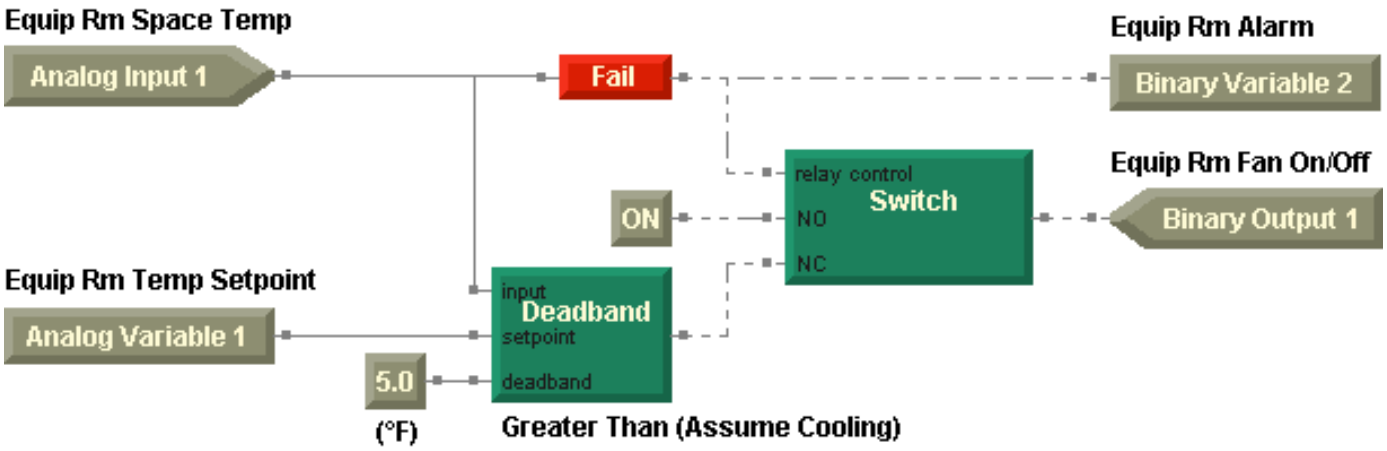
Applications Guide

Tracer Graphical Programming



CNT-APG001-EN

TRANE®

Applications Guide

Tracer Graphical Programming

CNT-APG001-EN
February 2002

Tracer Graphical Programming

Printed in the U.S.A.

CNT-APG001-EN

Special notifications and formats

The following notifications and formats may appear at appropriate locations throughout this manual. Read warnings and cautions carefully.

⚠WARNING

Indicates a potentially hazardous situation, which, if not avoided, could result in death or serious injury.

⚠CAUTION

Indicates a potentially hazardous situation, which, if not avoided, may result in minor or moderate injury. It may also be used to alert against unsafe practices.

CAUTION

Indicates a situation that may result in equipment or property-damage-only accidents.

IMPORTANT

Alerts installer, servicer, or operator to potential actions that could cause the product or system to operate improperly but will not likely result in potential for damage.

Note:

A note may be used to make the reader aware of useful information, to clarify a point, or to describe options or alternatives.

- This symbol precedes a procedure that consists of only a single step.

➤ ***Explanatory information***

Text in this format provides explanations that you can read for further information on a particular subject or concept. The information is not necessary to complete the task.

Contents

Chapter 4 Cooling tower with two-speed fan example 47

Figures

Chapter 3 Modifying the exhaust fan program 29

Chapter 4 Cooling tower with two-speed fan example 47

Chapter 7 Constant-volume AHU example 145

Chapter 8 Constant-volume AHU with warm-up, pre-cool, and communications 189

Programming best practices. 227

Summary-question answers. 235

Tables

Chapter 1

Using the Tracer Graphical Programming editor

Use the Tracer Graphical Programming (TGP) editor to create custom programs for the Tracer MP580/581 programmable controller.

About this book

This book is a tutorial with instructions and examples you can use to learn to write programs in the TGP editor. To get the most from this book, have a Tracer MP580/581 powered up and ready to be programmed. As you read through the chapters, follow the instructions. At the end of each chapter, you will have a configured and programmed Tracer MP580/581 controller for that chapter.

For each chapter, complete the following steps:

1. Read and analyze the sequence of operation.
2. Configure the inputs, outputs, and variables in the Tracer MP580/581 plug-in using the Rover service tool.
3. Follow the instructions and build the programs as you go through the tutorial.
4. Compile and download the programs to your controller.
5. Review the programs and answer the summary questions at the end of each chapter.

Note:
Many of the chapters in this book build on previous chapters, so be sure to complete the chapters in the order presented.

➤ ***Explanatory information***

Text in this format provides explanations that you can read for further information on a particular subject or concept. The information is not necessary to complete the task.

Rover service tool overview

The Rover service tool is the setup and configuration tool for the Tracer MP580/581. It is analogous to PCM Edit for the PCM and UPCM Edit for the UPCM. For more information about the Rover service tool, see the *Rover Operation and Programming* guide (EMTX-SVX01B-EN).

To access the controller through the Rover service tool, you must have the Tracer MP580/581 plug-in. The plug-in is a software file that Rover requires to display information and set up configuration for the controller. The plug-in also contains extensive online Help to help you access and change controller information.

The Tracer MP580/581 plug-in can be run with Rover Version 3 and higher. You cannot run the Tracer MP580/581 plug-in with an earlier version of Rover. Updated versions of the Tracer MP580/581 plug-in may be released independently from the Rover software. Contact your local sales office for the latest versions of the Rover device plug-ins.

About the Tracer MP580/581 controller

The Tracer MP581 programmable controller is a general-purpose input and output device. The controller provides direct digital control of a variety of HVAC equipment.

The Tracer MP580 is factory-installed on Modular Climate Changer, T-Series, and M-Series air handlers. The controller is factory wired to all sensors, actuators, valves, starters, and other items shipped with the air handler.

Use this guide as a tutorial to set up and write graphical programs to control the Tracer MP580/581 programmable controller.

Opening the TGP editor

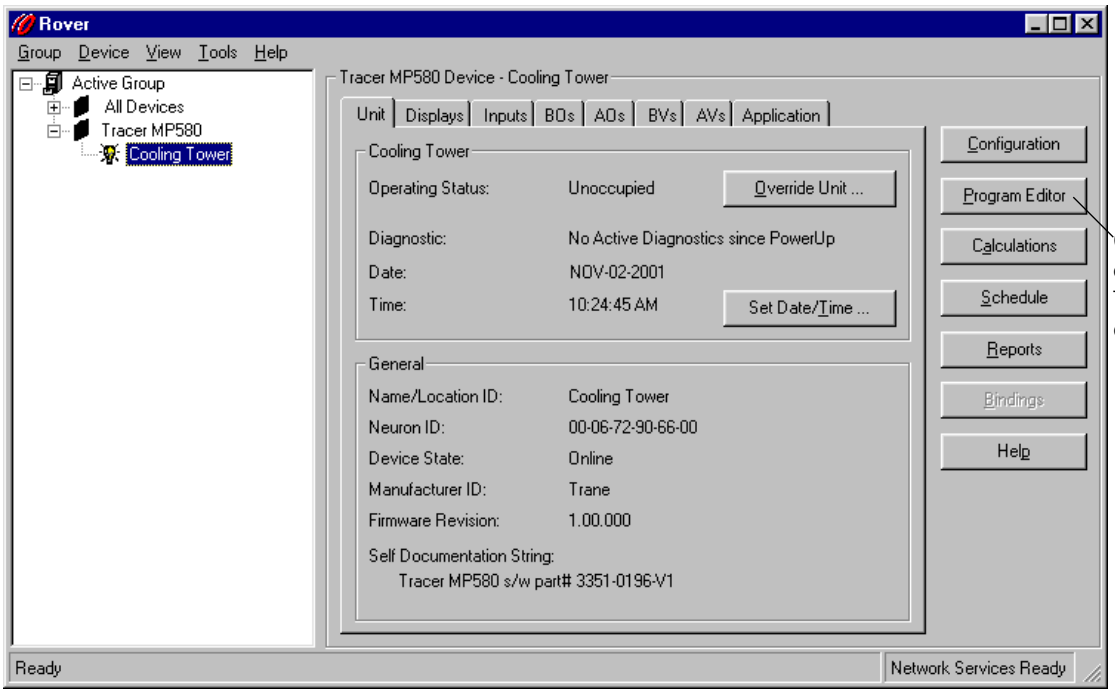
Open the TGP editor through the Rover service tool. First, verify that your laptop PC is connected to an active, LonTalk®, Comm5 communications link. See the *Rover Installation and Operation* guide (EMTX-SVX01B-EN) for information on how to connect to a Comm5 link.

To open the TGP editor:

1. Open the Rover service tool software.
2. In the tree, click the name of the controller you want to configure. The status information for the controller appears (Figure 1 on page 3).

Note:
The controller must be a Tracer MP580/581 to access the TGP editor.

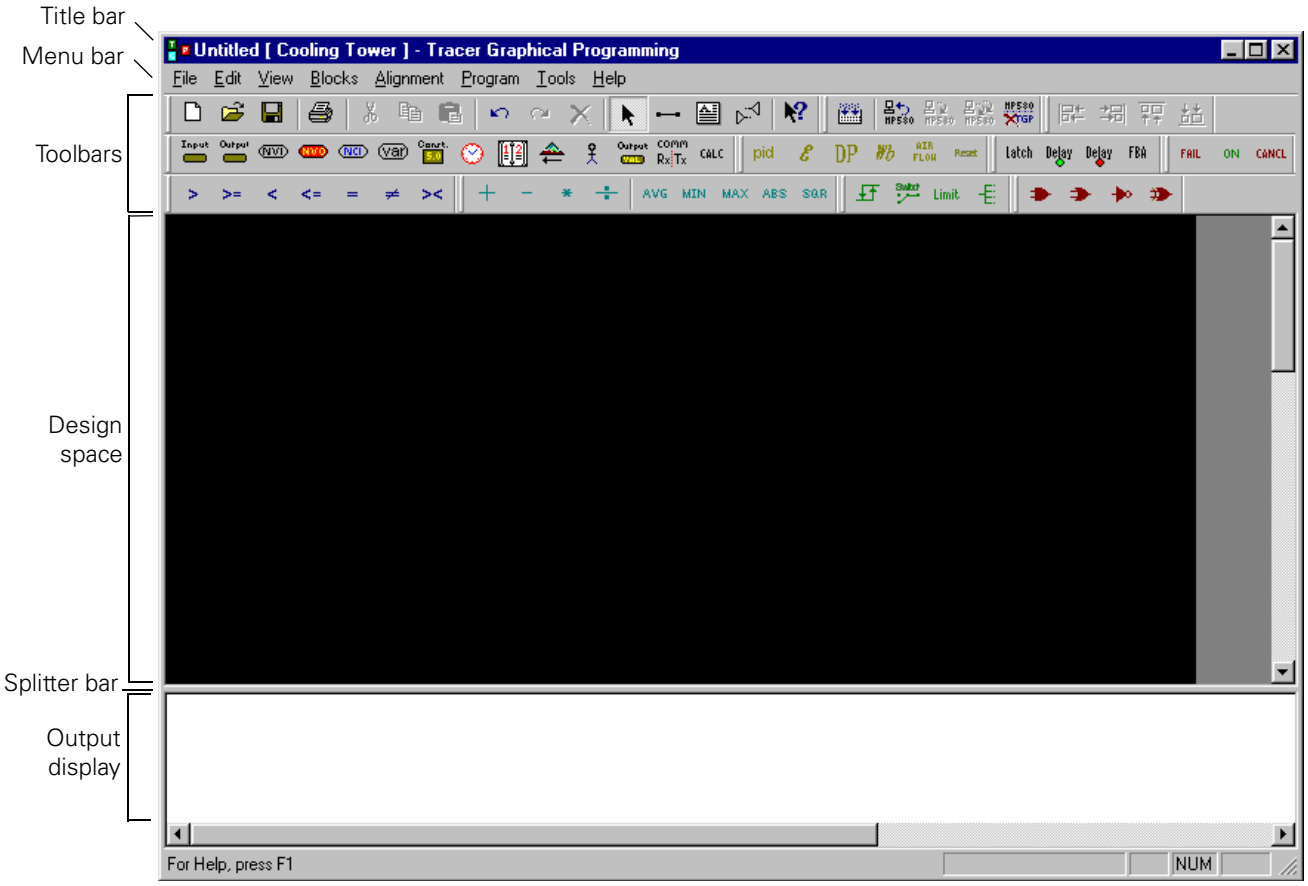
Figure 1: Rover application window



3. Click the Program Editor button. The TGP editor appears with a blank program in the design space (Figure 2 on page 4).

Note:
The TGP editor takes a minute or two to upload configuration information from the controller. The configuration information includes input, output, and variable data as well as any programs currently loaded on the controller.

Figure 2: TGP editor running in Rover service tool



TGP editor

The TGP editor screen includes the design space, output display, blocks, menu bar, toolbars, and shortcut menus.

Design space

The design space is the area in which you can create graphical programs.

Output display

The output display is the area in which results from building a graphical program appear and any programming errors are displayed.

To show or hide the output display:

- From the View menu, choose Output Display.

Note:
If you still cannot see the output display, the splitter bar may be too low. To move it up, click under the design space. Move the splitter bar up.

Blocks

Graphical programming blocks are the fundamental objects used to write a program in the TGP editor. Each block serves a specific purpose. Connecting these blocks in a given arrangement determines how the program behaves. A program consists of a combination of graphical programming blocks connected to perform a logical task.

Figure 3 illustrates the basic structure of a graphical programming block. The connection points on the left side of the block are called input ports. Input ports pass data into the block. Connections on the right side of the block are called output ports. Output ports pass data out of the block.

Figure 3: Block structure

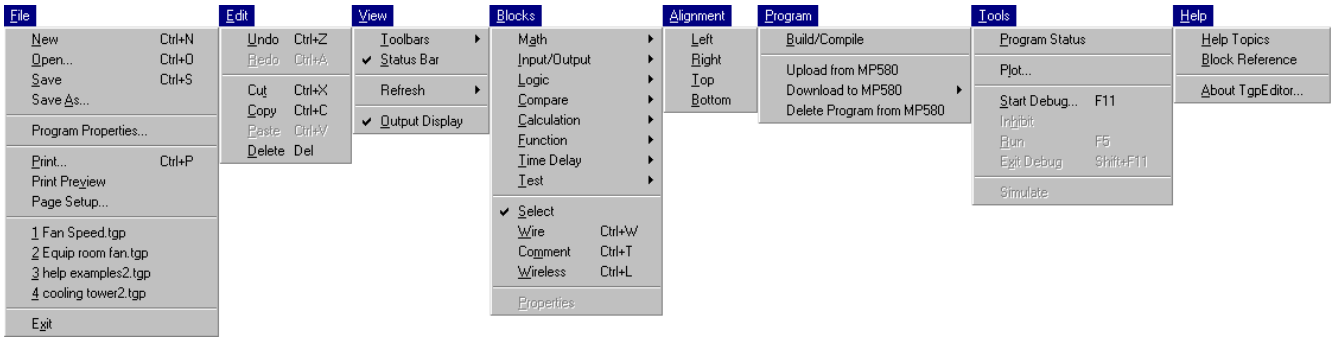
Note:
For further information on specific graphical programming blocks, see the blocks reference in the online Help.

Menu bar

The menu bar at the top of the TGP editor contains drop-down menus for working with TGP programs (Figure 4 on page 6).

- Use the File menu to open new and existing program files as well as to save programs and set program properties.
- Use the Edit menu to undo and redo the last actions made in the editor. This menu also includes options for cutting, copying, pasting, and deleting program elements.
- Use the View menu to set up the editor window.
- The Blocks menu includes options for placing blocks in the design space.
- Use the Alignment menu to align blocks in the design space.
- The Tools menu includes options for working with your program.
- Use the Help menu to access the extensive TGP online Help and to find more information about the TGP editor.

Figure 4: TGP editor menu bar



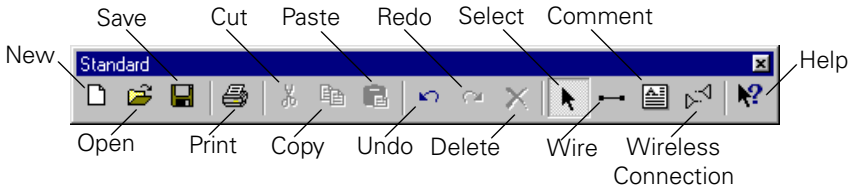
Toolbars

The TGP editor includes toolbars that provide buttons you can click to complete common tasks.

Standard toolbar

Use the Standard toolbar buttons (Figure 5) to open a new or existing program file or to save a file. Click one of the edit buttons to cut, copy, paste, or delete a block or group of blocks. You can undo or redo the last actions completed in the editor, add a wired or wireless connection, or print the program. Click the Help button and then click on a block in the design space to get information about that block.

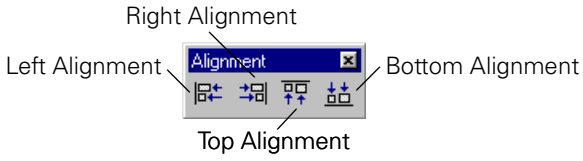
Figure 5: Standard toolbar



Alignment toolbar

Select the blocks in the design space you want to align and click an alignment button (Figure 6) to align the blocks in the design space. The last block selected controls the alignment.

Figure 6: Alignment toolbar

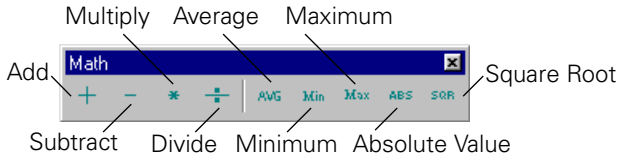


Block toolbars

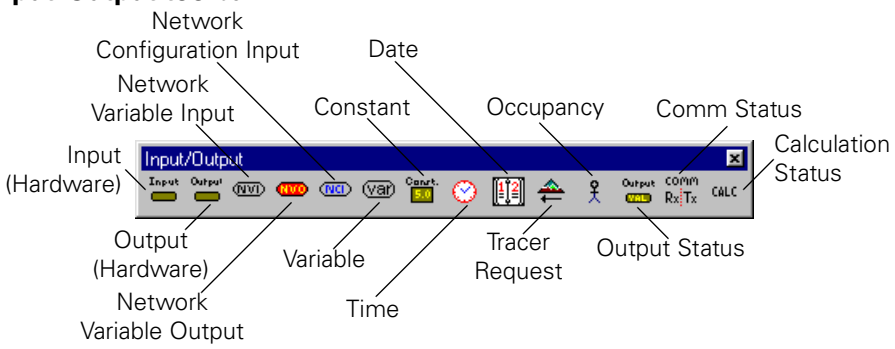
Use each of the following toolbars to add various blocks to your TGP program (Figure 7). For more information about each block, see the blocks reference in the online Help. The blocks are subdivided into eight categories. These categories are displayed in separate toolbars.

Figure 7: Block toolbars

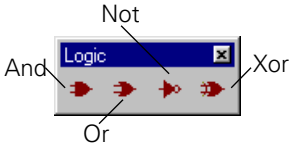
Math toolbar



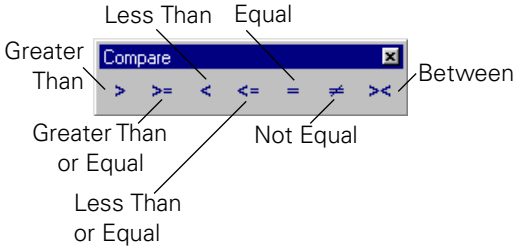
Input/Output toolbar



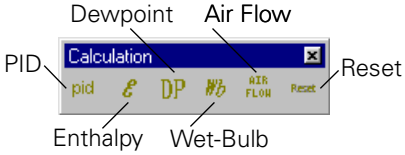
Logic toolbar



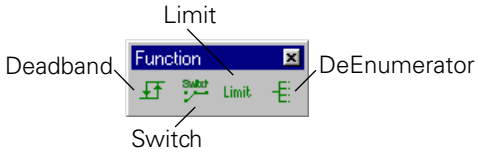
Compare toolbar



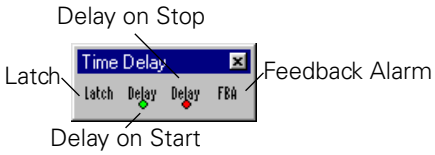
Calculation toolbar



Function toolbar



Time Delay toolbar

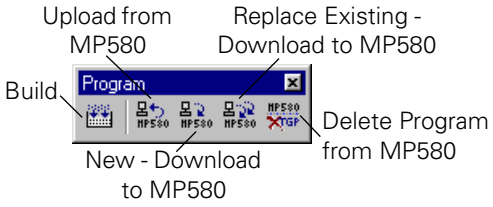


Test toolbar

Program toolbar

Use the Program toolbar to build programs and to control the programs on the Tracer MP580/581 controller (Figure 8).

Figure 8: Program toolbar



Showing or hiding toolbars

You can show or hide each toolbar in the TGP editor window. You can also move each toolbar in the window by clicking on the title bar of the toolbar and dragging it to a new position.

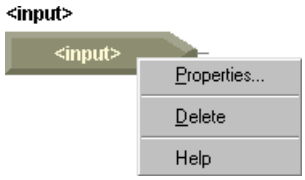
To show or hide a toolbar:

- From the View menu, choose Toolbars. From the Toolbars menu, choose the toolbar you want to view or hide. If a check mark is next to the toolbar name, that toolbar is shown in the window.

Shortcut menus

To view a shortcut menu, use your right mouse button (right-click) to click any block or port in the design space (Figure 9). Shortcut menus contain common commands you can use on the item you clicked. For example, right-click an input (hardware) block in the design space and choose Properties from the shortcut menu to edit the properties of the block.

Figure 9: Shortcut menu



Keyboard shortcuts

Use keyboard shortcuts (Table 1) in the TGP editor to work with program files and blocks.

Table 1: Keyboard shortcuts

Category	Function	Key stroke
File	New	Ctrl+N
	Open	Ctrl+O
	Save	Ctrl+S
	Print	Ctrl+P
Edit	Undo	Ctrl+Z
	Redo	Ctrl+Y
	Cut	Ctrl+X
	Copy	Ctrl+C
	Paste	Ctrl+V
	Delete	Delete
Blocks	Block (pop-up menu)	Ctrl+B
	Comment	Ctrl+T
	Wire	Ctrl+W
	Wireless	Ctrl+L
Program	Build	F7
Tools	Start Debug	F11
	Run	F5
	Exit Debug	Shift+F11

Using online Help

The Rover service tool includes online Help for each screen and dialog box in the plug-in and the TGP editor. The extensive online Help does not appear in this guide. To access help for a tab or dialog box, click the Help button. For information about a screen element, such as a field, option, or command button, click the What's This? help question mark icon and then click a field. For information about a specific block, complete one of the following:

- Click the Help icon on the Standard toolbar and then click a block.
- Right-click on a block and choose Help from the shortcut menu.
- From the Help menu, choose Block Reference and choose the block about which you want more information from the list.

Chapter 2

Writing the exhaust fan program

This chapter introduces the basics of Tracer graphical programming (TGP) by stepping you through the process of constructing a simple program. Graphical programming consists of drawing a picture that represents data and logic. In this chapter, you will construct a program to control an equipment room exhaust fan.

Note:
Many of the chapters in this book build on previous chapters, so be sure to complete the chapters in the order presented. See "About this book" on page 1 for additional instructions.

What you will learn

In this chapter, you will learn a variety of skills, concepts, and definitions.

Skills

You will learn how to:

- Start a new program
- Edit program properties
- Add a block to a program
- Edit block properties
- Move a block
- Align blocks
- Connect blocks
- Add a comment
- Save a program
- Compile a program
- Close a program

Concepts and definitions

You will understand the following concepts and definitions:

- Wire
- Input/Output blocks
- Compare blocks

Blocks

You will learn how to use the following blocks:

- Input (Hardware)
- Constant
- Output (Hardware)
- Greater Than
- Comment

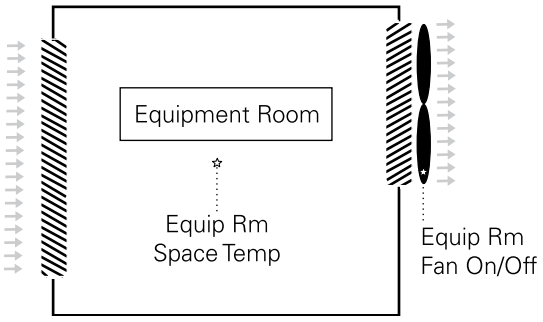
Reviewing the sequence of operation

To begin, review the sequence of operation to determine the necessary data definitions, such as the inputs, outputs, and variables. Then draw a wiring diagram.

In this scenario an equipment room contains machinery that generates a significant amount of heat. As a result, the temperature in the room rises. When the temperature exceeds 85°F, turn on the exhaust fan to draw outside air through the equipment room. When the temperature falls below 85°F, turn off the exhaust fan.

Analyze the scenario and its sequence of operation to determine the necessary inputs, outputs, and variables. Analyzing this scenario might result in Figure 10.

Figure 10: Equipment room exhaust fan data



A temperature sensor in the room supplies the space temperature as an analog value, so you need an analog input to read the temperature. Because the exhaust fan is either on or off, use a binary output to control the fan. The resulting data definition is presented in Table 2 on page 13, and a wiring diagram is presented in Figure 11 on page 14.

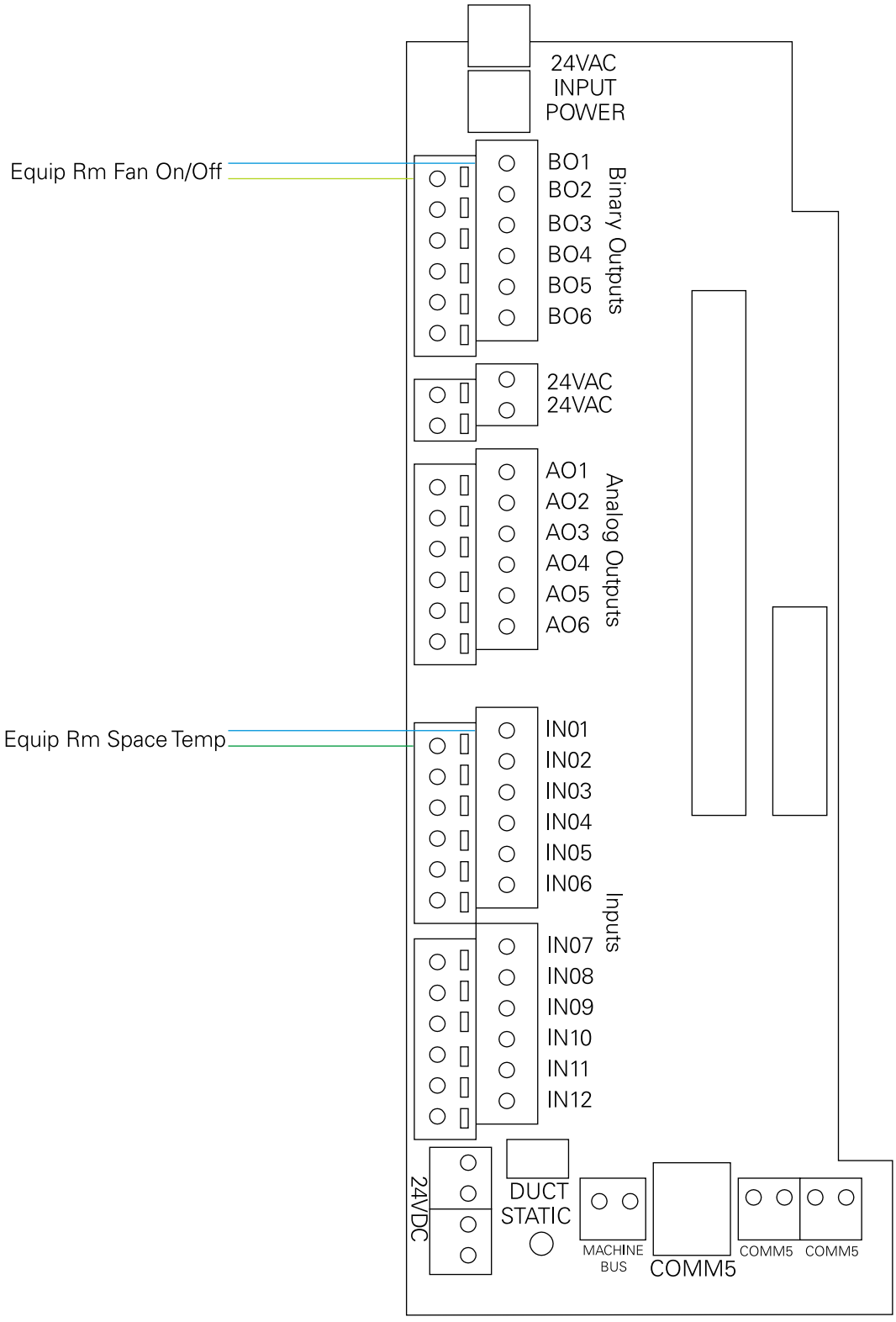
Before you start to write the program, configure the inputs and outputs as shown in Table 2 on page 13. For more information on setting up the wiring, see the *Tracer MP581 Programmable Controller Hardware Installation* guide (CNT-SVN01A-EN). For more information on configuring the

inputs and outputs, see the *Tracer MP580/581 Programmable Controller Programming* guide (CNT-SVP01A-EN).

Table 2: Equipment room exhaust fan data definition

Data	Type	Name	Notes
Input	Analog	Equip Rm Space Temp	Universal input configured as thermistor or RTD (Balco, Platinum)
Output	Binary	Equip Rm Fan On/Off	Set minimum on/off times to 2 minutes.

Figure 11: Equipment room exhaust fan wiring diagram



Opening a new program

To begin, open the TGP editor in the Rover service tool. When the editor opens, a new, blank program appears. If a program is already open, or you want to open a new program, choose New from the File menu. A blank program appears in the design space (Figure 2 on page 4).

Note:
Only one program can be open in the TGP editor at a time.

Editing program properties

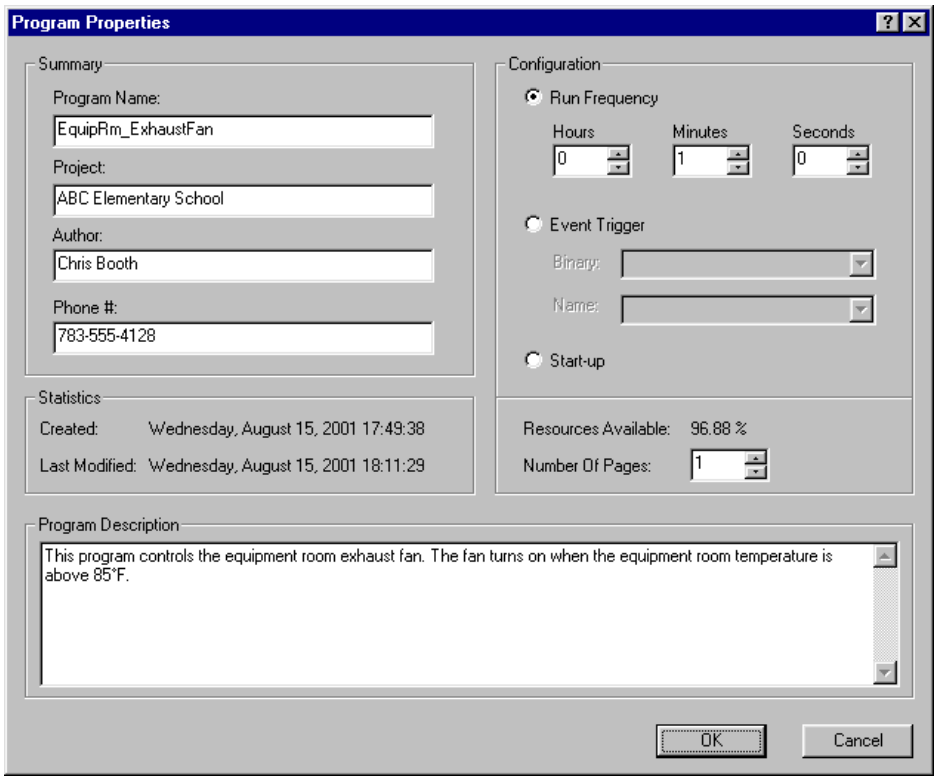
Give the program a name and define some of its basic properties. The properties of a program define how the program behaves. For example, Run Frequency is a property that defines how often the program executes.

Programming tip:
It is good practice to set the program properties before you begin to write a new program, but you can edit the properties at any time by opening the Program Properties dialog box.

To edit program properties:

1. From the File menu, choose Program Properties. The Program Properties dialog box appears (Figure 12 on page 16).

Figure 12: Program Properties dialog box



2. In the Program Name field, type:

   EquipRm_ExhaustFan

   The name may be up to 32 characters in length. Spaces and hyphens are not allowed, and the name cannot begin with a numeral.

3. Type the project name, your name, and your phone number in the appropriate fields.

4. Click the Run Frequency option to run the program at regular intervals.

5. In the Minutes field, type:

   1

   The program runs once per minute.

6. In the Number of Pages field, type:

   1

   The allowable range is one to five pages. This is a very simple program, so start with one page.

7. In the Program Description field, type:

   This program controls the equipment room exhaust fan. The fan turns on when the equipment room temperature is above 85°F.

Note:
To include the degree (°) symbol, press and hold the Alt key while pressing 0, 1, 7, 6 on the keypad. Then release the Alt key, and the symbol appears. (Alt+0176)

8. Click OK.

➤ ***Methods of program execution: Run frequency, event trigger, and start-up***

To select the appropriate method of program execution, ask yourself the following questions:

- Is the program required to run at regular time intervals? If so, click the Run Frequency option. Then specify the time interval in hours, minutes, and seconds.
- Is the program required to run on demand? If so, use the Event Trigger option to provide the program with a reference, which you specify, to a universal input configured as a binary input or to a binary variable. When the binary input or variable changes state, the program runs. (If the trigger direction, off to on or on to off, is a concern, you will have to add logic to another program to check it.)
- Is the program required to run only when the Tracer MP580/581 powers on? In this case, click the Start-up option.

Adding a block

For the program, start with an Input block to access the space temperature hardware input. Input blocks pass data into the program from hardware universal inputs.

To add a block:

Programming tip:
Place input blocks on the left and output blocks on the right so that your program reads from left to right.

1. From the Blocks menu, choose Input/Output. From the Input/Output menu, choose Input (Hardware). The cursor changes to a cross-hair (✛) when over the design space.
2. Click in the design space to place the block. The block appears at the cursor location in the design space (Figure 13 on page 18).

Figure 13: Input (Hardware) block

Editing block properties

In the TGP editor, edit the properties for blocks by using their properties dialog boxes. For example, the Type property for the Input block defines what type of hardware input, analog or binary, the block represents. Edit the block properties to set up the hardware input to read the space temperature from the analog input.

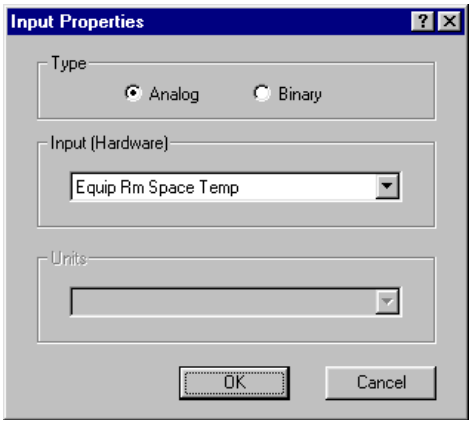
Programming tip:
Set the properties of each block as you place it in the program. This is especially important for blocks that can be set as analog or binary because the connections between blocks are dependent on the data type.

To edit block properties:

1. Click the Input block to select it. The block is outlined in yellow.
2. From the Edit menu, choose Properties. The Input Properties dialog box appears (Figure 14).

Note:
The properties dialog box for each block varies considerably depending on the options available for that block. A few blocks have no editable properties.

Figure 14: Input Properties dialog box



3. Under Type, click the Analog option.
4. In the Input list, click Equip Rm Space Temp.
5. Click OK. The block displays the name of the associated input and the input number (Figure 15 on page 19).

Figure 15: Equip Rm Space Temp input block

Using a Constant block

You want to check if the space temperature is above or below the setpoint of 85°F. Because you know the setpoint and do not want it to change, use a Constant block to represent it.

➤ ***Constants and variables***

When is it appropriate to use a constant? Or a variable? The primary distinction between constants and variables is that constants remain unchanged; whereas, variables change. You might ask, "How do these definitions apply to the Tracer MP580/581?"

Constants may be used in programs as unchanging values. They may be analog, binary, or time/date values. However, they cannot be known outside of the program in which they reside. In other words, a constant cannot be seen or changed unless you edit the program itself.

Variables, on the other hand, may be changed using a variety of methods. Variables can be communicated from the Tracer Summit system and changed using the Rover service tool. Variables may also be calculated in a program, or they may be made adjustable through the operator display. Ask the following questions when you are considering using a constant or a variable:

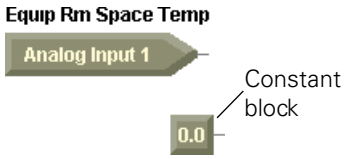
- Does the value change during program execution?
- Must the value be displayed on the operator display?
- Must the value be adjustable by the operator, through the operator display, the Tracer Summit system, or the Comm5 network, including the Rover service tool?
- Must the value be communicated (to Tracer Summit, Rover, or another Comm5 device)?

If you answered no to all of the above questions, then it is appropriate to use a constant. If you answered yes to any of the questions, then use a variable.

