = EM1
   SURPLUS-METER      = Sale
   EXH-LOOP           = Heating-Loop
   JAC-LOOP           = Heating-Loop  ..
```

Since the hospital is now consuming generated power on site and selling only the surplus, the generator's ELEC-METER is now specified to be EM1 so that the generator knows which on-site meter it should supply. Since EM1 is now controlling the generator, COGEN-TRACK-SCH is specified under this meter. The Sale meter now has no influence over this generator, other than to accept its surplus power.

Note that, if the Sale meter had other generators attached to it via their ELEC-METER keywords, then it should have a COGEN-TRACK-MODE or COGEN-TRACK-SCH, and the Sale meter would control those generators.

It is also possible for an ELECTRIC-SALE meter to have some generators attached to it through their ELEC-METER keywords and other generators attached to it through their SURPLUS-METER keywords. In this case, only the generators attached via their ELEC-METER keywords would be controlled by that meter; the SURPLUS-METER generators would only contribute their surplus power and would be controlled by other ELEC-METERS.

Example 3. Electric Tracking with Heat Recovery

A hospital has two generators and runs one or both generators to satisfy the on-site demands. No surplus power is generated or sold because the local utility rates are unfavorable. The plant has one 100 kW generator and one 200 kW generator. Because the generators are relatively inefficient at loads below 30% capacity, an EQUIP-CTRL sequence will be used to prevent the generators from running when the load is below 30 kW.

```
EM1 = ELEC-METER
   TYPE               = UTILITY
   COGEN-TRACK-MODE   = TRACK-ELEC
   EQUIP-CTRL         = Gen-Ctrl  ..

FM1 = FUEL-METER
   TYPE               = NATURAL-GAS  ..

MASTER-METERS
   MSTR-ELEC-METER    = EM1
   MSTR-FUEL-METER    = FM1  ..

Gen-100 = ELEC-GENERATOR
   TYPE               = GAS-TURBINE-GENERATOR
   CAPACITY           = 100                  kW
   ELEC-METER         = EM1
   EXH-LOOP           = Heating-Loop
   ..

Gen-200 = ELEC-GENERATOR
   TYPE               = GAS-TURBINE-GENERATOR
   CAPACITY           = 200                  kW
   ELEC-METER         = EM1
   EXH-LOOP           = Heating-Loop  ..

Gen-Ctrl = EQUIP-CTRL
   TYPE               = ELECTRICAL
   ELEC-METER         = EM1
   LOADS-THRU-1       = 30                   $ First 30 kW
                                             $ no equipment
   GENERATORS-2       = (Gen-100)            $ ~30-100 kW

   GENERATORS-3       = (Gen-200,            $ ~100+ kW
                         Gen-100)            $ ~200+ kW
   ..
```

In this example LOADS-THRU-1 in the EQUIP-CTRL command is simply a placeholder for the first 30 kW. Since no equipment is listed for GENERATORS-1, no equipment will operate in this range. Above 30 kW the small generator operates (as listed in GENERATORS-2). If the load is larger than what the small generator can handle, the small generator is shut down and the big generator started (as listed in GENERATORS-3). If the load is larger than what the big generator can handle, then the smaller generator will start and operate simultaneously with the big generator (also as listed in GENERATORS-3).

Note that neither LOADS-THRU-2 or LOADS-THRU-3 is explicitly specified. Instead, these load ranges default, respectively, to the hourly capacity of the equipment listed in the GENERATORS-2 and GENERATORS-3 keywords. This is the preferred method for specifying the load ranges since the capacity of gas-turbine generators varies with outdoor temperature. Allowing the load ranges to default ensures that there will be no gaps in generator operation.

Note also that the sequence numbers for GENERATORS-3 are not specified. By default, the first generator listed is given a sequence number of 1 and the second has a sequence number of 2.

If the EQUIP-CTRL sequence was not defined, then EM1 would, by default, attempt to give all loads below approximately 100 kW to Gen-100, all loads between 100 and 200 kW to "Gen-200", and run both generators above approximately 200 kW. Since this is the same sequence the EQUIP-CTRL command is doing, the EQUIP-CTRL command is not really necessary except to lock out the generators below 30 kW. Another way to lock out the generators below 30 kW would be to assign Gen-100 a MIN-RATIO of 0.30. By default, all loads below 100 kW would be assigned to Gen-100 but the generator would lock itself out below 30 kW (100 kW * 0.30 minimum part load ratio).

Example 4. Thermal Tracking with Surplus Sale

A facility has two boilers, each 1.5 MBtuh (0.44 MW). It also has two electric generators, one 100 kW and another 200 kW. Heat is recovered from the generators, and the generators are to thermal track the heating load. Because the generators are relatively inefficient at low loads, the generators will not be started until the heating load is at least 0.5 MBtuh (0.15 MW). On-site power demands will have first priority; surplus generator output will be sold back to the utility. (In this example the option of putting U-name in quotes is exercised; this allows U-names with spaces, like Heating Loop, to be indicated.)

```
"EM1" = ELEC-METER
   TYPE               = UTILITY
   COGEN-TRACK-MODE   = TRACK-THERMAL  ..

"Sale" = ELEC-METER
   TYPE               = ELECTRIC-SALE  ..

"FM1" = FUEL-METER
   TYPE               = NATURAL-GAS  ..

MASTER-METERS
   MSTR-ELEC-METER    = "EM1"
   MSTR-FUEL-METER    = "FM1"  ..

"Heating Loop" = CIRCULATION-LOOP
   TYPE               = HW
   HEAT-EQUIP-CTRL    = "Heating Ctrl"  ..

"Boiler 1" = BOILER
   TYPE               = HW-BOILER
   CAPACITY`          = 1.5
   HW-LOOP            = "Heating Loop"  ..

"Boiler 2" = BOILER
   LIKE               = "Boiler 1"
   ..

"Generator 100" = ELEC-GENERATOR
   TYPE               = GAS-TURBINE-GENERATOR
   CAPACITY           = 100  $ kW
   ELEC-METER         = "EM1"
   SURPLUS-METER      = "Sale"
   EXH-LOOP           = "Heating Loop"  ..


"Generator 200" = ELEC-GENERATOR
   TYPE               = GAS-TURBINE-GENERATOR
   CAPACITY           = 200  $ kW
   ELEC-METER         = "EM1"
   SURPLUS-METER      = "Sale"
   EXH-LOOP           = "Heating Loop"  ..
```

```
"Heating Ctrl" = EQUIP-CTRL
   TYPE               = HEATING
   CIRCULATION-LOOP   = "Heating Loop"
   LOADS-THRU-1       = 0.5
   BOILERS-1          = ("Boiler 1")          $<0.5 MBtu
   GENERATORS-2       = ("Generator 100")     $~0.5-0.99 MBtu
   GENERATORS-3       = ("Generator 200",     $~0.99+ MBtu
                         "Generator 100")     $~1.98+ MBtu
   BOILERS-3          = ("Boiler 1",          $~2.97+ MBtu
                         "Boiler 2")          $~4.47+ MBtu
   GENERATORS-SEQ-3   = (1, 2)
   BOILERS-SEQ-3      = (3, 4)
   ..
```

In the EQUIP-CTRL instruction, the first load range is the only range in which the load is explicitly defined; this is so a boiler will operate below 0.5 MBtu (0.15 MW) rather than a generator. The second load range defaults to the full load thermal capacity of "Generator 100", which is approximately 0.99 MBtu (0.29 MW). In the third load range, "Generator 200" runs first, followed by "Generator 100", and finally by the boilers.

Example 5. Track Greater of Electric or Thermal Demands with Surplus Sale

A facility has two 1.5 MBtuh (0.44 MW) boilers. It also has two generators, one 100 kW and one 200 kW. The generators are to track the greater of the electrical or thermal loads. Because the generators are relatively inefficient at low loads, the generators will not be started until the heating load is at least 0.5 MBtu (0.15 MW) and the electric load is at least 30kW. On-site power demands will have first priority; surplus generator output will be sold.

```
"EM1" = ELEC-METER
   TYPE               = UTILITY
   COGEN-TRACK-MODE   = TRACK-GREATER
   EQUIP-CTRL         = "Gen Ctrl"  ..

"Sale" = ELEC-METER
   TYPE               = ELECTRIC-SALE  ..

"FM1" = FUEL-METER
   TYPE               = NATURAL-GAS  ..

MASTER-METERS
   MSTR-ELEC-METER    = "EM1"
   MSTR-FUEL-METER    = "FM1"  ..

"Heating Loop" = CIRCULATION-LOOP
   TYPE               = HW
   HEAT-EQUIP-CTRL    = "Heating Ctrl"  ..

"Boiler 1" = BOILER
   TYPE               = HW-BOILER
   CAPACITY           = 1.5
   HW-LOOP            = "Heating Loop"  ..

"Boiler 2" = BOILER
   LIKE               = "Boiler 1"  ..

"Generator 100" = ELEC-GENERATOR
   TYPE               = GAS-TURBINE-GENERATOR
   CAPACITY           = 100                  $ kW
   ELEC-METER         = "EM1"
   SURPLUS-METER      = "Sale"
   EXH-LOOP           = "Heating Loop"  ..

"Generator 200" = ELEC-GENERATOR
   TYPE               = GAS-TURBINE-GENERATOR
   CAPACITY           = 200  $ kW
   ELEC-METER         = "EM1"
   SURPLUS-METER      = "Sale"
   EXH-LOOP           = "Heating Loop"  ..
```

```
"Gen Ctrl" = EQUIP-CTRL
   TYPE              = ELECTRICAL
   ELEC-METER        = "EM1"
   LOADS-THRU-1      = 30                  $ First 30
                                           $ no equipment
   GENERATORS-2      = ("Generator 100")   $ ~ 30-100 kW
   GENERATORS-3      = ("Generator 200",   $ ~100+ kW
                        "Generator 100")   $ ~200+ kW
   ..

"Heating Ctrl" = EQUIP-CTRL
   TYPE              = HEATING
   CIRCULATION-LOOP  = "Heating Loop"

   LOADS-THRU-1      = 0.5
   BOILERS-1         = ("Boiler 1")        $<0.5 MBtu

   GENERATORS-2      = ("Generator 100")   $~0.5-0.99 MBtu
   GENERATORS-3      = ("Generator 200",   $~0.99+ MBtu
                        "Generator 100")   $~1.98+ MBtu
   BOILERS-3         = ("Boiler 1",        $~2.97+ MBtu
                        "Boiler 2")        $~4.47+ MBtu
   GENERATORS-SEQ-3  = (1, 2)
   BOILERS-SEQ-3     = (3, 4)  ..
```

This example is a combination of the previous two examples. It illustrates how generators can be incorporated into EQUIP-CTRL sequences on both the thermal and electrical sides of a central plant.

Example 6. Peak Shaving

A facility has a 100-kW emergency generator that runs whenever the peak load exceeds 200 kW in order to reduce peak load charges. The generator will modulate as required to maintain the utility load at 200 kW but will not attempt to reduce it below this level. Because relatively short run periods are anticipated no heat will be recovered. The minimum load at which the generator can run is 25%.

```
"EM1" = ELEC-METER
   TYPE               = UTILITY
   COGEN-TRACK-MODE   = TRACK-ELEC
   EQUIP-CTRL         = "Gen Ctrl"  ..

"FM1" = FUEL-METER
   TYPE               = NATURAL-GAS  ..

MASTER-METER
   MSTR-ELEC-METER    = "EM1"
   MSTR-FUEL-METER    = "FM1"  ..

"Generator 100" = ELEC-GENERATOR
   TYPE               = GAS-TURBINE-GENERATOR
   CAPACITY           = 100                  $ kW
   ELEC-METER         = "EM1"
   MIN-RATIO          = 0.25
   ..

"Gen Ctrl" = EQUIP-CTRL
   TYPE = ELECTRICAL
   ELEC-METER         = "EM1"

   LOADS-THRU-1       = 200                  $ First 200 kW
                                             $ no equipment
   LOADS-THRU-2       = 999
   METER-MAX-LOAD-2   = 175
   METER-SEQ-2        = 1
   GENERATORS-2       = ("Generator 100")    $ 200+ kW
   GENERATORS-SEQ-2   = (2)
   ..
```

In the EQUIP-CTRL command the LOADS-THRU-1 is simply a placeholder for the first 200 kW. Since no equipment is listed for GENERATORS-1, no equipment will operate in this range and all of the load will be assigned to the meter.

The second load range is a little more complicated. The intent is that the meter pick up the first 200 kW, but note that the METER-MAX-LOAD-2 is listed as 175. This is because the effect of the generator's MIN-RATIO must be recognized. If the METER-MAX-LOAD-2 were 200 kW and the actual load were 220 kW, then the program would attempt to allocate 200 kW to the meter and 20 kW to the generator. However, since the minimum load the generator can handle is 25 kW, the generator won't run and all of the load would go to the meter. The load would have to be 225 kW for the generator to run.

To compensate, the METER-MAX-LOAD-2 is listed as 175 kW. The generator will then be able to start once the actual load is 200 kW and will be able to peak shave all loads above 200 kW. Note that, when the load is 200 kW, the generator picks up 25 kW of the load and passes only 175 kW on to the utility. While this is not exactly what is intended, it is the reality of having a generator with a 25% MIN-RATIO.

Example 7. Time and Temperature-Based Peak Shaving

A data processing facility has an emergency generator. The local electric utility has made the facility an attractive offer if they will "go off-line" and start and run their generator within an hour's notice during the utility's maximum peak load periods. These periods occur intermittently during the months of May through September and roughly correspond to times when the outdoor air temperature exceeds 93F (33.9C). No power is sold back to the utility.

```
"EM1" = ELEC-METER
   TYPE               = UTILITY
   COGEN-TRACK-MODE   = TRACK-ELECTRIC  ..

"FM1" = FUEL-METER
   TYPE               = NATURAL-GAS  ..

MASTER-METERS
   MSTR-ELEC-METER    = "EM1"
   MSTR-FUEL-METER    = "FM1"  ..

"Generator 100" = ELEC-GENERATOR
   TYPE               = ENGINE-GENERATOR
   CAPACITY           = 100  $ kW
   ELEC-METER         = "EM1"  ..

"Gen Ctrl Off" = EQUIP-CTRL
   TYPE               = ELECTRICAL
   ELEC-METER         = "EM1"
   LOADS-THRU-1       = 999.                $ all loads,
                                            $ no equipment
   ..

"Gen Ctrl On" = EQUIP-CTRL
   TYPE               = ELECTRICAL
   ELEC-METER         = "EM1"
   LOADS-THRU-1       = 999.                $ all loads
   GENERATORS-1       = ("Generator 100")
   ..

"No Peak-Shave DS" = DAY-SCHEDULE-PD
   TYPE               = FLAG
   VALUES             = (0)  ..

"Allow Peak-Shave DS" = DAY-SCHEDULE-PD
   TYPE               = FLAG
   VALUES             = (1)  ..

"No Peak-Shave WS" = WEEK-SCHEDULE-PD
   TYPE               = FLAG
   DAY-SCHEDULES      = ("No Peak-Shave DS")  ..

"Allow Peak-Shave WS" = WEEK-SCHEDULE-PD
   TYPE               = FLAG
   DAY-SCHEDULES      = ("Allow Peak-Shave WS")  ..
```

```
"Peak Shave Sch" = SCHEDULE-PD
   TYPE              = FLAG
   MONTH             = (4, 9, 12)
   DAY               = (30,30,31)
   WEEK-SCHEDULES    = ("No Peak-Shave WS",
                        "Allow Peak-Shave WS",
                        "No Peak-Shave WS")  ..

"Normal Mode" = LOAD-MANAGEMENT
   TYPE              = OA-TEMP
   QUAL-SCH          = "Peak Shave Sch"
   QUAL-SCH-FLAG     = 0
   TEMPS-THRU-1      = 200.
   EQUIP-CTRLS-1     = ("Gen Ctrl Off")  ..

"Go Off-Line" = LOAD-MANAGEMENT
   TYPE              = OA-TEMP
   QUAL-SCH          = "LM Sch"
   QUAL-SCH-FLAG     = 1
   TEMPS-THRU-1      = 93.
   EQUIP-CTRLS-1     = ("Gen Ctrl Off")
   CTRL-PRIORITY-1   = 1
   TEMPS-THRU-2      = 200.
   EQUIP-CTRLS-2     = ("Gen Ctrl On")
   CTRL-PRIORITY-2   = 1
   ..
```

Note that no EQUIP-CTRL sequence is explicitly listed in the "EM1" meter instruction. Therefore, "EM1" will attempt to allocate the electrical load to the generator by default, unless overridden by a LOAD-MANAGEMENT instruction. The two LOAD-MANAGEMENT instructions are designed to do just that. Based on the flag value in "Peak Shave Sch", one or the other of these instructions is active all hours.

The "Normal Mode" instruction activates the "Gen Ctrl Off" equipment control sequence during the winter months. "Gen Ctrl Off" is a dummy sequence that simply forces the program to use an equipment control command rather than allocating the electrical load by default. Since this sequence has no equipment listed, all of the electrical load will go to the utility whenever this sequence is activated.

While the "Normal Mode" instruction is of type OA-TEMP, it really could be of any type since its only purpose is to activate the "Gen Ctrl Off" sequence during the winter.

The "Go Off-Line" instruction is active only during the summer months. When the temperature is 93F (33.9C) or less, it activates the "Gen Ctrl Off" sequence, thereby preventing the generator from running. Above 93F (33.9C) it activates the "Gen Ctrl On" sequence and the generator starts and tracks the electrical load.

Example 8. Thermal Tracking Steam Turbine

A food-processing facility has design process load of 1.5 MBtu (0.44 MW). A steam turbine generator is used to produce electricity by reducing the high-pressure steam produced by the boiler to lower-pressure steam used in the manufacturing processes. Because the turbine efficiency is relatively low, the electricity generated is considered a windfall; the turbine tracks the process thermal load.

```
"EM1" = ELEC-METER
   TYPE               = UTILITY  ..

"Sale" = ELEC-METER
   TYPE               = ELECTRIC-SALE
   COGEN-TRACK-MODE   = TRACK-THERMAL  ..

"FM1" = FUEL-METER
   TYPE               = NATURAL-GAS  ..

MASTER-METERS
   MSTR-ELEC-METER    = "EM1"
   MSTR-FUEL-METER    = "FM1"  ..

"Feedwater Pump" = PUMP  ..

"Process Pump" = PUMP  ..

"Steam Loop" = CIRCULATION-LOOP
   TYPE               = HW
   LOOP-PUMP          = "Feedwater Pump"  ..

"Process Loop" = CIRCULATION-LOOP
   TYPE               = HW
   LOOP-PUMP          = "Process Pump"
   PROCESS-LOAD       = 1.5                  $MBtu
   PROCESS-SCH        = "Process Sch"  ..

"Steam Boiler" = BOILER
   TYPE               = STM-BOILER-W/DRAFT
   HW-LOOP            = "Steam Loop"  ..

"Dummy Boiler" = BOILER
   TYPE               = HW-BOILER
   HW-LOOP            = "Process Loop"  ..