To use a Constant block:

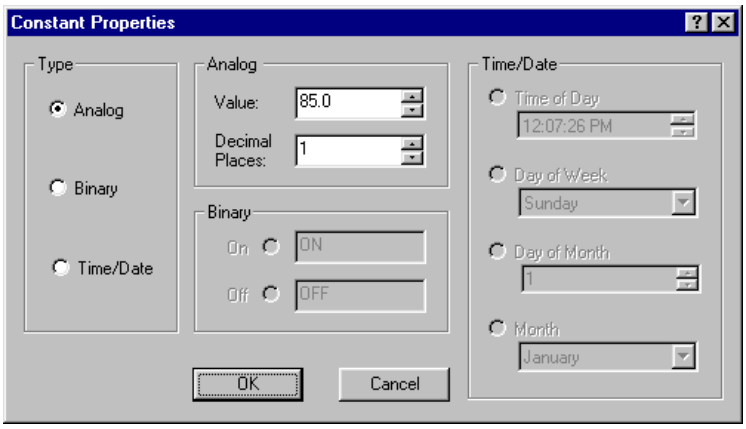
1. From the Blocks menu, choose Input/Output. From the Input/Output menu, choose Constant.
2. Click in the design space to place the Constant block (Figure 16 on page 20).

Figure 16: Constant block in design space



3. Double-click on the Constant block to open the Constant Properties dialog box (Figure 17).

Figure 17: Constant Properties dialog box



4. Under Type, click the Analog option.
5. In the Value field, type:

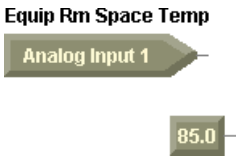
   85.0

6. To set the number of digits right of the decimal, in the Decimal Places field, type:

   1

7. Click OK. The block value is changed to 85.0 (Figure 18).

Figure 18: Constant block set to 85.0



Adding a comment

Use comments to make notes in your program, to annotate blocks, or to describe logic. Place a comment under the 85.0 constant block indicating that it is being used as the equipment room temperature setpoint and that it is in degrees Fahrenheit.

To add a comment:

1. From the Blocks menu, choose Comment and click in the design space to place the Comment block. The Comment dialog box appears.
2. Type:

   Equip Rm Temp Setpoint (°F)
3. Click OK. The comment appears in the design space (Figure 19).

Figure 19: Comment added to describe the Constant block

Arranging blocks

As the saying goes, “A picture is worth a thousand words.” In graphical programming, because the picture tells a story, the way the picture looks becomes very important. Move blocks and use the alignment options to arrange your blocks and design your programs.

Selecting and moving blocks

Practice selecting and moving blocks in the design space.

To select and move blocks:

1. Press and hold the left mouse button on a block and then move the cursor (click and drag) to move the block to a new position in the design space.
2. To select two or more blocks, press the Ctrl key while clicking to select blocks. The selected blocks are outlined in yellow.

Note:
You can also use a rubber-band selection to select more than one block. Click in the design space and drag the cursor so that the white line surrounds the blocks you want to select.

3. Click and drag the selected blocks to a new position in the design space.

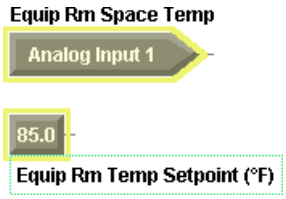
Aligning blocks

Align the Equip Rm Space Temp block, the 85.0 constant block, and the Equip Rm Temp Setpoint (°F) comment blocks.

To align blocks:

1. Select the Equip Rm Space Temp block, the 85.0 constant block, and the Equip Rm Temp Setpoint (°F) comment block. The *last* block selected is the reference block, so all of the selected blocks align according to the position of the *last* block selected. The selected blocks are outlined in yellow.
2. From the Alignment menu, choose Left. The blocks align left (Figure 20).

Figure 20: Blocks aligned left



Adding a Compare block

Compare blocks compare two analog values and output a binary true or false value, providing an analog to binary conversion. In this case, you want to add a Greater Than block to the program.

Use the Greater Than block to compare the space temperature with the setpoint. If the space temperature is greater than the setpoint, the output of the Greater Than block is true. Otherwise, if the space temperature is less than or equal to the setpoint, the output of the Greater Than block is false.

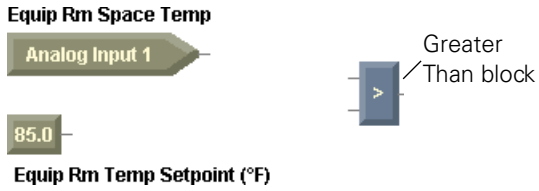
➤ *Inputs to a Compare block*

For some Compare blocks, the input port you choose for a value is important. For example, the Greater Than block compares the top input-port value to determine whether it is greater than the bottom input-port value. Remember this relationship when working with the Greater Than, Greater Than or Equal, Less Than, and Less Than or Equal blocks.

To add a Compare block:

1. From the Blocks menu, choose Compare. From the Compare menu, choose Greater-Than.
2. Click in the design space to place the Greater Than block (Figure 21 on page 23).

Figure 21: Greater Than block in program



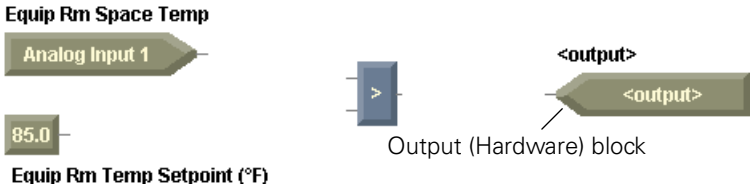
Adding an Output block

To apply the output of the Greater Than block to actually control the fan, add an Output block to the program to pass data to the Equip Rm Fan On/Off binary output. Output blocks pass data out of the program, controlling analog and binary hardware outputs.

To add an Output block:

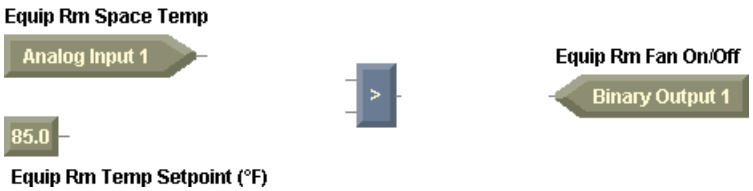
1. From the Blocks menu, choose Input/Output. From the Input/Output menu, choose Output (Hardware).
2. Click in the design space to place the Output (Hardware) block (Figure 22).

Figure 22: Output (Hardware) block in design space



3. Set the properties so that the type is binary and the designated output is Equip Rm Fan On/Off (Figure 23).

Figure 23: Equip Rm Fan On/Off output block



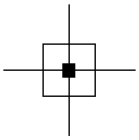
Connecting blocks using wired connections

Finally it is time to connect the blocks.

1. From the Blocks menu, choose Wire. The cursor changes to a cross-hair (✛) in the design space.
2. Hold the cursor over the output port of the Equip Rm Space Temp input block. The cursor changes from a simple cross-hair to a cross-hair with a square (Figure 24). This change in the cursor indicates that a connection may be made.

Note:
You may start wires on any block-input or -output port or on an existing wire. However, you cannot end a connection on a wire if that wire originates on another wire.

Figure 24: Cursor in wire mode over a valid connection



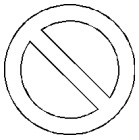
3. Click on the output port of the Equip Rm Space Temp input block. A solid line appears between the connection point and the cursor.

Note:
A solid wire between blocks represents analog data being passed. A dotted wire between blocks represents binary data being passed.

4. Move the cursor to the input port of the Equip Rm Fan On/Off binary-output block. The cursor changes to indicate an invalid connection because you are trying to connect an analog wire to a binary-output input port (Figure 25).

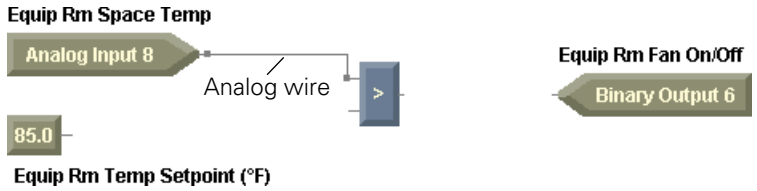
Note:
An analog-output port cannot be connected to a binary-input port, and a binary-output port cannot be connected to an analog-input port.

Figure 25: Cursor in wire mode on an invalid connection



5. Move the cursor and click on the upper input port of the Greater Than block. The wired connection is complete (Figure 26).

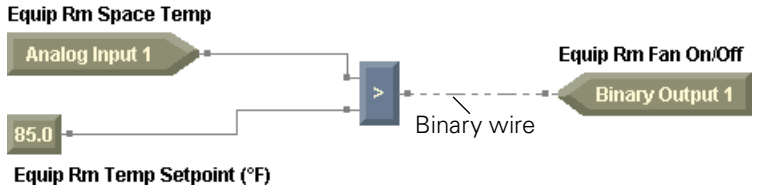
Figure 26: Analog wired connection



6. Click on the output port of the 85.0 constant block. Move the cursor and click again to create a bend in the wire. Click again to create another bend in the wire.
7. Click on the lower input port of the Greater Than block.
8. Click on the output port of the Greater Than block and connect it to the input port of the Equip Rm Fan On/Off binary-output block. The program is complete (Figure 27).

Note:
While drawing a wire, press the right mouse button (right-click) to cancel the wire.

Figure 27: Completed Equip Room Exhaust Fan program



Saving a program

Now that the program is complete, it is a good idea to save it.

To save a program:

1. From the File menu, choose Save. The Save As dialog box appears.

   The Save As location defaults to the c:\tgp\custom folder. This is a great place to save all of your custom TGP programs.
2. In the File name field, type:

   Exhaust Fan
3. Click Save. The graphical program file is saved.

   All files are saved with a file extension of *.tgp, which denotes the file as a Tracer graphical program.

Compiling a program

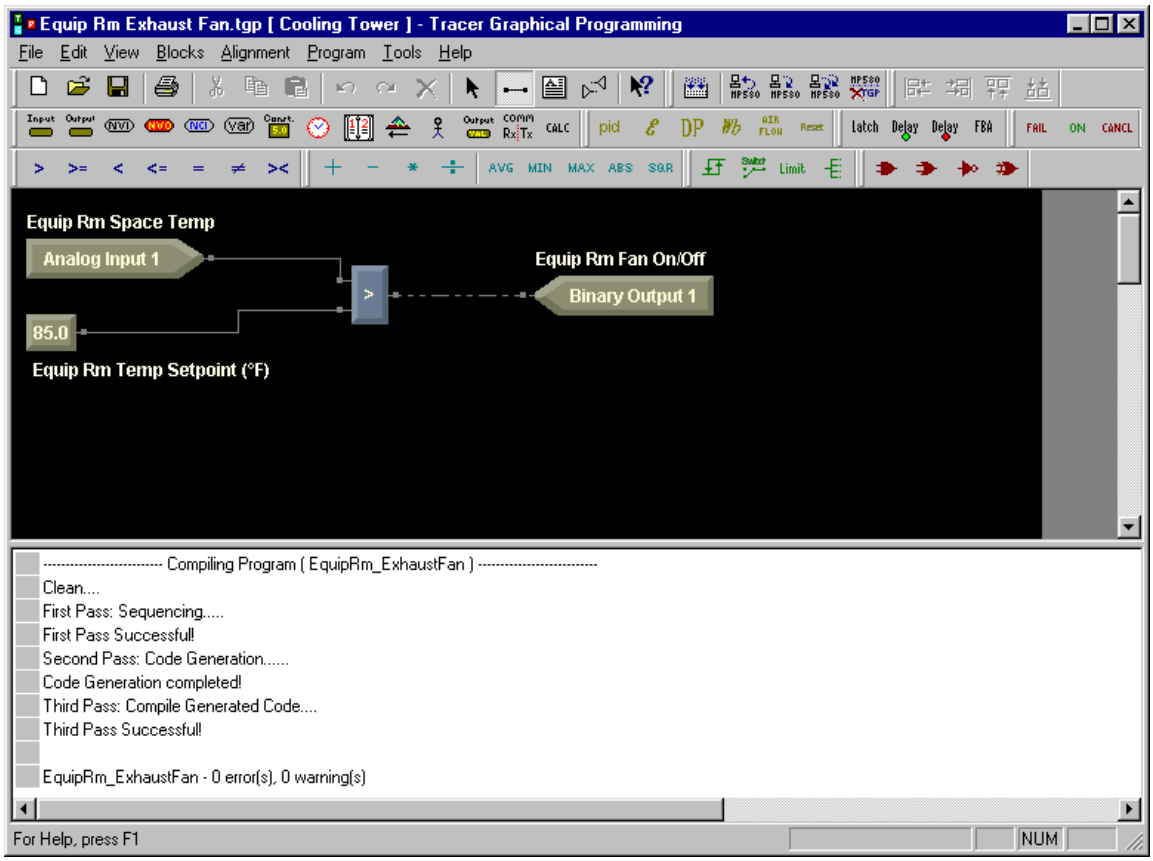
When you compile a program, the TGP editor checks the program for errors and prepares it for download.

To compile a program:

- From the Tools menu, choose Build. The system builds and compiles the program. The output display includes the results, including any applicable errors (Figure 28).

Programming tip:
Always save programs after making changes and compiling. Be sure to use meaningful file names.

Figure 28: Compilation results in output display



Closing a program

Now that your program is saved, close the program and exit the TGP editor.

To close a program:

- From the File menu, choose Exit. The Exhaust Fan.tgp program and the TGP editor closes.

Summary questions

Answer the following questions to review the skills, concepts, definitions, and blocks you learned in this chapter. The answers to these questions are on page 235.

1. When connecting two blocks, the connection points, or ports, must represent the same type of data. What are the two data types?

2. Do all blocks have an associated properties dialog box?

3. At what points may you start a wire? End a wire?

Chapter 3

Modifying the exhaust fan program

This chapter introduces the remaining basic tasks and principles of Tracer graphical programming (TGP) by stepping you through the process of building upon and modifying your first program.

Note:
Many of the chapters in this book build on previous chapters, so be sure to complete the chapters in the order presented. See "About this book" on page 1 for additional instructions.

What you will learn

In this chapter, you will learn a variety of skills, concepts, and definitions.

Skills

You will learn how to:

- Open an existing program
- Delete blocks and wires
- View compilation errors
- Print a program
- Download a program
- Debug a program
- Upload a program

Concepts and definitions

You will understand the following concepts and definitions:

- Function blocks
- Test blocks
- Logic blocks

Blocks

You will learn how to use the following blocks:

- Variable
- Deadband
- Fail

- Switch
- Or

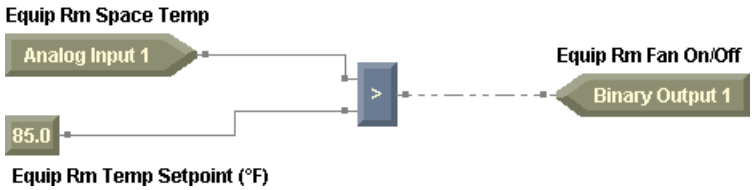
Opening an existing program

Begin by opening your first program, Equip Rm Exhaust Fan. You can have only one program open in the TGP editor at a time. If you try to open another program, the first program is closed.

To open an existing program:

1. From the File menu, choose Open. The Open dialog box appears.
2. Select Exhaust Fan.tgp.
3. Click Open. The graphical program opens in the design space (Figure 29).

Figure 29: Equipment room exhaust fan program



Reviewing the sequence of operation

In this scenario the equipment room remains as described in Chapter 2, “Writing the exhaust fan program.” The room contains machinery that generates a significant amount of heat. As a result, the temperature in the room rises. The following criteria apply to control of the equipment room exhaust fan.

- When the temperature exceeds 85°F, turn the exhaust fan on to draw outside air through the equipment room.
- When the temperature falls below 80°F, turn the exhaust fan off.
- An operator must be able to adjust the limiting setpoint from the operator display.
- If the temperature sensor fails, turn the fan on and indicate an alarm condition.

There are three important differences from the original sequence of operation:

- The sequence incorporates a deadband into control of the fan.
- The temperature setpoint at which the fan turns on is adjustable by the operator at the operator display.
- The sequence calls for an alarm indication when the temperature sensor input fails.

Analyzing this scenario might result in Figure 30. The corresponding data definition is presented in Table 3, and a wiring diagram is presented in Figure 11 on page 14.

Figure 30: Modified equipment room exhaust fan data

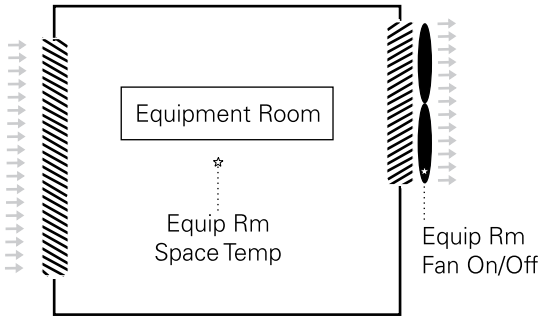


Table 3: Modified equipment room exhaust fan data definition

Data	Type	Name	Notes
Input	Analog	Equip Rm Space Temp	Universal input configured as thermistor or RTD (Balco, Platinum)
Output	Binary	Equip Rm Fan On/Off	Set minimum on/off times to 2 minutes.
Variables	Analog	Equip Rm Temp Setpoint	Analog variable, source set to operator display/service tool
	Binary	Equip Rm Alarm	Binary variable, source set to program

Before you actually start to write the program, verify that these inputs, outputs, and variables are configured in your Tracer MP580/581. For more information, see the *Tracer MP580/581 Programmable Controller Programming* guide (CNT-SVP01A-EN).

Deleting a block

In the Equip Room Exhaust Fan program, you used the Greater Than block to compare the space temperature and the temperature setpoint. Now you do not want to just compare the space temperature to the set-point; you want to incorporate a deadband. A 5-degree deadband will keep the fan from cycling on and off as the space temperature fluctuates around 85ºF. The fan should stay on until the space temperature falls below 80ºF. Replace the Greater Than block with a Deadband block.

To delete a block:

1. Click on the Greater Than block. The block is outlined in yellow.

2. From the Edit menu, choose Delete. The block and any wires connected to the block are removed from the design space.

Note:
To delete a block, you can also select the block and then press the Delete key.

Adding a deadband

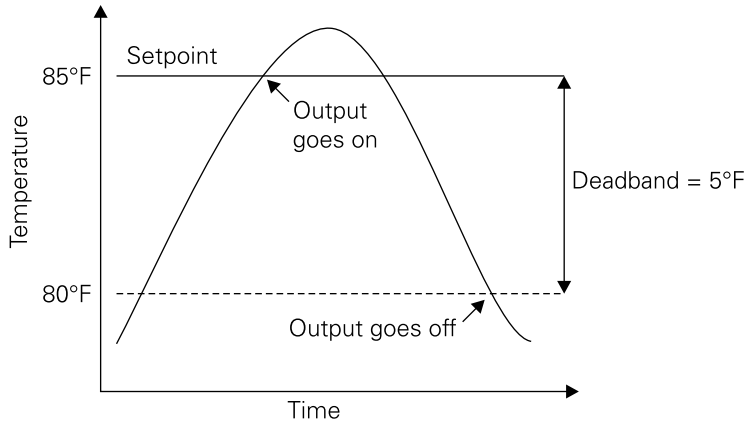
Add a Deadband block to the program. The deadband concept is very useful in HVAC control applications. Its primary purpose is to minimize equipment cycling. For the purposes of graphical programming, consider three possible deadband configurations.

- Greater than (assume cooling)
- Less than (assume heating)
- Centered (assume cooling)

Greater than (assume cooling)

In the greater than (assume cooling) mode, the output of the deadband function turns on when the measured value exceeds the setpoint. The output remains on until the measured value falls below the setpoint point minus the deadband (Figure 31).

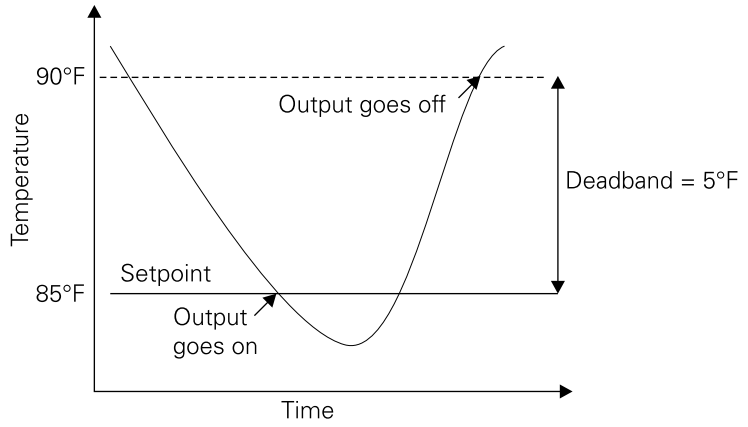
Figure 31: Deadband function with greater than (assume cooling) relationship



Less than (assume heating)

In the less than (assume heating) mode, the output of the deadband function turns on when the measured value falls below the setpoint. The output remains on until the measured value rises above the setpoint plus the deadband (Figure 32 on page 33).

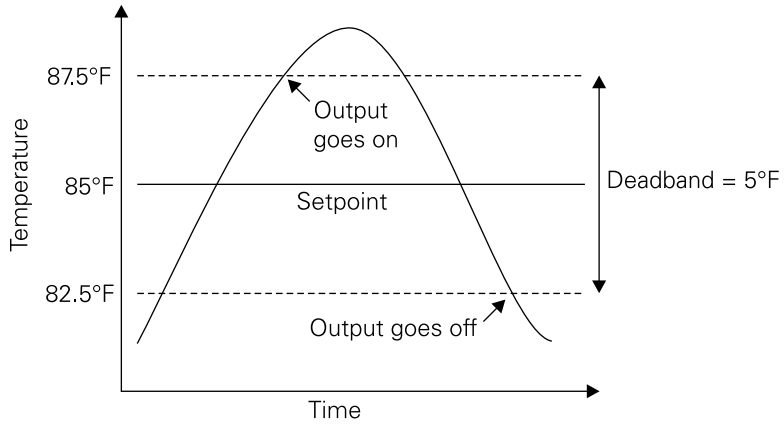
Figure 32: Deadband function with less than (assume heating) relationship



Centered (assume cooling)

In the centered mode, the output of the deadband function turns on when the measured value exceeds the setpoint plus one half of the deadband. The output remains on until the measured value falls below the setpoint minus one half of the deadband (Figure 33).

Figure 33: Deadband function with centered (assume cooling) relationship

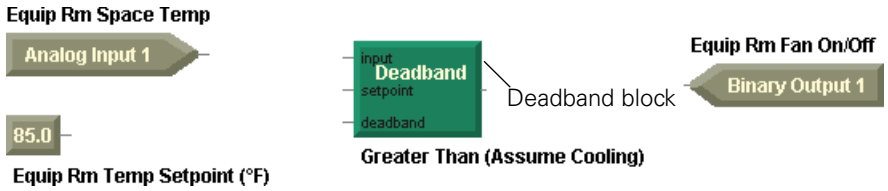


To add a deadband:

1. From the Blocks menu, choose Function. From the Function menu, choose Deadband.
2. Click in the design space to place the Deadband block.
3. Move the Deadband block to the former location of the Greater Than block.
4. Set the Deadband block as greater than (assume cooling) because we want the fan to stay on until the temperature falls below the setpoint minus the deadband (Figure 31 on page 32).

5. Add a comment below the Deadband block indicating its mode for documentation purposes (Figure 34).

Figure 34: Program with Deadband block



Adding a variable

In the first version of the Exhaust Fan program, you used a constant value as the temperature setpoint. In this scenario the operator must be able to adjust the setpoint using the operator display. Use an analog variable to incorporate this functionality. For more information on using constants and variables, see “Constants and variables” on page 19.

➤ *Reading from and writing to a variable using the variable block*

Read and write privileges depend on the selected variable and its source. *Read* refers to the ability of a program to know the value of the variable. On the other hand, *write* refers to the ability of a program to change the value of a variable. Table 4 summarizes the rules regarding reading from and writing to variables in graphical programming. See the *Tracer MP580/581 Programmable Controller Programming* guide (CNT-SVP01A-EN) for more information about variables and control sources.

Table 4: Variable block read or write rules for each control source

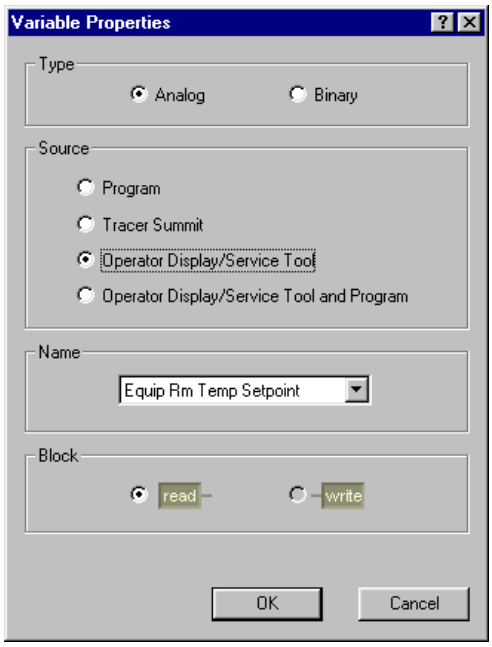
Source	Variable — Read	— Variable Write
Program	X	X
Tracer Summit	X	
Operator display/service tool	X	
Operator display/service tool and program	X	X

To add a variable:

1. Move the 85.0 constant block and the Equip Rm Temp Setpoint comment down to make space for a variable block.
2. From the Blocks menu, choose Input/Output. From the Input/Output menu, choose Variable.

3. Click in the design space to place the Variable block between the Equip Rm Space Temp input block and the 85.0 constant block.
4. Double-click on the block to set the properties. The Variable Properties dialog box appears (Figure 35).

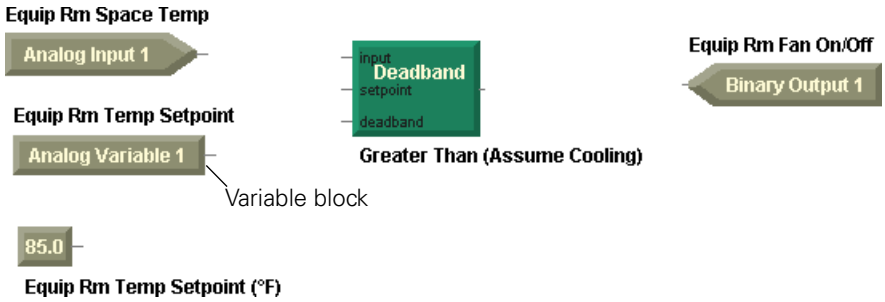
Figure 35: Variable Properties dialog box



5. Under Type, click the Analog option to set the variable as an analog variable.
6. Under Source, click the Operator Display/Service Tool option.
7. In the Name list, click Equip Rm Temp Setpoint.
8. Under Block, click the Read option.

   Because the Equip Rm Temp Setpoint variable source is the operator display/service tool, it is a read-only variable.
9. Click OK. The Variable block is assigned to the Equip Rm Temp Setpoint value (Figure 36).

Figure 36: Equip Rm Temp Setpoint variable block



Using a Constant block for a deadband value

Because we added a variable block to serve as the setpoint, we do not need the constant block for this purpose. However, the sequence of operation does call for a 5.0°F deadband. Use the Constant block for the 5.0 value.

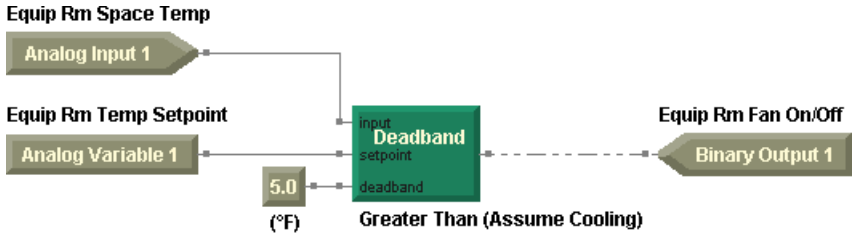
To use a Constant block for a deadband value:

1. Change the 85.0 constant block value to 5.0.
2. Modify the associated comment to read "(°F)."

Connecting blocks to a Deadband block

1. Connect the Equip Rm Space Temp input block to the Input port of the Deadband block.
2. Connect the Equip Rm Temp Setpoint variable block to the Setpoint port of the Deadband block.
3. Connect the 5.0 constant block to the Deadband port of the Deadband block.
4. Connect the output port of the Deadband block to the Equip Rm Fan On/Off output block.(Figure 37).

Figure 37: Program with deadband



Adding alarm indication

The program meets all specifications of the sequence of operation with one exception: If the Equip Rm Space Temp input fails, turn the fan on and indicate an alarm condition.

Adding an alarm variable

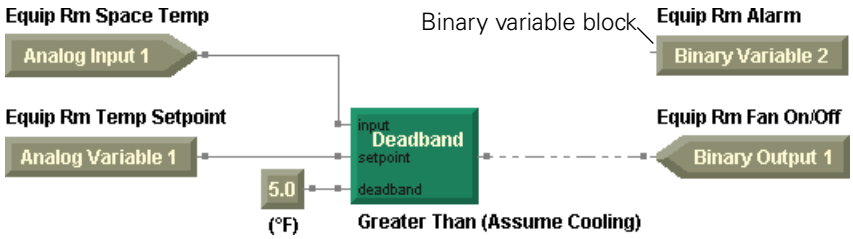
The configured, binary variable, Equip Rm Alarm, serves as the alarm indicator.

Note:
Variables that indicate status, such as the Equip Rm Alarm variable, may be added to custom displays so that they appear on the operator display. See the *Tracer MP580/581 Programmable Controller Programming* guide (CNT-SVP01A-EN) for more information about custom displays.

To add an alarm variable:

1. Add another Variable block to the design space above the Equip Rm Fan On/Off output block.
2. Set the variable to the binary variable, Equip Rm Alarm, which is sourced from the program.
3. Set the block to write to the Equip Rm Alarm variable (Figure 38).

Figure 38: Equip Rm Alarm binary variable in the design space



Implementing a test for sensor failure

Use a Fail block to check the Equip Rm Space Temp input and send information to the Equip Rm Alarm variable. When the temperature input fails, the output of the Fail block is true, and the binary variable, Equip Rm Alarm, indicates the failure by turning on.

For the Fail block to detect properly an input failure, it must be applied to analog inputs configured as thermistor, Balco, Platinum, current, voltage, or resistance. The Fail at End of Range option must also be selected for the input in the Configuration dialog box.

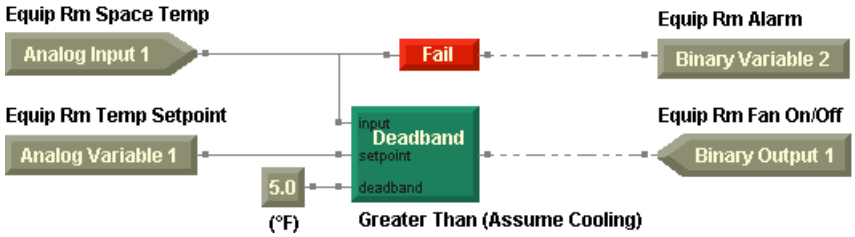
To implement a test for sensor failure:

1. From the Blocks menu, choose Test. From the Test menu, choose Fail.
2. Click in the design space to add a Fail block above the Deadband block.
3. Connect the wire leaving the Equip Rm Space Temp input block to the input port of the Fail block.

Note:
You can draw more than one connection from an output port.

4. Connect the output port of the Fail block to the input port of the Equip Rm Alarm variable block (Figure 39 on page 38).

Figure 39: Program with Fail block



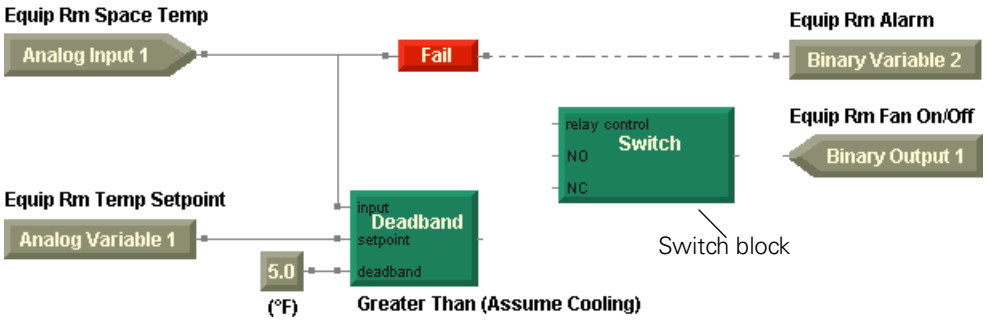
Using a Switch block

The sequence requires that the fan be on when the input is failed. The mechanism, known as the Switch block, provides the ability to choose between two data paths based on a controlling binary value. See the block reference in the online Help for more information about the Switch block.

Adding a Switch block

1. Move the Equip Rm Temp Setpoint variable, the 5.0 constant, and the Deadband blocks down.
2. Move the Equip Rm Alarm variable and Equip Rm Fan On/Off output blocks to the right to make room for the Switch block.
3. Delete the connection between the Deadband block and the Equip Rm Fan On/Off output block.
4. From the Blocks menu, choose Function. From the Function menu, choose Switch.
5. Click in the design space to add a Switch block (Figure 40).
6. Set the Switch block property to binary.

Figure 40: Program with switch block



Connecting the Switch block

The Switch block requires three inputs. The top input, relay control, controls the switch mechanism. When this input is on (true), the switch passes the normally open input to the output. When the relay control input is off (false), the switch passes the normally closed input to the output.

In this case, the output of the Fail block controls the switch. When the Fail block output is false, the fan is controlled using the deadband logic. When the Fail block output is true, the fan is on.

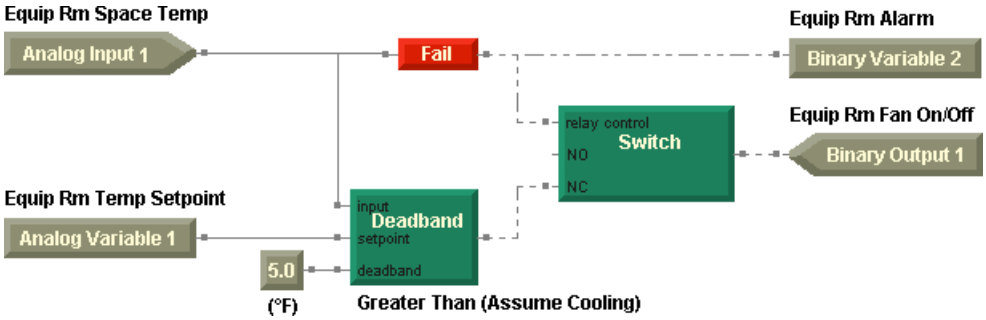
To connect the Switch block:

1. Connect the output port of the Fail block to the Relay Control port of the Switch block.

> **Note:**
> You can draw more than one connection from the Fail output port.

2. Connect the output port of the Deadband block to the NC (normally closed) port of the Switch block.
3. Connect the output port of the Switch block to the Equip Rm Fan On/Off output block (Figure 41).

Figure 41: Program with Switch block connected



4. Compile the program. The compilation results appear in the output display (Figure 42 on page 40) and an error is shown in the design space (Figure 43 on page 40).

   When you compile a program containing any errors, the errors are called out in the output display. Click on the error message in the output display to highlight the error in the program. In this case, the error message indicates that a block input is not connected.

Figure 42: Program compilation results in output display

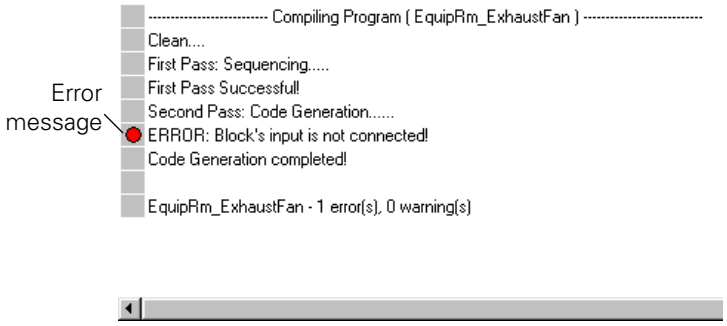
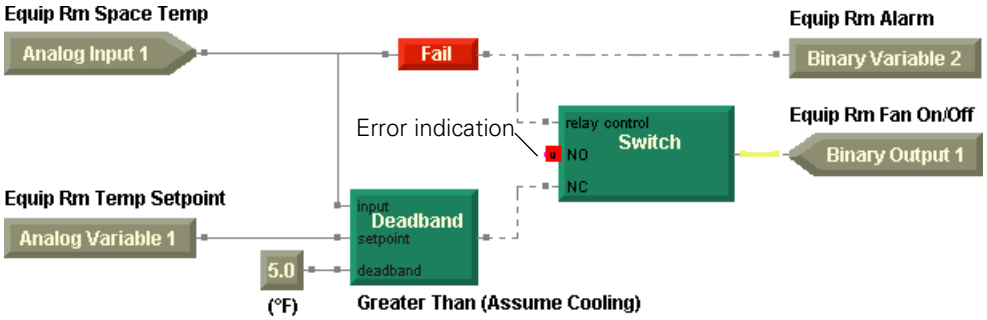


Figure 43: Program with error indication



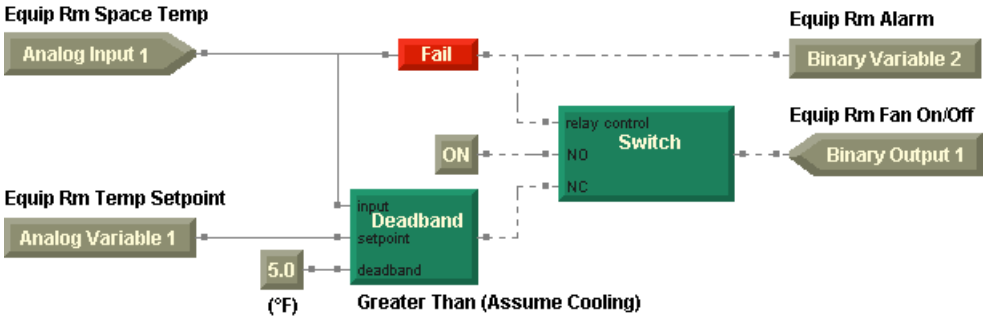
Completing the Switch block connections

What value does the Switch block use when the relay control input is true? The sequence of operation specifies that the fan must be on in the event that a Equip Rm Space Temp input failure exists. Use another constant block to pass the value true (on).

To complete the Switch block connections:

1. Add a Constant block to the design space between the Fail block and the Deadband block.
2. Set the block as a binary constant with a value of ON.
3. Connect the ON constant block to the NO (normally open) port of the Switch block (Figure 44 on page 41).

Figure 44: Program with Switch block connected correctly



4. Compile the program. No errors should appear.
5. From the File menu, choose Save As to save the program under a new name.

The program illustrated in Figure 44 completely fulfills the sequence of operation. The Deadband block compares the Equip Rm Space Temp and the Equip Rm Temp Setpoint and turns the fan on and off as required. If the Equip Rm Space Temp input fails, the Equip Rm Alarm indicates the failure and the fan remains on.

Simplifying the program with an Or block

Logic blocks perform basic logic operations. The Or block may be used as a simpler alternative to the Switch block in the Exhaust Fan program. Use the Or block to check the logic of the Fail block and the Deadband block. If either the Fail block is true or the Deadband block is true, the Equip Rm Fan On/Off output is true (on).

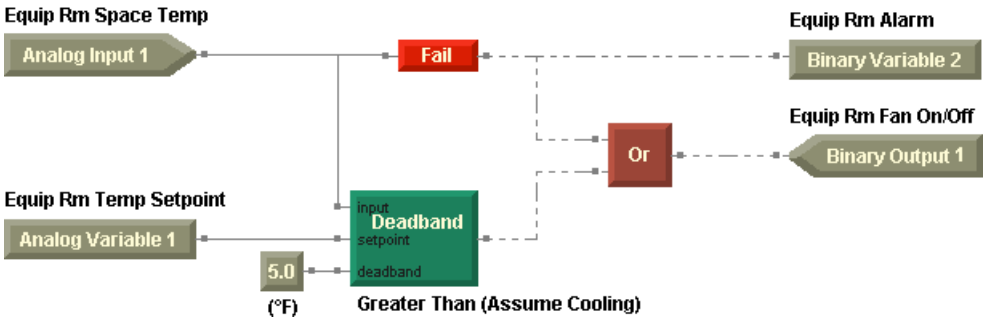
Programming tip:
Keep programs as simple as possible.

To simplify the program with an Or block:

1. Delete the Switch block.
2. Delete the ON constant block.
3. From the Blocks menu, choose Logic. From the Logic menu, choose Or.
4. Click in the design space to place the Or block where the Switch block used to be.
5. Connect the wire leaving the Fail block to the upper input port of the Or block.
6. Connect the output port of the Deadband block to the lower input port of the Or block.

7. Connect the output port of the Or block to the Equip Rm Fan On/Off output block (Figure 45).

Figure 45: Program with Or block



8. Compile the program.
9. Save the program under a new name.

This program meets the requirements of the same sequence of operation as the program pictured in Figure 44 on page 41 but in a simpler manner and with fewer blocks. So there is more than one way to solve the same problem.

Printing a program

If you want a hard copy version of your program, just print it.

To print a program:

1. From the File menu, choose Print. The Print dialog box appears.
2. Select the printer and set the print range and the number of copies.
3. Click the Properties button to select the paper size and orientation.
4. Click OK. The graphical program is printed.

Downloading a program

Now that you have created a graphical program, apply it to a Tracer MP580/581 controller. Remember to compile your program before downloading it to the controller.

To download your program to the Tracer MP580/581 controller:

1. From the Program menu, choose Download to MP580.
2. From the Download to MP580 menu, choose New. The program download begins. Upon completion of the download, a message appears stating that the program downloaded successfully.

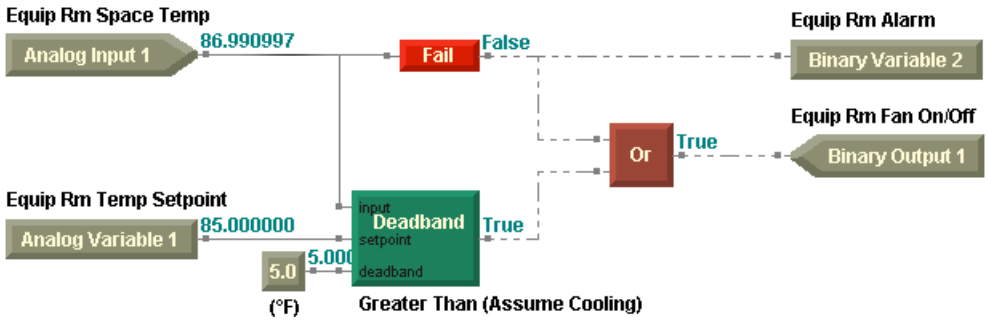
   You would choose Replace Existing if the program already existed in the controller.
3. Click OK.

Debugging a program

To debug a program:

1. Compile the program and download the program you want to debug if you have not done so already.
2. From the Tools menu, choose Start Debug. The Select a Program dialog box appears.
3. Click the program name you want to debug.
4. Click OK. The screen changes color to indicate that debug mode is active.
5. From the Tools menu, choose Inhibit if you want to prevent the execution of the program from controlling outputs and changing variables.
6. From the Tools menu, choose Run. The program runs. Upon completion, the output value of each block is displayed (Figure 46). Use these values to determine if your program logic is correct.

Figure 46: TGP program in debug run



7. From the Tools menu, choose Exit Debug. A message appears stating that the program was exited normally.
8. Click OK.

Uploading a program

To upload a program from the Tracer MP580/581 controller:

1. From the Program menu, choose Upload from MP580. The Select a Program dialog box appears.
2. Click the program name you want to upload.
3. Click OK. The program uploads and appears in the workspace. A message may appear asking you to save changes to the current program, if you already had one open.

Summary questions

Answer the following questions to review the skills, concepts, definitions, and blocks you learned in this chapter. The answers to these questions are on page 235.

1. Can a single Variable block read *and* write the value of an analog or binary variable?

2. What property of an analog or binary variable determines the read or write capability of the associated Variable block?

3. When the relay control input to the Switch block is true, which input value passes to the output of the block?

4. Which input port on the Switch block is always binary, regardless of the Switch block type?

5. For the Fail block to work properly, it must be connected to what types of blocks?

6. What property must be set in the input configuration for the Fail block to work properly?

7. What minor modification to the program pictured in Figure 44 on page 41 would allow you to remove the ON constant block yet still provide an input to the Switch block)?

Chapter 4

Cooling tower with two-speed fan example

This chapter and the chapters that follow build upon your fundamental skills and knowledge of Tracer graphical programming (TGP) by stepping you through the writing of graphical programs of increasing complexity. In this chapter, you will create a program to control a cooling tower with a two-speed fan.

Note:
Many of the chapters in this book build on previous chapters, so be sure to complete the chapters in the order presented. See "About this book" on page 1 for additional instructions.

What you will learn

In this chapter, you will learn a variety of skills, concepts, and definitions.

Skills

You will learn how to better interpret a sequence of operation and transfer it to a working graphical program.

Concepts and definitions

You will understand the following concepts and definitions:

- Time Delay blocks
- Expandable blocks
- Math blocks

Blocks

You will learn how to use the following blocks:

- Delay on Start
- Add
- Less Than or Equal
- Latch

Reviewing the sequence of operation

In this scenario a cooling tower with a two-speed fan delivers condenser water to a small chiller plant. The following specifications apply to control of the cooling tower.

Condenser water pump

When condenser water is requested by the chiller plant, command the condenser water pump to start. If condenser water flow fails to be confirmed within 30 seconds, command the pump to stop, indicate a pump failure at the operator display, and turn on the alarm output. A user must be able to reset the alarm at the operator display.

Note:
The instructions for writing the condenser water pump module are not included in this chapter. If you are working through the book chapter by chapter, you will write this module in Chapter 5. Otherwise, see "Writing the condenser water pump module" on page 94.

Cooling tower fan

If condenser water flow is established and if the condenser water temperature rises 2.5°F above the setpoint, start the fan at low speed. Switch the fan to high speed if the condenser water temperature rises to 5.0°F above the setpoint. Turn off the fan when the condenser supply water temperature falls below the setpoint minus 2.5°F. A minimum 30-second delay is required between starting the fan at low speed and switching to high speed.

Sump heater

Turn on the sump heater if the sump temperature falls below 40°F (this value is adjustable at the operator display). If the outdoor air temperature falls below 32°F, turn on the sump heater continuously. If the sump temperature remains below 36°F (this value is adjustable at the operator display) for 15 minutes, or if the sump temperature falls below 32°F, indicate an alarm at the operator display and turn on the alarm output.

Alarms

In addition to the alarm requirements mentioned in the previous sequence of operation components, indicate an alarm at the operator display and turn on the alarm output when any temperature sensor fails. The user must be able to reset the alarms at the operator display.

Analysis of this scenario results in Figure 47 on page 49. The corresponding data definition is presented in Table 5 on page 49, and a wiring diagram is presented in Figure 48 on page 51.

Figure 47: Cooling tower with two-speed fan drive data

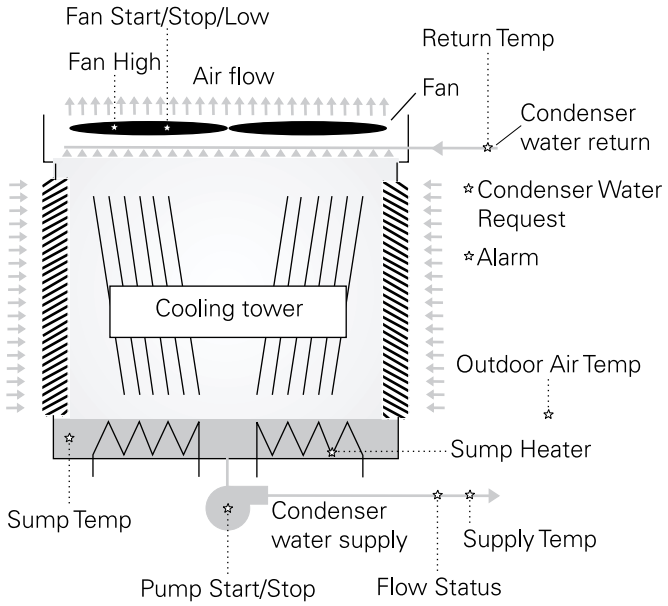


Table 5: Cooling tower with two-speed fan drive data definition

Data	Type	Name	Notes
Inputs	Analog	Supply Temp	Universal input configured as thermistor or RTD (Balco, Platinum)
		Return Temp	Universal input configured as thermistor or RTD (Balco, Platinum)
		Sump Temp	Universal input configured as thermistor or RTD (Balco, Platinum)
		Outdoor Air Temp	Universal input configured as thermistor or RTD (Balco, Platinum)
	Binary	Flow Status	
		Condenser Water Request	
Outputs	Binary	Fan Start/Stop/Low	Apply sufficient minimum on/off timers to prevent excessive fan cycling.
		Fan High	Apply sufficient minimum on/off timers to prevent excessive fan cycling.
		Sump Heater	Apply sufficient minimum on/off timers to prevent excessive heater cycling.
		Pump Start/Stop	Apply sufficient minimum on/off timers to prevent excessive pump cycling.
		Alarm	

* The binary variable, Alarm Reset, is sourced from both the operator display/service tool and the program. This allows the user to turn on the Alarm Reset variable, and once the alarm is reset, the program turns the Alarm Reset variable off.

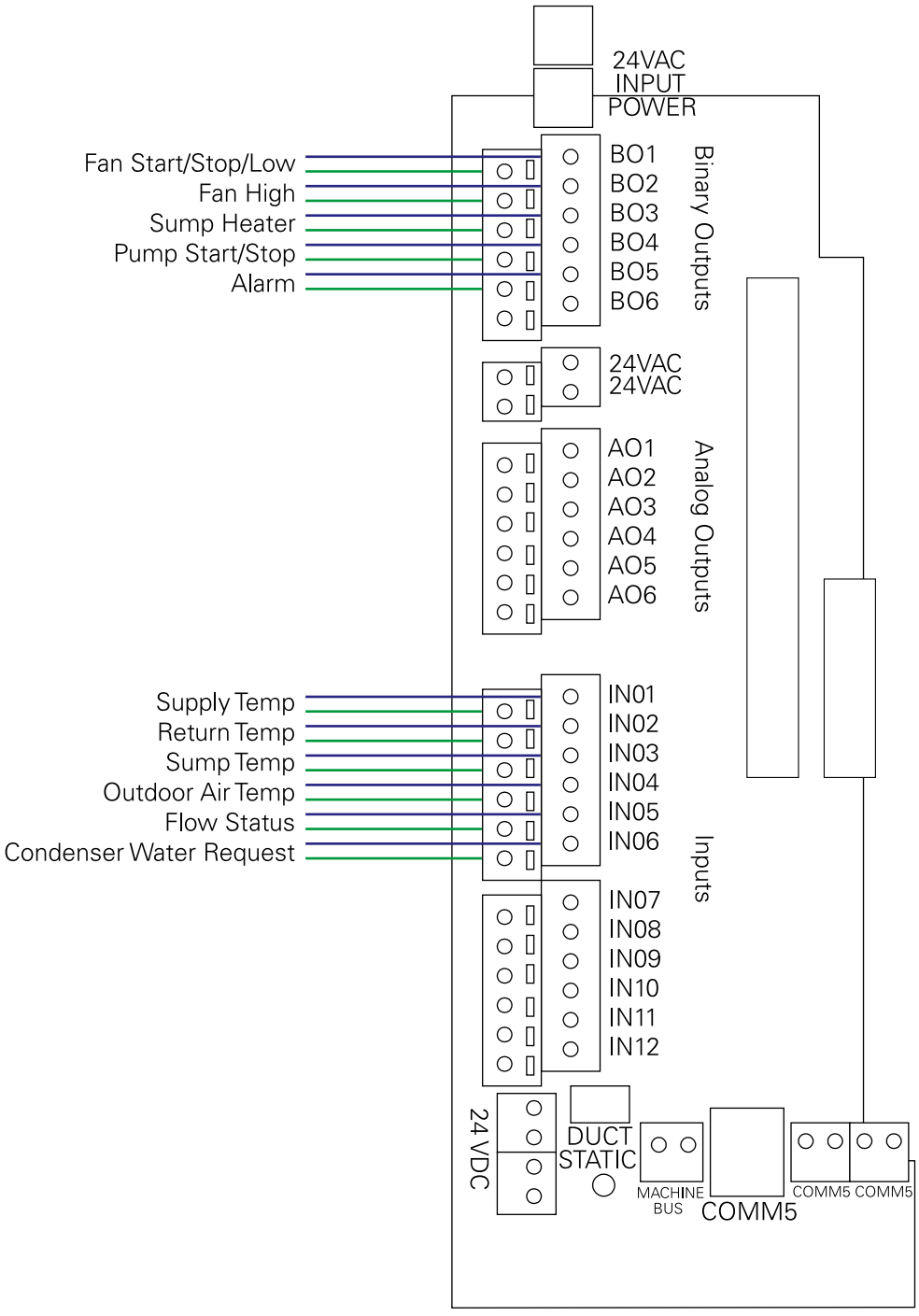
Table 5: Cooling tower with two-speed fan drive data definition (continued)

Data	Type	Name	Notes
Variables	Analog	Supply Temp Setpoint	Source set to operator display/service tool
		Sump Heater Setpoint	Source set to operator display/service tool
		Sump Alarm Setpoint	Source set to operator display/service tool
	Binary	Pump Fail	Source set to program
		Alarm Reset	Source set to operator display/service tool and program*

* The binary variable, Alarm Reset, is sourced from both the operator display/service tool and the program. This allows the user to turn on the Alarm Reset variable, and once the alarm is reset, the program turns the Alarm Reset variable off.

Before you write the program, configure these inputs, outputs, and variables in your Tracer MP580/581 controller. Then open the TGP editor.

Figure 48: Cooling tower with two-speed fan drive wiring diagram



Determining a programming approach

Dividing the sequence of operation into logical program modules makes programming easier. In a single program, you can write several program modules. This sequence can be subdivided into the following four modules:

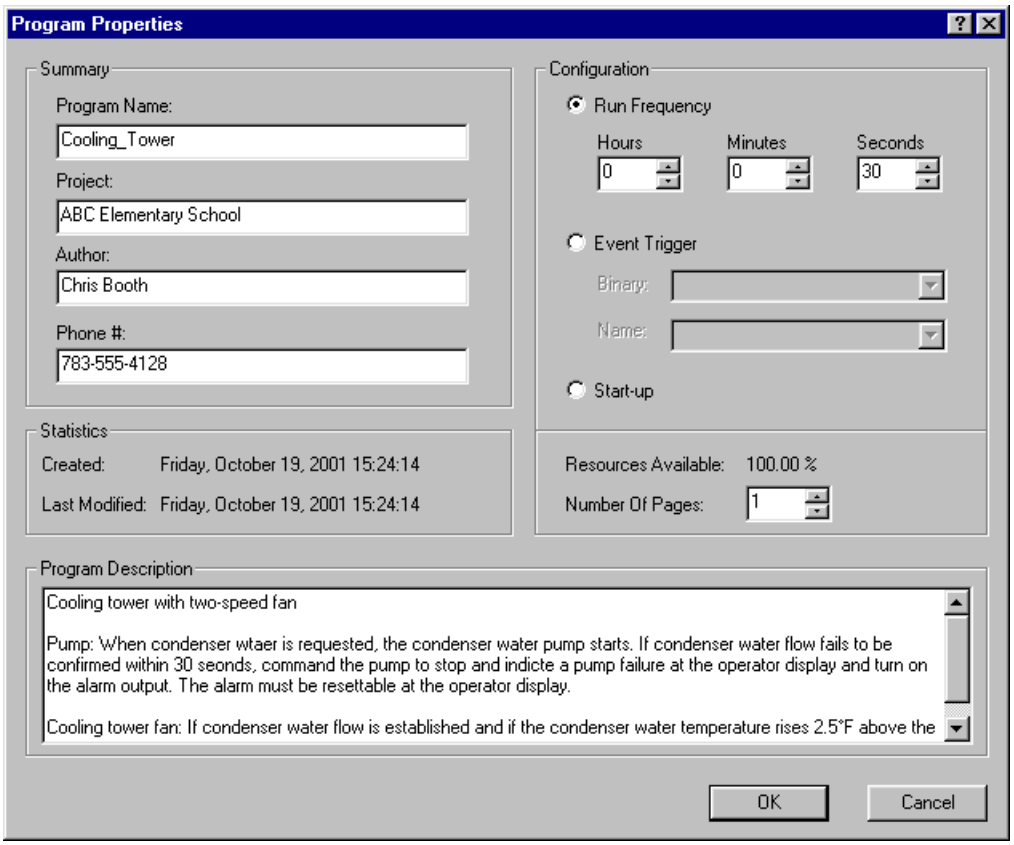
- Alarms
- Sump heater
- Cooling tower fan
- Condenser water pump (Assume for this chapter that the chiller does this.)

Programming tip:
When possible, subdivide graphical programs according to the sequence of operation. Doing so makes the program easier to read and understand.

Setting the program properties

Set the program properties as shown in Figure 49 on page 53. Set the program run frequency to 30 seconds, which is appropriate for this cooling tower application.

Figure 49: Cooling tower with two-speed fan drive program properties



Writing the alarms module

Start by compiling the alarms requirements from the various parts of the sequence of operation, including the following:

- If the condenser water pump fails, indicate a general alarm. A user must be able to reset the alarm at the operator display.
- If the sump temperature remains below 36°F (this value is adjustable at the operator display) for 15 minutes, or if the sump temperature falls below 32°F, indicate an alarm at the operator display and turn on the Alarm output.
- Indicate an alarm when any temperature sensor fails. These temperatures include Sump Temp, Supply Temp, and the Outdoor Air Temp inputs.
- Alarms must be resettable at the operator display.

Note the following important points and questions about the alarm control requirements:

- The pump failure is determined by another part of the program. The alarms module merely uses the result.

- Alarms related to monitoring the sump temperature require time-based control. How is time-based control implemented in graphical programming?
- All temperature sensors must be monitored for failure.
- How is an alarm-reset function incorporated in a program?

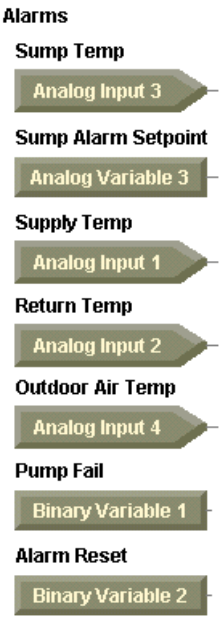
Adding the input blocks

Begin by adding blocks to represent the inputs to this part of the program.

To add the input blocks:

1. Place four Input (Hardware) blocks in the design space and assign the following analog inputs to them:
   - Sump Temp
   - Supply Temp
   - Return Temp
   - Outdoor Air Temp
2. Place three Variable blocks in the design space and assign the following variables to them (Figure 50):
   - Sump Alarm Setpoint (analog, sourced from the operator display/service tool)
   - Pump Fail (binary, sourced from the program)
   - Alarm Reset (binary, sourced from the operator display/service tool and the program)

Figure 50: Input blocks for the alarms module



Adding the output block

- Place an Output (Hardware) block in the design space and assign the binary output Alarm to it (Figure 51).

Figure 51: Alarms module output

Monitoring the sump temperature

First, focus on the requirement dealing with monitoring the sump temperature. Be sure to pay attention to the key words shown in *italics*. They help indicate which blocks to use.

> If the sump temperature *remains below* 36°F (this value is adjustable at the operator display) *for 15 minutes, or* if the sump temperature *falls below* 32°F, indicate an alarm at the operator display and *turn on* the alarm output.

Comparing the sump temperature with the sump alarm setpoint and the freezing point

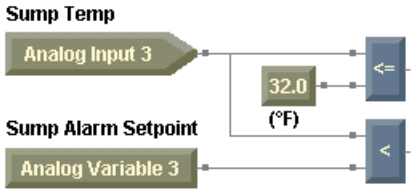
Apply a Less Than block to determine if the Sump Temperature is less than the Sump Alarm Setpoint. And apply a Less Than or Equal block to compare the Sump Temperature to the freezing point, 32°F. Use a Constant block to represent the freezing point.

To compare the sump temperature with the sump alarm setpoint and the freezing point:

1. Add a Constant block to the design space and assign the value 32.0 to it to represent the freezing point.
2. Add a Less Than block and a Less Than or Equal block to the design space.

3. Connect the Sump Temp input block and the 32.0 constant block to the Less Than or Equal block.
4. Connect the Sump Temp input block and the Sump Alarm Setpoint variable block to the Less Than block (Figure 52).

Figure 52: Comparing the sump temperature to the alarm setpoint and the freezing point



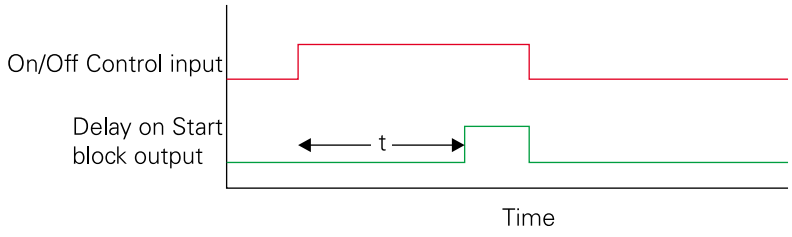
Timing the sump temperature alarm

From the sequence of operation, you know that the sump temperature must be less than the Sump Alarm Setpoint for 15 minutes before indicating an alarm. For applications involving time-based control, take advantage of the Time Delay blocks.

Time Delay blocks provide time-based control functions using program cycles to measure time. These blocks require a program run frequency to operate correctly. For further definitions of the time delay blocks behavior, see the online Help.

For example, you could use the Delay on Start block to delay a binary on signal for a specified amount of time based on a binary-input on signal. The delay timer starts when a change in state (from off to on) occurs in the controlling binary-input signal. After the delay time passes, the output signal follows the input signal. In the Delay on Start block properties dialog box, set the delay time units as seconds, minutes, or hours and use another Constant or Variable block to represent the amount of time. Figure 53 illustrates how the Delay on Start block works. The letter “t” represents the delay time.

Figure 53: Delay on start timing diagram



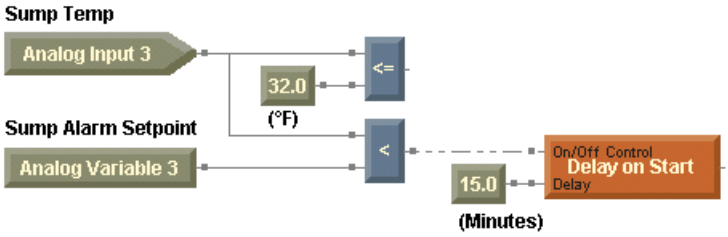
Add a Delay on Start block and connect its On/Off Control port to the output of the Less Than block. Use a Constant block to supply the time interval value. Remember to use the Delay on Start block properties dialog box to set the delay time interval units to minutes.

In this application, when the output of the Less Than block is true, the delay timer begins counting down. If the output of the Less Than block remains true throughout the delay time, when the delay timer expires, the output of the Delay on Start block becomes true as well.

To time the sump temperature alarm:

1. From the Blocks menu, choose Time Delay. From the Time Delay menu, choose Delay on Start.
2. Click in the design space to place the Delay on Start block.
3. Set the Delay on Start block delay time units to minutes.
4. Add a Constant block to the design space and assign the value 15.0 to it to represent the delay time.
5. Connect the Less Than block to the On/Off Control port of the Delay on Start block.
6. Connect the 15.0 constant block to the Delay port of the Delay on Start block (Figure 54).

Figure 54: Implementing the Delay on Start block



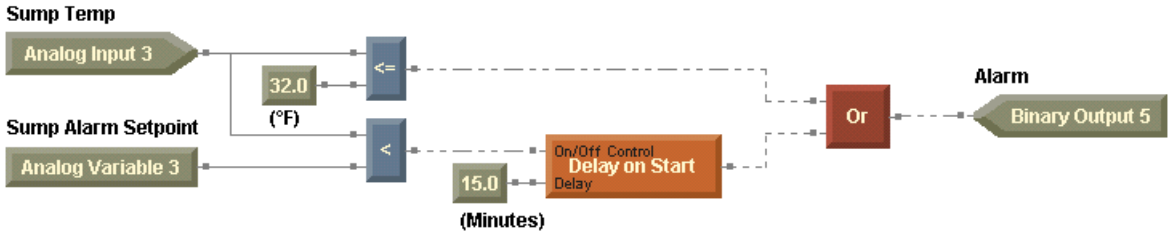
Controlling the sump temperature alarm

The requirement calls for the program to turn on the alarm output when the “sump temperature remains below 36°F (this value is adjustable at the operator display) for 15 minutes, or if the sump temperature falls below 32°F.” Use an Or block to complete the picture, connecting the Delay on Start and the Less Than or Equal blocks to the Or block. Then connect the Or block to the Alarm output block. The result is pictured in Figure 55 on page 58.

To control the sump temperature alarm:

1. Place an Or block in the design space.
2. Connect the Delay on Start block and the Less Than or Equal block to the Or block.
3. Connect the Or block to the Alarm output block (Figure 55 on page 58).

Figure 55: Sump temperature alarm logic



Indicating an alarm for any temperature sensor failure

You want to indicate an alarm when any temperature sensor fails. These sensor-input temperatures include Sump Temp, Supply Temp, Return Temp, and Outdoor Air Temp. Add a Fail block to monitor temperature sensors for failure and make the appropriate connections. Because there are four temperatures, use one Fail block and expand it to accommodate all four temperature inputs.

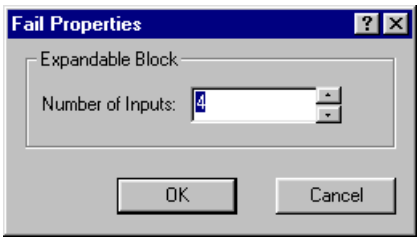
➤ ***Expandable blocks***

Several blocks expand to accommodate a number of inputs. Each of the properties dialog boxes for these blocks allows you to specify the number of inputs. The minimum number of inputs depends on the block. All expandable blocks expand to a maximum of eight input ports.

To indicate an alarm for any temperature sensor failure:

1. Place a Fail block in the design space.
2. Double-click the Fail block. The Fail Properties dialog box appears (Figure 56).

Figure 56: Fail Properties dialog box

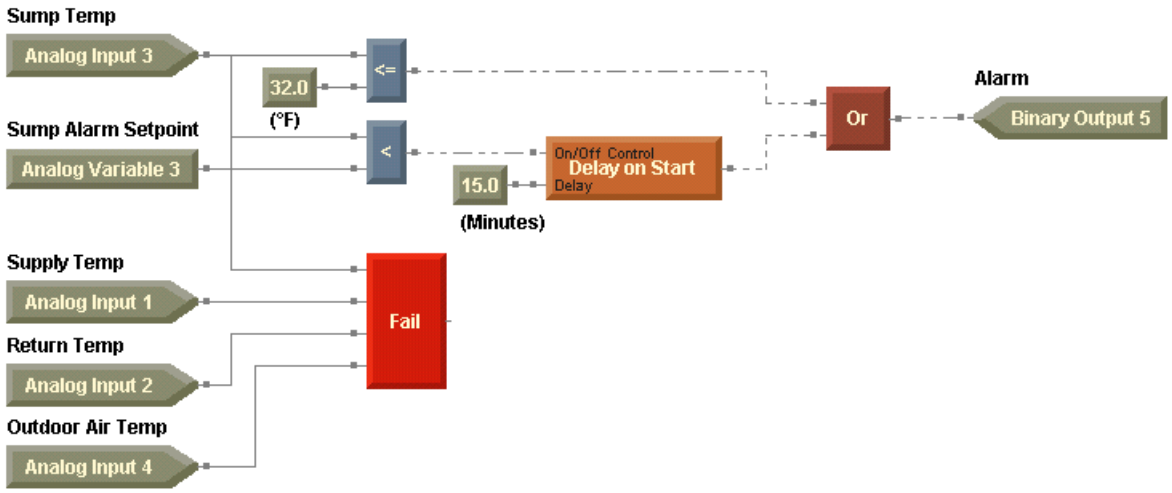


3. In the Number of Inputs field, type:

   4

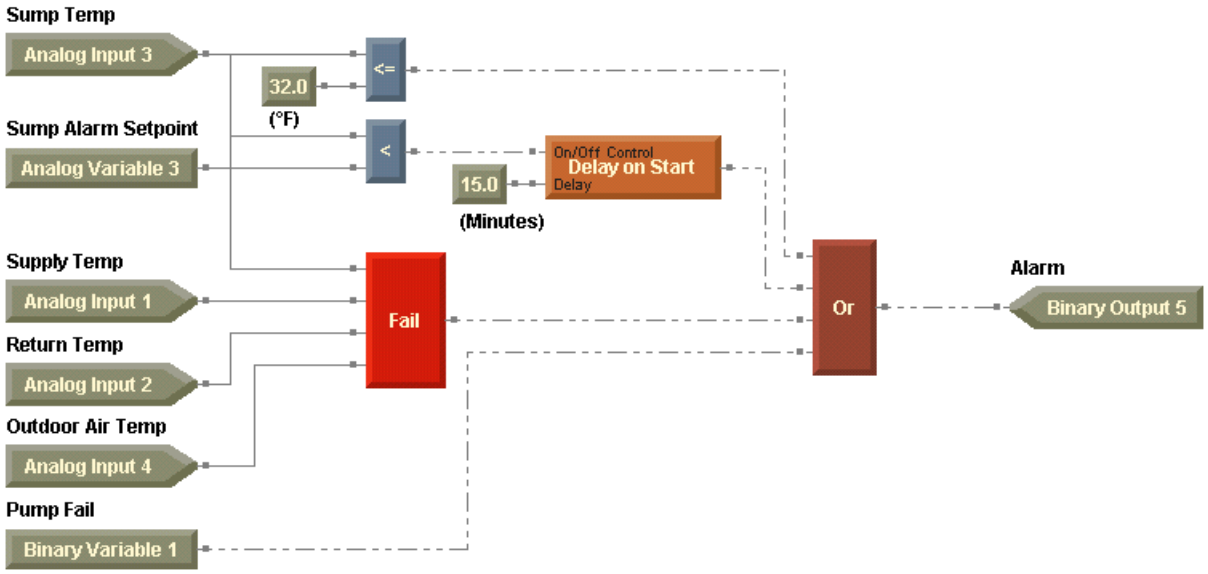
4. Click OK. The Fail block changes to include four input ports.
5. Connect the Sump Temp, Supply Temp, Return Temp, and Outdoor Air Temp input blocks to the Fail block (Figure 57 on page 59).

Figure 57: Fail block added to test temperature sensors



6. Expand the Or block to accommodate two additional input ports.
7. Connect the Fail block and the Pump Fail variable block to the Or block (Figure 58).

Figure 58: Test for temperature sensor failure complete



Implementing the alarm reset function

The alarm reset function requires the introduction of a new block known as the Latch block. The Latch block, a member of the Time Delay blocks group, maintains a binary-output on signal for a specified amount of time or until it is cancelled. It may be configured as timed (Figure 59) or manual (Figure 60).

Figure 59: Latch block in timed mode

Figure 60: Latch block in manual mode

For example, the Latch block in timed mode is often used to implement a timed override. You could use the Latch block in timed mode to control a binary signal for a specified amount of time. A trigger causes the countdown clock to begin and the signal to change. The trigger is a change in state (from off to on) of the controlling binary-input signal.

Using the properties dialog box for the Latch block, select the units of time as seconds, minutes, or hours. In the timed configuration, the latch function may be resettable or non-resettable (Figure 63 on page 62). If the Latch block is set as resettable, it resets its countdown clock every time the input signal changes state (from off to on). If it is set as non-resettable, the Latch block maintains its countdown clock despite repeated input signal changes in state.

Or you could use the Latch block in manual mode to output a binary signal indefinitely based on a change in state (from off to on) of a controlling binary-input signal. In either mode, the Cancel input turns the output of the Latch block off. See Figure 61 on page 60 and Figure 62 on page 61 for sample timing diagrams using the Latch block. The letter “t” represents the time interval.

Figure 61: Latch block timing diagram, relationship between trigger and output

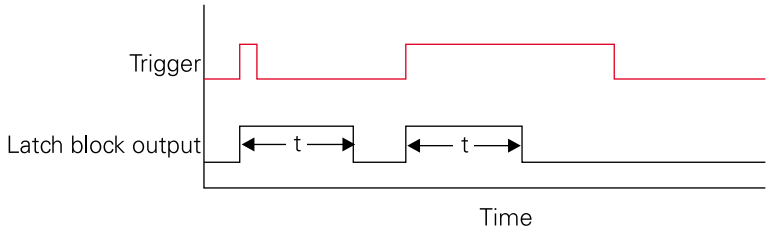
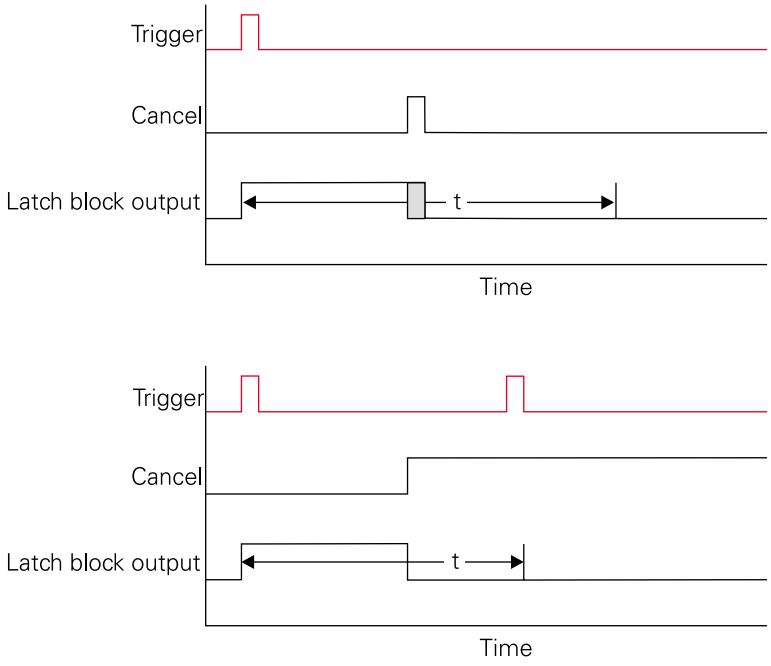


Figure 62: Latch block timing diagram, relationship between trigger, cancel, and output



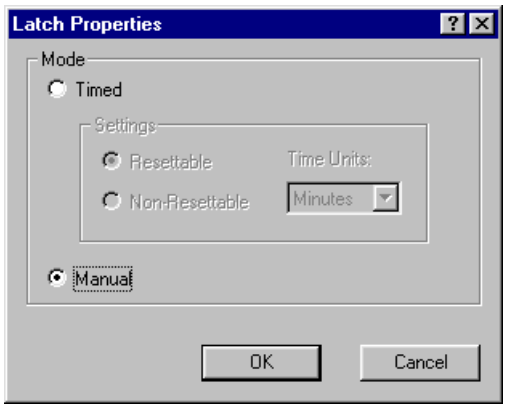
Adding a Latch block to control the alarm

In this program, the Alarm output is to be turned on if any of the alarm conditions is true. And it is to be turned off by the user using the binary variable, Alarm Reset. The Latch block in manual mode provides this functionality. Placing it between the Or block and the Alarm output block results in the following logic: Whenever the output of the Or block is true, the Latch will turn on and subsequently turn on the alarm output. Then when Alarm Reset is turned on, the Latch will turn off and subsequently turn off the Alarm output.