"Steam Generator"= ELEC-GENERATOR
   TYPE               = STEAM-TURBINE-GENERATOR
   CAPACITY           = 50.                  $kW
   ELEC-METER         = "EM1"
   SURPLUS-METER      = "Sale"
   STEAM-LOOP         = "Steam Loop"
   EXH-LOOP           = "Process Loop"
   STEAM-ENT-PRES     = 250                  $psig
   STEAM-EXH-PRES     = 15                   $psig
   MECH-EFF           = 0.60  ..
```

Given the entering and exhaust pressures and the mechanical efficiency, the program will calculate a theoretical steam rate of 21.4 lbs/kWh, and an actual steam rate of 35.6 lbs/kWh (16.2 kg/kWh). The enthalpies corresponding to these pressures are 1209 Btu/lb (781 Wh/kg) entering and 221 Btu/lb (143 Wh/kg) returning to the boiler (saturated condensate). The recoverable heat will be 31 kBtu/kWh (9.1 kWh/kWh), so a 50 kW generator will be sufficient to provide all of the heat required for the process loads.

Since "EM1" is thermal tracking, the generator will track the load on the "Process Loop". In turn, the turbine will place a demand on the "Steam Loop" which will be satisfied by the "Steam Boiler".

The "Feedwater Pump" will be sized to 3.55 gpm (35.6 lbs/kWh * 50 kWh / (8.34 lbs/gal * 60 min/hr)) (13.4 l/min) and a head of 575 ft (250 psig * 2.3 ft/psig) (175 m). The steam loop flow will also be 3.55 gpm (13.4 l/min), with a design capacity of 1.77 MBtu (0.52 MWh). This results in a temperature differential between the supply and return of approximately 997F (554K). The steam loop temperatures are, therefore, meaningless and should be disregarded. Obviously thermal losses cannot be simulated for the "Steam Loop".

The "Dummy Boiler" is never used in this example. However, it is required by the program because the program insists that each loop have at least one primary equipment unit attached, and heat recovery does not fall in this category.

ELEC-METER SUB-HOUR DEMAND INTERVALS

Most electric utilities base their demand charges on either 15-minute or 30-minute time intervals. DOE-2.1E calculated peak demands based on a one-hour interval, which is consistent with the one-hour time step used in the simulation. As a result, the program would consistently underestimate the peak demands associated with start-up loads of relatively short duration. For example, a zonal DX system might run continuously for the first half-hour after startup, and then cycle intermittently the rest of the hour. A one-hour demand interval would average the start-up load over the entire hour, thereby reducing the effective peak, whereas an electric meter with a 15-minute demand interval would record the higher peak that actually occurred during start-up.

This program partly alleviates this problem. For zonal equipment, such as electric reheat coils, electric baseboards, DX cooling and heat pumps, the program now calculates the electric load distribution throughout the hour for each component. This hourly profile is aggregated with the profiles of other components on the same meter to derive a net profile on a meter-by-meter basis. This capability is fully implemented for the following system types: SZRH, VAVS, RHFS, HP, CBVAV, PSZ, PVAVS, PTAC, and PVVT. It is not implemented for any other system types.

Note that for the water-loop heat-pump system (HP) the program calculates the peak distribution for the heat-pump compressor energy, but does not calculate this distribution for any electric boiler defined to serve this system. Future program versions may extend this capability to lights, miscellaneous equipment, and central plant equipment such as boilers and chillers.

The program calculates demands for both fixed-interval and sliding windows. The demand interval may range from 5 to 60 minutes, in 5-minute increments. You define the demand parameters in the UTILITY-RATE command; these parameters apply only to the utility rate, block charge and ratchet calculations and reports. The peak demands reported in all other reports, such as LS-D, SS-D, PS-B, PS-E, and PS-F, are the maximum hourly demand, as before.

THERMAL-STORAGE

You define a thermal energy storage (TES) device using the THERMAL-STORAGE command and attach it to circulation loops using the CHRG-LOOP and DCHRG-LOOP keywords. Compared to other primary equipment components, the rules for loop attachments are a somewhat different for TES devices since they can charge from and discharge to different loops. The rules are:

1. A loop can be supplied by at most one storage device.

2. A loop can charge one or more storage devices.

The loop that charges a TES device can be the same loop that the device discharges to, or it can be a different loop.

Energy-based Algorithms

When using TES devices it is important that you understand their modeling limitations and how those limitations affect your analysis. The program uses *energy-based* models of hot-water and chilled-water storage tanks that are essentially identical to the original algorithms in DOE-2.1E. The models are called energy-based because they assume that a unit of energy put into storage can be taken out again in an undegraded form. Actual storage tanks are usually stratified, with the warmest fluid at the top and the coldest at the bottom. The degree of stratification depends on the circulation loop flow rate and supply and return temperature. A tank on a constant-flow loop may destratify fairly rapidly, even when the load on the tank is small. Once destratified, the tank may not be able to meet the supply temperature requirement of the loop even though, in theory, it has half or more of its storage energy left.

The current algorithms assume *perfect stratification* under all flow conditions. In addition, the models do not take the temperature at which they are charged or discharged into account. For example, the program allows you to charge a hot-water storage tank from a loop operating at 110F (43.3C) and discharge the tank to a loop operating at 180F (82.2C). This violates the Second Law of Thermodynamics. Future versions of the program may alleviate these shortcomings.

Pumping Energy

Furthermore, energy-based models do not allow the flow and pumping head into and out of the tank to be calculated. For this reason, the tank is the only primary equipment component that does not allow you to attach a pump directly to the tank. However, pumping energy can be a critical factor in the cost effectiveness of a thermal energy storage system. This is especially true when a tank at atmospheric pressure is coupled to a loop having high static heads.

For example, assume a chilled water storage tank is buried under a parking lot at a university campus. The tank is at atmospheric pressure. The chilled water loop must supply air handlers located in penthouse mechanical rooms as high as the eighth story in one or more buildings. The required residual static pressure at these air handlers is 15 psig (1034 mbarg) . The net static head is then

(8 stories)*(14 ft/story) + (15 psig)*(2.3 ft/psig) = 146.5 ft (44.7 m)

This static head is in addition to the dynamic (friction) head and is constant all hours of operation. If a pump is to directly inject fluid from the storage tank into the loop it must overcome this head and may consume considerable energy doing so. If the pump is variable speed it may have to run at 85% speed or greater at all flows just to overcome the injection head (the head capacity of a pump varies as the speed squared). Thus we see that pumping energy may be an important factor in the evaluation of a TES system. Until these algorithms are upgraded, you can approximate the pump energy using the THERMAL-STORAGE:AUX-KW and AUX-MODE keywords.

DW-HEATER SUPPLYING SPACE HEATING

Normally, hot water coils used for space heating are attached to a circulation loop of TYPE = HW or PIPE-2 (see CIRCULATION-LOOP command in the *DOE-2.2 Dictionary*). However, some systems use the heat from a domestic hot water heater to provide space heating. These systems are typically in homes or apartments in climates requiring relatively little heating.

To simulate this type of system, specify HEAT-SOURCE, ZONE-HEAT-SOURCE, PREHEAT-SOURCE and/or BASEBOARD-SOURCE in the SYSTEM command to be of type DHW-LOOP. Then attach the heating coil to a domestic hot water loop using the SYSTEM:DHW-LOOP keyword or, for a zonal coil, the ZONE:DHW-LOOP keyword.

The program will take into account the required coil capacity when sizing the loop and auto-sizing the domestic hot water heaters attached to the loop. You should review the resulting sizing parameters to make sure they are acceptable. Also, you should make sure that the entering water temperature of the heating coil is at least 10F (5.6K) warmer than the leaving air temperature. Otherwise the program will not be able to size the coil correctly.

Domestic hot water loops without heating coils or recirculation do not need to be pumped; the program will assume that the water-main pressure is sufficient. Adding heating coils to the loop requires the loop to be pumped; otherwise water cannot circulate between the water heater and the coil. For domestic hot water loops the pump is located on the return side of the loop; it does not pump water that is used for dishwashing or other domestic hot water uses, only water that returns to the water heaters. The pump is attached within the circulation loop component, or, alternatively, to each of the water heaters attached to the loop.

POSITIVE VS. NEGATIVE HEATING AND COOLING VALUES

The DOE-2.1E SYSTEMS program required that heating quantities be entered as negative values and cooling quantities as positive values. On the other hand, the PLANT program required that heating values (such as boiler size) be entered as a positive value. The PLANT convention makes more sense since equipment manufacturers do not list the capacity of boilers, furnaces, etc. as negative numbers.

Now, all of the keywords associated with plant heating equipment can be entered using either a positive or negative value. In either case the program will convert the entry to a negative value for consistency with the SYSTEMS algorithms. Existing SYSTEMS keywords, such as HEATING-CAPACITY still require a negative value.

Note that, internally in the program, heating quantities are always negative numbers and cooling quantities are positive numbers. This is because heat fluxes into and out of a space are defined from the viewpoint of the space. Therefore, a heat flux into a space is defined as a positive value (cooling load) and a heat flux out of a space is a negative value (heating load). For this reason, hourly reports will still show heating loads as negative numbers and cooling loads as positive.

Section

6

Economic Components

LIFE-CYCLE COSTING

The Economics sub-program calculates the life-cycle cost of a building, which includes energy cost and non-energy cost (capital, operation and maintenance cost). Energy costs are calculated from the building's energy consumption using user-specified utility rate schedules. Non-energy costs are calculated from user-specified costs for the purchase, operation and maintenance of building components.

The capital, operations, maintenance and energy costs of a building are calculated over the user-specified lifetime of the building. In addition, to allow you to compare design alternatives for new and retrofit buildings, a few numbers, called *investment statistics*, are calculated that measure the cost-effectiveness of a design relative to a baseline case.

The program divides non-energy costs into *plant* costs (the cost of plant components, such as boilers and chillers) and non-plant, or *building*, costs (the cost of envelope components, such as walls and windows, and the cost of secondary system components, such as fans and ducts). Plant costs are entered using MATERIALS-COST commands and the COST-DATA keyword in plant commands CHILLER, BOILER, COOLING-TWR, ELEC-GENERATOR, etc. Building costs are entered using COMPONENT-COST commands.

The Economics program adds plant, building and energy costs to arrive at an overall life-cycle cost. It also computes the following investment statistics:

- Investment
- Energy cost savings
- Non-energy cost savings
- Energy use savings
- Ratio of total (energy plus non-energy) cost savings to investment
- Ratio of energy use savings to investment
- Payback period

These quantities are calculated by comparing costs and energy use for a design with those you input for a baseline case using the BASELINE command.

Life-Cycle Costing Methodology

We describe here the terminology and methods used to calculate life-cycle costs and investment statistics.

The *project lifetime* is the period, in years, over which the life-cycle cost analysis is done. This can be from 1 to 25 years. The *life-cycle cost* of an item is the total cost of an item over the project lifetime. For a cost that recurs every year, such as energy cost, the life-cycle cost (LCC) is

$$LCC = \sum_{n=1}^{N} C_n$$

where N is the project lifetime in years and Cn is the cost in the nth year.

Rather than use the actual cost in year n, the program calculates the *present value* of the cost, which is given by

$$C_n = C\left(\frac{1+i}{1+d}\right)^n$$

where C is the cost in current (i.e., today's) dollars, i is the cost inflation rate relative to general inflation, and d is the discount rate. For example, if:

11. Project lifetime = 25 years
12. Discount rate = 10%
13. Energy inflation rate = 5% (above general inflation)
14. C = $1000 (annual energy cost in today's dollars),

then the present value of the energy cost in year 10, say, would be

$$C_{10} = \$1000\left(\frac{1+0.05}{1+0.10}\right)^{10} = \$1000(0.955)^{10} = \$1000(0.628) = \$628$$

The present value of the energy cost over the project's life cycle would then be

$$LCC = \sum_{n=1}^{25} C\left(\frac{1+i}{1+d}\right)^n = \sum_{n=1}^{25} 1000\left(\frac{1.05}{1.10}\right)^n = 1000(14.436) = 14{,}436$$

Note that without discounting (i.e., without present-valuing), the life-cycle cost would be 25x1000 = $25,000.

Non-energy costs are broken down into three categories:

- First cost
- Operations cost
- Replacement cost

The *first cost* is the purchase price of an item, including installation. I includes expenses, such as maintenance and overhauls, required to keep the item in working condition. The *replacement cost* is the cost (including installation) of replacing an item at the end of its useful life. The residual value (the fraction of the remaining useful life times the capital cost) of an item at the end of the life-cycle analysis period is not considered.

A primary application of the economics sub-program is cost-benefit analysis of energy conservation measures (ECMs). In this type of analysis, a *baseline building* is first analyzed to determine its costs and energy use. The building is then modified to conserve energy and the simulation is repeated. If the baseline energy use and cost data from the first run are entered in the second run (using the BASELINE command), then the program will assess the cost-effectiveness of the ECM.

The quantities that enter the cost effectiveness analysis are the following:

- Investment

- Incremental investment
- Cost savings
- Energy savings
- Savings to investment ratio (SIR)
- Energy savings to investment ratio
- Discounted payback period

The ECM *investment* is the sum of the life-cycle first cost and replacement cost for all plant and non-plant cost items:

ECM investment = (first cost) + (replacement cost) for all components

Subtracting from this the baseline first cost and baseline replacement cost gives the *incremental investment*:

Incremental investment = (ECM investment) - (baseline first + replacement cost)

The *cost savings* is the difference between the life-cycle energy and operations cost of the baseline and that of the ECM:

$Cost\ savings = (energy + operations\ cost)_{baseline} - (energy + operations\ cost)_{ECM}$

The energy use is calculated both at the site (i.e., at the building boundary) and at the source. The energy savings, the difference between baseline energy use and ECM energy use, is calculated for both site energy and source energy.

The *savings to investment ratio* (SIR) is just the cost savings divided by the incremental investment:

SIR = (cost savings) / (incremental investment)

An SIR greater than 1.0 means that the ECM is cost effective. The higher the SIR, the more cost effective is the ECM.

The energy savings to investment ratio gives the lifetime energy savings per dollar invested:

Energy savings to investment ratio = (energy savings) / (incremental investment)

Finally, the *discounted payback period* is the number of years it takes the accumulated cost savings to equal the incremental investment. For example, if the present value of the cost savings in years 1 to 5 is $1000, $900, $800, $700 and $600, respectively, and the incremental investment is $2700, then the payback period is 3 years.

A cost-effective ECM has a payback period that is less than the project lifetime. The shorter the payback period, the more cost effective is the ECM.

Economic Evaluation Methods

The program can be used in three different ways to perform an economic analysis:

- Method I — Ranking by life-cycle cost: The total life-cycle cost of different design alternatives is compared. Investment statistics such as payback period and savings-to-investment ratio do not apply. This method may be used to rank new building or retrofit alternatives.

- Method II — Ranking retrofit designs using investment statistics: Retrofit alternatives are ranked on the basis of investment statistics. The building before retrofit is defined as the baseline. The investment is the cost of energy-saving modifications to the baseline building. The savings are given by the difference in life-cycle energy and operation costs of the modified building relative to the baseline. A run is made on the baseline and on each of the alternatives.

- Method III — Ranking new-building designs using investment statistics: This is similar to Method II, except that new-building alternatives (rather than retrofit alternatives) are compared using investment statistics. In this method, the alternative with the smallest investment is used as the baseline.

In the following, these methods are described in more detail.

Method I — Ranking by life-cycle cost

A separate simulation run is made for each building design as follows.

- Enter cost data for each building design

- Enter plant costs using MATERIALS-COST commands and COST-DATA keywords for the various pieces of plant equipment. For existing buildings, specify HOURS-USED in the MATERIALS-COST command, otherwise the equipment will be considered to be new, and the first cost of the equipment will be included in the life-cycle cost calculation.

- Enter cost and escalation rate using UTILITY-RATE commands for the different energy types used in the building.