To add a Latch block to reset the alarm:

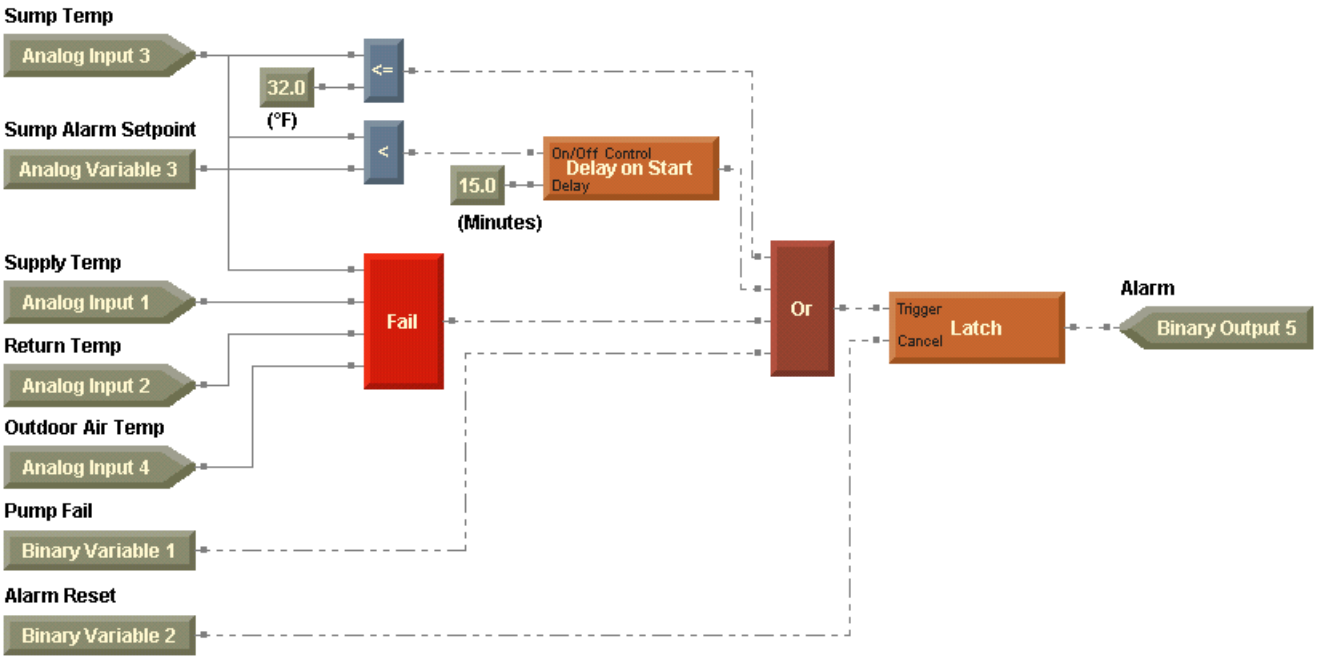
1. Delete the connection between the Or block and the Alarm output block.
2. Place a Latch block in the design space.
3. Double-click the Latch block. The Latch Properties dialog box appears (Figure 63 on page 62).

Figure 63: Latch Properties dialog box



4. Under Mode, click the Manual option.
5. Click OK. The Latch block changes to include only Trigger and Cancel ports.
6. Connect the Or block to the Trigger port of the Latch block.
7. Connect the Alarm Reset variable block to the Cancel port of the Latch block.
8. Connect the Latch block to the Alarm output block (Figure 64).

Figure 64: Alarm reset implemented



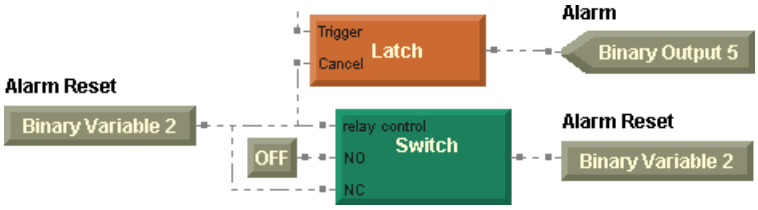
Turning the alarm reset off automatically

The source for the Alarm Reset binary variable is both the operator display/service tool and the program. The ability to apply both control sources to a variable makes what you are about to do possible. To reset an alarm condition, the operator changes the value of the Alarm Reset variable from off to on. In the current form of the program, after the alarm resets, the operator is required to change the Alarm Reset variable from on to off. If the operator neglects this last step, subsequent alarms will reset automatically without being addressed. But you can eliminate this extra step with a little additional programming. See Figure 65.

To turn the alarm reset off automatically:

1. Place a Constant block in the design space and assign the binary value OFF to it.
2. Place a Switch block in the design space and set it as binary.
3. Place a Variable block in the design space and assign the binary variable Alarm Reset to it.
4. Set the Alarm Reset variable block to be a write variable.
5. Connect the first Alarm Reset variable block to the Relay Control port of the Switch block.
6. Connect the OFF constant block to the NO port of the Switch block.
7. Connect the output port of the Alarm Reset variable block to the NC port of the Switch block.
8. Connect the Switch block to the second Alarm Reset variable block (Figure 65).
9. Compile and save your program to check for errors and to preserve your work.

Figure 65: Alarm reset module



As you go through the logic, make the initial assumptions that no alarm conditions exist and that the Alarm Reset variable is off. Upon the detection of an alarm condition, such as a temperature sensor failure, the Trigger input of the Latch block becomes true. This turns on the Alarm binary output. Because the Alarm Reset remains off at this point, the Switch block passes the value of the Alarm Reset variable itself, essentially its own value.

The operator may observe the alarm at this point and, after correcting the situation that caused the alarm, changes the Alarm Reset to on. First, the Cancel port of the Latch block becomes true, turning the Alarm output

off. Then the Switch block passes the value of the binary constant (off) to the Alarm Reset variable. The Alarm Reset variable value changes from on to off. Figure 66 and Table 6 illustrate the logic.

Note:
Place alarm reset logic within the program calculating the failure. The reset action may be erratic if done in another program operating at a different program run frequency.

Figure 66: Alarm reset module

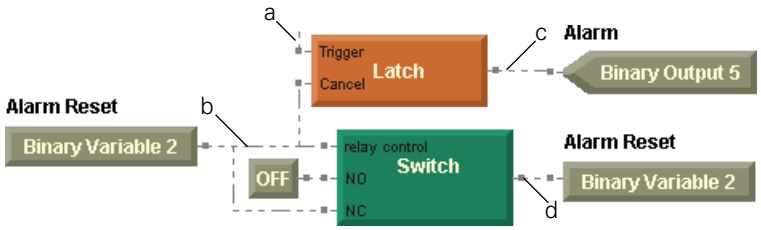


Table 6: Alarm reset module state table

Run	a	b	c	d	Comments
1	F	F	F	F	Initially, no alarm conditions exist and Alarm Reset is false (off).
2	T	F	T	F	The program detects an alarm condition.
3	F	F	T	F	The operator corrects the alarm condition, but the Alarm remains true (on).
4	F	T	F	F	The operator turns on the Alarm Reset, which cancels the Alarm.
5	F	F	F	F	No alarm conditions exist and Alarm Reset is false (off).

➤ *Using state tables*

Table 6 is an example of a state table. Keep the following in mind when reading state tables in this guide.

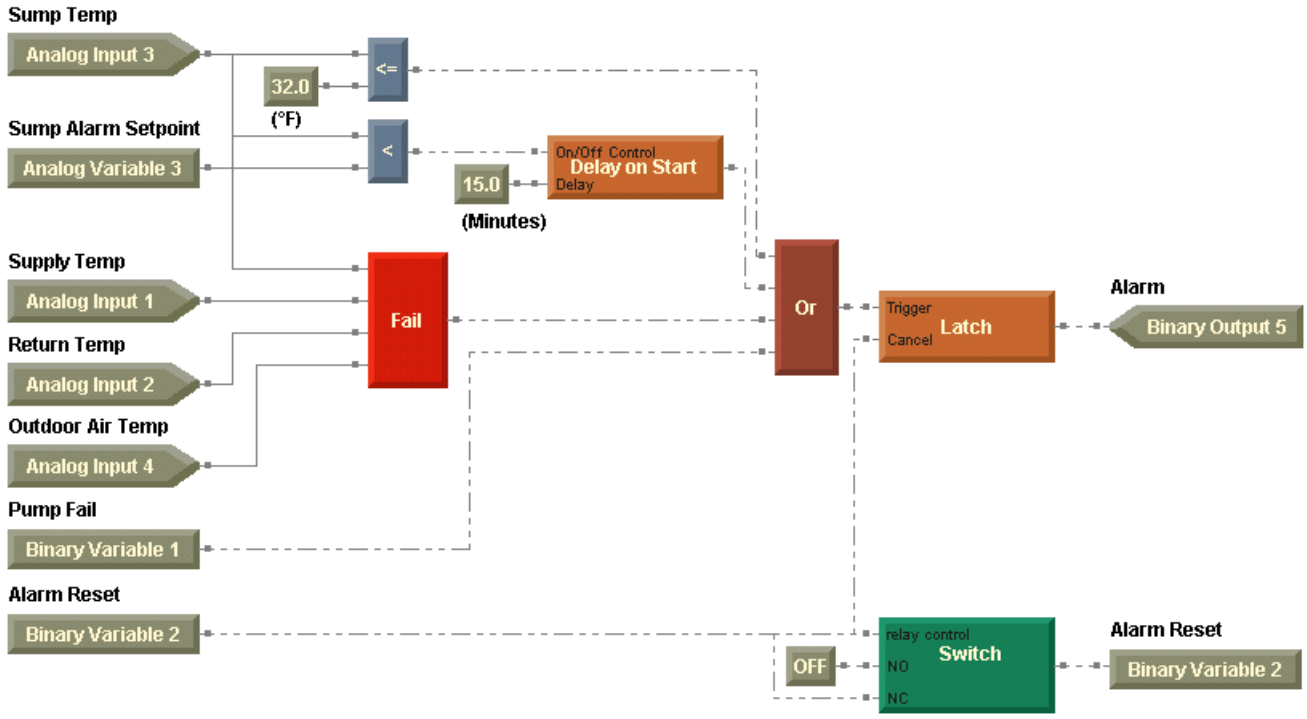
- Each row represents a discrete point in time in which a program runs once.
- F is equivalent to false, off, or zero (0).
- T is equivalent to true, on, or one (1).

➤ *Reading from and writing to the same variable*

How do you ensure that blocks within a program execute in the right order? The graphical programming compiler determines the order of execution and compiles the program accordingly. For example, in the alarm reset module in Figure 65 on page 63, the Alarm Reset variable is both read and written to. The compiler, by checking for dependencies, determines that the write to the variable depends on the read, so the write must occur after the read.

Programming tip:
Write to a variable only once in a program.

Figure 67: Alarms module completed



Adding pages to your program

At this point in the program, you may be running out of design space. To increase the space, add more pages to your program.

To add pages to your program:

1. From the File menu, choose Program Properties. The Program Properties dialog box appears.
2. In the Number Of Pages field, type the number of pages you want in your program.
3. Click OK.

Writing the sump heater module

Take another look at the part of the sequence of operation dealing with control of the sump heater.

> Cycle the sump heater if the sump temperature falls below 40°F (this value is adjustable at the operator display). If the outdoor air temperature falls below 32°F, turn on the sump heater continuously.

The programming elements required to accomplish this task, including the following:

- Use a deadband to cycle the sump heater under normal operating conditions. A deadband prevents excessive heater cycling.
- Use a comparison to keep the sump heater on continuously when the outdoor air temperature is below the freezing point.

Adding the input blocks

Begin by adding blocks to represent the inputs to this part of the program.

Adding wireless connections

You used Sump Temp and Outdoor Air Temp in the alarms module. Instead of adding these blocks again or trying to connect to them from another module, use wireless connections.

➤ ***Connecting blocks using wireless connections***

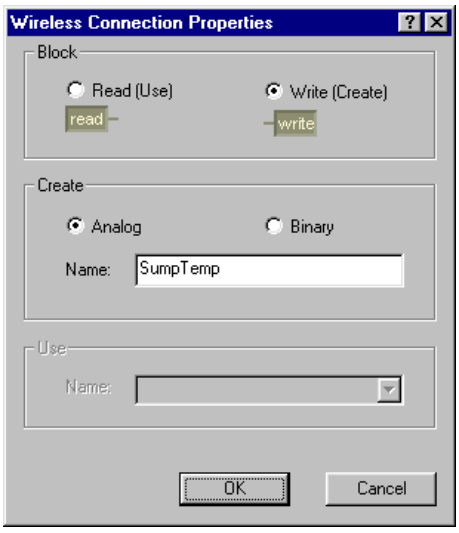
Use wireless connections to pass data from block to block when wired connections are impractical. In general, it is good programming practice to use an input in a program only once. By doing so, you will have an easier time debugging your program. By using the wireless connection, you can also prevent long and overlapping wired connections that are difficult to follow.

First create a wireless connection for the Sump Temp input.

To add a wireless connection:

1. From the Blocks menu, choose Wireless. The cursor changes to a cross-hair (✛) in the design space.
2. Click in the design space to place a Wireless block. The Wireless Connection Properties dialog box appears (Figure 68 on page 67).

Figure 68: Creating Wireless connection block

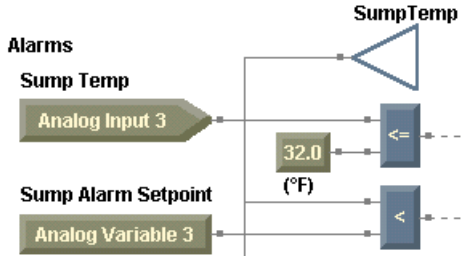


3. Under Block, click the Write (Create) option to create a Wireless connection block.
4. Under Create, click the Analog option to create an analog connection.
5. In the Name field, type:

   SumpTemp

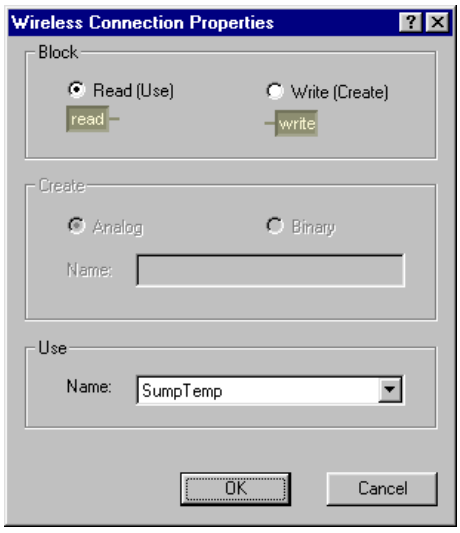
   The name may be up to 16 characters in length. Spaces are not allowed.
6. Click OK. The Wireless write block appears in the design space.
7. Connect the Wireless write block with a wired connection to the Sump Temp input block (Figure 69).

Figure 69: Wireless write connection



8. From the Blocks menu, choose Wireless. and click in the design space to place a Wireless block. The Wireless Connection Properties dialog box appears (Figure 70 on page 68).

Figure 70: Using Wireless connection block



9. Under Block, click the Read (Use) option to use a Wireless connection block.
10. Under Use, in the Name list, click SumpTemp.
11. Click OK. The Wireless read block appears in the design space (Figure 71).

Figure 71: Wireless read

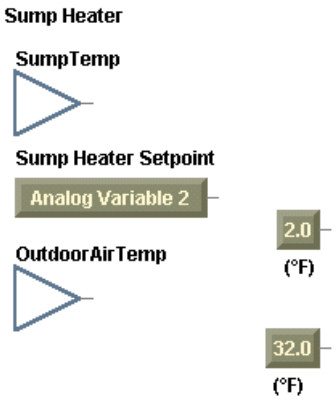
Adding the other input blocks

Add the remaining input blocks needed for the sump heater module.

To add the input blocks:

1. Place a Variable block in the design space and assign the analog variable Sump Heater Setpoint (sourced from the operator display/service tool) to it.
2. In the alarms module, create a Wireless write block, name it OutdoorAirTemp, and connect it to the Outdoor Air Temp input block.
3. Place a Constant block in the design space and assign the value 2.0 to it to represent a 2.0°F deadband.
4. Place another Constant block in the design space and assign the value 32.0 to it to represent the freezing point (Figure 72 on page 69).

Figure 72: Input blocks for sump heater module



Adding the output block

- Place an Output (Hardware) block in the design space and assign the binary output, Sump Heater, to it as shown in Figure 73.

Figure 73: Output block for sump heater module

Controlling the sump heater under normal conditions

Use the Deadband block to control the sump heater under normal conditions.

To control the sump heater under normal conditions:

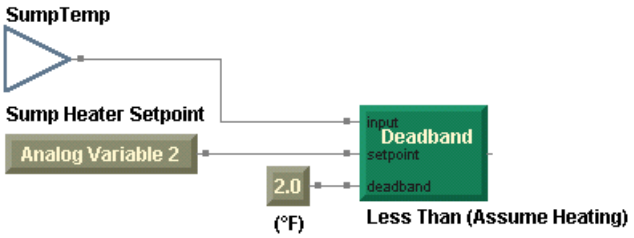
1. Place a Deadband block in the design space and set it to assume heating.

   Choose assume heating because you want the heater to come on if the Sump Temp falls below the Sump Heater Setpoint (40°F). You want the heater to stay on until the Sump Temp rises to the Sump Heater Setpoint plus the deadband constant (2.0°F).

2. Connect the SumpTemp wireless read block to the Input port of the Deadband block.
3. Connect the Sump Heater Setpoint variable block to the Setpoint port of the Deadband block.

4. Connect the 2.0 constant block to the Deadband port of the Deadband block (Figure 74).

Figure 74: Deadband incorporated in sump heater module



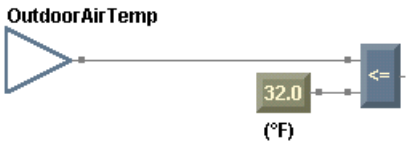
Comparing the outdoor air temperature with the freezing point

Use a Less Than or Equal block to maintain the sump heater on continuously when the outdoor air temperature is less than or equal to the freezing point.

To compare the outdoor air temperature with the freezing point:

1. Place a Less Than or Equal block in the design space.
2. Connect the OutdoorAirTemp wireless read block and the 32.0 constant block to the Less Than or Equal block (Figure 75).

Figure 75: Less Than or Equal comparison in sump heater module



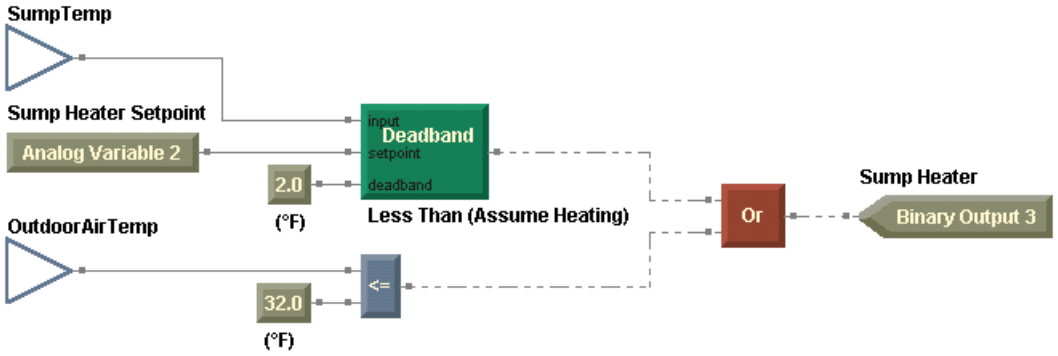
Controlling the sump heater on or off

Add an Or block to combine the two paths so that if the sump temperature falls below 40°F or the outdoor air temperature is less than or equal to the freezing point, the sump heater is controlled on.

To control the sump heater on or off:

1. Place an Or block in the design space and connect the Deadband block and the Less Than or Equal block to the Or block.
2. Connect the Or block to the Sump Heater On/Off output block (Figure 76 on page 71).
3. Compile and save your program to check for errors and to preserve your work.

Figure 76: Sump heater module completed



Writing the cooling tower fan module

Review the portion of the sequence of operation concerning the cooling tower fan.

> If condenser water flow is established and if the condenser water temperature rises 2.5°F above the setpoint, start the fan at low speed. Switch the fan to high speed if the condenser water temperature rises to 5.0°F above the setpoint. Turn off the fan when the condenser supply water temperature falls below the setpoint minus 2.5°F. A minimum 30-second delay is required between starting the fan at low speed and switching to high speed.

The following important points may be drawn from the above sequence.

- Condenser water flow is a prerequisite to starting the fan.
- This part of the sequence involves a concept known as staging. Multiple deadbands may be used to control the fan stages.
- The fan must remain at low speed for 30 seconds prior to transitioning to high speed. You need to incorporate time-based control.

Adding the input blocks

Two inputs (hardware) and one variable are required for control of the cooling tower fan.

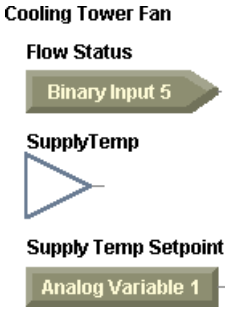
- Flow Status
- Supply Temp
- Supply Temp Setpoint

To add the input blocks:

1. Place an Input (Hardware) block in the design space and assign the binary input, Flow Status to it.
2. Create a Wireless write block, name it SupplyTemp, and connect it to the Supply Temp input block.
3. Place the SupplyTemp wireless read block in the design space.

4. Place a Variable block in the design space and assign the analog variable, Supply Temp Setpoint (sourced from the operator display/service tool), to it (Figure 77).

Figure 77: Input blocks for the cooling tower fan module



Adding the output blocks

The cooling tower fan module requires two outputs.

To add the output blocks:

- Place two Output (Hardware) blocks in the design space and assign the following binary outputs to them (Figure 78):
  - Fan Start/Stop/Low
  - Fan High

Figure 78: Output blocks for the cooling tower fan module

Starting the fan at low speed

Dissecting the sequence of operation into ever smaller sections, you find that starting the fan at low speed depends on two criteria.

- Condenser water flow is established.
- The condenser supply temperature is 2.5°F above the setpoint.

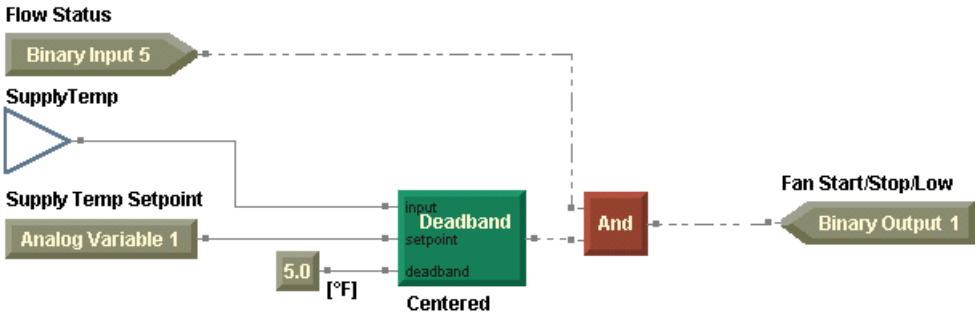
Use a Deadband block to start and stop the fan. Apply a centered deadband with a deadband value of 5.0°F. According to the sequence, the upper limit, or setpoint, for the Deadband block is equal to the condenser water setpoint plus 2.5°F.

Use an And block to account for both conditions necessary to start the fan at low speed.

To start the fan at low speed:

1. Place a Deadband block in the design space and set it as Centered.
2. Place a Constant block in the design space and assign the value 5.0 to it.
3. Connect the SupplyTemp wireless read block to the Input port of the Deadband block.
4. Connect the Supply Temp Setpoint variable block to the Setpoint port of the Deadband block.
5. Connect the 5.0 constant block to the Deadband port of the Deadband block.
6. Place an And block in the design space.
7. Connect the Flow Status input block and the Deadband block to the And block.
8. Connect the And block to the Fan Start/Stop/Low output block (Figure 79).

Figure 79: Start the fan at low speed



Transitioning the fan to high speed

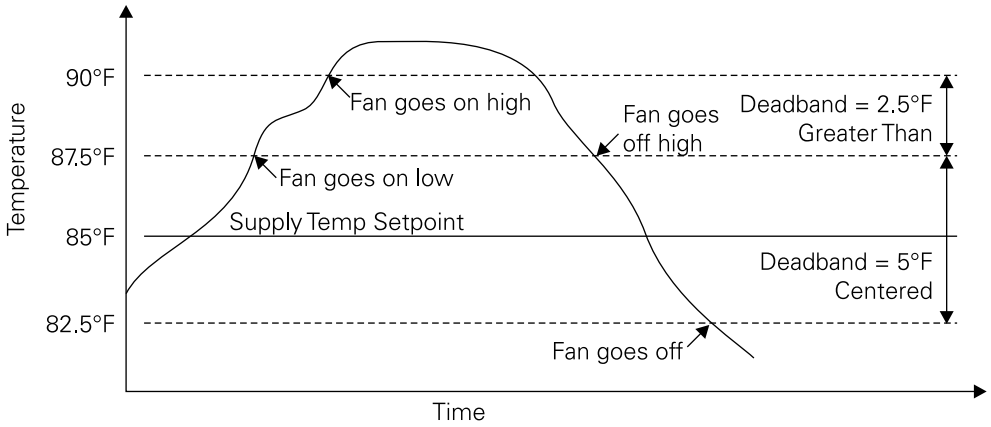
The transitioning the fan to high speed depends on two additional criteria.

- The condenser supply temperature is 5.0°F above the setpoint.
- The fan is on at low speed, and it has been on at low speed for at least 30 seconds.

Transitioning the fan based on supply temp

Apply a second deadband to activate the fan at high speed based on a combined deadband and offset from the setpoint of 5.0°F. See Figure 80 on page 74 for more information about how the deadbands work together.

Figure 80: Deadbands used to control fan speed

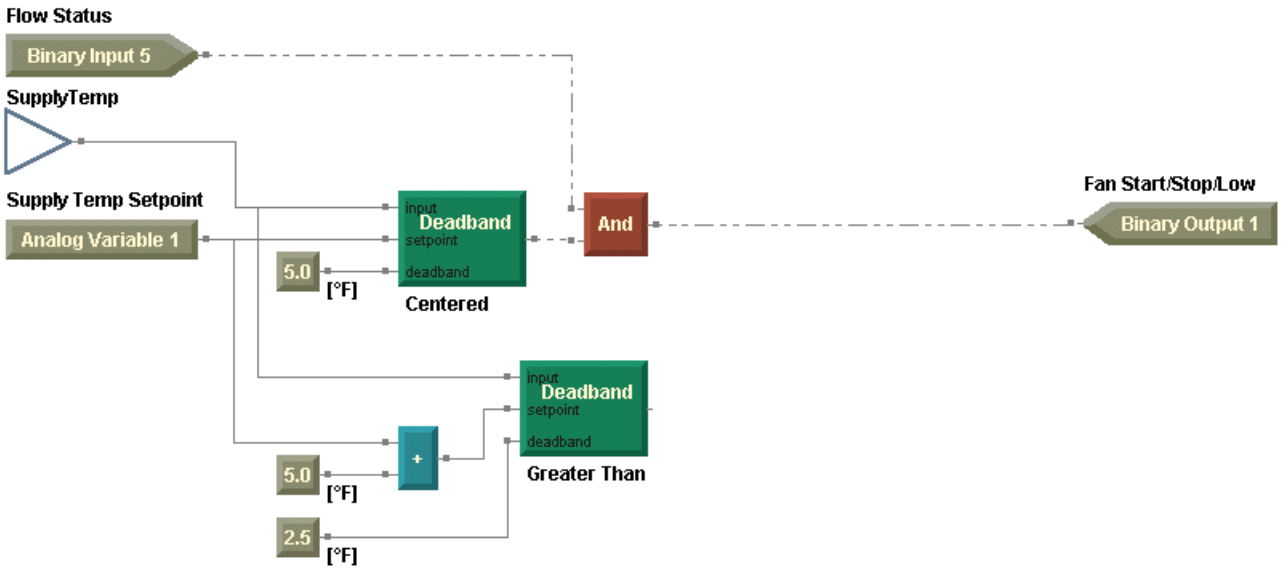


Use a Math block to incorporate this functionality. Math blocks perform standard math operations and functions. Specifically, use an Add block to adjust the setpoint value by adding 5.0°F. The result of the addition serves as the setpoint input to the Deadband block.

To transition the fan based on supply temp:

1. Place a Deadband block in the design space and set it as Greater Than.
2. From the Blocks menu, choose Math. From the Math menu, choose Add.
3. Click in the design space to place an Add block.
4. Place a Constant block in the design space and assign the value 5.0 to it.
5. Connect the SupplyTemp wireless read block to the Input port of the Deadband block.
6. Connect the Supply Temp Setpoint variable block and the 5.0 constant block to the Add block.
7. Connect the Add block to the Setpoint port of the Deadband block.
8. Place a Constant block in the design space and assign the value 2.5 to it.
9. Connect the 2.5 constant block to the Deadband port of the Deadband block (Figure 81 on page 75).

Figure 81: Adding the fan at high speed



Transitioning the fan based on time

To implement the time delay, use the Delay on Start block. The on/off control input of the Delay on Start block comes from the output of the And block. The delay countdown begins when the fan starts at low speed. Use another And block to ensure that both of the previously mentioned criteria apply to control of the binary output, Fan High.

To transition the fan based on time:

1. Place a Delay on Start block in the design space and assign its units to seconds.
2. Place a Constant block in the design space and assign the value 30.0 to it.
3. Place another And block in the design space.
4. Connect the first And block to the On/Off Control port of the Delay on Start block.
5. Connect the 30.0 constant block to the Delay port of the Delay on Start block.
6. Connect the Delay on Start block and the second Deadband block to the second And block.
7. Connect the second And block to the Fan High output block.

8. Arrange your blocks with comments in the design space as shown in Figure 82.

Note:
Even in graphical programming, comments are especially helpful in describing the logic used in the program. Place specific comments near the blocks they are describing. Otherwise, place general program notes in one location as in Figure 82.

9. Compile and save your program to check for errors and to preserve your work.

Figure 82: Cooling tower fan module complete

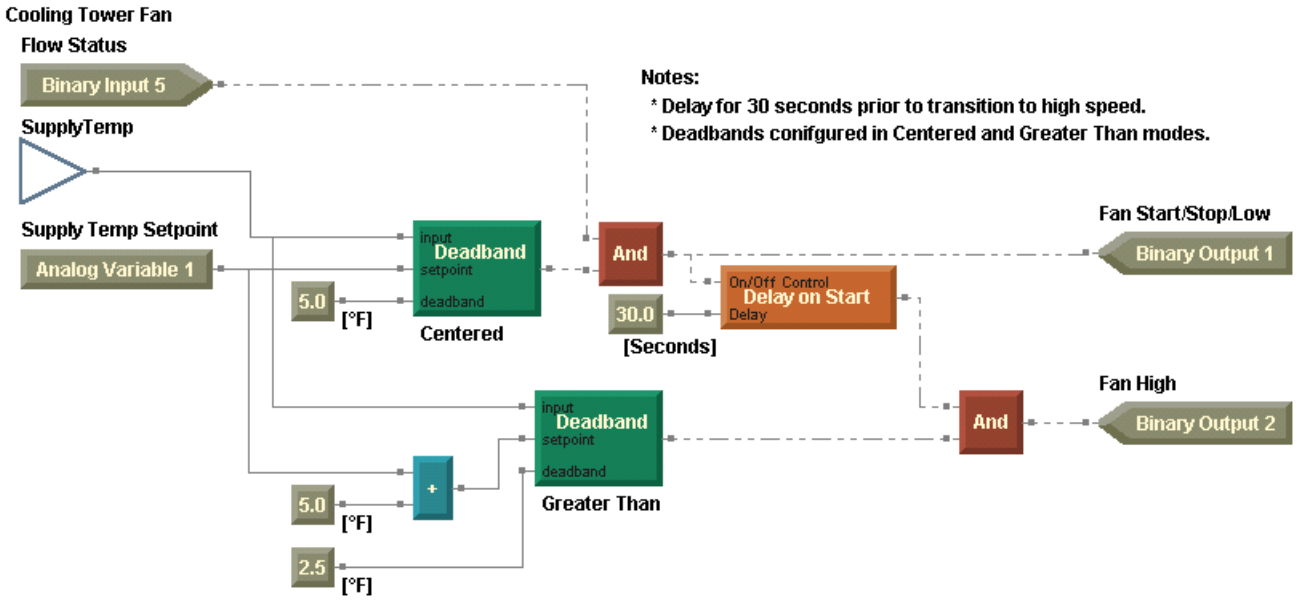
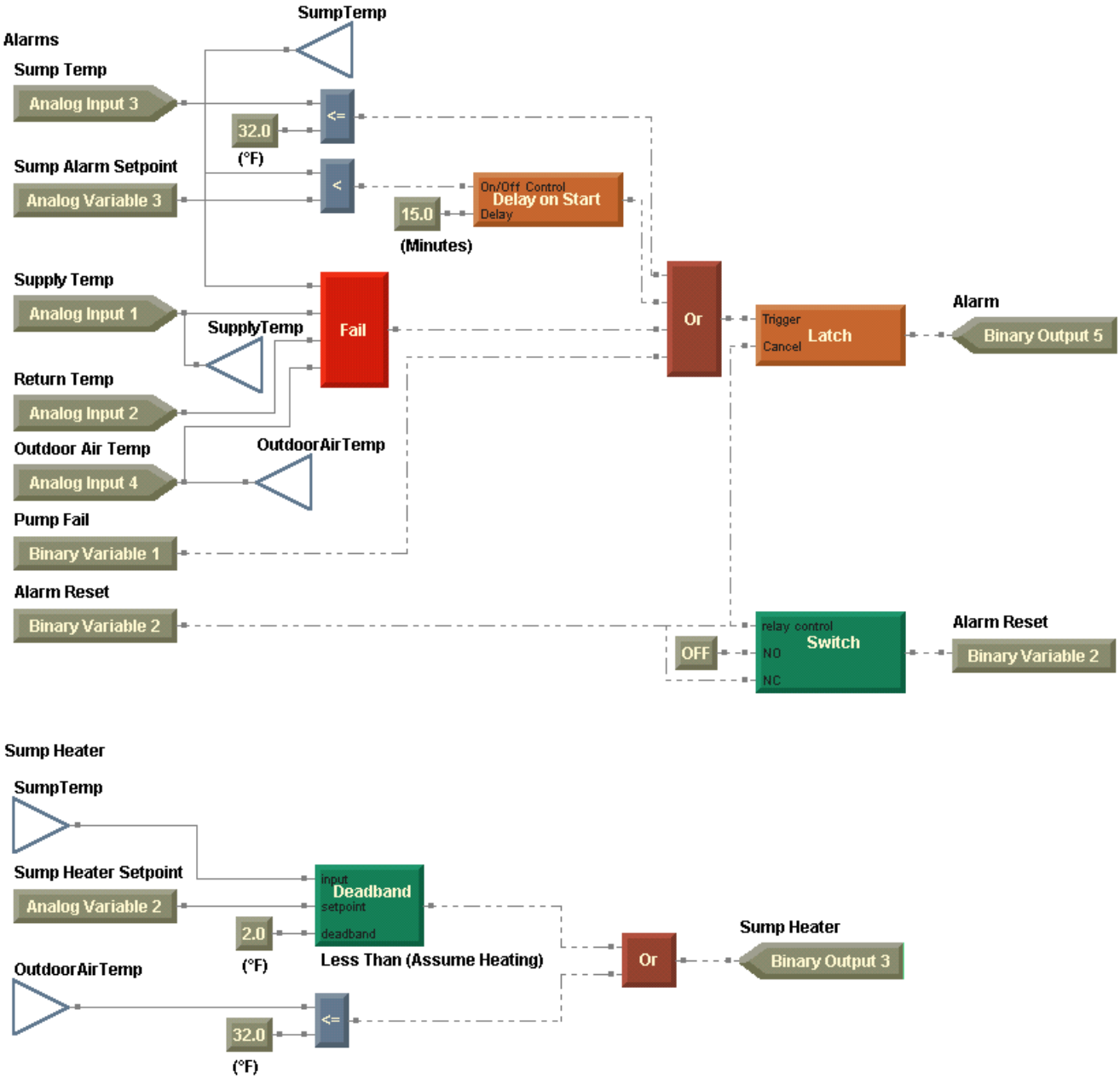
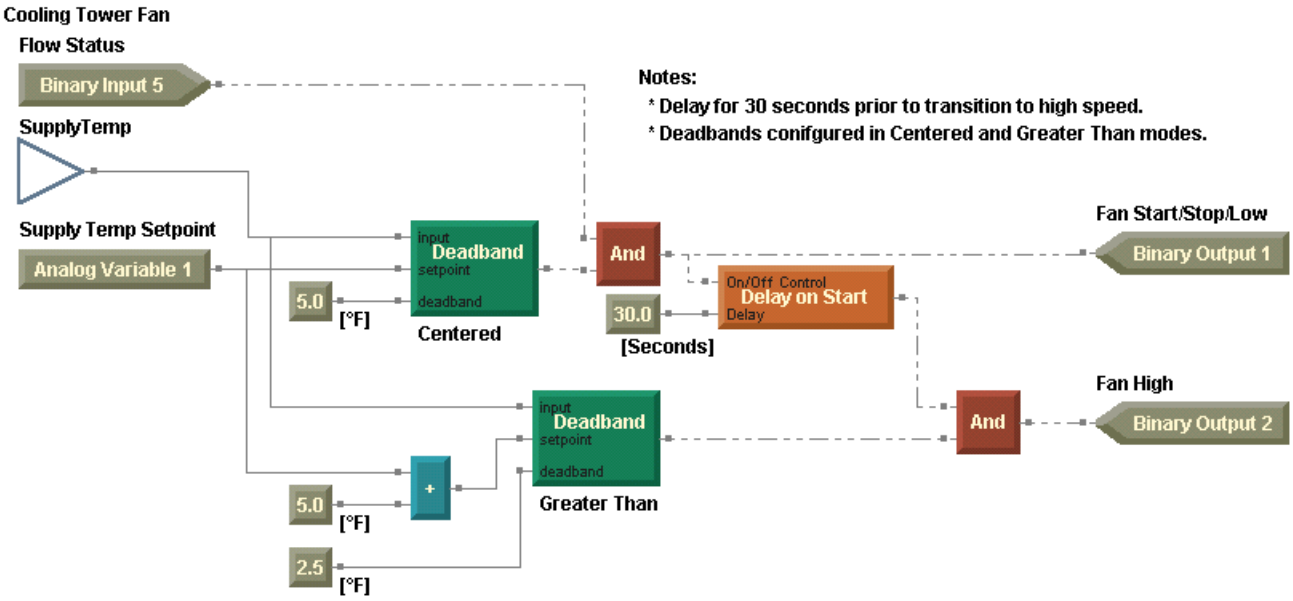


Figure 83 on page 77 is the cooling tower with two-speed fan program. Go to Chapter 5, “Cooling tower with variable-speed fan example” to add the condenser water pump module and to modify your program.