- Enter project lifetime, discount rate, etc. using the BASELINE command.

- Enter all other costs using COMPONENT-COST commands. These non-plant costs include those for components of the building's envelope and its lighting systems, secondary HVAC systems, etc.

- If you want the overall life-cycle cost of the building, you must enter all relevant costs. However, if you just want the difference in life-cycle cost between alternatives, then enter only the cost items that differ between the alternatives.

- Run the program for each design

- Compare total life-cycle costs as given in Report ES-B.

Note: for this method, baseline data need not be specified. Also, the cost savings, investment, savings to investment ratio, etc., given in Report ES-C, do not apply.

Cost Minimization using Parametric Runs
This method may be used to evaluate any parameter of the design by graphing the total life-cycle cost for several different values of the parameter. Sufficient runs should be made to produce a smooth curve in the neighborhood of the minimum total life-cycle cost. The optimum value for the parameter occurs at this minimum.

Method II — Ranking retrofit designs using investment statistics

This method applies to an existing building that is being modified to make it energy efficient. The primary concern is whether the investment to modify the building is cost effective. In other words, are the energy and operations cost savings over the project lifetime greater than the investment, or is the payback period short enough to be financially attractive. To use this method, the program is first run on a baseline building before modifications are made. The program is then run a second time on the modified building. Selected numbers from the first run are input in the BASELINE command in the second run. Report ES-C of the second run then gives the investment statistics.

- For the baseline run, enter costs as in Method I. Specify HOURS-USED for each plant component since this is an existing building. Get Reports ES-A, ES-B and ES-C.

- Repeat step 1 for the modified building, with one exception: some numbers from ES-A, ES-B and ES-C of the baseline run (see description of BASELINE command) must be entered in the BASELINE command for the modified building.

- The investment statistics that determine whether the modification is cost effective are given in Report ES-C.

Special Case
If the modification involves adding or replacing plant equipment, that equipment must be entered using COMPONENT-COST in the input of the modified building. In this case, to avoid double counting, it is important to set to zero the corresponding costs in the MATERIALS-COST command and COST-DATA keywords for the modified building.

Method III — Ranking new-building designs using investment statistics

- Run the program on the alternative with the smallest investment. This is the baseline run.

- Run the program on each of the remaining alternatives, using the results of step 1 to obtain the appropriate input for the BASELINE instruction (see description of BASELINE command).

- The investment statistics that determine whether the alternative is cost effective, relative to the design with the smallest investment, are given in Report ES-C.

Section

7

Weather

INTRODUCTION

The loads and HVAC simulations in require hourly weather data, which are contained in weather files. These weather files are created from source hourly data by the (separate) weather processor program doewth. This topic document describes the weather variables used by the program, the types and formats of source data, the various ways to use doewth, and how to process measured weather data or data in nonstandard formats.

Weather Variables

The weather variables used by the program and present on the weather file are as follows.

Hourly Variables

The weather file contains hourly data for 1 year (8760 hours). Leap years are ignored - all program weather files are 365 days long.

Dry bulb Temperature (°F)
Wet bulb Temperature (°F)
Atmospheric Pressure (inches of Hg times 100)
Wind Speed (knots)
Wind Direction (compass points 0-15, with 0 being north, 1 NNE, etc.)
Cloud Amount (0 - 10, with 0 clear and 10 totally cloudy)
Cloud Type (0, 1, or 2)
    0 is cirrus or cirrostratus, the least opaque;
    1 is stratus or stratus fractus, the most opaque; and
    2 is all other cloud types, of medium opacity
Humidity Ratio (pounds of water per pound of dry air)
Density of Air (lb/ft3)
Specific Enthalpy (Btu/lb)
Rain Flag (0 means it is not raining; 1 means it is)
Snow Flag (0 means it is not snowing; 1 means it is)

Hourly Solar Variables

The program can accept two types of weather files: those containing hourly solar values and those without. In the case of the files without solar data, the program calculates solar values using the ASHRAE clear sky model and the clearness numbers, cloud amounts, and cloud types from the program weather file. The solar weather files contain the following hourly values.

Total Horizontal Solar Radiation (btu/hr-ft2)
Direct Normal Solar Radiation (btu/hr-ft2)

Monthly Variables

- Clearness Number (dimensionless - this is the ASHRAE clearness number; see for example page 27.12, figure 7, 1993 Fundamentals Handbook)

- Ground Temperature (°Rankine)

Header Data

In the header record on the weather file are contained the file identification, latitude, longitude, and time zone.

Source Data and Formats

Source weather data for building energy simulation programs can be broken into two major classes: historical data and typical weather years. Historical data is just "real" data: usually measured (but sometimes modeled) data from a particular location for a given period of record. Typical years are ersatz years assembled to match the long term data from a particular location using a particular statistical measure.

The primary source for historical weather data is the National Climatic Data Center (NCDC)[11] . NCDC can provide hourly historical data for thousands of locations around the world. This data may not always be complete; data items or periods of record may be missing. A highly reliable source of historical data for US locations is the Solar and Meteorological Surface Observational Network (SAMSON) data set assembled by the National Renewable Energy Laboratory (NREL)[12] . This contains a 30 year (1961 to 1990) period of record for 239 locations.

There are several sets of typical year data that are widely used in the program's simulations. The TMY2 data set was derived from the SAMSON database by NREL. There are typical years for 239 locations in the US and its territories. NREL also developed the WYEC2 typical year data set for ASHRAE. The selection criteria were different for TMY2 and WYEC2, with TMY2 weighting the solar data more heavily. The California Energy Commission has defined 16 climate zones with corresponding typical year data sets (CTZ2) for use in Title 24 code compliance. NRC Canada has funded WATSUN Energy Laboratory at the University of Waterloo to create typical weather years for Canada. There are files currently available for 51 locations.

Formats: TMY2 and WYEC2

Source data comes in various formats: typically the files are ASCII, but the data items, units, item location, and record length vary from format to format. NCDC can provide historical data in a variety of formats: TD-3280, TD-9950, TD-1440(CD144). Of these, the weather processor can only deal with CD144, usually called TD-1440 by NCDC. Typical years now come in two formats: TMY2 and WYEC2. Note that TMY2 and WYEC2 are names of both typical year data sets and data formats.

The weather processor can process files in either TMY2 or WYEC2 format. The NREL TMY2 typical weather year data sets are in TMY2 format. The ASHRAE WYEC2, the Canadian CWEC, and the CEC CTZ2 typical weather year data sets are all in WYEC2 format. One other format is worth mentioning: TRY. This is the format of an old, typical year data set, which did not include solar radiation data. The format can still be useful, however, as a format for measured data. The weather processor can process data in this format and add modeled solar data to the weather file. The format is described below.

[11] National Climatic Data Center, Federal Building, 151 Patton Avenue, Asheville, NC 28801-5001, http://www.ncdc.noaa.gov.

[12] National Renewable Energy Laboratory, 1617 Cole Boulevard, Golden, CO ;80401-3393, http://www.nrel.gov.

TMY2

The TMY2s are data sets of hourly values of solar radiation and meteorological elements for a one-year period. Their intended use is for computer simulations of solar energy conversion systems and building systems to facilitate performance comparisons of different system types, configurations, and locations in the United States and its territories. Because they represent typical rather than extreme conditions, they are not suited for designing systems to meet the worst-case conditions occurring at a location.

To distinguish between the old and new TMY data sets, the new TMY data sets are referred to as TMY2s. TMY and TMY2 data sets cannot be used interchangeably because of differences in time (solar versus local), formats, elements, and units. Unless they are revised, computer programs designed for TMY data will not work with TMY2 data.

The TMY2 data sets and manual were produced by the National Renewable Energy Laboratory's (NREL's) Analytic Studies Division under the Resource Assessment Program, which is funded and monitored by the U.S. Department of Energy's Office of Solar Energy Conversion.

A format description is on NREL's website at http://rredc.nrel.gov/solar/old_data/nsrdb/tmy2/

WYEC2

(*excerpt from the Watsun Simulation Lab website at http://dial.uwaterloo.ca/~watsun/ cwed.htm*)
Canadian weather for energy calculations (CWEC) files are typical year sets of meteorological data in WYEC2 format; they include such quantities as solar radiation (global, diffuse, direct), dry bulb, and dew point temperatures, wind speed and direction, atmospheric pressure, etc., on an hourly basis. They were developed by the Watsun Simulation Laboratory under the auspices of the National Research Council of Canada.

(*excerpt from the Watsun Simulation Lab website at http://dial.waterloo.ca//~watsun/ cwecovvw.htm*)
The CWEC files are created by concatenating twelve Typical Meteorological Months selected from a database of, in most cases, 30 years of data. The method is similar to TMY procedure developed in the 1980s by Sandia National Laboratory. The months are chosen by statistically comparing individual with long-term monthly means for daily total global radiation, mean-, minimum- and maximum- dry bulb temperatures, mean-, minimum- and maximum- dew point temperatures, and mean and maximum wind speeds. The composite index used to select the most 'typical' months uses the following weights (%):

Parameter	Dry Bulb Max	Dry Bulb Min	Dry Bulb Mean	Dew Point Max	Dew Point Min	Dew Point Mean	Wind Speed Max	Wind Speed Mean	Daily Solar Rad.
Weight(%)	5	5	30	2.5	2.5	5	5	5	40

Additional consideration is given, in the selection process, to the statistics and persistence structures of the daily mean dry bulb temperature and daily total radiation. A complete description of the procedure used can be found in: D.L. Siurna, L.J. D'Andrea, K.G.T. Hollands, "A Canadian Representative Meteorological Year for Solar System Simulation," Proceedings of the 10th annual conference of the Solar Energy Society of Canada (SESCI '84), August 1-6, 1984, Calgary, Alberta, Canada.

In the CWEC files, no missing values will be found in the following WYEC2 fields: extraterrestrial irradiance (101), global horizontal irradiance (102), direct normal irradiance (103), diffuse horizontal irradiance (104), weather (204), station pressure (205), dry bulb temperature (206), dew point temperature (207), wind direction (208), wind speed (209), total sky cover (210), opaque sky cover (211), snow cover (212). The original long-term data sets (up to 40 years of data) from which the CWEC files were derived can also be obtained directly from Environment Canada

WEATHER PROCESSOR

The weather processor is a batch or command line program called doewth or doewth.exe, depending on the computing environment. The primary function of the weather processor is to read hourly weather data in a variety of formats, extract the data needed by the simulation engine, and write a packed binary weather file which is used by the simulation program. In addition to its primary function (called packing) the weather processor can produce hourly listings of raw or packed weather files in a readable format and can produce a summary report of the data on a packed weather file.

Input and Output

The weather processor requires 2 input files. One, called WEATHR.TMP, is the hourly weather data. This will be either an ASCII (text) file, if raw weather data is being packed, or a binary file, if a packed weather file is being listed or summarized. The second input file, INPUT.TMP, is a short ASCII file that tells the weather processor what functions it is to perform and supplies any additional information needed to perform the task. This file is described in detail below.

Output is on two files. The file OUTPUT. contains any listings, reports, and error messages. The file NEWTH.TMP contains the packed weather file, if one is being created.

Copying, renaming, and saving these files will normally be performed by a procedure file specific to the computing environment being used.

Input Description

Here we describe the input file (INPUT.TMP) content needed to perform the 3 functions:

- Creating a weather file (packing)
- Listing a weather file
- Producing a statistical summary.

PACK

Line 1:	The word PACK in columns 1-4.
Line 2:	The station name in columns 1-20. This name will be written on the output file as identification. The entry here is for the user only and is arbitrary.
Line 3:	The data on this card is entered as shown in Table 16below. When the format is shown as L, it signifies that the datum must be left justified in the columns indicated. The format R signifies that the data must be right justified in the columns indicated, and the format D means that the value should be entered with a decimal point (neither right or left justification is required). For those with FORTRAN background: L corresponds to A6, R to I6, and D to F6.1.
Line 4:	Contains the 12 clearness numbers (one per month) in D format in column intervals 1-6, 7-12, 13-18, etc. (skip for TMY2; unused for WYEC2, can be just 1.0).
Line 5:	Contains the 12 ground temperatures (one per month in F) in D format in column intervals 1-6, 7-12, 13-18, etc. (skip for TMY2). A value of –999 will flag

the program to calculate the ground temperature using the method of Kusuda and Achenbach (ASHRAE Trans. 41 (1965) p. 61).

Table 16 Line 3 Format

Columns	Format	Description
1-6	L	A code-word specifying the unpacked file type. Options are TMY2, WYEC2, CD144, TRYSLM[13], TD9685, and OTHER[14].
7-12	R	Weather station number. This is required.
13-18	R	The year of the weather data (e.g., 1972). This is required for CD144 and TD9685 files (which can contain several years of weather data). For other files, –999 should be input.
19-24	R	Time zone (as in the SITE-PARAMETERS instruction)
25-30	D	Latitude (degrees)
31-36	D	Longitude (degrees)
37-42	L	A code-word specifying the number of bits per word to be used in packing the output file. The options are 60-BIT or 30-BIT (for 32 bit machines)
43-48	L	A code-word specifying the type of output file. The options are NORMAL and SOLAR. NORMAL produces a packed weather file with no solar data. SOLAR produces a file containing solar information.
49-54	R	Interpolation interval. The program fills in missing data by linear interpolation between the last and the next value present, if the number of hours of missing data is less than or equal to the interpolation interval. If more hours of data are missing than the interpolation interval, it still does interpolation up to 24 hours and a warning message is issued. If more than 24 hours are missing, the previous value is used. The interpolation interval must be less than 24 [15].
55-60	D	This sets the maximum dry bulb temperature change allowed in one hour. Changes larger than this will cause a warning message to be printed.
61-66	D	Soil thermal diffusivity (ft2/hr). Used for calculating monthly ground temperatures. A value of 0.010 can be used for dry soil, 0.025 for average soil, and 0.050 for wet soil.
67-72	D	Altitude (feet), used in CD144 and TD9685.
73-78	R	Location needed only for TRYSLM to choose a cloud cover model. See Table 17.

[13] TRYSLM reads a file in TRY format and adds ersatz solar data using the ASHRAE clear sky, SOLMET cloud cover, and the Erbs, Klein and Duffie direct normal models.

[14] If OTHER is chosen, the user must first have written a subroutine with the name OTHER to be placed in the code. This subroutine will interpret the format on the unpacked tape to the Weather Processor. The user must have access to the source code of the Weather Processor to see how to do this.

[15] The weather processor makes no evaluation of the data to see that it is internally consistent, except that during interpolation it never allows the wet bulb temperature to exceed the dry bulb temperature, or the dew point temperature to exceed the wet bulb temperature.

Table 17 ILOC and Station Name

ILOC	STATION NAME	ILOC	STATION NAME	ILOC	STATION NAME
1	APALACHICOLA, FL	10	DODGE CITY, KS	19	MIAMI, FL
2	ALBUQUERQUE, NM	11	EL PASO, TX	20	NASHVILLE, TN
3	BISMARCK, ND	12	ELY, NV	21	NEW YORK, NY
4	BOSTON, MA	13	FORT WORTH, TX	22	NORTH OMAHA, NE
5	BROWNSVILLE, TX	14	FRESNO, CA	23	PHOENIX, AZ
6	CAPE HATTERAS, NC	15	GREAT FALLS, MT	24	SANTA MARIA, CA
7	CARIBOU, ME	16	LAKE CHARLES, LA	25	SEATTLE-TACOMA, WA
8	CHARLESTON, SC	17	MADISON, WI	26	WASHINGTON, DC
9	COLUMBIA, MO	18	MEDFORD, OR	27	

LIST

Line 1: The word LIST in columns 1-4.

Line 2: As shown in Table 18

Table 18 Line 2 Format

Colums	Format	Description
1-6	L	Input type or file type. Options are PACKED, OTHER, TRY, WYEC2, CD144, TMY2, TD9685, and TRYSLM.
7-12	R	Year of weather data (e.g., 1972). Required for CD144 and TD9685 files. TRY, TMY2, WYEC2, TMYSLM, and PACKED should have –999.
13-18	R	Station number. PACKED can have –999 entered here.
19-24	R	Beginning month of listing (1-12 for January - December)
25-30	R	Ending month of listing (1-12 for January - December)

STAT

Only one card is necessary for the STAT option with the word STAT in columns 1-4.

Multiple Options

To exercise several options in the same computer run simply concatenate the input for several functions. The last card in the entire file--whether running one or several options must have the word END in columns 1-3.

Examples

1. To PACK, LIST for 12 months, and STAT a Mexico City CD144 file.

```
         1         2         3         4         5         6
123456789012345678901234567890123456789012345678901234567890123456789
PACK
MEXICO CITY 90
CD144  76679  1990     6 19.24 99.0930-BITNORMAL     4    20 .025
1.0   1.0   1.0   1.0   1.0   1.0   1.0   1.0   1.0   1.0   1.0   1.0
-999.
LIST
PACKED  -999  -999     1    12
STAT
END
```

2. To LIST the month of January 1990 from a Mexico City CD144 file.

```
         1         2         3         4         5         6
12345678901234567890123456789012345678901234567890123456789012345678
LIST
CD144   1990 76679     1     1
END
```

ASCII WEATHER FILES

Coverting between binary and ASCII data for DOE-2

Our last topic involves the transfer of packed weather files from one computer system to another. The weather files are packed, binary files and, therefore, they cannot be transferred from computer to computer unless the computers and operating systems are identical. Frequently, the original raw data used to produce the packed files has been lost or discarded. When a new computer system is installed, or when it is necessary to do runs at another site, a method is needed to preserve and transfer the data contained in the packed files. To do this, we run a small program called BIN2TXT, which reads a packed binary weather file and writes out an ASCII file containing the same information. The ASCII file can then be written to disk, along with a program called TXT2BIN, which reverses the process (i.e., read the ASCII weather tape and output a packed binary compatible file suitable for the new computer system). The disk recipient just has to read the disk, compile TXT2BIN and execute it with the ASCII weather file as input. Both BIN2TXT and TXT2BIN are in FORTRAN and will compile and execute on a PC. For other systems, the code may need minor changes (to the OPEN statements, for instance).

The ASCII weather file can be examined and edited on a terminal, using the local editor. This means that the technique can be used when you want to make changes to a packed weather file. The format of the ASCII file has been chosen to be easily readable by humans.

With reference to the programs that follow, TXT2BIN reads a one-line input file (INPUT.TMP) in addition to the file containing the weather data. INPUT.TMP tells TXT2BIN what type of packed binary file to produce. Note that only the first line is used; the subsequent lines are explanatory. The numbers on the first line must be in columns 13 and 31, respectively.

INPUT.TMP for TXT2BIN

```
----+----1----+----2----+----3----+----4----+----5----+----6----+----7
WORD SIZE = 2     FILE TYPE = 2
 WORD SIZE = 1 MEANS 60-BIT, 2 MEANS 30-BIT
 FILE TYPE = 1 MEANS OLD, 2 MEANS NORMAL (NO SOLAR DATA),
 3 MEANS THE DATA HAS SOLAR DATA
```

TXT2BIN

```
      PROGRAM TXT2BIN
C
C     THIS PROGRAM READS A FORMATTED WEATHER FILE (WEATHER.FMT)
C     AND A FORMATTED INPUT FILE (FMTWTH.INP) AND WRITES A
C     PACKED BINARY DOE2 WEATHER FILE (WEATHER.BIN)
C
      DIMENSION CLN(12),GT(12),MDAYS(12),IDAT(1536),IWDID(5)
C
      DATA MDAYS / 31,28,31,30,31,30,31,31,30,31,30,31 /
C
      OPEN(UNIT=12,FILE='INPUT.DAT')
      OPEN (UNIT=11,FILE='WEATHER.FMT',blocksize=710000)
      OPEN (UNIT=10,FILE='WEATHER.BIN',FORM='UNFORMATTED',
     $      recordtype='variable',recl=6200,blocksize=148992)
C
C     IWSZ           WORD SIZE          1 = 60-BIT, 2 = 30-BIT
C     IFTYP          FILE TYPE          1 = OLD, 2 = NORMAL (NO SOLAR),
C                                       3 = THE DATA HAS SOLAR
C     IWDID          LOCATION I.D.
C     IWYR           YEAR
C     WLAT           LATITUDE
C     WLONG          LONGITUDE
C     IWTZN          TIME ZONE NUMBER
C     IWSOL          SOLAR FLAG         FUNCTION OF IWSZ + IFTYP
C     CLN            CLEARNESS NO.
C     GT             GROUND TEMP.       (DEG R)
C     KMON           MONTH              (1-12)
C     KDAY           DAY OF MONTH
C     KH             HOUR OF DAY
C     WBT            WET BULB TEMP      (DEG F)
C     DBT            DRY BULB TEMP      (DEG F)
C     PATM           PRESSURE           (INCHES OF HG)
C     CLDAMT         CLOUD AMOUNT       (0 - 10)
C     ISNOW          SNOW FLAG          (1 = SNOWFALL)
C     IRAIN          RAIN FLAG          (1 = RAINFALL)
C     IWNDDR         WIND DIRECTION     (0 - 15; 0=N, 1=NNE, ETC)
C     HUMRAT         HUMIDITY RATIO     (LB H2O/LB AIR)
C     DENSTY         DENSITY OF AIR     (LB/CU FT)
C     ENTHAL         SPECIFIC ENTHALPY  (BTU/LB)
C     SOLRAD         TOTAL HOR. SOLAR   (BTU/HR-SQFT)
C     DIRSOL         DIR. NORMAL SOLAR  (BTU/HR-SQFT)
C     ICLDTY         CLOUD TYPE         (0 - 2)
C     WNDSPD         WIND SPEED         KNOTS
C
      IWSZ  = 0
      IFTYP = 0
      REWIND 12
      READ (12,9001,END=12) IWSZ,IFTYP
 9001 FORMAT(12X,I1,17X,I1)
   12 REWIND 11
      READ (11,9002) (IWDID(I),I=1,5),IWYR,WLAT,WLONG,IWTZN,IWSOL
 9002 FORMAT(5A4,I5,2F8.2,2I5)
      PRINT 3,(IWDID(I),I=1,5)
    3 FORMAT(' FMTWTH2: Processing weather data for: ',5A4)
      IF(IWSZ .eq. 0) THEN
```

```
        PRINT 1, IWSOL
    1   FORMAT('          Using file type found on WEATHER.FMT of ',I1)
      ELSE
        IWSOL = IWSZ + (IFTYP-1)*2 - 1
        PRINT 2, IWSOL
    2   FORMAT('          Using file type found on INPUT.TMP of ',I1)
      ENDIF
      PRINT 4
    4 FORMAT('          (5 = Solar data on file, 3= No solar on file)')
      READ (11,9003) (CLN(I),I=1,12)
      READ (11,9004) (GT(I),I=1,12)
 9003 FORMAT(12F6.2)
 9004 FORMAT(12F6.1)
      DO 1000 IM=1,12
      IDE = MDAYS(IM)
      DO 1000 ID=1,IDE
      IRECXO = IM*2 + (ID-1)/16 - 1
      IDXO = MOD(ID-1,16) + 1
      DO 500 IH=1,24
      READ (11,9005) KMON, KDAY, KH, WBT, DBT, PATM, CLDAMT, ISNOW,
     1               IRAIN, IWNDDR, HUMRAT, DENSTY, ENTHAL, SOLRAD,
     2               DIRSOL, ICLDTY, WNDSPD
 9005 FORMAT(3I2,2F5.0,F6.1,F5.0,2I3,I4,F7.4,F6.3,F6.1,2F7.1,I3,F5.0)
      ISOL = INT(SOLRAD + .5)
      IDN = INT(DIRSOL + .5)
      IWET = INT(WBT+99.5)
      IDRY = INT(DBT+99.5)
      IPRES = INT(PATM*10.-149.5)
      ICLDAM = INT(CLDAMT)
      IWNDSP = INT(WNDSPD+0.5)
      IHUMRT = INT(HUMRAT*10000.+0.5)
      IDENS = INT(DENSTY*1000.-19.5)
      IENTH = INT(ENTHAL*2.0+60.5)
      IP1 = (IDXO-1)*96 + IH*4 - 3
      IDAT(IP1) = IPRES*65536 + IWET*256 + IDRY
      IDAT(IP1+1) = ISOL*1048576 + IDN*1024 +
     1              ICLDAM*64 + ISNOW*32 + IRAIN*16 + IWNDDR
      IDAT(IP1+2) = IHUMRT*128 + IDENS
      IDAT(IP1+3) = IENTH*2048 + ICLDTY*128 + IWNDSP
  500 CONTINUE
      IF ((ID .NE. 16) .AND. (ID .NE. IDE))  GO TO 1000
      WRITE (10) IWDID,IWYR,WLAT,WLONG,IWTZN,IRECXO,IDE,CLN(IM),
     1           GT(IM),IWSOL,IDAT
 1000 CONTINUE
      END
```

BIN2TXT

```
      PROGRAM BIN2TXT
C
C     THIS PROGRAM READS A PACKED BINARY DOE-2 WEATHER FILE AND
C     CREATES A FORMATTED WEATHER FILE AS OUTPUT.  THE INPUT
C     FILE IS TAPE10, THE OUTPUT FILE IS TAPE11.
C
      DIMENSION CLN(12),GT(12),MDAYS(12),IDAT30(1536),
     _          IWDID(5),IWTH(14)
      DIMENSION XMASK(16,2), CALC(16)
      INTEGER IDUM
C
      DATA MDAYS / 31,28,31,30,31,30,31,31,30,31,30,31 /
      DATA IWDID /5*4H    /
      DATA XMASK / -99., -99., 15., 0., 0., 0., 0., 0., .02, -30., 0.,
     1             .0, .0, .0, .0, 10.,
     2              1., 1., .1, 1., 1., 1., 1., .0001, .001, .5,
     3             1., 1., 1., 1., 0., 0. /
C
      OPEN (UNIT=11,FILE='WEATHER.FMT',blocksize=710000)
      OPEN (UNIT=10,FILE='WEATHER.BIN',FORM='UNFORMATTED',
     $      recordtype='variable',recl=6200,blocksize=148992)
      REWIND 10
      DO 100 IM1=1,12
      READ (10) (IWDID(I),I=1,5),IWYR,WLAT,WLONG,IWTZN,LRECX,NUMDAY,
     _          CLN(IM1),GT(IM1),IWSOL
      READ (10) IDUM
  100 CONTINUE
      REWIND 10
      LRECX = 0
      WRITE (11,9001) (IWDID(I),I=1,5),IWYR,WLAT,WLONG,IWTZN,IWSOL
      WRITE (11,9002) (CLN(I),I=1,12)
      WRITE (11,9003) (GT(I),I=1,12)
 9001 FORMAT(5A4,I5,2F8.2,2I5)
 9002 FORMAT(12F6.2)
 9003 FORMAT(12F6.1)
      DO 1000 IM2=1,12
      IDE = MDAYS(IM2)
      DO 1000 ID=1,IDE
      DO 1000 IH=1,24
  105 IRECX = IM2*2 + (ID-1)/16 - 1
      IDX = MOD(ID-1,16) + 1
      IF (IRECX-LRECX) 200,400,300
  200 IDIF = LRECX - IRECX + 1
      DO 220 I=1,IDIF
      BACKSPACE 10
  220 CONTINUE
  300 READ (10) IWDID,IWYR,WLAT,WLONG,IWTZN,LRECX,NUMDAY,CLRNES,
     _          TGRND,IDUM,IDAT30
      GO TO 105
  400 CONTINUE
      IP1 = 96*(IDX-1) + 4*IH - 3
      IWTH(3) = IDAT30(IP1)/65536
      IWTH(1) = MOD(IDAT30(IP1),65536)/256
      IWTH(2) = MOD(IDAT30(IP1),256)
      IWTH(11) = IDAT30(IP1+1)/1048576
```