Figure 83: Completed cooling tower with two-speed fan program





Summary questions

Answer the following questions to review the skills, concepts, definitions, and blocks you learned in this chapter. The answers to these questions are on page 236.

1. Which Time Delay block optionally applies a timer based on its configuration?

2. How would you change the cooling tower fan module of the program (Figure 82 on page 76) if the fan motor controller accounts for the delay between low and high speeds?

Chapter 5

Cooling tower with variable-speed fan example

In this chapter, you will build upon the cooling tower program that you constructed in Chapter 4. Some changes to the cooling tower that you must accommodate in your program include a variable-speed fan (instead of a two-speed fan) and calculation of some critical performance parameters for the cooling tower.

Note:
Many of the chapters in this book build on previous chapters, so be sure to complete the chapters in the order presented. See "About this book" on page 1 for additional instructions.

What you will learn

In this chapter, you will learn a variety of skills, concepts, and definitions.

Skills

You will learn how to interpret increasingly complex sequences of operation.

Concepts and definitions

You will understand Calculation blocks.

Blocks

You will learn how to use the following blocks:

- Output Status
- Subtract
- Wet-Bulb
- Min
- Max
- Limit
- PID
- Feedback Alarm
- Not

Reviewing the sequence of operation

In this scenario a cooling tower with a variable-speed fan delivers condenser water to a small chiller plant. The following specifications apply to control of the cooling tower.

Condenser water pump

When condenser water is requested by the chiller plant, command the condenser water pump to start. If condenser water flow fails to be confirmed within 30 seconds, command the pump to stop and indicate a pump failure at the operator display and turn on the alarm output. A user must be able to reset the alarm at the operator display.

Cooling tower fan

Upon successful confirmation of condenser water flow, and when the cooling tower supply water temperature exceeds the setpoint by 2.5°F, turn on the fan. Modulate the fan to maintain the condenser water supply temperature according to setpoint. When the cooling tower supply water temperature is 2.5°F below the setpoint, turn off the fan. Note that the setpoint is adjustable from the operator display and that it is limited to a minimum of 65°F and a maximum of 95°F.

Sump heater

Cycle the sump heater if the sump temperature falls below 40°F (this value is adjustable at the operator display). If the outdoor air temperature falls below 32°F, turn on the sump heater continuously. If the sump temperature remains below 36°F (this value is adjustable at the operator display) for 15 minutes, or if the sump temperature falls below 32°F, indicate an alarm at the operator display and turn on the alarm output.

Alarms

In addition to the alarm requirements mentioned above, indicate an alarm at the operator display and turn on the alarm output when any sensor, including temperature and humidity, fails. Alarms must be resettable at the operator display.

Calculations

Calculate and display the change in water temperature across the cooling tower, the ambient wet-bulb temperature, and the approach temperature. (The approach temperature is defined as the difference between the ambient wet-bulb temperature and the condenser water supply temperature.)

The specifications for the pump and the fan have changed. The alarms have changed a little bit with the addition of the humidity sensor failure. The calculations section is completely new.

Analysis of this scenario results in Figure 84. The corresponding data definition is presented in Table 7, and a wiring diagram is presented in Figure 85 on page 83.

Figure 84: Cooling tower with variable-speed fan drive data

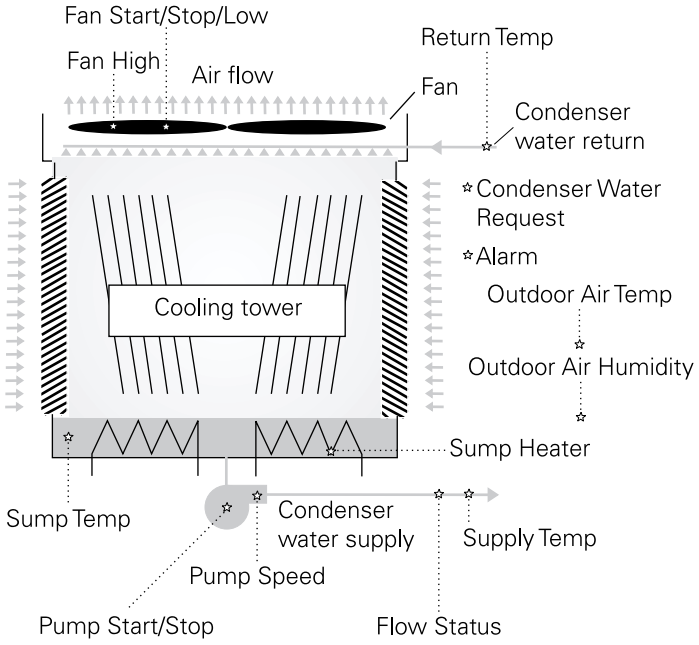


Table 7: Cooling tower with variable-speed fan drive data definition

Data	Type	Name	Notes
Inputs	Analog	Supply Temp	Universal input configured as thermistor or RTD (Balco, Platinum)
		Return Temp	Universal input configured as thermistor or RTD (Balco, Platinum)
		Sump Temp	Universal input configured as thermistor or RTD (Balco, Platinum)
		Outdoor Air Temp	Universal input configured as thermistor or RTD (Balco, Platinum)
		Outdoor Air Humidity	Universal input configured as a current input
	Binary	Flow Status	
		Condenser Water Request	

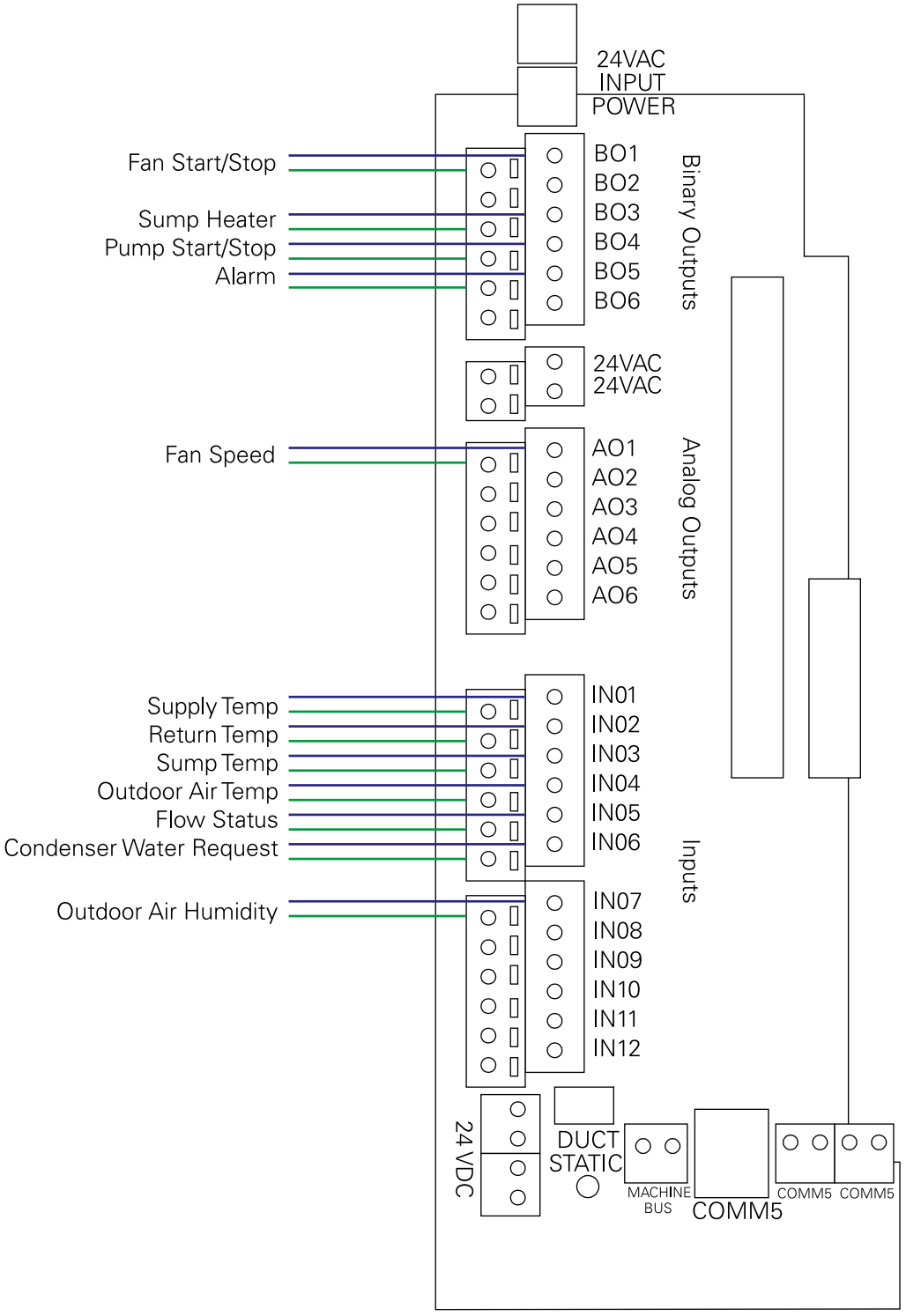
Table 7: Cooling tower with variable-speed fan drive data definition (continued)

Data	Type	Name	Notes
Outputs	Analog	Fan Speed	Analog output configured as voltage, 2–10 V
	Binary	Fan Start/Stop	Apply sufficient minimum on/off timers to prevent excessive fan cycling.
		Sump Heater	Apply sufficient minimum on/off timers to prevent excessive heater cycling.
		Pump Start/Stop	Apply sufficient minimum on/off timers to prevent excessive pump cycling.
		Alarm	
Variables	Analog	Supply Temp Setpoint	Analog variable, source set to operator display/service tool
		Sump Heater Setpoint	Analog variable, source set to operator display/service tool
		Sump Alarm Setpoint	Analog variable, source set to operator display/service tool
		Delta Temp	Analog variable, source set to program
		Wet-Bulb Temp	Analog variable, source set to program
		Approach Temp	Analog variable, source set to program
	Binary	Pump Fail	Binary variable, source set to program
		Alarm Reset	Binary variable, source set to operator display/service tool and program*

* The binary variable, Alarm Reset, is sourced from both the operator display/service tool and the program. This allows the user to turn on the reset. After the alarm is reset, the program turns the Alarm Reset off.

Before you start to write the program, configure these inputs, outputs, and variables in your Tracer MP580. Then open the TGP editor.

Figure 85: Cooling tower with variable-speed fan drive wiring diagram



Refreshing the TGP editor

If you already have the TGP editor open and you make changes to the controller configuration, you do not have to close the editor. Use the refresh function to update the TGP editor. For example, if you are writing a program and you realize that you need to set up an analog variable, go to the Rover application to set up the variable. Then return to the TGP editor and refresh the analog variables to implement the variable you just configured.

To refresh the TGP editor:

1. From the View menu, choose Refresh.
2. From the Refresh menu, choose the set of data you want to refresh or choose All. The configuration data uploads to the editor.

Determining a programming approach

Review the sequence of operation to determine the changes that are required to the cooling tower program in the module that controls the cooling tower fan. The new sequence of operation calls for modulating control of the fan. Also, a new module is required to calculate critical performance parameters for the cooling tower and to control the condenser water pump. For the purposes of this tutorial, work on the modules in the following order:

1. Alarms
2. Calculations
3. Cooling tower fan
4. Condenser water pump

Editing the program properties

Set the program properties, including the program name and its run frequency. Use a run frequency of 30 seconds.

To edit the program properties:

1. Open the cooling tower with two-speed fan program.
2. Save the program under a new name.
3. Open the Program Properties dialog box and set the properties, along with an updated description of the program.
4. Click OK.

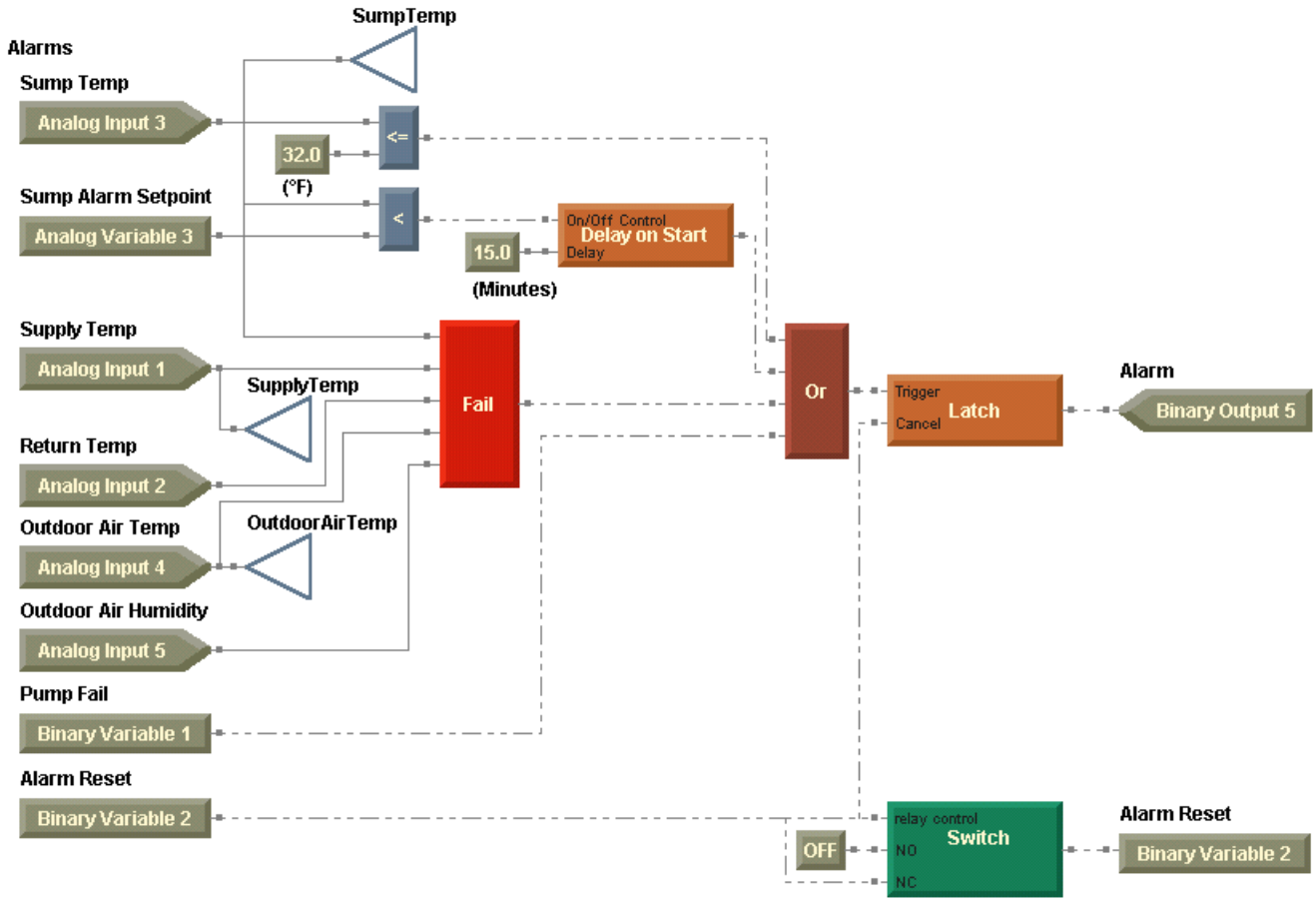
Modifying the alarms module

Using the Alarms module that you created in Chapter 4, expand the Fail block to check the new, outdoor air relative humidity input for failure.

To modify the alarms module:

1. Add an Input (Hardware) block to the design space and assign the analog input, Outdoor Air Humidity, to it.
2. Expand the Fail block.
3. Connect the Outdoor Air Humidity input block to the Fail block (Figure 86).

Figure 86: Alarms module completed



Writing the calculations module

According to the sequence of operation, the program must perform three calculations:

- Change in water temperature across the cooling tower
- Ambient wet-bulb temperature
- Approach temperature

Calculating change in water temperature across the cooling tower

Start with the simplest of the three calculations and assemble the blocks needed to calculate the change in water temperature across the cooling tower.

To calculate the change in water temperature across the cooling tower:

1. Create a Wireless write block, name it ReturnTemp, and connect it to the Return Temp input block.
2. Place a ReturnTemp and a SupplyTemp wireless read block in the design space.
3. Place a Subtract block in the design space and connect it so that the Supply Temp is subtracted from the Return Temp.

➤ ***Inputs to a Math block***

For some Math blocks, the input port you choose for a value is important. For example, the Subtract block subtracts the bottom input-port value from the top input-port value. Remember this relationship when working with the Subtract and Divide blocks.

4. Place a Variable block in the design space and assign the analog variable, Delta Temp, to it. Make sure to set the block so that you can write to the variable.
5. Connect the Subtract block to the Delta Temp variable block (Figure 87).

Figure 87: Change in temperature calculation

Calculating the ambient wet-bulb temperature

The second calculation requirement calls for calculation of the ambient wet-bulb temperature. Use the Wet-Bulb block, a member of the Calculation blocks. Calculation blocks perform calculations unique to HVAC control applications. Use these blocks to determine air-flow rate, enthalpy, dewpoint temperature, or wet-bulb temperature. You can set Calculation block properties with the units for the inputs and, in some cases, for the output.

The wet-bulb temperature of (moist) air is a function of the dry-bulb temperature and the relative humidity.

To calculate the ambient wet-bulb temperature:

1. Create a Wireless write block, name it OAHumidity, and connect it to the Outdoor Air Humidity input block.
2. Place an OutdoorAirTemp and an OAHumidity wireless read block in the design space.
3. From the Blocks menu, choose Calculation. From the Calculation menu, choose Wet-Bulb.
4. Click in the design space to place the Wet-Bulb block and assign its units to degrees Fahrenheit (°F).
5. Connect the OutdoorAirTemp wireless read block to the Dry-Bulb Temp port of the Wet-Bulb block.
6. Connect the OAHumidity wireless read block to the Rel Humidity port of the Wet-Bulb block.
7. Place a Variable block in the design space and assign the analog variable, Wet-Bulb Temp, to it.
8. Connect the Wet-Bulb block to the Wet-Bulb Temp variable block so that the calculated value is written to the variable (Figure 88).

Figure 88: Wet-bulb temperature calculation

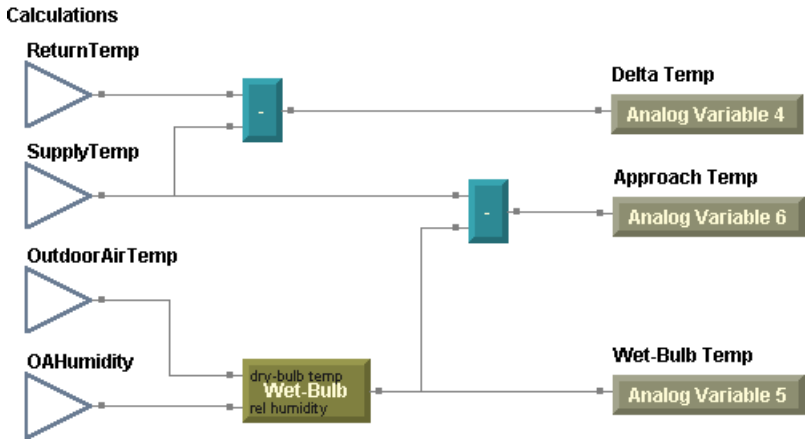
Calculating the approach temperature

Finally, calculate the approach temperature, or the difference between the ambient wet-bulb temperature and the condenser water supply temperature. This calculation is very similar to the calculation used to determine the change in water temperature across the cooling tower. The inputs to this calculation may be obtained from existing blocks.

To calculate the approach temperature:

1. Place a Subtract block in the design space and connect it so that the Wet-Bulb Temp is subtracted from the Supply Temp.
2. Place a Variable block in the design space and assign the analog variable, Approach Temp, to it.
3. Connect the Subtract block to the Approach Temp variable block so that the calculated value is written to the variable (Figure 89).

Figure 89: Calculations module complete



Writing the cooling tower fan module

The new sequence of operation calls for modulation of the fan speed to maintain the condenser water supply temperature.

> Upon successful confirmation of condenser water flow, and when the cooling tower water supply temperature exceeds the setpoint by 2.5°F, turn on the fan. Modulate the fan to maintain the condenser water supply temperature according to setpoint. When the cooling tower supply water temperature is 2.5°F below the setpoint, turn off the fan. Note that the setpoint is adjustable from the operator display and is limited to a minimum of 65°F and a maximum of 95°F.

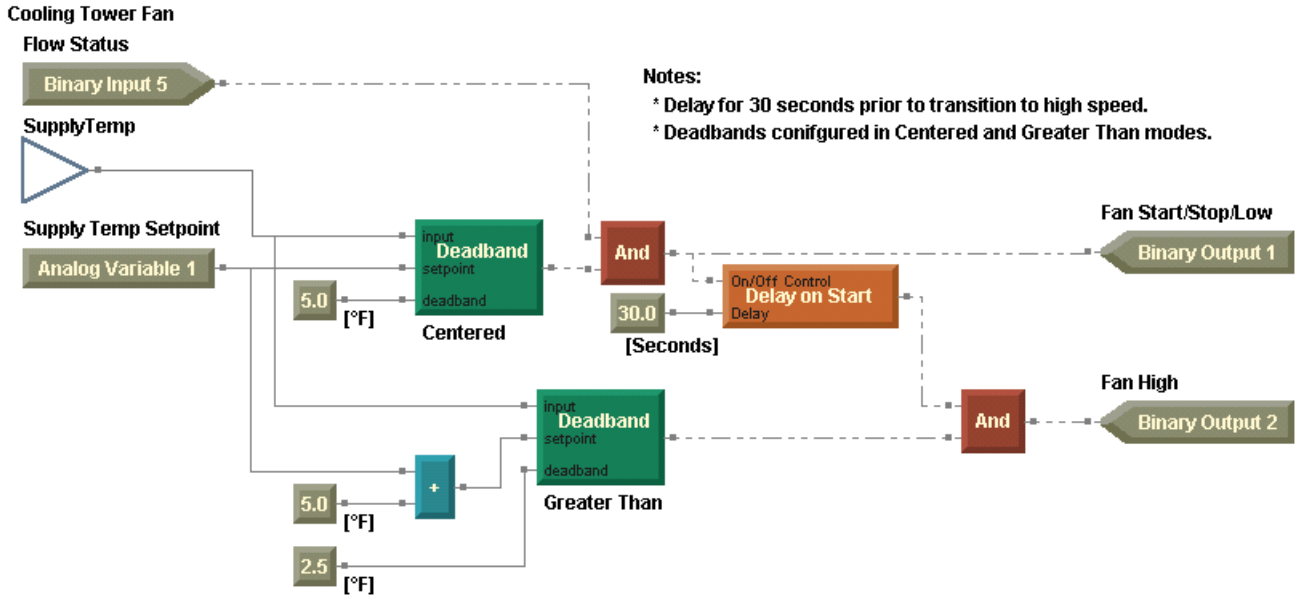
Starting and stopping the fan

As in Chapter 4, a deadband is required to start and stop the fan. Use a centered deadband to start the fan when the water temperature is 2.5°F above the setpoint and to stop the fan when the water temperature is 2.5°F below setpoint. Remember that flow must be confirmed in order for the fan to start.

To start and stop the fan:

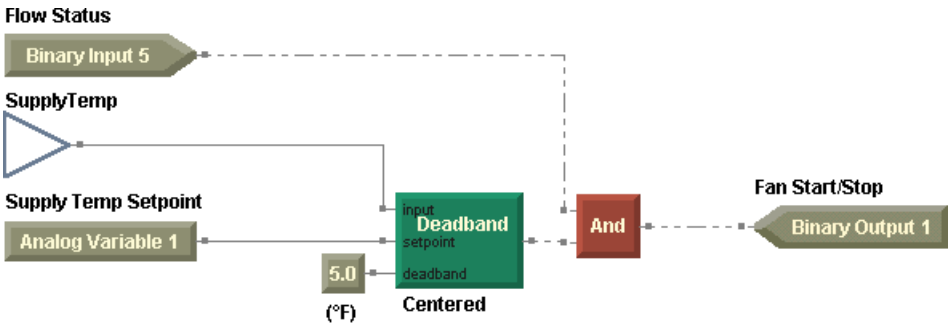
1. Return to the cooling tower fan module of your program (Figure 90).

Figure 90: Previous cooling tower fan module



2. Delete all the blocks implementing the transition of the fan to high speed (Figure 91).

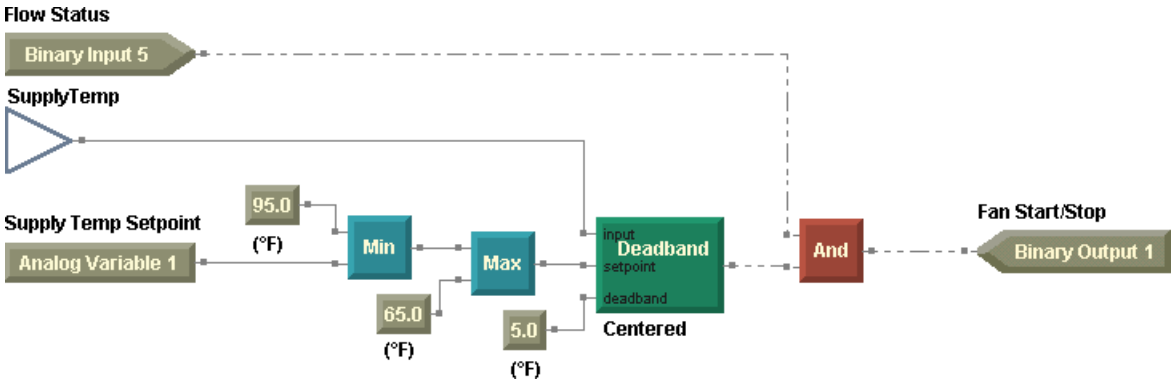
Figure 91: Using a deadband to start and stop the fan



Imposing limits

The program must check the setpoint against the setpoint limits per the sequence of operation. You have two options as to which blocks to use. The limit may be applied using a combination of the Min and Max blocks, or you may use the Limit block. Figure 92 on page 90 is an example of how you could use the Min and Max blocks.

Figure 92: Checking a setpoint against limits using the Min and Max blocks

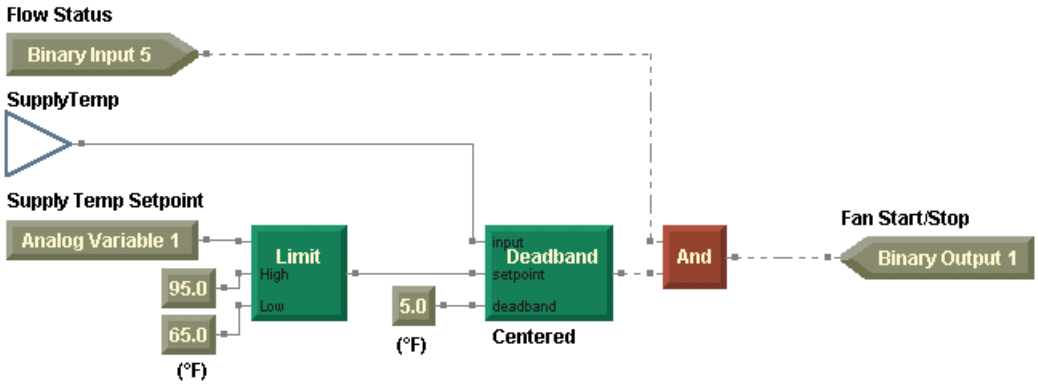


Implement the simpler module using the Limit block in your program. The Limit block compares a value with high and/or low limit values, depending on how you set the properties for the block. If the value is higher than the high limit, the high-limit value is passed out of the block. If the value is lower than the low limit, the low-limit value is passed. If the value is within the limits, the value itself is passed.

To impose limits:

1. Delete the connection between the Supply Temp Setpoint variable block and the Deadband block.
2. From the Blocks menu, choose Function. From the Function menu, choose Limit.
3. Click in the design space to place the Limit block.
4. Place two constant blocks in the design space and assign 95.0 to one for the high limit and 65.0 to the other for the low limit.
5. Connect the Limit block so that it compares the Supply Temp Setpoint to the low and high limit and passes the correct value to the Deadband block (Figure 93).

Figure 93: Checking a setpoint against limits using the Limit block



Implementing PID control

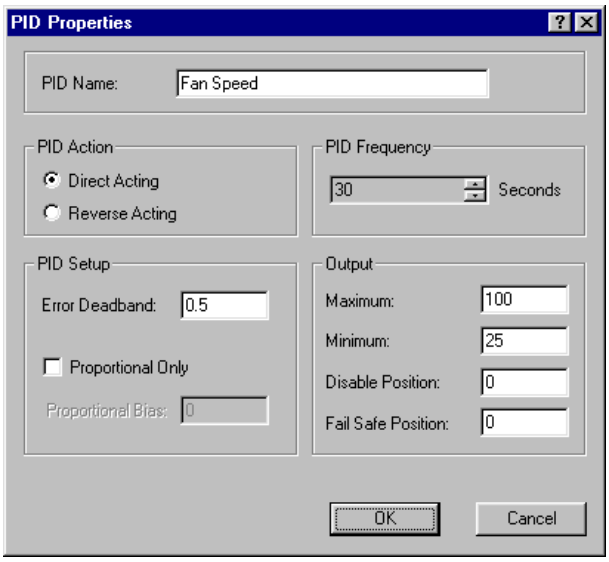
Modulation of fan speed to maintain the condenser water setpoint calls for proportional, integral, derivative (PID) control. Implement PID control in graphical programming using the PID block.

Note:
For an in-depth explanation of PID control, see the online help for the block or the applications guide, *PID Control in Tracer Controllers* (CNT-APG002-EN).

Setting up the PID block properties

1. From the Blocks menu, choose Calculation. From the Calculation menu, choose PID.
2. Click in the design space to place the PID block.
3. Double-click the PID block. The PID Properties dialog box appears (Figure 94).

Figure 94: PID Properties dialog box



4. In the PID name field, type:

   Fan Speed

   The name is used to select a PID loop for troubleshooting and plotting purposes.

5. Under PID Action, click the Direct Acting option.

   The action of a PID loop determines how it reacts to a change in the measured variable, or the process variable. A controller using direct action increases the output when the measured variable increases. A controller using reverse action decreases the output when the mea-

sured variable increases. In this program, the fan speed increases when the cooling tower supply water temperature increases.

6. In the Error Deadband field, type:

   0.5

   Error deadband prevents the PID from changing when the measured value is within plus or minus this value of the setpoint.

7. Click to clear the Proportional Only check box. The Proportional Bias field remains unavailable.

8. Under PID Frequency, verify the value is 30 seconds.

   The PID Frequency defaults to the same frequency as the program. For this program, do not change the default frequency. But note that you may run a PID loop at the same rate as its parent program or at a slower rate. The time interval must be an integer multiple of the program run frequency.

9. In the Maximum field, type:

   100

   This field sets the maximum output of the PID loop to 100%.

10. In the Minimum field, type:

    25

    This field sets the minimum output of the PID loop to 25%.

11. In the Disable Position field, type:

    0

    When the Output Enable/Disable port of the PID block receives a value of false, the PID loop outputs the disable-position value. Otherwise, the PID loop outputs its calculated value.

12. In the Fail Safe Position field, type:

    0

    When the Fail port of the PID block receives a value of true, the PID loop outputs the fail-safe-position value. Otherwise, the PID loop outputs its calculated value.

13. Click OK.

Incorporating the PID block

Add the necessary intermediate blocks and then make the PID block connections.

To incorporate the PID block:

1. Place a Fail block in the design space and connect it so that it checks the measured variable, the supply temperature, for failure.

Note:
Wireless connections with input blocks do not pass the fail flag to the Fail block, so add an Input (Hardware) block for the supply temperature.

2. Place three constant blocks to serve as the proportional, integral, and derivative gains for the PID block.
3. Set the constant values to 4.0, 1.0, and 0.0, respectively.

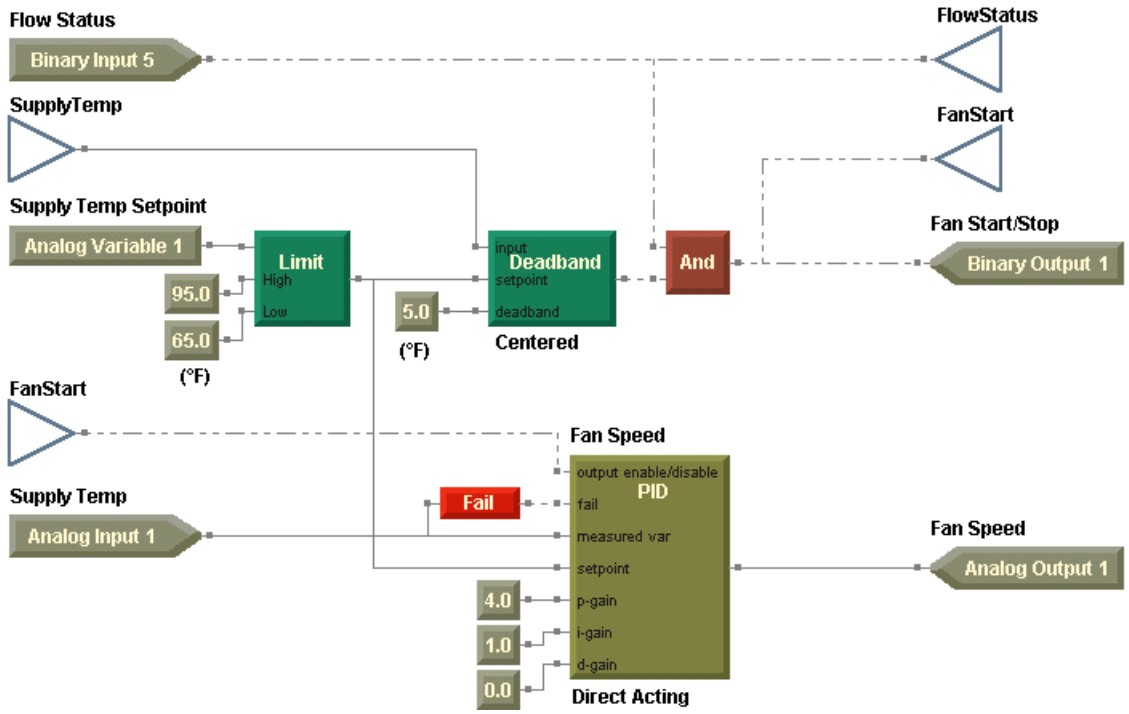
   These values are place holders for the proportional, integral, and derivative gains. When you are writing a program for a specific job, determine the best values for that job. You could also use Variable blocks here so that you can tune the PID loop from the operator display.
4. Create a Wireless write block, name it FanStart, and connect it to the And block.

   Remember that Wireless blocks can be used to pass any type of analog or binary data. Use a Wireless block to pass a hardware input value or to pass the result of some arbitration.
5. Place a FanStart wireless read block in the design space and connect it to the Output Enable/Disable port of the PID block.

   When the PID block receives a true value here, it outputs its calculated result. When the PID block receives a false value here, it outputs its disable-position value.
6. Connect the Fail block to the Fail port of the PID block.

   When the PID block receives a false value here, it outputs its calculated result. When the PID block receives a true value here, it outputs its fail-safe-position value.
7. Connect the Supply Temperature input block to the Measured Variable port of the PID block.
8. Connect the Limit block to the Setpoint port of the PID block.
9. Connect the Constant blocks to the appropriate ports of the PID block.
10. Connect the PID block to the NO port of the Switch block.
11. Place comments in the design space to explain the logic of the module (Figure 95 on page 94).
12. Compile and save your program to check for errors and to preserve your work.

Figure 95: Cooling tower fan module complete



Writing the condenser water pump module

The remaining portion of the sequence of operation concerns the condenser water pump.

> When condenser water is requested by the chiller plant, command the condenser water pump to start. If condenser water flow fails to be confirmed within 30 seconds, command the pump to stop and indicate a pump failure at the operator display and turn on the alarm output. A user must be able to reset the alarm at the operator display.

First, note the following observations about the sequence.

- Control of the pump is dependent on flow status; however, flow status is also dependent on the pump control.
- The actual alarm is handled by the alarms module of the program. This module simply determines whether or not to indicate a pump failure.

Adding the input blocks

Start with the inputs to this portion of the program.

To add the input blocks:

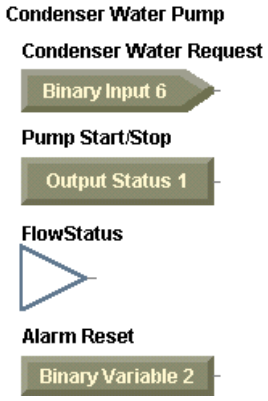
1. Place a Input (Hardware) block in the design space and assign the Condenser Water Request binary input to it.
2. Place an Output Status block in the design space and assign the Pump Start/Stop binary out to it. Use the output status to allow overrides of the pump without indicating a pump failure.

➤ ***Using the Output Status block***

Use the Output Status block to use the current value of an analog or binary output in a program. The Output Status block only reads the status of an output; it does not control the selected analog or binary output.

3. Create a Wireless write block, name it FlowStatus and connect it to the Flow Status input block.
4. Create a Wireless write block, name it AlarmReset and connect it to the first Alarm Reset variable in the alarms module.
5. Place the FlowStatus and AlarmReset wireless read blocks in the design space.
6. Arrange your blocks with comments in the design space as shown in Figure 96.

Figure 96: Input blocks for the condenser water pump module



Adding the output blocks

Place the output blocks of the program on the right side of the design space.

To add the output blocks:

1. Place an Output (Hardware) block in the design space and assign the binary output, Pump Start/Stop, to it.

2. Place a Variable block in the design space and assign the binary variable, Pump Fail (sourced from the program and a write variable), to it (Figure 97).

Figure 97: Output blocks for the condenser water pump module

Determining when to start and stop the pump

To simply start the pump based on the request, you could connect the Condenser Water Request directly to the Pump Start/Stop output. However, control of the pump is also dependent on the flow status. If flow is not confirmed in 30 seconds following the request for flow, the pump must be controlled to off. Furthermore, an alarm indication is required. A block exists specifically for this purpose—the Feedback Alarm block.

Adding a Feedback Alarm block

The Feedback Alarm block compares the status of a binary input, output, variable, or other binary value to a requested state. The binary output of the Feedback Alarm block is determined by the selected relationship between the status and the request. Using the properties dialog box for the Feedback Alarm, configure the block with one of the following three relationships:

- Request XOR Status
- Request AND NOT(Status)
- NOT(Request) AND Status

Table 8 explains how the delay timer acts based on the selected feedback alarm relationship.

Table 8: Feedback alarm relationships

Relationship	Delay timer
Request XOR Status	The delay timer starts when the request and status are not the same.
Request AND NOT(Status)	The delay timer starts when the request is true and the status is false.
NOT(Request) AND Status	The delay timer starts when the request is false and the status is true.

When the delay timer is zero, the output of the Feedback Alarm block changes to true (on). Then when the reset input to the Feedback Alarm block changes from off to on, the block output resets to false (off).

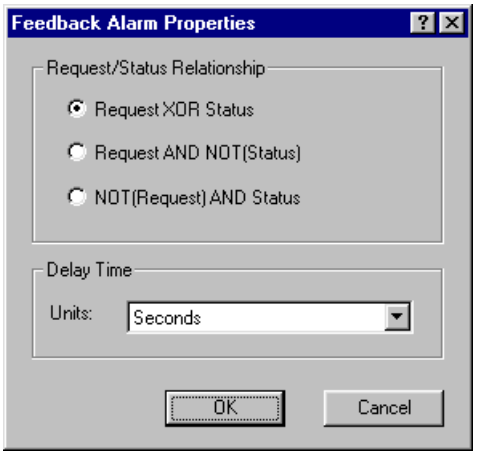
Add a Feedback Alarm block to the program to compare the condenser water request to the flow status. Set the Feedback Alarm block to the Request XOR Status relationship. Also, set the units for the delay time interval to seconds.

Next, add a Constant block and set it to an analog value of 30. The constant provides the delay time interval. Make the appropriate connections to the Feedback Alarm block.

To add a Feedback Alarm block:

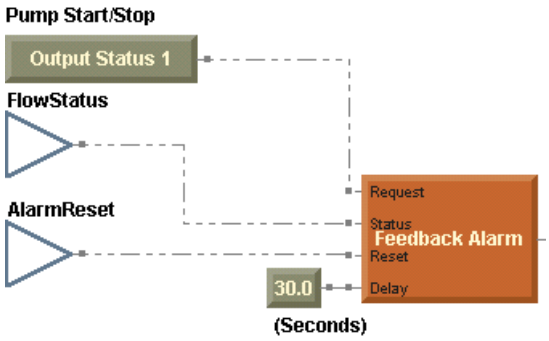
1. From the Blocks menu, choose Time Delay. From the Time Delay menu, choose Feedback Alarm.
2. Click in the design space to place the Feedback Alarm block.
3. Double-click the Feedback Alarm block. The Feedback Alarm Properties dialog box appears (Figure 98).

Figure 98: Feedback Alarm Properties dialog box



4. Under Request/Status Relationship, click the Request XOR Status option.
5. In the Units list, click Seconds.
6. Click OK.
7. Place a Constant block in the design space and assign the value 30.0 to it.
8. Connect the Pump Start/Stop output status block to the Request port of the Feedback Alarm block.
9. Connect the FlowStatus wireless read block to the Status port of the Feedback Alarm block.
10. Connect the AlarmReset wireless read block to the Reset port of the Feedback Alarm block.
11. Connect the 30.0 constant block to the Delay port of the Feedback Alarm block (Figure 99).

Figure 99: Feedback Alarm block



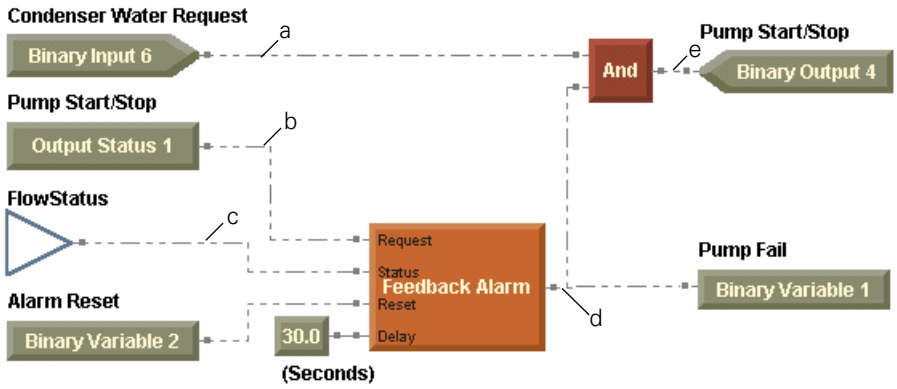
Checking conditions to start and stop the pump

Because control of the pump is dependent on two conditions, both the condenser water request and the flow status, add an And block just before the Pump Start/Stop output block.

To check conditions to start and stop the pump:

1. Place an And block in the design space and connect it so that it checks the state of both the Condenser Water Request input block and the Feedback Alarm block.
2. Connect the And block to the Pump Start/Stop output block.
3. Connect the Feedback Alarm block to the Pump Fail variable block (Figure 100 on page 99).

Figure 100: Pump start/stop module



4. See Table 9 for an analysis of this logic.

Table 9: Pump start/stop module state table

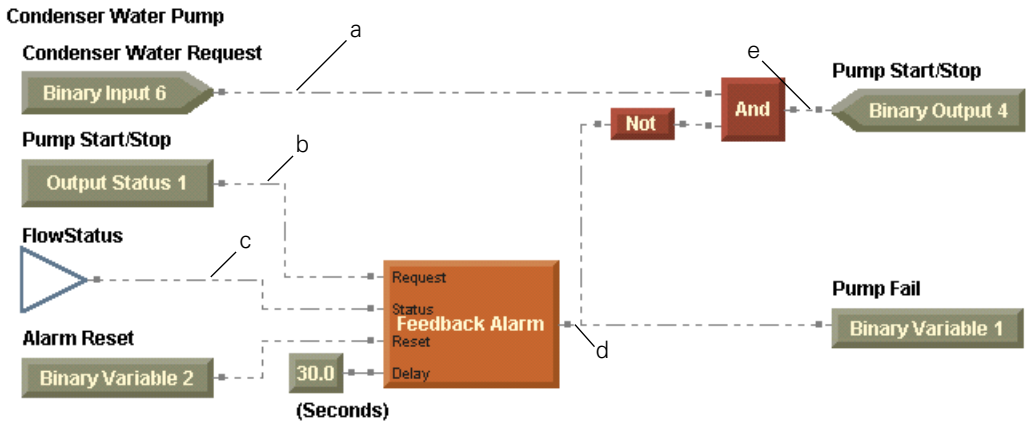
Run	a	b	c	d	e	Comments
1	F	F	F	F	F	Initially, Request is off, and Status is off.
2	T	F	F	F	F	The chiller plant calls for condenser water, so Request is on. Flow is not confirmed yet. The Feedback Alarm delay countdown begins.
3	T	F	F	T	T	The Feedback Alarm delay expires and its output transitions to on, controlling the pump on and indicating a pump failure.
4	T	T	T	T	T	Flow status is confirmed. The Feedback Alarm remains on because it has not been reset.

What is wrong with the sequence presented in Table 9? The pump did not start until the Feedback Alarm delay expired. Thus, flow is never confirmed during the delay period of 30 seconds. But the sequence of operation calls for the pump to start, allowing 30 seconds for flow confirmation. If after the 30-second delay period, flow status is not confirmed, a pump failure results.

Only one block is required to fix this problem, but which one? The answer is the Not block. The Not block changes the logic of the data to the opposite state. For example, if the Feedback Alarm block sends a true message, the Not block changes the message to false.

5. Delete the connection between the Feedback Alarm block and the And block.
6. From the Blocks menu, choose Logic. From the Logic menu, choose Not.
7. Click in the design space to place a Not block.
8. Connect the Feedback Alarm block to the Not block.
9. Connect the Not block to the And block (Figure 101 on page 100).

Figure 101: Adjusted pump start/stop module



10. See Table 10 and Table 11 to analyze the resulting logic for both a successful start and a failure to confirm flow.

Table 10: Pump start/stop module with successful start

Run	a	b	c	d	e	Comments
1	F	F	F	F	F	Initially, the Request is off, and the Status is off.
2	T	F	F	F	T	The chiller plant calls for condenser water, Request is on. The pump starts, but flow is not confirmed yet. The Feedback Alarm delay counter begins its countdown.
3	T	T	T	F	T	Flow status is confirmed. The Feedback Alarm delay countdown ceases and the pump remains on.

Table 11: Pump start/stop module with failure to confirm flow

Run	a	b	c	d	e	Comments
1	F	F	F	F	F	Initially, the Request is off, and the Status is off.
2	T	F	F	F	T	The chiller plant calls for condenser water, Request is on. The pump starts, but flow is not confirmed yet. The Feedback Alarm delay counter begins its countdown.
3	T	T	F	T	F	Flow status is not confirmed. The Feedback Alarm delay expired, and its output changes to true, resulting in a pump failure. The pump is controlled off.

11. Compile and save your program to check for errors and to preserve your work (Figure 102 on page 101).

Figure 102: Condenser water pump module completed

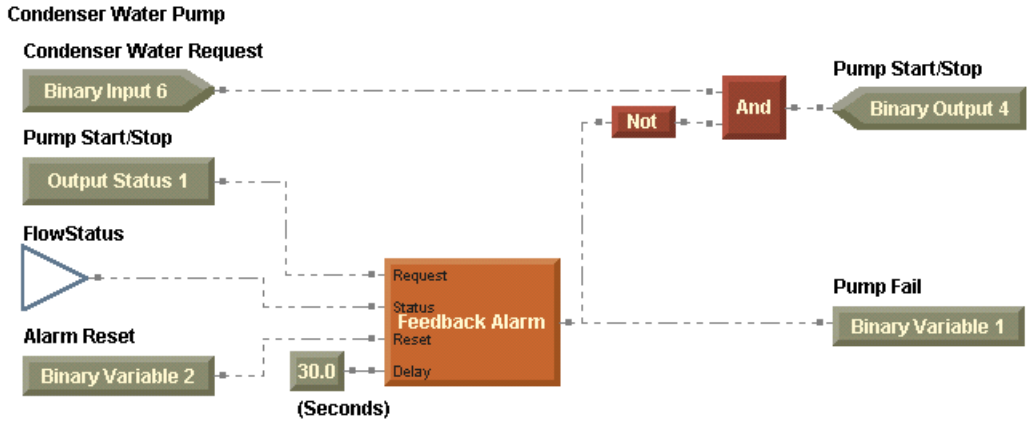
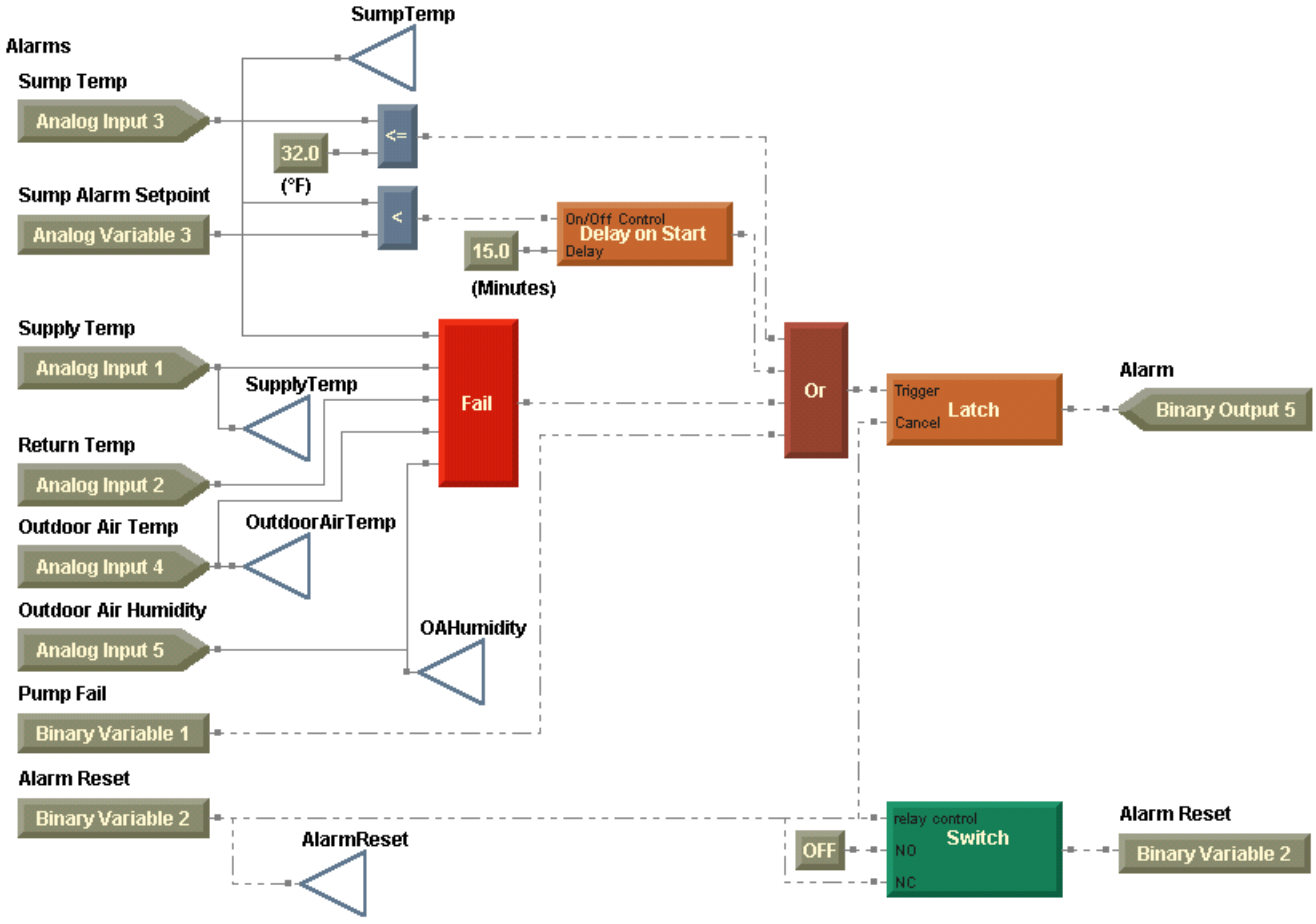
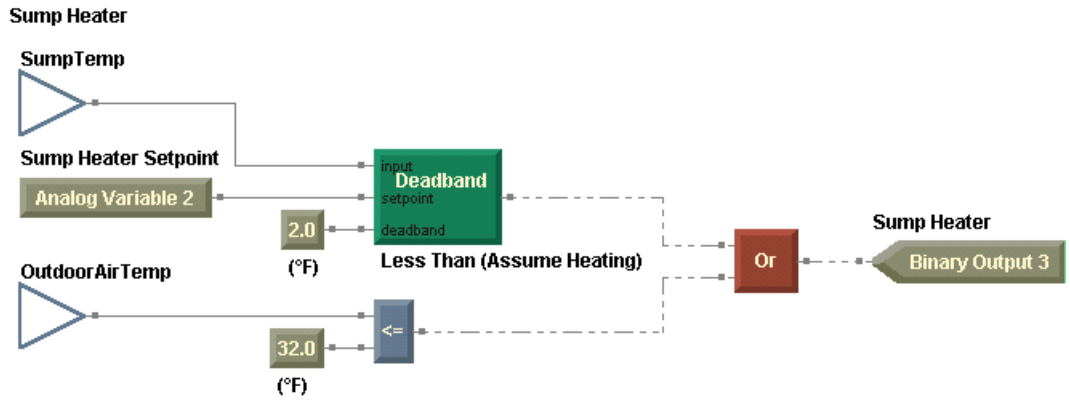
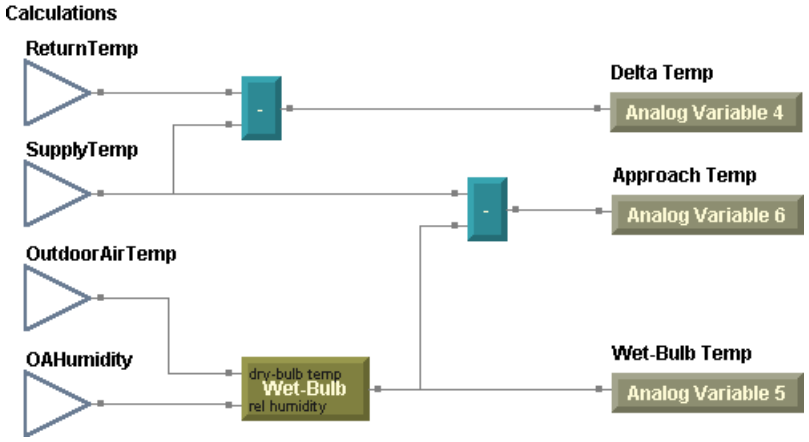


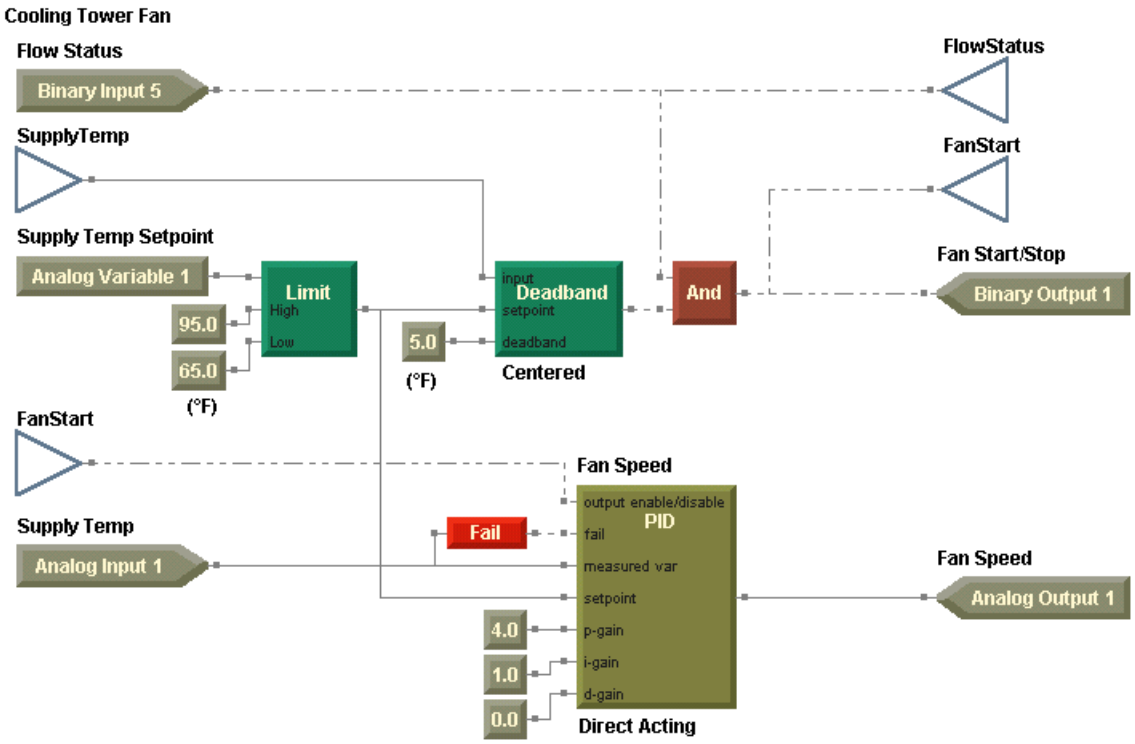
Figure 103 is the cooling tower with variable-speed fan program.

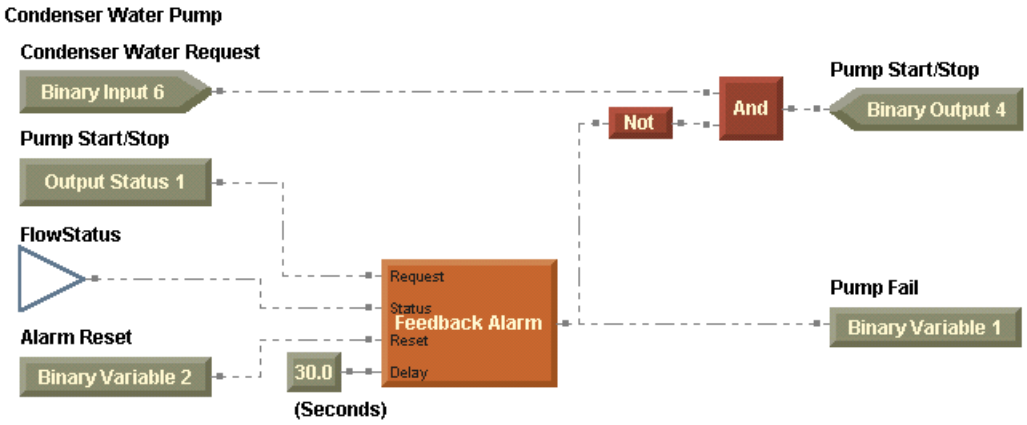
Figure 103: Complete cooling tower with variable-speed fan program




Sump Heater
SumpTemp
Sump Heater Setpoint
Analog Variable 2
input
Deadband
setpoint
deadband
2.0
(°F)
Less Than (Assume Heating)
OutdoorAirTemp
32.0
(°F)
<=
Or
Sump Heater
Binary Output 3


Calculations
ReturnTemp
SupplyTemp
OutdoorAirTemp
OAHumidity
-
-
dry-bulb temp
Wet-Bulb
rel humidity
Delta Temp
Analog Variable 4
Approach Temp
Analog Variable 6
Wet-Bulb Temp
Analog Variable 5


Cooling Tower Fan
Flow Status
Binary Input 5
FlowStatus
SupplyTemp
FanStart
Supply Temp Setpoint
Analog Variable 1
Limit
High
Low
95.0
65.0
(°F)
input
Deadband
setpoint
deadband
5.0
(°F)
Centered
And
Fan Start/Stop
Binary Output 1
FanStart
Fan Speed
output enable/disable
PID
fail
measured var
setpoint
p-gain
i-gain
d-gain
Fail
Supply Temp
Analog Input 1
4.0
1.0
0.0
Direct Acting
Fan Speed
Analog Output 1


Condenser Water Pump
Condenser Water Request
Binary Input 6
Not
And
Pump Start/Stop
Binary Output 4
Pump Start/Stop
Output Status 1
FlowStatus
Request
Status
Feedback Alarm
Reset
Delay
Alarm Reset
Binary Variable 2
30.0
(Seconds)
Pump Fail
Binary Variable 1

Summary questions

Answer the following questions to review the skills, concepts, definitions, and blocks you learned in this chapter. The answers to these questions is on page 236.

1. The logic contained in the Feedback Alarm block is a combination of several other blocks. Can you replicate this logic without using the block itself? Focus on one Feedback Alarm relationship, XOR. First construct a sequence of operation. Then write a graphical program.

2. How could you reset the Alarm Reset without using the Switch block?

Chapter 6

VAV AHU example

In this chapter, you will expand your programming skills. You will use what you learned in Chapter 1 through Chapter 5 to program a variable-air-volume (VAV) air handler. Because you are more familiar with graphical programming, this chapter proceeds at a faster pace. Instead of presenting the programs block-by-block, they are presented in a more modular form. Use the sequence of operation to guide you as you create modules that fit into larger programs.

Note:
Many of the chapters in this book build on previous chapters, so be sure to complete the chapters in the order presented. See "About this book" on page 1 for additional instructions.

What you will learn

In this chapter, you will learn a variety of skills, concepts, and definitions.

Skills

You will learn how to:

- Interpret increasingly complex sequences of operation, with a focus on VAV air handler applications
- Use logic blocks (And, Or, Xor, Not) in combination with each other

Concepts and definitions

You will understand how to apply graphical programming language to an air-handling unit.

Blocks

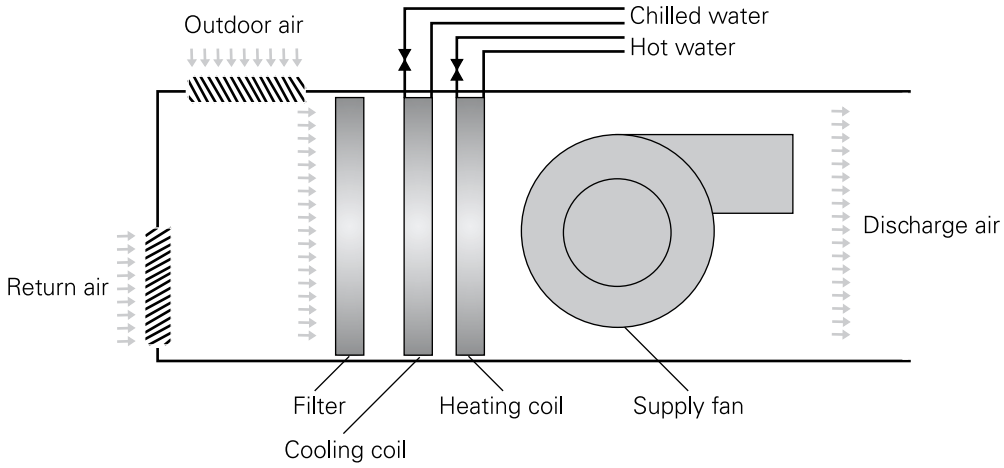
You will learn how to use the following blocks:

- Occupancy
- De-Enumerator
- Network Configuration Input

Reviewing the sequence of operation

In this scenario a VAV air handler provides comfort heating and cooling to a space. The air handler contains a variable-speed fan, both cooling and heating coils, and an outdoor air damper. The air handler is a stand-alone unit (Figure 104).

Figure 104: VAV air handler



Modes and setpoints

The following parameters for the sequence of operation are presented according to the occupancy mode, the heat/cool mode, or the setpoint.

Occupied mode

The occupied mode is initiated by the Tracer MP580/581 local schedule or by a unit override. When the air handler is in occupied mode, operate the supply fan continuously. Keep the outside air damper at minimum position, unless the air handler is economizing. Modulate the cooling valve, heating valve, and outside air damper to maintain the discharge air temperature.

Unoccupied mode

The unoccupied mode is initiated by the Tracer MP580/581 local schedule or by a unit override. When the air handler is in unoccupied mode, turn the supply fan off and fully close the outside air damper and the cooling and heating valves. Open the heating valve completely if the outdoor air temperature falls below the freeze avoidance setpoint, 35°F (adjustable).

Space setpoints

For occupied, occupied standby, and occupied bypass modes, calculate effective cooling and heating setpoints based on a single space tempera-

ture setpoint and the configured, default occupied and occupied standby setpoints.

Note:
Calculate the effective cooling and heating setpoints as in other Comm5 devices. See "Calculating the effective space setpoints" on page 135 for more information.

Discharge air setpoints

For discharge air control, use the configured, default discharge air temperature cooling and heating setpoints.

Heat/Cool arbitration

Determine the heat/cool decision of the air handler based on the effective occupied cooling and heating setpoints. Transition the air handler to cooling if the space temperature exceeds the cooling setpoint plus 1°F. Transition the air handler to heating if the space temperature falls below the heating setpoint minus 1°F.

Control

The following parameters for the sequence of operation are presented according to the equipment that must be controlled.

Supply fan

Operate the supply fan whenever the air handler is in occupied mode and modulate the fan speed to maintain the duct static pressure setpoint, 1.5 in. wg (adjustable). Turn the supply fan off whenever one of the following occurs:

- The air handler is unoccupied.
- The run/stop interlock is open.
- The mixed air temperature is too cold and the Low Temperature Detection input is closed.
- The supply fan status indicates a fan failure after a 1-minute delay.

If the duct static pressure exceeds 4 in. wg, shut down the air handler immediately.

Outdoor air damper

When the economizer function is enabled and the outdoor air temperature is less than the economizer changeover setpoint, modulate the outdoor air damper between the adjustable minimum position and fully open to maintain the discharge air cooling setpoint. Modulate the outdoor air damper closed, overriding the minimum position, to maintain the mixed air temperature at or above the mixed air setpoint.

If the economizer function is disabled, or the air handler is in the heat mode, control the outdoor air damper to its minimum position. If the outdoor air temperature falls below a low outdoor air temperature limit, control the outdoor air damper to its closed position. In unoccupied mode,

control the outdoor air damper to its closed position. Also, if the supply fan is off or the status of the mixed air temperature sensor is failed, control the outdoor air damper to its closed position.

Exhaust fan

Coordinate exhaust fan operation with the unit supply fan and outdoor air damper position. Turn on the exhaust fan whenever the supply fan is on and the outdoor air damper is open beyond 30%. The exhaust fan remains on until the outdoor air damper closes to below 20% open or the supply fan is turned off.

Cooling valve

Modulate the cooling valve to maintain the discharge air temperature at the discharge air cooling setpoint. If the economizer function is enabled and the outdoor air damper is not open at least 90%, control the cooling valve to its closed position. Also, close the cooling valve if the air handler is in heat mode, the supply fan is off, or the discharge air temperature sensor is failed.

Heating valve

Modulate the heating valve to maintain the discharge air temperature at the discharge air heating setpoint. Close the heating valve if the air handler is in cool mode, the supply fan is off, or the discharge air temperature sensor is failed. Open the heating valve if the supply fan is off and the outdoor air temperature falls below the adjustable freeze avoidance setpoint.

Alarms

In addition to the alarm requirements mentioned above, indicate an alarm at the operator display and turn on the alarm output when any sensor fails. Alarms must be resettable at the operator display. Diagnostic conditions include the following:

- Dirty filter
- Duct static pressure high limit
- Exhaust fan failure
- Low outdoor air temperature
- Low (mixed air) temperature detection
- Sensor failure (including discharge air temperature, mixed air temperature, outdoor air temperature, space temperature, and duct static pressure)
- Supply fan failure

The following failures shutdown the air handler and require a manual reset:

- Discharge air or mixed air temperature sensor failure
- Fan failure
- High duct static pressure
- Low mixed air temperature

The corresponding data definition is presented in Table 12 on page 109 and Table 13 on page 110, and a wiring diagram is presented in Figure 105 on page 112.

Table 12: VAV AHU inputs and outputs data definition

Inputs and outputs		Type	Name	Notes
Inputs	1	Analog	Space Temp	Universal input configured as thermistor or RTD
	2	Analog	Thumbwheel SP	Universal input configured as thermistor or RTD
	3	Analog	Mixed Air Temp	Universal input configured as thermistor or RTD
	4	Analog	Discharge Air Temp	Universal input configured as thermistor or RTD
	5	Analog	Outdoor Air Temp	Universal input configured as thermistor
	6	—	Not Used	
	7	Binary	Low Temp Detect	
	8	Binary	Run/Stop Interlock	
	9	Binary	Occupancy/Generic	
	10	Binary	Supply Fan Status	
	11	Binary	Filter Status	
	12	Binary	Exhaust Fan Status	
	Pressure		Duct Static Pressure	There are no configurable properties for the pressure input.
Binary outputs	1		Supply Fan Start/Stop	
	2		Exhaust Fan Start/Stop	
	3		Not Used	
	4		Not Used	
	5		Not Used	
	6		Alarm Output	Status/custom alarm indicator
Analog outputs	1		Supply Fan Speed	Analog output configured as voltage, 0–10 V
	2		Cooling Valve Position	Analog output configured as voltage, 0–10 V
	3		Heating Valve Position	Analog output configured as voltage, 0–10 V
	4		Face and Bypass Damper*	Analog output configured as voltage, 0–10 V
	5		OA Damper Position	Analog output configured as voltage, 0–10 V
	6		Not Used	

* This output is not used in this program but is often used in air-handler programs.

Note: The inputs and outputs in this table are configured so that a Tracer AH540 main logic board could be replaced with a Tracer MP580/581 board, except for the Mixed Air Temp, which is input 6 on the Tracer AH540.

Table 13: VAV AHU variables data definition

Variable	Name	Notes
Tracer Summit binary variables	Not Used	
Local binary variables	Supply Fan Failure	Sourced from a program
	Exhaust Fan Failure	Sourced from a program
	Sensor Failure	Sourced from a program
	Duct Static High Limit	Sourced from a program
	Low OA Temp Alarm	Sourced from a program
	Fan Failure Reset	Sourced from operator display and a program
	Alarm Reset	Sourced from operator display and a program
	Manual Reset Alarm	Sourced from a program
	Info Alarm Active	Sourced from a program
	Wired Setpoint Enable	Sourced from operator display
	Economizer Enable	Sourced from operator display
	Heat/Cool Mode	Sourced from a program
	OK to Economize	Sourced from a program
	Maintenance Required	Sourced from a program
	Maintenance Timer Reset	Sourced from operator display and a program
Tracer Summit analog variables	Not Used	
Local analog variables	Space Temp SP (OD)	Setpoint sourced from the operator display
	Eff Occ Cool SP	Sourced from a program
	Eff Occ Heat SP	Sourced from a program
	Effective Cool SP	Sourced from a program
	Effective Heat SP	Sourced from a program
	DA Cooling SP	Sourced from a program
	DA Heating SP	Sourced from a program
	Low OA Temp SP	Setpoint sourced from the operator display
	OA Damper Min Pos	Setpoint sourced from the operator display

Before you write the program, configure these inputs, outputs, and variables in your Tracer MP580/581 controller. Also, because you are programming a VAV air-handling unit, set the controller to use the LonMark® Discharge Air Controller (DAC) profile. The DAC profile provides the configuration data listed in Table 14.

Table 14: Network configuration inputs for the DAC profile

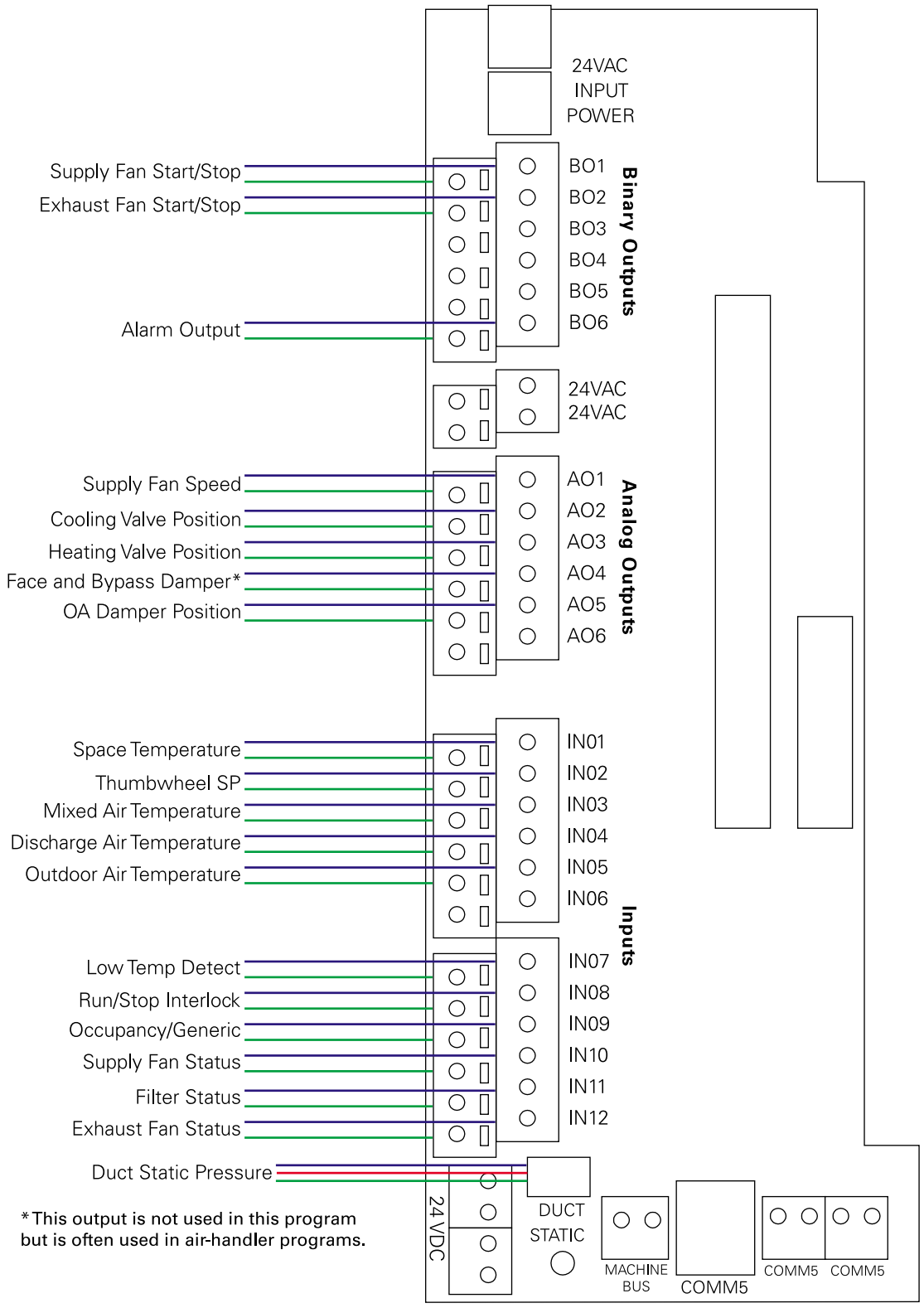
nci	SNVT	Notes
nciBypassTime	SNVT_time_min	Occupied bypass time
nciDACISP	SNVT_temp_p	Discharge air cooling setpoint
nciDAHtSP	SNVT_temp_p	Discharge air heating setpoint
nciDuctStatSP	SNVT_press_p	Duct static pressure setpoint
nciDuctStatLim	SNVT_press_p	Duct static pressure limit
nciBldgStaticSP	SNVT_press_p	Building static pressure setpoint

Table 14: Network configuration inputs for the DAC profile (continued)

nci	SNVT	Notes
nciMALowLimitSP	SNVT_temp_p	Mixed air low limit setpoint
nciOATSP	SNVT_temp_p	Outdoor air temperature (economizer changeover) setpoint
nciSetpoints	SNVT_temp_setpt	Provides default unoccupied, occupied, and occupied standby cooling and heating setpoints

Then make sure that the TGP editor is open and ready to go.