```
      IWTH(12) = MOD(IDAT30(IP1+1),1048576)/1024
      IWTH(4) = MOD(IDAT30(IP1+1),1024)/64
      IWTH(5) = MOD(IDAT30(IP1+1),64)/32
      IWTH(6) = MOD(IDAT30(IP1+1),32)/16
      IWTH(7) = MOD(IDAT30(IP1+1),16)
      IWTH(8) = IDAT30(IP1+2)/128
      IWTH(9) = MOD(IDAT30(IP1+2),128)
      IWTH(10) = IDAT30(IP1+3)/2048
      IWTH(13) = MOD(IDAT30(IP1+3),2048)/128
      IWTH(14) = MOD(IDAT30(IP1+3),128)
      DO 500 I=1,14
      CALC(I) = FLOAT(IWTH(I))*XMASK(I,2) + XMASK(I,1)
  500 CONTINUE
      ISNOW = INT(CALC(5) + .01)
      IRAIN = INT(CALC(6) + .01)
      IWNDDR = INT(CALC(7) + .01)
      ICLDTY = INT(CALC(13) + .01)
C
C              IM2            MOMTH              (1-12)
C              ID             DAY OF MONTH
C              IH             HOUR OF DAY
C              CALC(1)        WET BULB TEMP      (DEG F)
C              CALC(2)        DRY BULB TEMP      (DEG F)
C              CALC(3)        PRESSURE           (INCHES OF HG)
C              CALC(4)        CLOUD AMOUNT       (0 - 10)
C              ISNOW          SNOW FLAG          (1 = SNOWFALL)
C              IRAIN          RAIN FLAG          (1 = RAINFALL)
C              IWNDDR         WIND DIRECTION     (0 - 15; 0=N, 1=NNE, ETC)
C              CALC(8)        HUMIDITY RATIO     (LB H2O/LB AIR)
C              CALC(9)        DENSITY OF AIR     (LB/CU FT)
C              CALC(10)       SPECIFIC ENTHALPY  (BTU/LB)
C              CALC(11)       TOTAL HOR. SOLAR   (BTU/HR-SQFT)
C              CALC(12)       DIR. NORMAL SOLAR  (BTU/HR-SQFT)
C              ICLDTY         CLOUD TYPE         (0 - 2)
C              CALC(14)       WIND SPEED         KNOTS
C
  900 WRITE (11,9005) IM2, ID, IH, CALC(1), CALC(2), CALC(3), CALC(4),
     1               ISNOW, IRAIN,IWNDDR, CALC(8), CALC(9), CALC(10),
     2               CALC(11), CALC(12), ICLDTY, CALC(14)
 9005 FORMAT(3I2,2F5.0,F6.1,F5.0,2I3,I4,F7.4,F6.3,F6.1,2F7.1,I3,F5.0)
 1000 CONTINUE
      ENDFILE 11
      END
```

PROCESSING NONSTANDARD WEATHER DATA

The weather processor is capable of processing raw weather data in a variety of formats into a compatible form. Quite frequently, however, users obtain weather data in a format that is unknown to the weather processor. The user then has two alternatives: convert the data into a known format; or process the raw weather data directly by filling in the empty subroutine OTHER in the weather processor.

Use of the subroutine OTHER

OTHER is a typical weather data processing subroutine in the weather processor - it just doesn't contain any code! But like the other such routines (TRYDCD, TMYDCD, etc.) it is called once every 24 hours by subroutine PACKER, and its use can be triggered by the weather processor input. Putting OTHER in the first 5 columns of the 3rd record of the PACK input sequence informs the weather packer that subroutine OTHER will be used for reading in and processing the raw data. It is up to the user to then supply the code in OTHER that will do this. Basically, the arrays in the common block /RAWDAT/ must be filled for each call to OTHER. The arrays are dimensioned 24, and are all integers. They are

IDRY	dry bulb temperature in °F, rounded to the nearest whole degree.
IWET	wet bulb temperature in °F, rounded to the nearest whole degree.
IDEW	dew point temperature in °F, rounded to the nearest whole degree.
IPRESS	atmospheric pressure in inches of Hg times 100 (so 29.92 will be 2992, etc.)
IWNDSP	wind speed, in knots (!) (nearest whole knot).
ICLAMT	cloud amount (sky cover), 0 - 10; 0 = no clouds, 10 = totally cloudy.
ISOL,	total solar on a horizontal surface and direct normal (beam) solar radiation, both in Btu/ft2-hr,
IDN	nearest whole unit.
IWNDIR	wind direction in compass points (0 - 15). 0 is north, 15 is NNW.
ICLTY	cloud type; takes the values 0, 1, and 2. Type 0 is the most transparent cloud category. It contains TRY types 8 and 9 (cirrus and cirrostratus or cirrocumulus). Type 1 is the most opaque. It contains TRY type 2 (stratus or fractus stratus). Type 2 is of intermediate transparency. It contains all other types of clouds, and is a good default if no cloud type information is available.
ICLTY1	cloud type - UNUSED
IRN, ISN	the rain and snow flags. Set to 1 if raining or snowing, 0 otherwise. IRN and ISN are never used, but it is nice to set them anyway. When printed out along with the other weather variables with the LIST option of the weather processor, they can help explain some otherwise odd looking weather or solar data.

Frequently the raw data will include dry bulb and relative humidity, instead of dry bulb, wet bulb, and dew point as required by the weather processor. The user should use the procedure given in 1993 ASHRAE Fundamentals, page

6.14, situation number 3. A slightly different procedure is used in the example OTHER subroutine at the end of this article.

In the case of solar data, the weather processor needs total horizontal and direct normal. The data is often in the form of direct and diffuse on a horizontal surface. The cosine of the solar zenith angle (or the sine of the solar altitude) must then be calculated in order to obtain the direct normal from the direct horizontal. This calculation involves knowing the solar declination angle and the equation of time; it is complicated by the fact that the cosine of the zenith angle must usually be averaged over a one hour time bin, since the solar data point is usually the average over 1 hour of a number of data points taken at less than 1 hour intervals. In this case it is best to simply follow the procedure shown in the example subroutine. As in the example, it is usually necessary to do a limit check on the resulting direct normal, particularly at sunrise and sunset, where the data is frequently bad.

Sometimes only total horizontal solar data is available. A model must then be employed to obtain the direct radiation from the total. We recommend the model of Erbs, Klein, and Duffie, described in Solar Energy, volume 28, page 293 (1982). See the FORTRAN fragment following the example OTHER subroutine.

Example of subroutine OTHER

```
      SUBROUTINE OTHER                                                  OTHER    2
C                                                                       OTHER    3
C              INSERT YOUR ROUTINE TO DECODE SPECIAL TAPE FORMAT        OTHER    4
C                                                                       OTHER    5
      COMMON  /LOCALD/   STALAT,STALON,SSTALA,CSTALA,TSTALA,HRSLON      /LOCALD/ 2
C                                                                       /LOCALD/ 3
      COMMON  /CONST/    DTOR, PIOVR2, PIOVR4, PIOV12, IBLNK            /CONST/  2
C                                                                       /CONST/  3
      COMMON  /TIMES/    IMNTH,IDAY,IHOUR,IRECXO,IDXO,IDAYL,ITIM        073b     2
C                                                                       /TIMES/  3
      COMMON  /MONTHC/   BEFORE(12),MDAYS(12),MNAMES(12)                /MONTHC/ 2
      INTEGER            BEFORE                                         /MONTHC/ 3
C                                                                       /MONTHC/ 4
      COMMON  /FILES/    INFIL,OUTFIL,INWTH,OUTWTH,SOLWTH,STOUT         /FILES/  2
      INTEGER            OUTFIL,OUTWTH,SOLWTH,STOUT                     /FILES/  3
C                                                                       /FILES/  4
      COMMON  /GETCRC/   IEOF                                           /GETCRC/ 2
      LOGICAL            IEOF                                           /GETCRC/ 3
C                                                                       /GETCRC/ 4
      COMMON  /PACKEC/   NTZ,ISTAT,XLAT,XLONG,                          /PACKEC/ 2
     1                   IYR,INTINT,DDBT,IBEGH,NBS,IDOYSH(5),ISSHFT(5), /PACKEC/ 3
     2                   ISHFT,ALT,SKYCOV(12),ILOC                      073a     1
C                                                                       /PACKEC/ 5
      COMMON  /PARAMS/   STOPIT,VERS,FIRST,NEWPAK,CALCGT                /PARAMS/ 2
      LOGICAL            STOPIT,VERS,FIRST,NEWPAK,CALCGT                /PARAMS/ 3
C                                                                       /PARAMS/ 4
      COMMON  /PCKINT/   KYR,KSTAT,CN(12),TG(12),KTIM,KARD(500),KYRL    -089     1
C                                                                       /PCKINT/ 3
      COMMON  /RAWDAT/   IDRY(24),IWET(24),IDEW(24),IPRESS(24),         /RAWDAT/ 2
     1                   IWNDSP(24),ICLAMT(24),ISOL(24),IDN(24),        /RAWDAT/ 3
     2                   IWNDIR(24),ICLTY(24),IRN(24),ISN(24),          CD144S   1
     3                   IOPQCA(24),IRELH(24),ICLTY1(24)                CD144S   2
      DIMENSION IRAW(24,14)                                             CD144S   3
      EQUIVALENCE        (IDRY(1),IRAW(1,1))                            /RAWDAT/ 6
      COMMON /LSTIME/   LSTHRS(24)                                      /RAWDAT/ 7
C                                                                       /RAWDAT/ 8
      COMMON  /REPORC/   IBEGM,IENDM                                    /REPORC/ 2
C                                                                       /REPORC/ 3
      COMMON  /UNDEF/    IUNDEF,UNDEF                                   /UNDEF/  2
C                                                                       /UNDEF/  3
C                                                                       OTHER   14
      DIMENSION DEABC(5)                                                MSWTH    5
C             DAY OF YEAR                                               MSWTH    6
      IDOY = BEFORE(IMNTH) + IDAY                                       MSWTH    7
C             GET SUN PARAMETERS                                        MSWTH    8
      CALL SUNPRM(IDOY,DEABC)                                           MSWTH    9
C             SOLAR CONSTANT                                            MSWTH   10
```

```
      SOLCON = 435.2*(1.+0.033*COS(DTOR*360.*FLOAT(IDOY)/365.))                 MSWTH   11
C             LOOP OVER HOURS IN THE DAY                                        MSWTH   12
      DO 1000 IH=1,24                                                           MSWTH   13
      DIRN = 0.                                                                 MSWTH   14
      DIF = 0.                                                                  MSWTH   15
      DIRH = 0.                                                                 MSWTH   16
C             READ IN WEATHER DATA                                              MSWTH   17
      READ (INWTH,9001) IHR,ID,IM,IWYR,TDRY,RELH,PRESMB,SOLH,                   MSWTH   18
     1                  WS,WD,SKYFR                                             MSWTH   19
 9001 FORMAT(3I2,I4,F10.1,F10.2,F10.1,F10.0,F10.1,F10.0,F10.1)                  MSWTH   20
C             CONVERT DRYBULB FROM CENTIGRADE TO FAHRENHEIT                     MSWTH   21
      TDRYF = 1.8*TDRY + 32.                                                    MSWTH   22
C             CALCULATE WETBULB AND DEWPOINT                                    MSWTH   23
C             SATURATED VAPOR PRESSURE                                          MSWTH   24
      PS = PPWVMS(TDRYF)                                                        MSWTH   25
C             PARTIAL PRESSURE                                                  MSWTH   26
      PW = RELH*PS                                                              MSWTH   27
C             CONVERT PRESSURE FROM MILLIBARS TO INCHES OF HG                   MSWTH   28
      PRESHG = .02953*PRESMB                                                    MSWTH   29
C             HUMIDITY RATIO                                                    MSWTH   30
      HUMRAT = 0.622*PW/(PRESHG-PW)                                             MSWTH   31
C             SPECIFIC ENTHALPY                                                 MSWTH   32
      ENTH = 0.24*TDRYF + (1061.+0.444*TDRYF)*HUMRAT                            MSWTH   33
      TWETF = WBF(ENTH,PRESHG)                                                  MSWTH   34
      Y = LOG(PW)                                                               MSWTH   35
      IF (PW .LE. 0.1836) THEN                                                  MSWTH   36
          TDEWF = 71.98 + 24.873*Y + 0.8927*Y*Y                                 MSWTH   37
      ELSE                                                                      MSWTH   38
          TDEWF = 79.047 + 30.579*Y + 1.8893*Y*Y                                MSWTH   39
      END IF                                                                    MSWTH   40
C             CONVERT WINDSPEED FROM M/S TO KNOTS                               MSWTH   41
      WSKNOT = 1.9438*WS                                                        MSWTH   42
C             TOTAL HORIZONTAL SOLAR; CONVERT FROM W/M**2 TO                    MSWTH   43
C             BTU/(FT**2)(HR)                                                   MSWTH   44
      SOLHOR = .31721*SOLH                                                      MSWTH   45
C             GET HOUR ANGLE AT UPPER AND LOWER HOUR BIN EDGE                   MSWTH   46
C             THIS IS LOCAL TIME!                                               MSWTH   47
      UL     = FLOAT(IH) - 12. + FLOAT(NTZ) + DEABC(2) - XLONG/15.              MSWTH   48
      BL     = UL - 1.                                                          MSWTH   49
C             SUNRISE AND SUNSET HOUR ANGLES                                    MSWTH   50
      SSHA   = ACOS(-TAN(STALAT)*TAN(DEABC(1)))/PIOV12                          MSWTH   51
      SRHA   = -SSHA                                                            MSWTH   52
C             RESET BIN BOUNDARIES TO ALLOW FOR SUNRISE AND SET                 MSWTH   53
      IF ((UL .LE. SRHA) .OR. (BL .GE. SSHA)) GO TO 500                         MSWTH   54
        IF (SRHA .GT. BL)  BL = SRHA                                            MSWTH   55
        IF (SSHA .LT. UL)  UL = SSHA                                            MSWTH   56
        IF ((UL-BL) .LT. 0.02)  GO TO 500                                       MSWTH   57
        IF (SOLHOR .EQ. 0) GO TO 500                                            MSWTH   58
        A = SIN(DEABC(1))*SIN(STALAT)                                           MSWTH   59
        B = COS(DEABC(1))*COS(STALAT)                                           MSWTH   60
C               INTEGRATE SOLAR Z DIREC. COSINE OVER BIN                        MSWTH   61
        COSZIN = A*(UL-BL) + B*(SIN(PIOV12*UL)-SIN(PIOV12*BL))/PIOV12           MSWTH   62
C               AVERAGE COSINE OF THE SOLAR ZENITH ANGLE FOR THE HOUR           MSWTH   63
        COSZAV = COSZIN/(UL-BL)                                                 MSWTH   64
C               EXTRATERRESTRIAL SOLAR HORIZONTAL                               MSWTH   65
        SOLEXH = SOLCON*COSZIN                                                  MSWTH   66
C               K sub T IS THE RATIO OF TERRESTRIAL TO EXTRATERRESTRIAL SOLAR   MSWTH   67
        RKT = AMIN1(SOLHOR/SOLEXH,0.9)                                          MSWTH   68
C               GET DIFFUSE COMPONENT FROM ERBS, KLEIN, AND DUFFIE              MSWTH   69
C               CORRELATION                                                     MSWTH   70
        IF (RKT .LE. 0.22)  DIF = SOLHOR*(1.-0.09*RKT)                          MSWTH   71
        RKT2 = RKT*RKT                                                          MSWTH   72
        IF ((RKT .GT. 0.22) .AND. (RKT .LE. 0.8))  DIF = SOLHOR*                MSWTH   73
     1   (.9511-.1604*RKT+4.388*RKT2-16.638*RKT*RKT2+12.336*RKT2*RKT2)          MSWTH   74
        IF (RKT .GT. 0.8)  DIF = 0.165*SOLHOR                                   MSWTH   75
C               DIRECT HORIZONTAL                                               MSWTH   76
        DIRH = AMAX1(0.,SOLHOR-DIF)                                             MSWTH   77
C               DIRECT NORMAL                                                   MSWTH   78
        DIRN = DIRH/COSZAV                                                      MSWTH   79
C               CHECK FOR MAXIMUM DIRECT NORMAL                                 MSWTH   80
        CALL MAXDIR(COSZAV,SOLCON,DIRMAX)                                       MSWTH   81
```

```
        DIRN = AMIN1(DIRN,DIRMAX)                                       MSWTH   82
  500 CONTINUE                                                          MSWTH   83
C             FILL THE DATA ARRAYS                                      MSWTH   84
      IDRY(IH) = IROUND(TDRYF)                                          MSWTH   85
      IWET(IH) = MIN0(IDRY(IH),IROUND(TWETF))                           MSWTH   86
      IDEW(IH) = MIN0(IWET(IH),IROUND(TDEWF))                           MSWTH   87
      IPRESS(IH) = IROUND(100.*PRESHG)                                  MSWTH   88
      IWNDSP(IH) = IROUND(WSKNOT)                                       MSWTH   89
      IWNDIR(IH) = IROUND(.0444444*WD)                                  MSWTH   90
      IF (IWNDIR(IH) .EQ. 16) IWNDIR(IH) = 0                            MSWTH   91
      IF (SKYFR.EQ.-999.) THEN                                          MSWTH   92
        ICLAMT(IH) = 5                                                  MSWTH   93
      ELSE                                                              MSWTH   94
        ICLAMT(IH) = IROUND(10.*SKYFR)                                  MSWTH   95
      ENDIF                                                             MSWTH   96
      ISOL(IH) = IROUND(SOLHOR)                                         MSWTH   97
      IDN(IH) = IROUND(DIRN)                                            MSWTH   98
      ICLTY(IH) = 2                                                     MSWTH   99
      ICLTY1(IH) = 2                                                    MSWTH  100
      IRN(IH) = 0                                                       MSWTH  101
      ISN(IH) = 0                                                       MSWTH  102
 1000 CONTINUE                                                          MSWTH  103
      RETURN                                                            OTHER   15
      END                                                               OTHER   16
```

Fortran fragment for direct radiation

```
C
C               Fortran fragment to calculate direct and diffuse
C               solar radiation from total solar and to get wet bulb
C               and dew point from dry bulb, relative humidity
C               and atmospheric pressure.
C
C     Start with: SOLHOR - total horiz. solar in btu/hr-area
C                 TDRYF  - dry bulb temperature in degrees Fahrenheit
C                 PRESHG - atmospheric pressure in inches of mercury
C                 RELHUM - relative humidity in percent
C
C     Define:     DTOR   - degrees to radians = pi/180
C                 PIOV12 - pi/12
C                 IH     - hour of the day (1 - 24)
C                 IDOY   - day of year (1 - 365)
C                 STALAT - weather station latitude in radians
C                 STALAT - station longitude in radians
C                 XLONG  - station longitude in degrees
C                 NTZ    - time zone (PST=8, EST=5, Greenwich=0)
C
C     Externals:  SUNPRM
C                 MAXDIR
C                 PPWVMS
C                 WBF
C                           These are all available from WTHPRC, the
C                           weather processor.
C
      DIMENSION DEABC(5)
C
      DIRN = 0.
      DIF = 0.
      DIRH = 0.
C
C               GET SOLAR CONSTANTS.  DEABC(1) IS THE SOLAR
C               DECLINATION ANGLE; DEABC(2) IS THE EQUATION
C               OF TIME.
      CALL SUNPRM(IDOY,DEABC)
C               GET SOLAR CONSTANT.  THIS FORMULA IS FROM KREIDER
C               AND RABL PAGE 237
C               BECKMAN, PAGE 7.
      SOLCON = 435.2*(1. + 0.033*COS(DTOR*360.*FLOAT(IDOY)/365.))
C               GET HOUR ANGLE AT UPPER AND LOWER HOUR BIN EDGE
C               THIS IS LOCAL TIME!
      UL     = FLOAT(IH) - 12. + FLOAT(NTZ) + DEABC(2) - XLONG/15.
      BL     = UL - 1.
C               SUNRISE AND SUNSET HOUR ANGLES
      SSHA   = ACOS(-TAN(STALAT)*TAN(DEABC(1)))/PIOV12
      SRHA   = -SSHA
C               RESET BIN BOUNDARIES TO ALLOW FOR SUNRISE AND SET
      IF ((UL .LE. SRHA) .OR. (BL .GE. SSHA))  GO TO 1100
      IF (SRHA .GT. BL)  BL = SRHA
      IF (SSHA .LT. UL)  UL = SSHA
      IF ((UL-BL) .LT. 0.02)  GO TO 1100
      IF (SOLHOR .EQ. 0) GO TO 1100
      A      = SIN(DEABC(1))*SIN(STALAT)
```