Figure 105: VAV AHU wiring diagram



Determining a programming approach

As mentioned previously, programming instructions in this chapter are in a modular form. Also, remember that this sequence of operation could result in a large number of variations in program content and programming technique.

From the sequence of operation, determine the basic control functions required. In this case, the following control functions must be addressed.

- Effective space setpoint calculation
- Discharge air setpoint validation
- Mode determination
- Supply fan control
- Duct static pressure control
- Exhaust fan control
- Mixed air/outdoor air damper control
- Cooling valve control
- Heating valve control
- Alarm management

Not every control function fits into one program. Determine the number of programs required by grouping control functions together that require the same information, and/or the same run frequency as shown in Table 15.

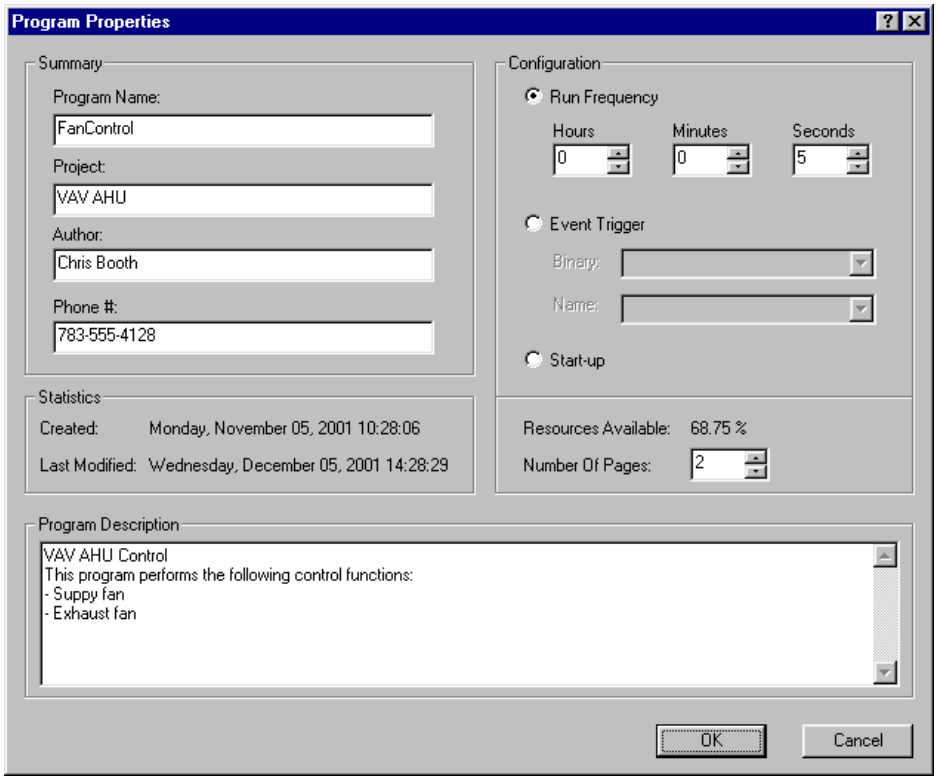
Table 15: Control functions within each program

Program name	Control functions
FanControl	Supply fan control
	Duct static pressure control
	Exhaust fan control
DischargeAirControl	Mixed air/outdoor air damper control
	Cooling valve control
	Heating valve control
Alarms	Alarm management
ModeAndSetpoints	Effective space setpoint calculation
	Discharge air setpoint validation
	Mode determination

Writing the fan control program

Create a new program and setting its properties as shown in Figure 106.

Figure 106: Fan control program properties



When complete, this program performs the following tasks:

- Supply fan control
- Duct static pressure control
- Exhaust fan control

Controlling the supply fan

Control the supply fan on and off according to the sequence of operation. Turn the supply fan on when the unit is occupied. Turn the supply fan off when *any* of the following is true:

- The unit is unoccupied.
- The run/stop interlock is open.
- The low temperature detection is closed.
- The supply fan status indicates a fan failure after a 1-minute delay.
- The duct static pressure exceeds 4 in. wg.

Starting and stopping the supply fan follows a pattern similar to starting and stopping the condenser water pump on the cooling tower. Use the module shown in Figure 107 on page 116 to control the supply fan.

- Use the Occupancy and the De-Enumerator blocks to interpret the occupancy mode of the controller.

➤ *Using the Occupancy block*

Use the Occupancy block to provide a program with the occupancy state of the controller. The output of the Occupancy block is the result of arbitration among various schedule-related inputs and results in the appropriate enumerated value for the current occupancy state. Enumerations of the Occupancy block output are as follows:

0 = Occupied
1 = Unoccupied
2 = Occupied bypass
3 = Occupied standby

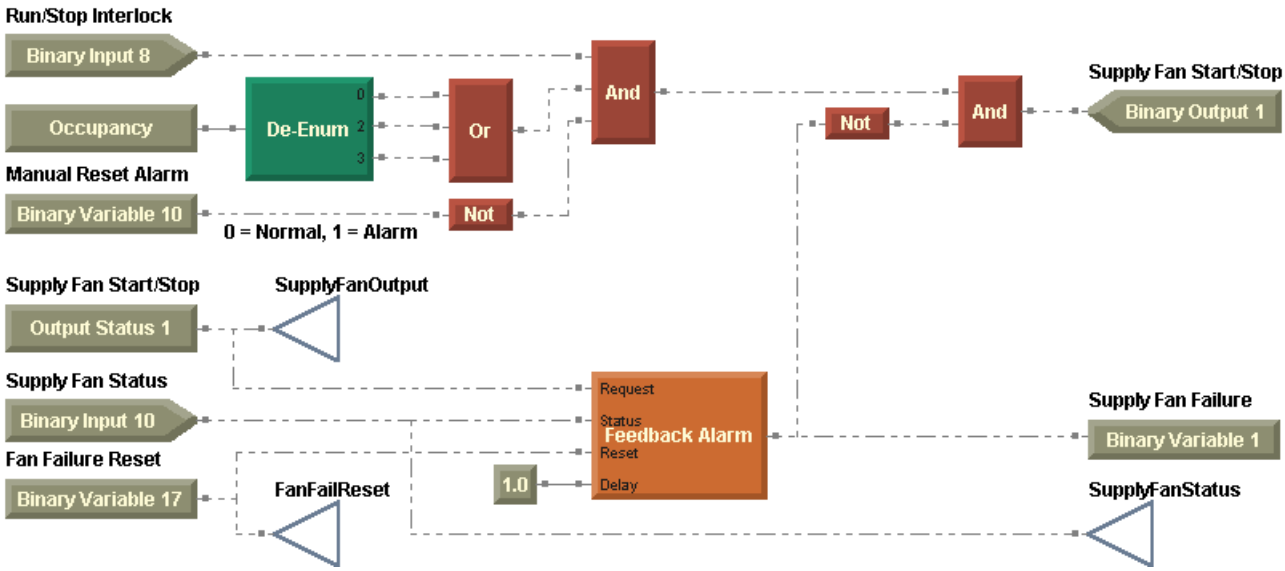
For more information about the Occupancy block, see the *Tracer MP580/581 Programmable Controller Programming* guide (CNT-SVP01A-EN).

➤ *Using the De-Enumerator block*

Use the De-Enumerator block to translate an analog enumeration to one of several possible binary output values. If the analog value input is equal to one of the enumerations for which the block tests, the appropriate binary output is set to true. All other binary outputs are set to false. Use the properties dialog box to select the configuration and specify the required outputs.

- Use the binary variable, Manual Reset Alarm, to carry the alarm status associated with manual reset alarm conditions, including low mixed air temperature and high duct static pressure. If this alarm is active, it prevents the fan from starting.
- Use the Feedback Alarm block to start the fan and confirm fan status. A fan failure occurs if status cannot be confirmed after a 1-minute delay.
- Connect the actual Output Status block to the Feedback Alarm block. Use the output status to allow overrides of the fan without indicating a fan failure.

Figure 107: Supply fan control



Controlling the duct static pressure

After starting the supply fan, modulate the supply fan speed to maintain the duct static pressure setpoint, 1.5 in. wg (adjustable). Use a PID loop, as shown in Figure 108.

- Use a Network Configuration Input (nci) block to access the duct static pressure setpoint associated with the DAC profile.

➤ ***Using the Network Configuration Input (nci) block***

Use a network configuration input (nci) in a program with this block. Select the specific nci in the properties dialog box.

- In the duct static pressure control programming, set the PID loop properties as shown in Figure 109 on page 117.

Figure 108: Duct static pressure control

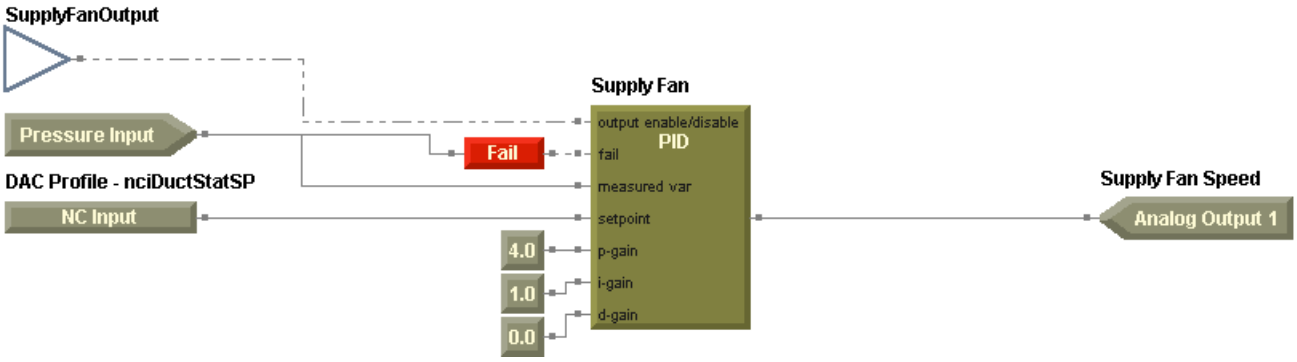
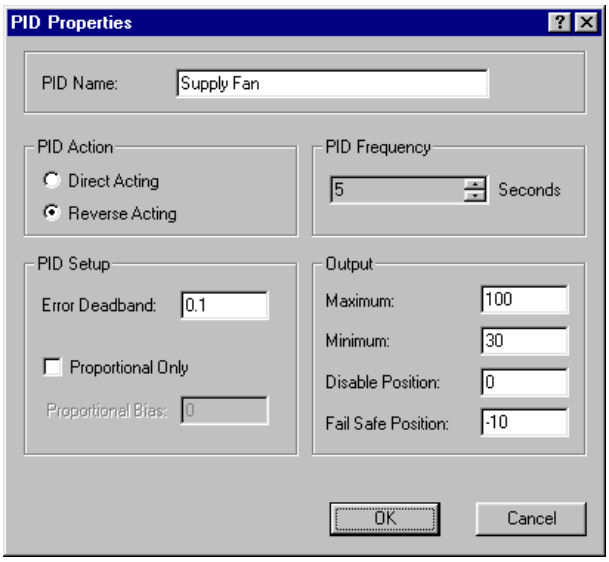


Figure 109: Supply fan PID properties



Controlling the exhaust fan

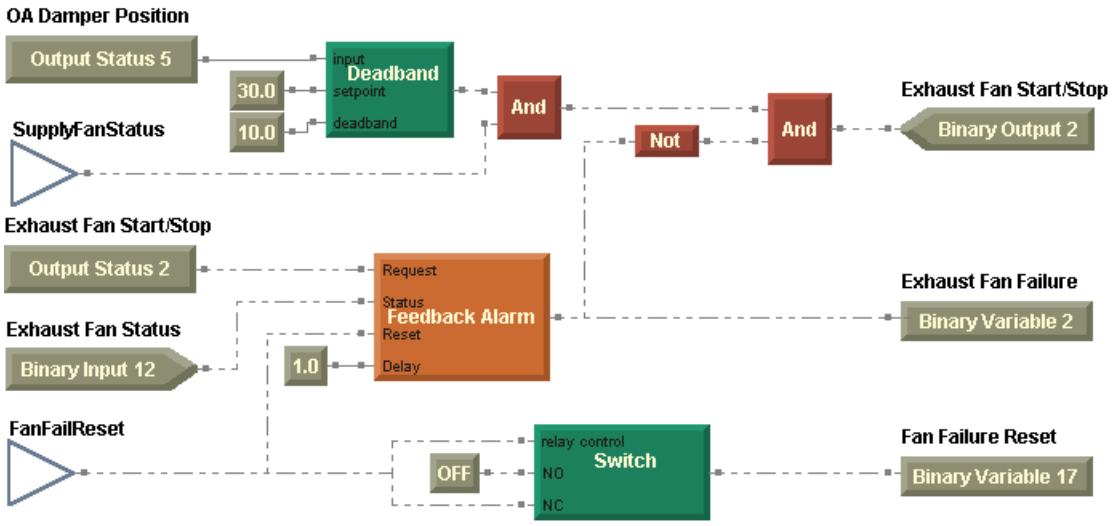
This program also controls the exhaust fan. From the sequence of operation, you know to:

- Turn the exhaust fan on when the supply fan is on and the outdoor air damper is open beyond 30%.
- Turn the exhaust fan off when the outdoor air damper closes to below 20% open or the supply fan is turned off.

Start and stop the fan with a deadband as shown in Figure 110 on page 118.

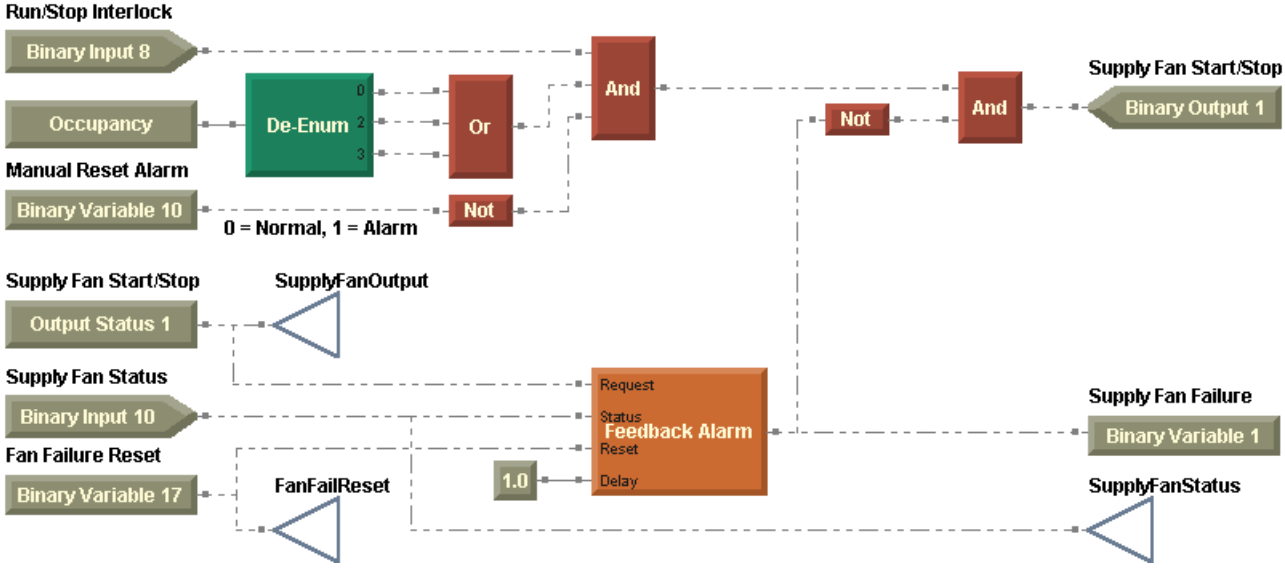
- Use a Deadband block to respond to the position of the outdoor air damper.
- Use the Feedback Alarm block to start the fan and confirm fan status. Implement a fan failure if status cannot be confirmed after a 1-minute delay.
- Connect the actual Output Status block to the Feedback Alarm block. Use the output status to allow overrides of the fan without indicating a fan failure.
- Use the Switch block to set the binary variable, Fan Failure Reset, to off after it has been set to on.

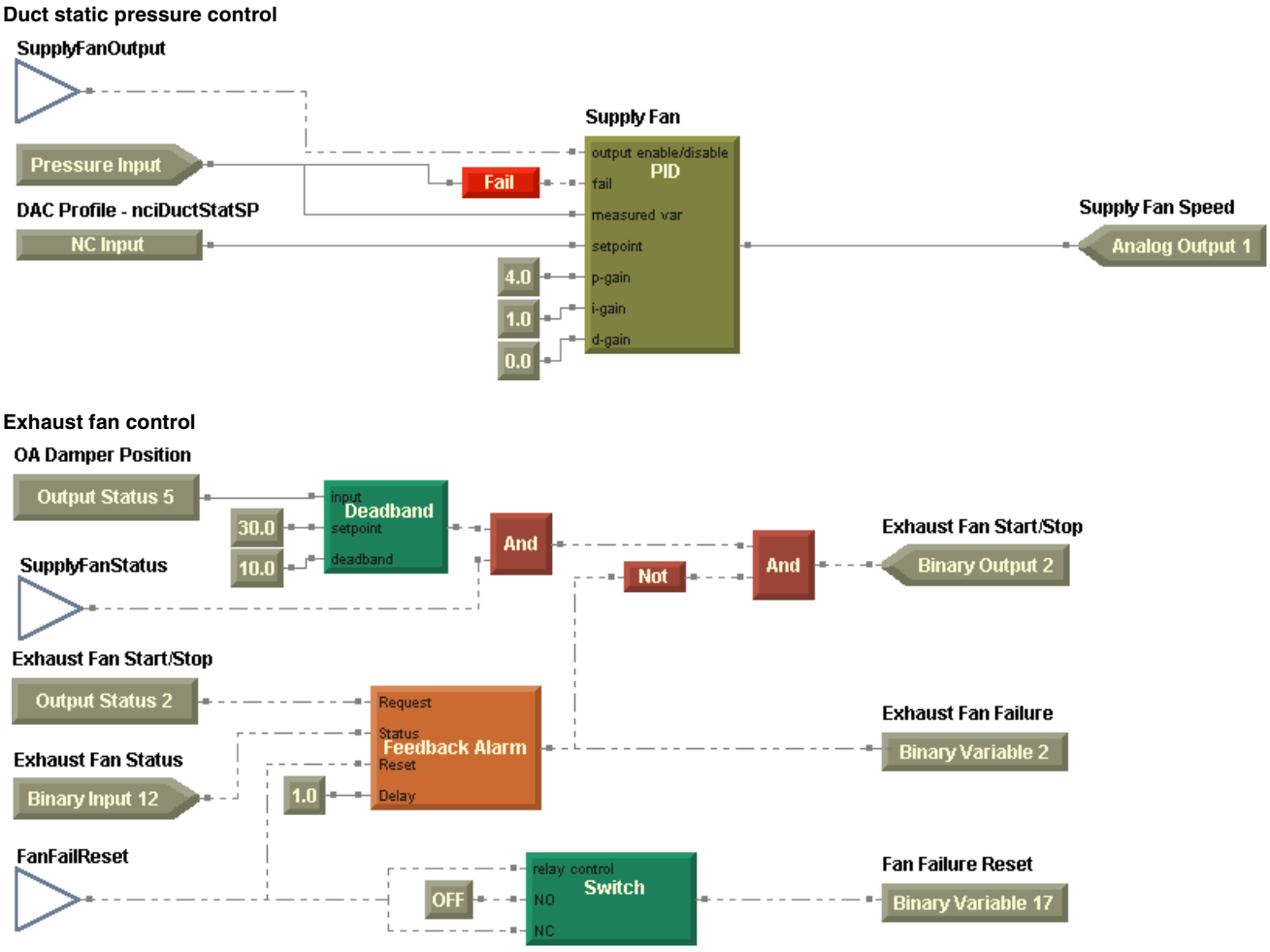
Figure 110: Exhaust fan control



Compile and download the fan control program (Figure 111) to your Tracer MP580/581 controller.

Figure 111: Complete fan control program





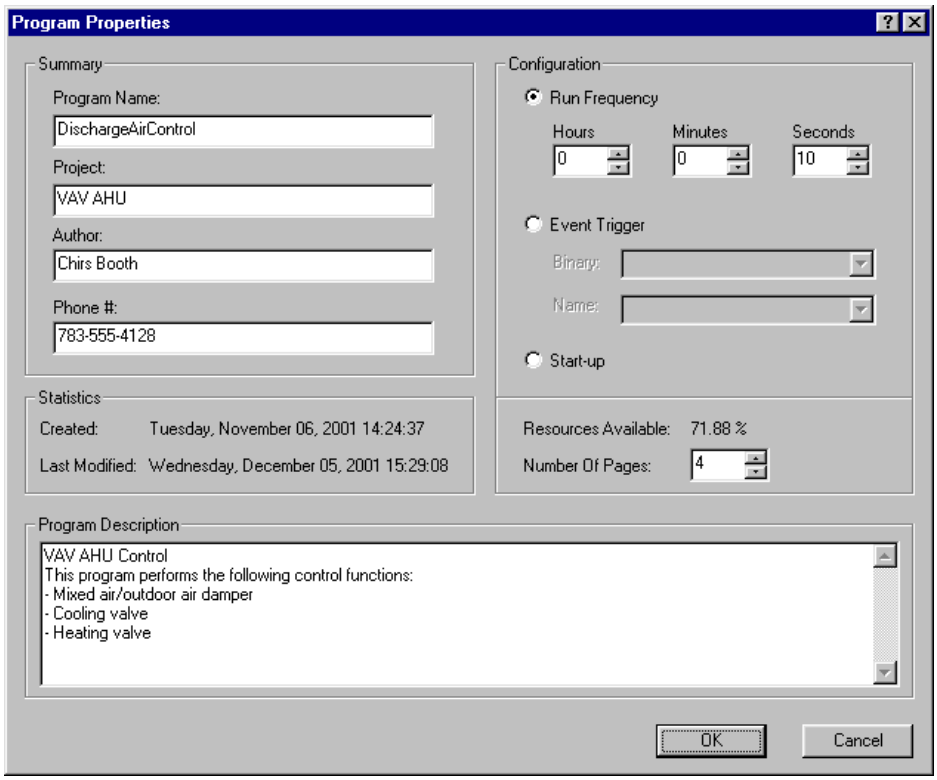
Writing the discharge air control program

Write a program to control the discharge air temperature that performs the following tasks:

- Mixed air and outdoor air damper control
- Cooling valve control
- Heating valve control

Create a new program and set its properties as shown in Figure 112 on page 120.

Figure 112: Discharge air control program properties



Controlling the mixed air and outdoor air damper

Operation of the outdoor air damper and subsequent mixed air temperature control must satisfy a number of scenarios according to the sequence of operation.

Modulate the outdoor air damper between the adjustable minimum position and fully open to maintain the discharge air cooling setpoint when *all* of the following are true:

- The economizer is enabled.
- The outdoor air temperature is less than the economizer changeover (or outdoor air temperature) setpoint.
- The air handler is in cooling mode.

Modulate the outdoor air damper closed, overriding the minimum position, to maintain the mixed air temperature at or above the mixed air setpoint.

Control the outdoor air damper to its minimum position when *either* of the following is true:

- The economizer function is disabled.
- The air handler is in heating mode.

Close the outdoor air damper completely when *any* of the following is true:

- The unit is in unoccupied mode.
- The outdoor air temperature falls below the low setpoint for the outdoor air temperature.
- The supply fan is off.
- The sensor for the mixed air temperature is failed.

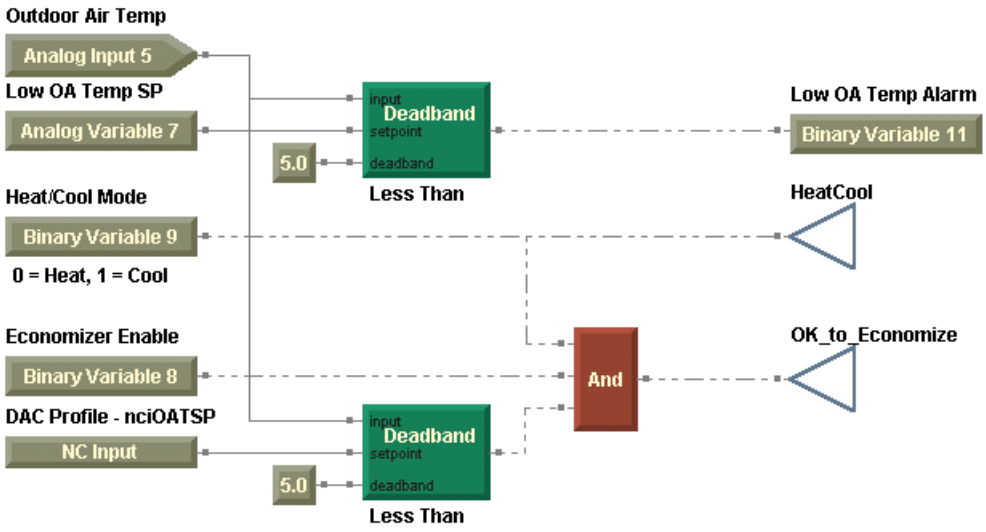
Determining whether to economize

Because the control of the outdoor air damper is potentially complex, you may simplify the program by separating the decision to economize from the actual control of the damper.

Complete the first part of this program, as shown in Figure 113, to determine some factors important to control of the outside air damper.

- Use a Deadband block to check the outside air temperature against the low outside air temperature setpoint. You will use this information later in the program.
- Use a second Deadband block to check the outside air temperature against the economizer changeover, or outside air temperature, setpoint. Combine the economizer enable setting and the heat/cool mode with the result of this deadband to determine whether conditions warrant economizer operation.
- Use a Network Configuration Input (nci) block to access the outdoor air temperature setpoint associated with the DAC profile.

Figure 113: Determining whether to economize

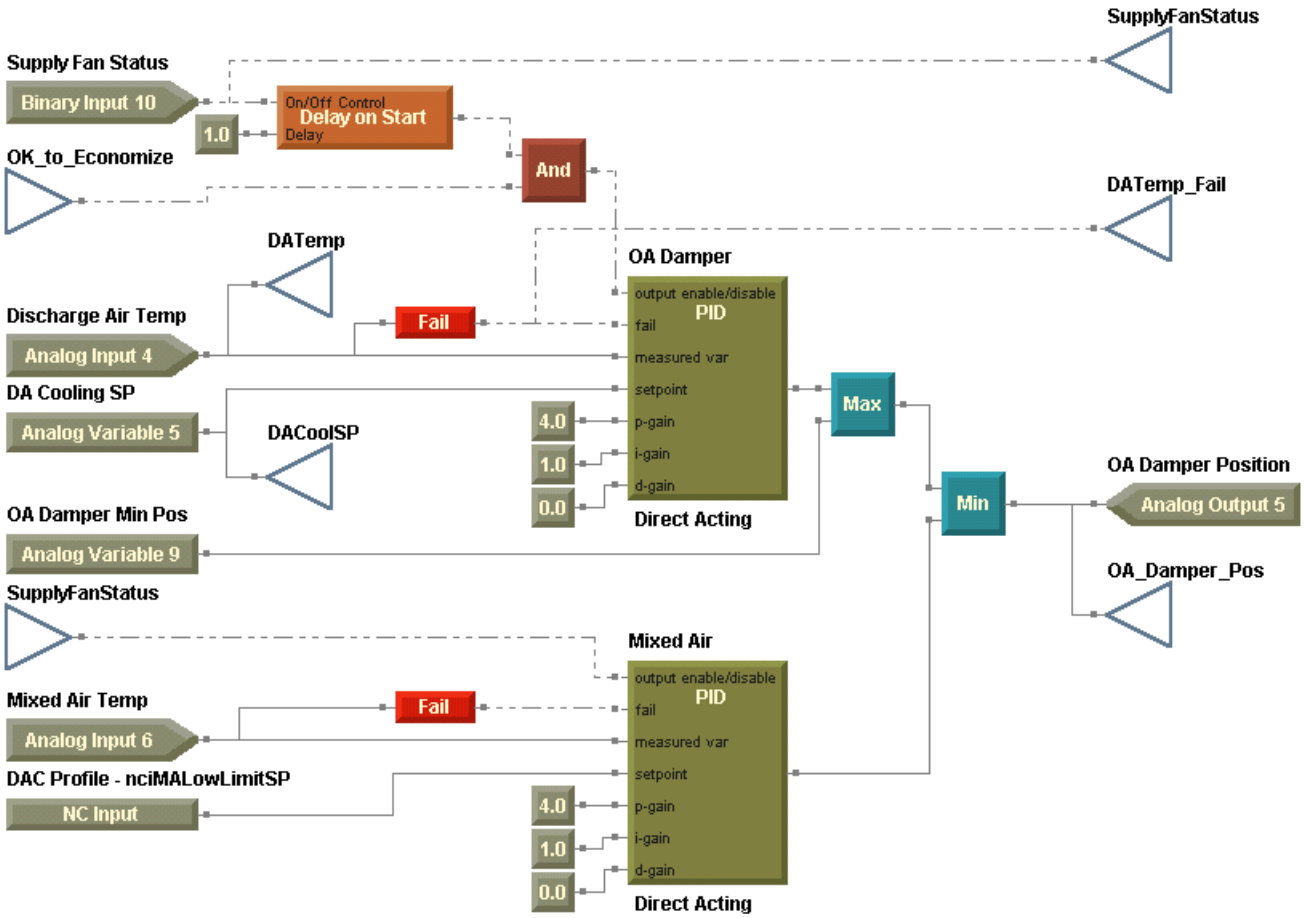


Determining the outdoor air damper position

The next piece of the puzzle involves determining the required position of the outdoor air damper. Use the TGP module in Figure 114.

- Use a Delay on Start block to implement a 1-minute delay prior to allowing the outdoor air damper to modulate.
- Use a PID loop to control the outdoor air damper to maintain the discharge air setpoint when economizing is permitted.
- Add a second PID loop as backup protection to monitor the mixed air temperature. It determines the necessary outdoor air damper positions to maintain the mixed air temperature at the mixed air temperature (low limit) setpoint.
- Use a Network Configuration Input (nci) block to access low limit setpoint for the mixed air associated with the DAC profile.
- Compare the calculated positions from the PID loops. And allow mixed air control to overrule discharge air control when necessary to prevent freezing of the heating or cooling coils.

Figure 114: Controlling outdoor air damper position



Controlling the cooling valve

Take a closer look at the sequence of operation with regard to the cooling valve.

Modulate the cooling valve to maintain the discharge air temperature at the discharge air cooling setpoint when *all* of the following are true:

- The supply fan is on.
- The air handler is in cooling mode.
- The economizer function is not enabled.

Or when *all* of the following are true:

- The supply fan is on.
- The air handler is in cooling mode.
- The economizer function is enabled.
- The outdoor air damper is open at least 90%.

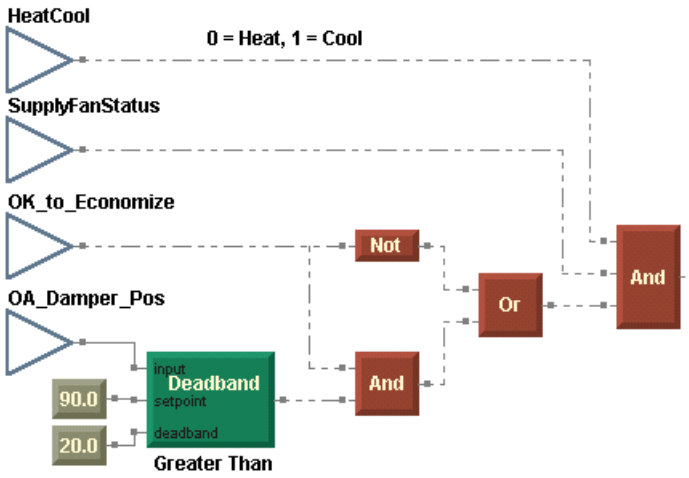
Close the cooling valve when *any* of the following is true:

- The air handler is in heating mode.
- The supply fan is off.
- The discharge air temperature sensor is failed.

It may help to think of control of the cooling valve in two parts. First, program the decision to operate the cooling valve as shown in Figure 115.

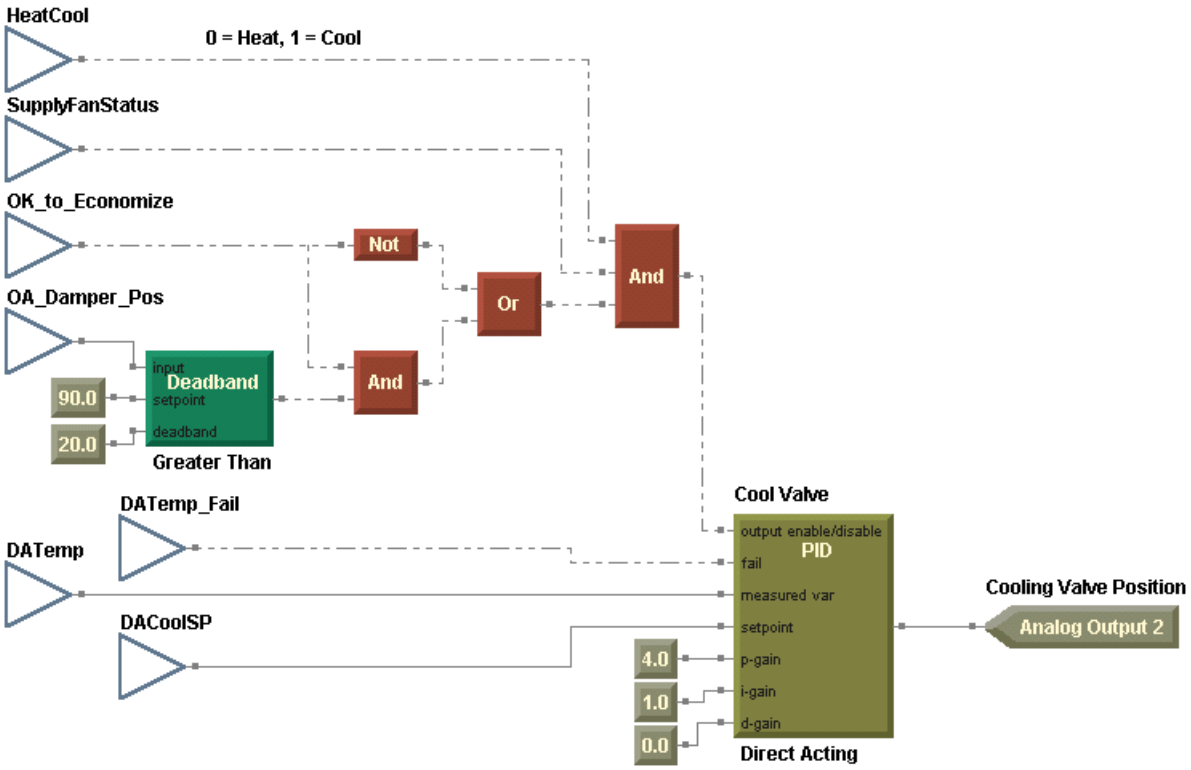
- Use a combination of logic blocks to determine whether the cooling valve operates.
- Apply the output of the economizer decision here.
- Use a Deadband block and the position of the outdoor air damper.

Figure 115: Operate the cooling valve?



The output of this decision feeds into the actual control of the cooling valve using a PID loop (Figure 116 on page 124).

Figure 116: Controlling the cooling valve



Controlling the heating valve

Consider the following specifications from the sequence of operation when writing the module for the heating valve.

Modulate the heating valve to maintain the discharge air temperature at the discharge air heating setpoint when *both* of the following are true:

- The supply fan is on.
- The air handler is in heating mode.

Open the heating valve completely when *both* of the following are true:

- The supply fan is off.
- The outdoor air temperature falls below the adjustable freeze avoidance setpoint.