```
      B      = COS(DEABC(1))*COS(STALAT)
C               INTEGRATE SOLAR Z DIREC. COSINE OVER BIN
      COSZIN = A*(UL-BL) + B*(SIN(PIOV12*UL)-SIN(PIOV12*BL))/PIOV12
C               AVERAGE COSINE OF THE SOLAR ZENITH ANGLE FOR THE HOUR
      COSZAV = COSZIN/(UL-BL)
C               EXTRATERRESTRIAL SOLAR HORIZONTAL
      SOLEXH   = SOLCON*COSZIN
C               K sub T IS THE RATIO OF TERRESTRIAL TO EXTRATERRESTRIAL SOLAR
      RKT = AMIN1(SOLHOR/SOLEXH,0.9)
C               GET DIFFUSE COMPONENT FROM ERBS, KLEIN, AND DUFFIE
C               CORRELATION
      IF (RKT .LE. 0.22)  DIF = SOLHOR*(1.-0.09*RKT)
      RKT2   = RKT*RKT
      IF ((RKT .GT. 0.22) .AND. (RKT .LE. 0.8))  DIF = SOLHOR*
     1   (.9511-.1604*RKT+4.388*RKT2-16.638*RKT*RKT2+12.336*RKT2*RKT2)
      IF (RKT .GT. 0.8)  DIF = 0.165*SOLHOR
C               DIRECT HORIZONTAL
      DIRH   = AMAX1(0.,SOLHOR-DIF)
C               DIRECT NORMAL
      DIRN = DIRH/COSZAV
C               CHECK FOR MAXIMUM DIRECT NORMAL
      CALL MAXDIR(COSZAV,SOLCON,DIRMAX)
      DIRN = AMIN1(DIRN,DIRMAX)
 1100 CONTINUE
C
C               CALCULATE WET BULB AND DEW POINT.  THE PROCEDURE IS
C               BASICALLY THAT GIVEN ON PAGE 6.16, ASHRAE FUNDAMENTALS
C               1989, SITUATION NO. 3
C
C                SATURATED VAPOR PRESSURE
      PS = PPWVMS(TDRYF)
C                PARTIAL PRESSURE
      PW = .01*RELHUM*PS
C                HUMIDITY RATIO
      HUMRAT = .622*PW/(PRESHG-PW)
C                SPECIFIC ENTHALPY
      ENTH = .24*TDRYF + (1061.+.444*TDRYF)*HUMRAT
C                WET BULB TEMPERATURE
      TWETF = WBF(ENTH,PRESHG)
C                DEW POINT TEMPERATURE
      Y = LOG(PW)
      IF (PW .LE. .1836) THEN
          TDEWF = 71.98 + 24.873*Y + .8927*Y*Y
      ELSE
          TDEWF = 79.047 + 30.579*Y + 1.8893*Y*Y
      END IF
```

Using Standard Formats for Measured Data

As an alternative to using the OTHER subroutine for measured or nonstandard data, one can convert the measured/nonstandard data into one of the standard formats. For weather data with no solar data, use the TRY format and pack using the TRYSLM option. Only the following TRY data fields need be filled (all unfilled fields should contain 9's).

hour
day
month
year
station number
dry bulb temperature
wet bulb temperature
dew point temperature
wind direction
wind speed
station pressure
total sky cover
type of lowest cloud layer
weather

For data sets with measured solar data, use the TMY2 format. The only hourly fields that need to be filled are:

hour
day
month
year
global horizontal radiation
direct normal radiation
opaque sky cover
dry bulb temperature
dew point temperature
atmospheric pressure
wind direction
wind speed
present weather

In the TMY2 header record, all the data items should be filled.

TRY Format

Table 19 TRY Format

FILE FIELD NUMBER	COLUMNS	ELEMENT
001	01 - 05	STATION NUMBER
002	06 - 08	DRY BULB TEMPERATURE
003	09 - 11	WET BULB TEMPERATURE
004	12 - 14	DEW POINT TEMPERATURE
005	15 - 17	WIND DIRECTION
006	18 - 20	WIND SPEED
007	21 - 24	STATION PRESSURE
008	25	WEATHER
009	26 - 27	TOTAL SKY COVER
010	28 - 29	AMOUNT OF LOWEST CLOUD LAYER
011	30	TYPE OF LOWEST CLOUD OR OBSCURING PHENOMENA
012	31 - 33	HEIGHT OF BASE OF LOWEST LAYER
013	34 - 35	AMOUNT OF SECOND CLOUD LAYER
014	36	TYPE OF CLOUD - SECOND LAYER
015	37 - 39	HEIGHT OF BASE OF SECOND LAYER
016	40 - 41	SUMMATION AMOUNT OF FIRST TWO LAYERS
017	42 - 43	AMOUNT OF THIRD CLOUD LAYER
018	44	TYPE OF CLOUD - THIRD LAYER
019	45 - 47	HEIGHT OF BASE OF THIRD LAYER
020	48 - 49	SUMMATION AMOUNT OF FIRST THREE LAYERS
021	50 - 51	AMOUNT OF FOURTH CLOUD LAYER
022	52	TYPE OF CLOUD - FOURTH LAYER
023	53 - 55	HEIGHT OF BASE OF FOURTH LAYER
024	56 - 59	SOLAR RADIATION [16]
025	60 - 69	BLANK3
026	70 - 73	YEAR
027	74 - 75	MONTH
028	76 - 77	DAY
029	78 - 79	HOUR
030	80	BLANK

[16] The DOE-2 weather processor recognizes the following solar data in TRY format:
Columns 57-59 Total horizontal radiation in Btu/ft2-hr
Columns 61-53 Direct normal radiation in Btu/ft2-hr

Table 20 TMY Format (Continued)

File Field Numer	File Positions	Element	File Configuration	Code Definitions and Remarks
001	01 - 05	STATION NUMBER	01001 - 98999	Unique number used to identify each station. Usually a WBAN number, but occasionally a WMC or other number system.
002	06 - 08	DRY BULB TEMP	000 - 140	Specified temperature in whole °F
003	09 - 11	WET BULB TEMP	-01 - -80	000 - 140 = 0° - +14°F -01 - -80 = -l - -80°F 999 = Missing
004	12 - 14	DEW POINT TEMP	999	
005	15 - 17	WIND DIRECTION	000 – 360 999	Direction from which the wind is blowing in whole degrees. 000 = Calm 001-360 = 001° - 360° 999 = Missing Note: Prior to 1964, direction was recorded to only 16 intervals (points of the compass). The following scheme was used to convert these values to whole degrees. FILE ORIGINAL CODE 000 = Calm 360 = North 023 = North Northeast 045 = Northeast 068 = East Northeast 090 = East 113 = East Southeast 135 = Southeast 158 = South Southeast 180 = South 203 = South Southwest 225 = Southwest 248 = West Southwest 270 = West 293 = West Northwest 315 = Northwest 338 = North Northwest
006	18 - 20	WIND SPEED	000 - 230 999	Wind speed in whole knots. 000 = Calm 001-230 = l-230 knots 999 = Missing
007	21 - 24	STATION PRESSURE	1900 - 3999 9999	Pressure at station in inches and hundredths of Hg. 1900-3999 = 19.00 - 39.99 in Hg. 9999 = Missing
008	25	WEATHER	0 - 9	Occurrence of weather at the time of

File Field Numer	File Positions	Element	File Configuration	Code Definitions and Remarks
				observation. 0 = No weather or obstructions 1 = Fog 2 = Haze 3 = Smoke 4 = Haze and smoke 5 = Thunderstorm 6 = Tornado 7 = Liquid precipitation (rain, rain showers, freezing rain, drizzle, freezing drizzle) 8 = Frozen precipitation (snow, snow showers, snow pellets, snow grains, sleet, ice pellets, hail) 9 = Blowing dust, blowing sand, blowing spray, dust Note: Original observations may contain combinations of these ele-ments. Whenever this occurred, a priority was assigned for the pur-pose of indicating weather in this file (1) - Liquid precip – 7 (2) - Frozen precip – 8 (3) - Obstructions to vision - l, 2, 3, 4, 9 (4) - Thunderstorm (no precip) – 5 (5) - Tornado (no precip) - 6
009	26 - 27	TOTAL SKY COVER	00 - 10	Amount of the celestial domecovered by clouds or obscuring phenomena in tenths. 00-10 = 0-10 tenths 99 = Missing
010	28 - 29	AMT OF LOWEST CLOUD LAYER	99	
013	34 - 35	AMT OF SECOND CLOUD LAYER		
016	40 - 41	SUM OF FIRST TWO LAYERS		
017	42 - 43	AMT OF THIRD CLOUD LAYER		
020	48 - 49	SUM OF FIRST THREE LAYERS		
021	50 - 51	AMT OF FOURTH CLOUD LAYER		
011	30	TYPE OF LOWEST CLOUD OR	0 - 9	Generic cloud type or obscuring phenomena

File Field Numer	File Positions	Element	File Configuration	Code Definitions and Remarks
		OBSCURING PHENOMENA		0 = Clear 1 = Fog or other obscuring phenomena 2 = Stratus or Fractus Stratus 3 = Stratocumulus 4 = Cumulus or Cumulus Fractus 5 = Cumulonimbus or Mammatus 6 = Altostratus or Nimbostratus 7 = Altocumulus 8 = Cirrus 9 = Cirrostratus or Cirrocumulus 9 = Unknown if the amount of cloud is 99
014	36	TYPE OF CLOUD - SECOND LAYER		
018	44	TYPE OF CLOUD - THIRD LAYER		
022	52	TYPE OF CLOUD - FOURTH LAYER		
012	31 - 33	HEIGHT OF BASE OF LOWEST LAYER	000 - 760	Height of base of clouds or obscuring phenomena in hundreds of feet 000-760 = 0-76,000 feet 777 = Unlimited - clear 888 = Cirroform clouds of unknown height 999 = Missing
015	37 - 39	HEIGHT OF BASE OF SECOND LAYER	777	
019	45 - 47	HEIGHT OF BASE OF THIRD LAYER	888	
023	53 - 55	HEIGHT OF BASE OF FOURTH LAYER	999	
024	56 - 59	SOLAR RADIATION	0000 - 1999 9999	Total solar radiation in Langleys to tenths. Values are for the hour ending at time indicated in Field 029. 0000-1999 = 0-199.9 Langleys 9999 = Missing
025	60 - 69	BLANK		Blank field--reserved for future use.
026	70 - 73	YEAR	1948 - 1980	Year
027	74 - 75	MONTH	01 - 12	Month of year 01 = Jan 02 = Feb etc.
028	76 - 77	DAY	01 - 31	Day of month
029	78 - 79	HOUR	00 - 23	Hour of observation in Local Standard Time. 00-23 = 0000-2300 LST
030	80	BLANK		Blank field, reserved for future use.

CD144 Format

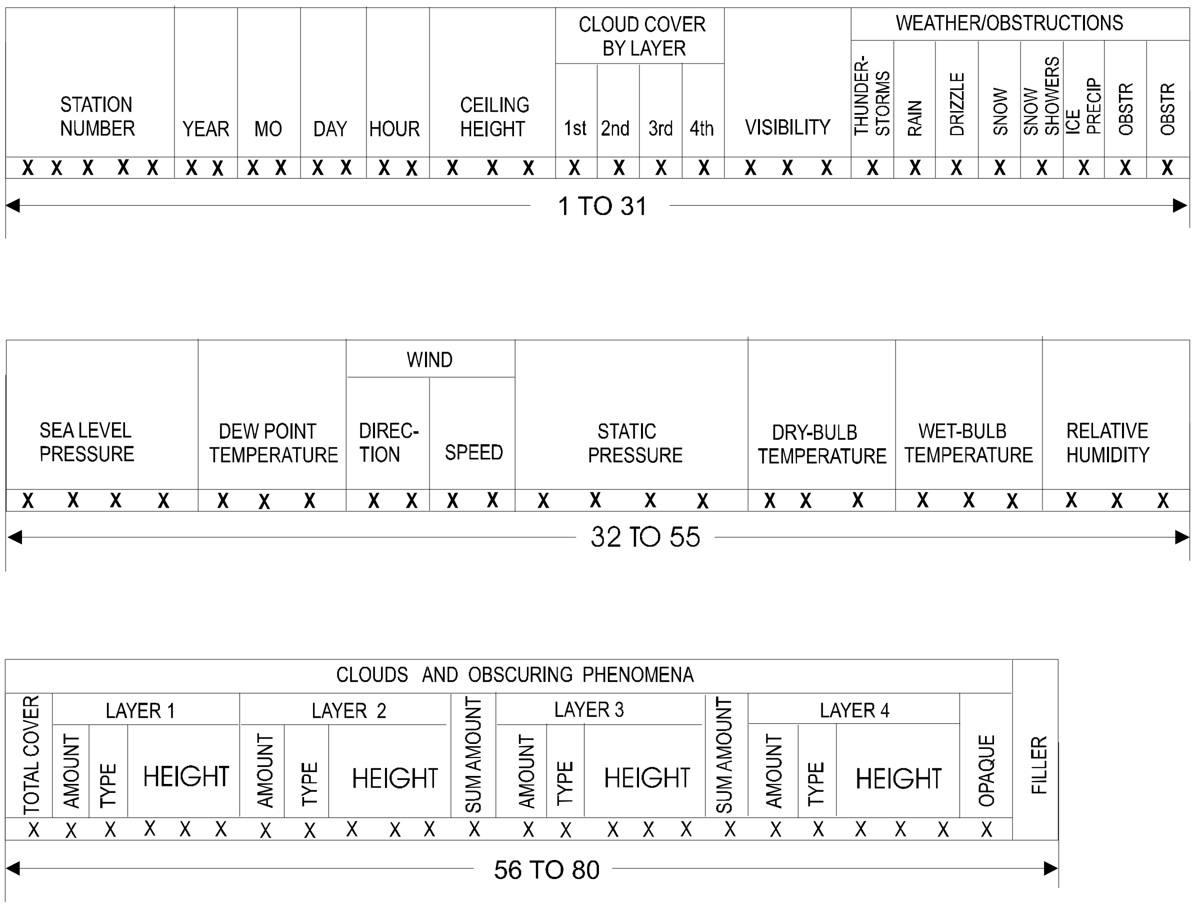


Figure 92 Record format for CD144 weather FILEs

Example Inputs

CD144 input:

```
PACK
Xian      1440/1988
CD144  57036  1988    -8 34.18-108.630-BITNORMAL     9 20.  .025  3000.
.98   .98   .98   .98   .98   .98   .98   .98   .98   .98   .98   .98
-999.
LIST
PACKED  -999  -999     1    12
STAT
END
```

TMY2 input:

```
PACK
Atlanta GA TMY2
TMY2   13874
STAT
END
```