Close the heating valve when *any* of the following is true:

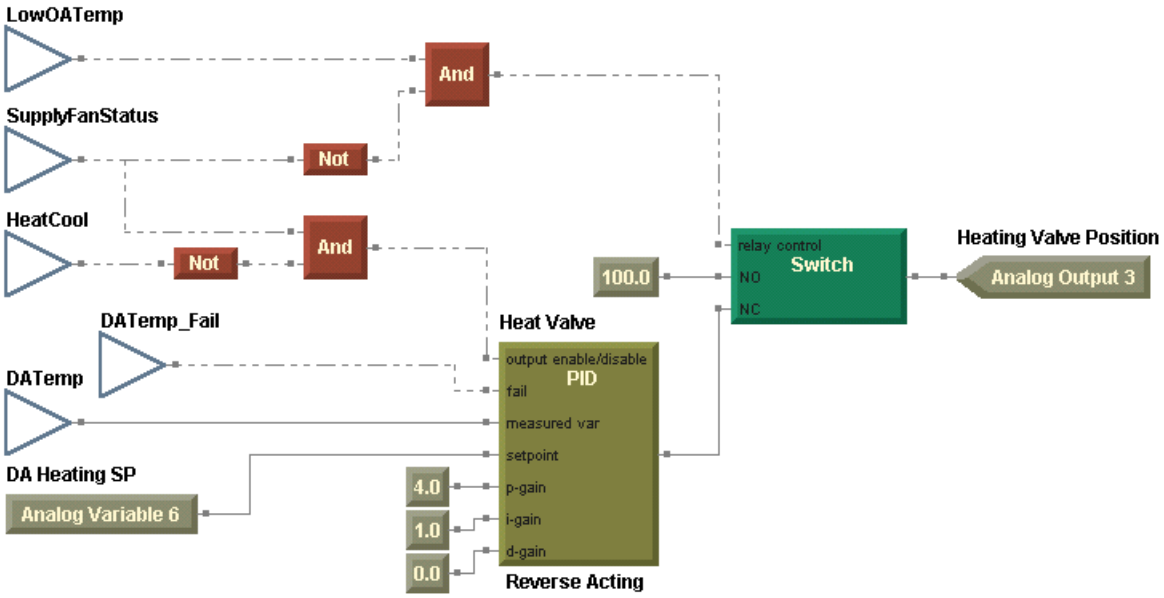
- The air handler is in cooling mode.
- The supply fan is off.
- The discharge air temperature sensor is failed.

Like the cooling valve, the control of the heating valve consists of two parts: the decision and the control. However, a third requirement calls for the heating valve to open when the outdoor air temperature falls below the adjustable, low setpoint for the outdoor air temperature.

Complete the module in Figure 117 to control the heating valve.

- Use logic blocks to implement the decision to operate the heating valve.
- Add a PID loop to control the heating valve output.
- When the outdoor air temperature falls below the adjustable, low outdoor air temperature setpoint, use a Switch block to control the output to its fully open position.

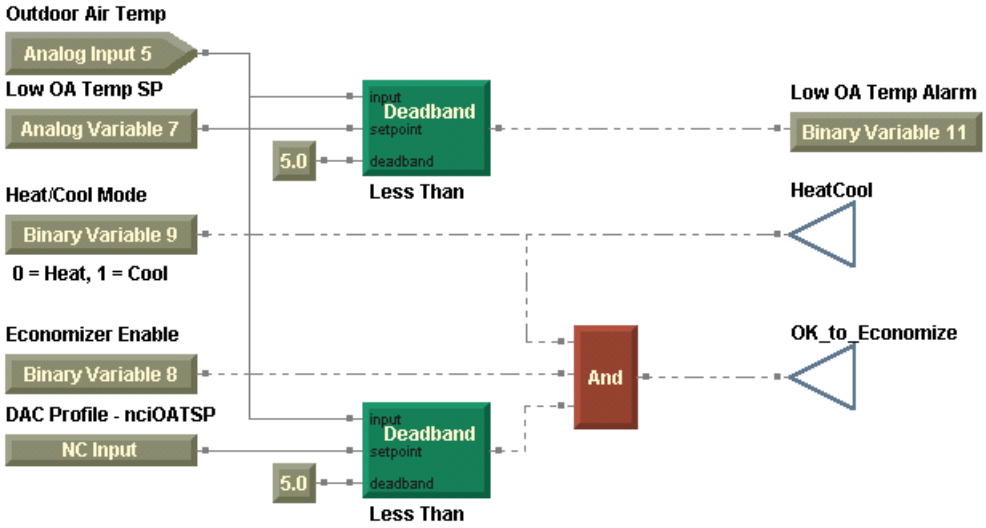
Figure 117: Controlling the heating valve



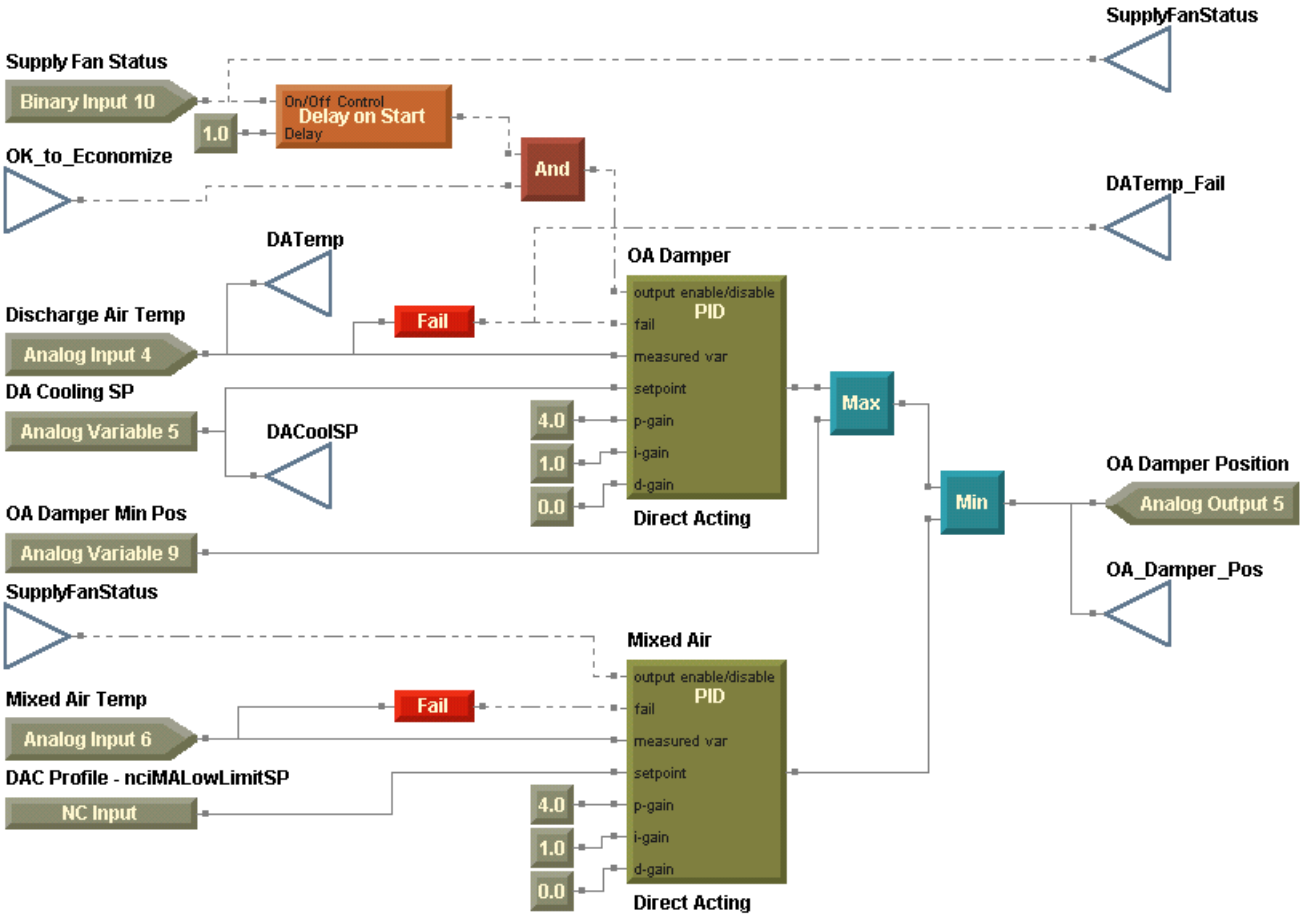
The heating valve module completes the discharge air control program as shown in Figure 118 on page 126. Compile and download the program to your Tracer MP580/581 controller.

Figure 118: Complete discharge air program

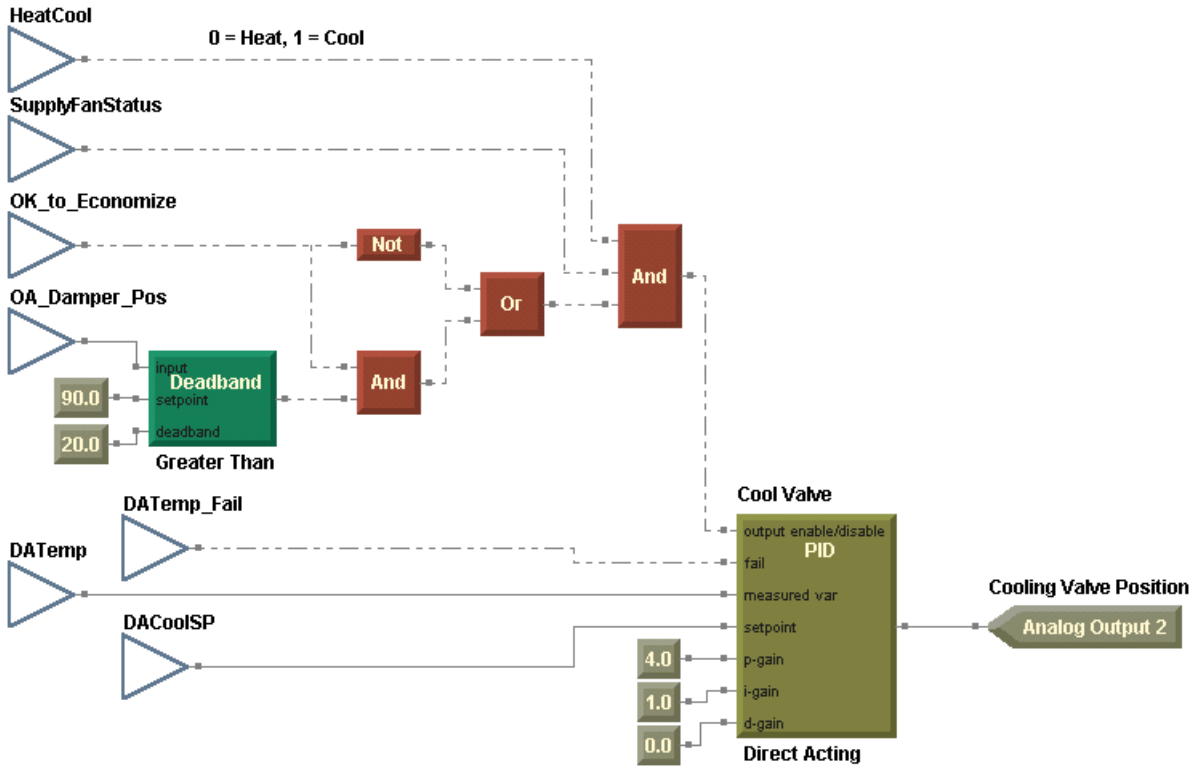
Determining whether to economize



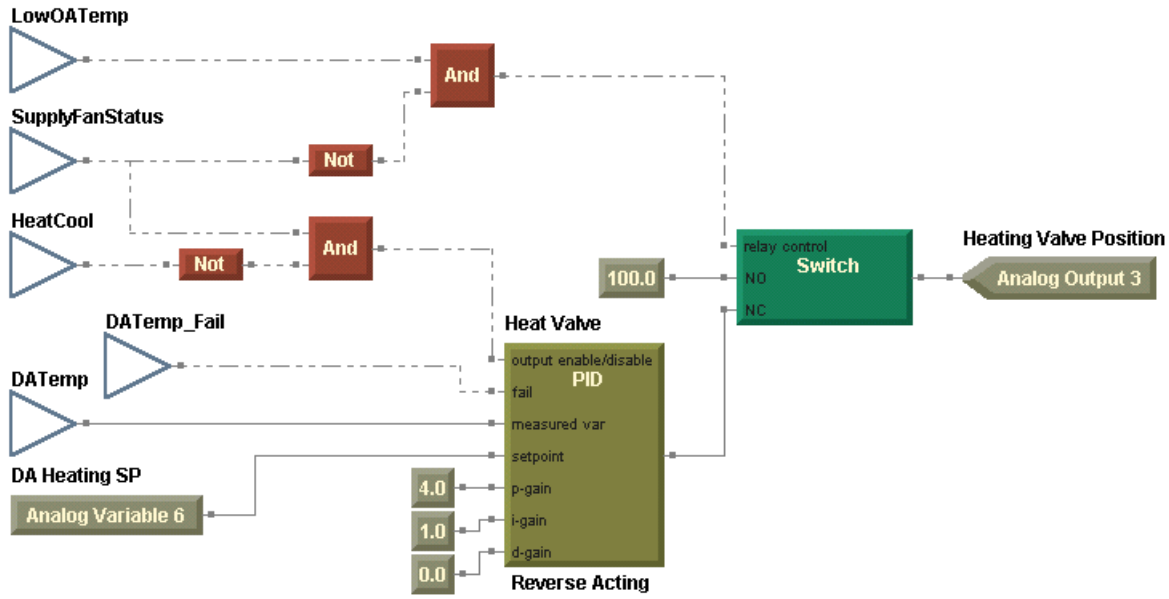
Controlling outdoor air damper position



Controlling the cooling valve



Controlling the heating valve



Writing the alarms program

The alarms program indicates diagnostic conditions to the operator and protects the equipment from potential harm when such a condition exists. According to the sequence of operation, the following diagnostic conditions require an alarm indication:

- Dirty filter
- Duct static pressure high limit
- Exhaust fan failure
- Low outdoor air temperature
- Low (mixed air) temperature detection
- Sensor failure (including discharge air temperature, mixed air temperature, outdoor air temperature, space temperature, and duct static pressure)
- Supply fan failure

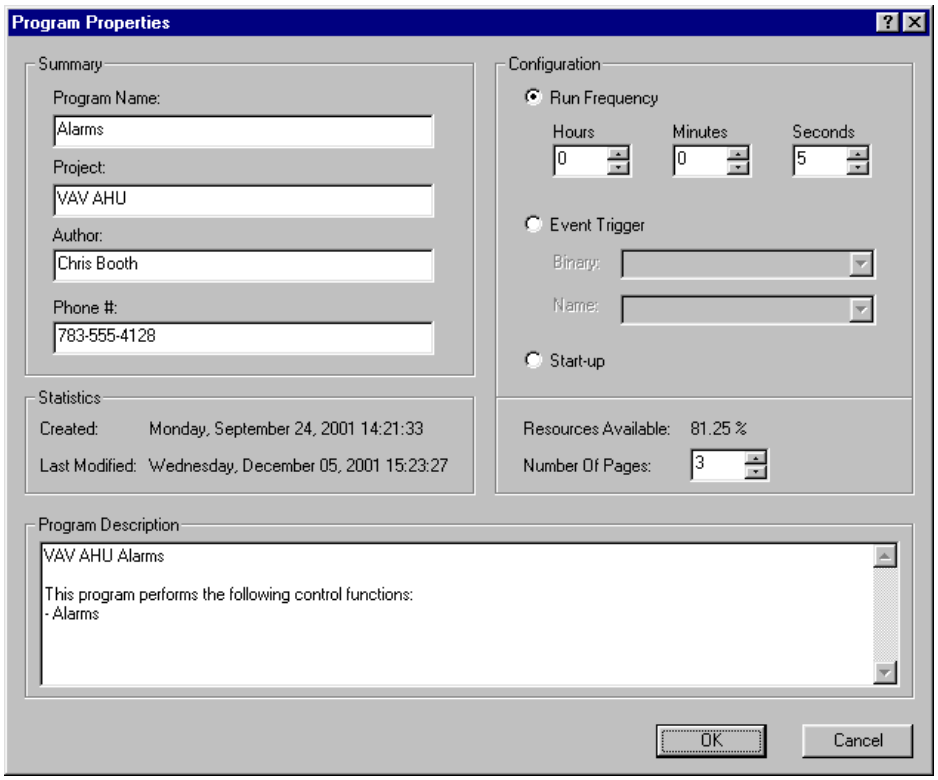
The following failures shutdown the air handler and require a manual reset:

- Discharge air or mixed air temperature sensor failure
- Fan failure
- High duct static pressure
- Low mixed air temperature

All alarms must be resettable at the operator display.

First, set the program properties as shown in Figure 119 on page 129.

Figure 119: Alarms program properties



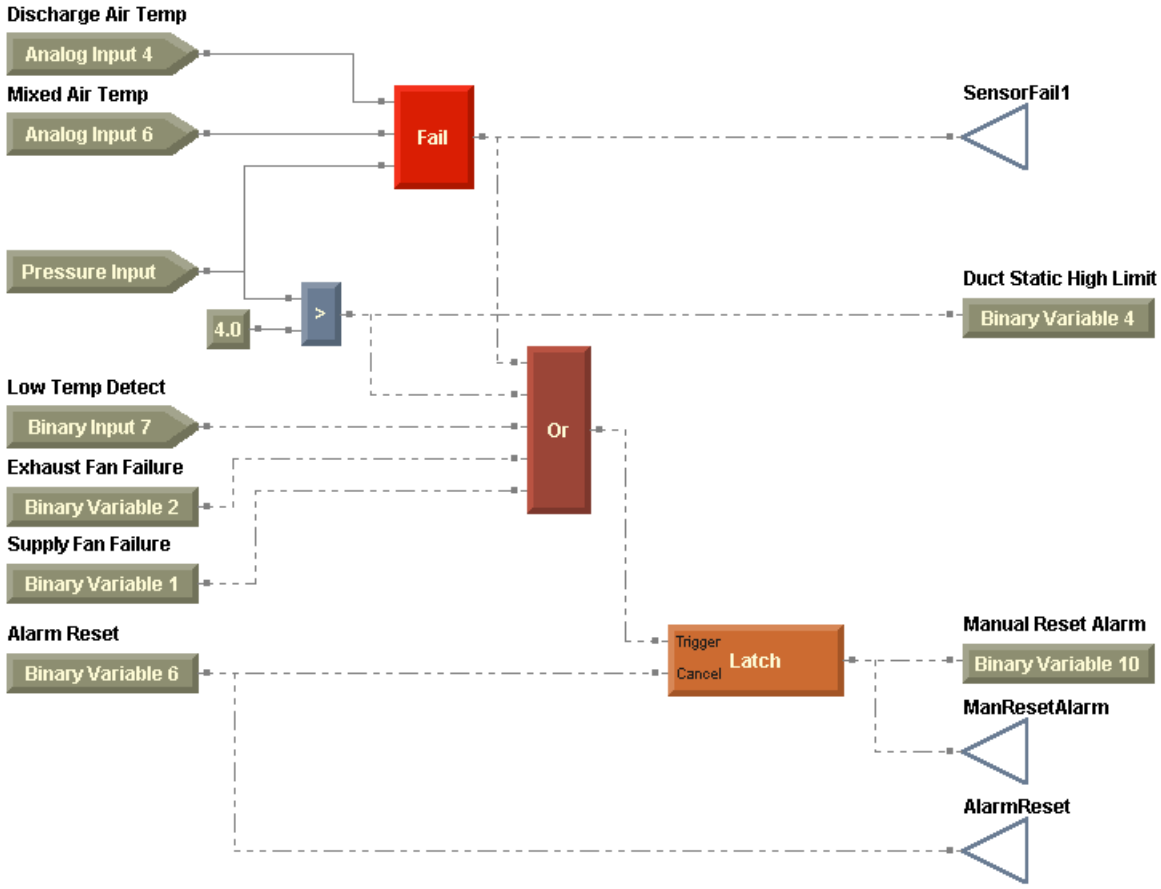
Note that some alarms require the air handler to shut down, and others simply require an alarm indication. An alarm reset at the operator display resets all alarms. Taking this into consideration, create the following three modules within the alarms program:

- Manual reset alarms
- Auto-reset alarms
- Alarm indication and reset

Indicating manual reset alarms

Complete the first module to accommodate the diagnostic conditions that require manual reset alarms (Figure 120). Because manual reset alarms require both shutdown and manual reset, use a binary variable, Manual Reset Alarm, to pass the existence of manual reset alarms to other programs. This variable is used in the fan control program to control the supply fan off.

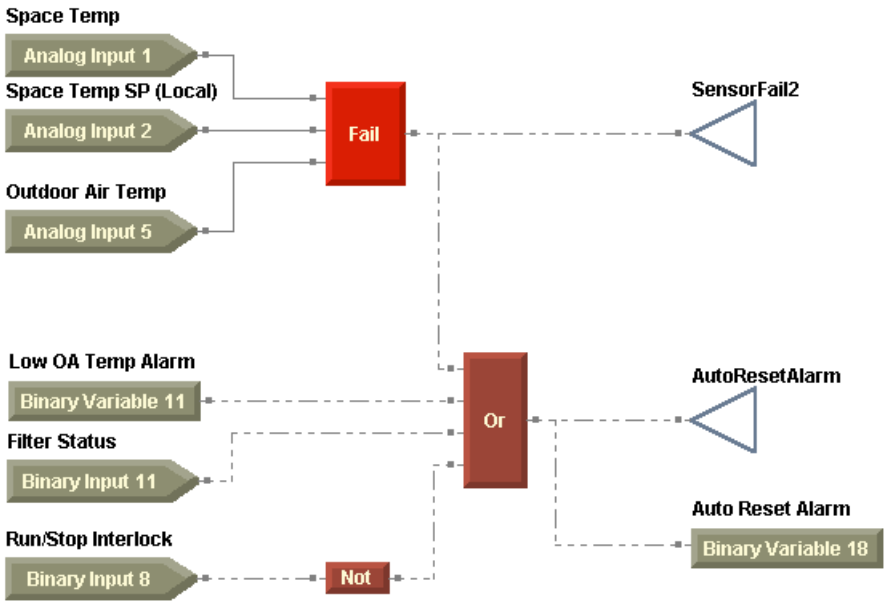
Figure 120: Manual reset alarms



Indicating auto-reset alarms

The remaining diagnostic conditions require auto-reset alarms. Use the TGP module in Figure 121.

Figure 121: Auto-reset alarms

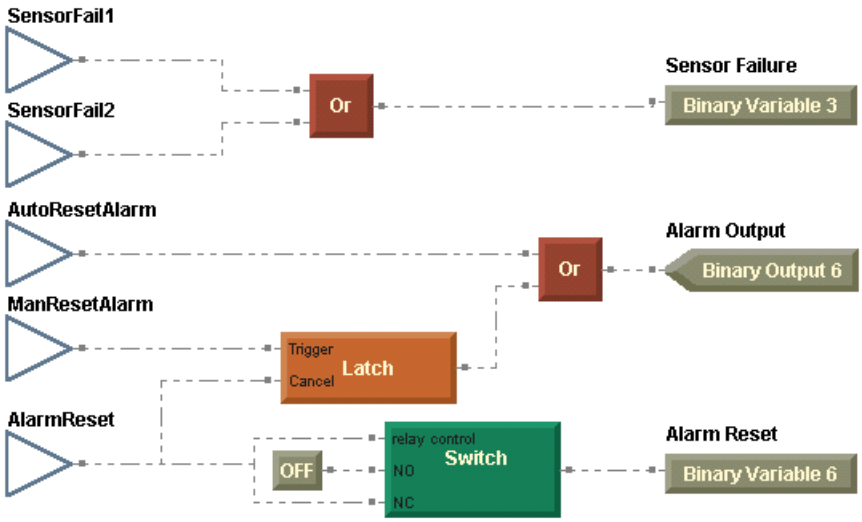


Controlling alarm indication and reset

Based on the status of both auto-reset and manual reset alarms, the last module manages alarm indication and reset (Figure 122 on page 132).

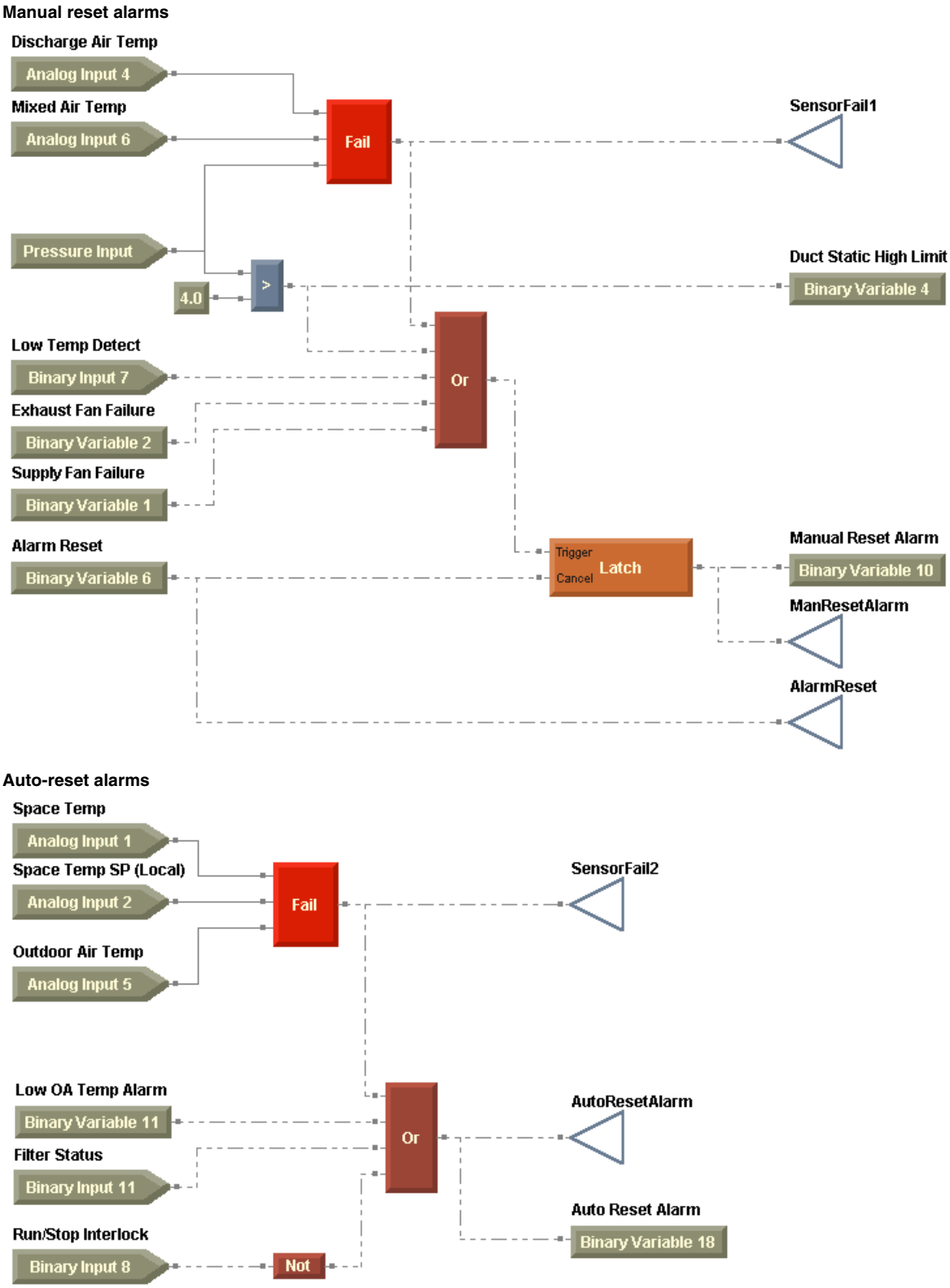
- Use a Latch block to indicate the alarm status.
- Be sure that auto-reset alarms clear automatically, but manual reset alarms require a manual reset.
- Use a Switch block to automatically return the binary variable, Alarm Reset, to a value of off (as presented in the cooling tower program in Chapter 4, “Cooling tower with two-speed fan example”).

Figure 122: Alarm indication and reset

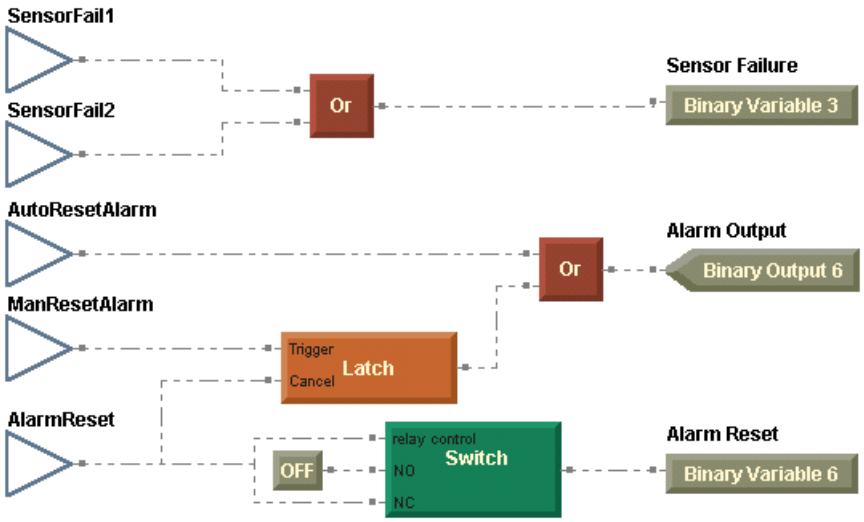


Compile and download your program (Figure 123 on page 133) to the Tracer MP580/581 controller.

Figure 123: Completed alarms program



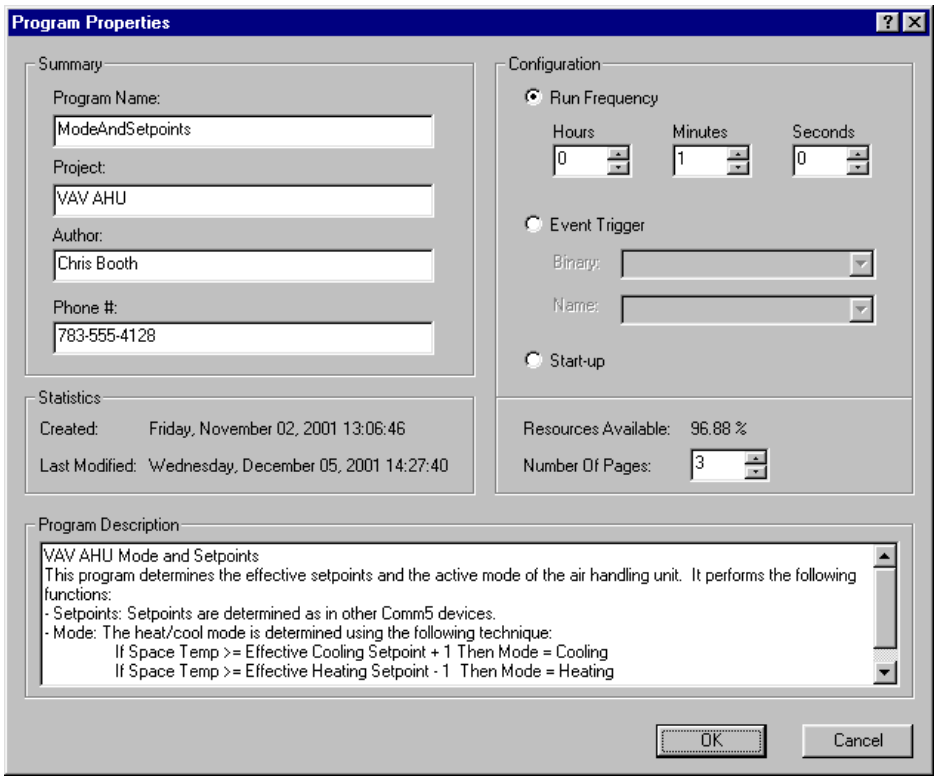
Alarm indication and reset



Writing the mode and setpoints program

The mode and setpoints program calculates setpoints and determines the heat/cool mode. Set the program properties as shown in Figure 124 on page 135.

Figure 124: Mode and setpoints program properties



When complete, this program will perform the following tasks:

- Effective space setpoint calculation
- Discharge air setpoint validation
- Heat/Cool mode determination

Calculating the effective space setpoints

Study the sequence of operation to determine the logic to use to construct your program. In this case, the following logical points apply to setpoint calculation.

- Calculate the effective occupied cooling and heating setpoints as in other Comm5 devices.
- In occupied mode, use the default occupied cooling and heating setpoints to determine an offset.
- In occupied standby mode, use the default occupied standby cooling and heating setpoints to determine an offset.
- In either mode, apply the calculated offset to the space temperature setpoint to determine the effective cooling and heating setpoints.
- Because this is a VAV air-handling unit, controlling the discharge air temperature is not based on the effective occupied cooling and heating setpoints but on the cooling and heating setpoints for the dis-

charge air. Use the effective heating and cooling setpoints to determine heat or cool mode.

To calculate the effective setpoints:

1. Calculate the average of the cooling and heating setpoints.
2. Subtract the average from the cooling setpoint to determine the offset.
3. Add the calculated offset to the space temperature setpoint to determine the effective cooling setpoint.
4. Subtract the calculated offset from the space temperature setpoint to determine the effective heating setpoint.

These are the basic steps in calculating the effective setpoints. See the following sections for more details and instructions.

Calculating the offset values

Some familiarity with how Comm5 devices determine effective setpoints may be helpful here. Use the following six default setpoints to determine offset values:

- Unoccupied cooling
- Occupied standby cooling
- Occupied cooling
- Occupied heating
- Occupied standby heating
- Unoccupied heating

These default setpoints are included in the DAC profile as a network configuration input, nciSetpoints. Determine the offset values using the following relationships.

For occupied mode,

$$Offset_{Occ} = CSP_{Occ} - \frac{(CSP_{Occ} + HSP_{Occ})}{2}$$

where,
CSP_{Occ} = Occupied cooling default setpoint
HSP_{Occ} = Occupied heating default setpoint

And for occupied standby mode,

$$Offset_{OccStby} = CSP_{OccStby} - \frac{(CSP_{OccStby} + HSP_{OccStby})}{2}$$

where,
$CSP_{OccStby}$ = Occupied standby cooling default setpoint
$HSP_{OccStby}$ = Occupied standby heating default setpoint

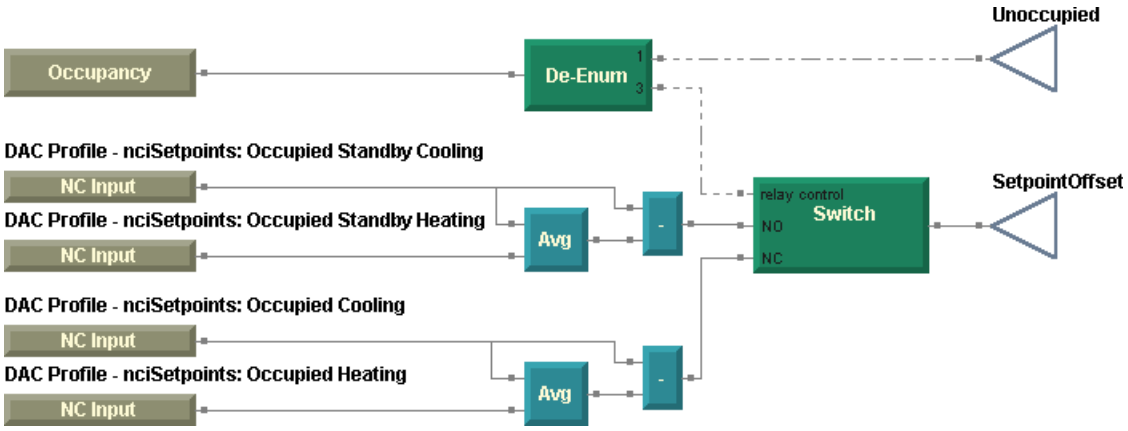
In unoccupied mode, no offset calculation is required. Instead, set the effective cooling and heating setpoints to the default unoccupied cooling and heating setpoints, respectively.

What does this look like in graphical programming? Use the module in Figure 125 on page 137 to complete the first part of the program that cal-

culates the offset values and uses the occupancy mode to determine which offset value to use.

- Use the Occupancy and De-Enumerator blocks to determine the mode: occupied, occupied standby, or unoccupied. In this case, the first output on the De-Enumerator block is true when the Occupancy block yields a value of unoccupied (1). The second output on the De-Enumerator block is true when the Occupancy block yields a value of occupied standby (3).
- Use Network Configuration Input (nci) blocks to access the default setpoints associated with the DAC profile.
- Use the Switch block to switch between the occupied and occupied standby offset values.
- Use wireless connections to transfer the unoccupied status and the final offset value to other parts of the program.

Figure 125: Offset calculation in TGP



Calculating the effective cooling and heating setpoints

Use the calculated offset and the space temperature setpoint to determine the effective occupied cooling and heating setpoints. Remember, the space temperature setpoint is the one setpoint that the building operator uses to adjust the space temperature. The setpoint origin could be any of the following:

- Tracer Summit
- Comm5 network variable (through custom binding)
- Operator display
- Local wired thumbwheel setpoint

In this case, the space temperature setpoint originates at the operator display. Obtain the effective cooling and heating setpoints using the following relationships.

$$CSP_{Effective} = STS + Offset$$

$$HSP_{Effective} = STS - Offset$$

where,

$CSP_{Effective}$ = Effective cooling setpoint
$HSP_{Effective}$ = Effective heating setpoint
STS = Space temperature setpoint

Increasing or decreasing the space temperature setpoint affects the effective occupied and unoccupied setpoints. However, the default setpoints remain constant and are unaffected by changes to the space temperature setpoint. Remember the following about the setpoints calculation:

- For occupied and occupied standby modes, the default setpoints determine the offset between effective cooling and heating setpoints.
- For unoccupied mode, the default and effective setpoints are the same.
- Changes to the space temperature setpoint increase or decrease effective occupied and occupied standby setpoints in a coordinated manner.

Effective setpoint calculation examples

The following tables show an example of effective setpoint calculation. In this example, the setpoints have been assigned the values in Table 16.

Table 16: Default and adjustable setpoint values

Setpoints		
Space temperature setpoint		72.0°F
Default setpoints	Unoccupied cooling	85.0°F
	Occupied standby cooling	78.0°F
	Occupied cooling	74.0°F
	Occupied heating	71.0°F
	Occupied standby heating	67.0°F
	Unoccupied heating	60.0°F

First, calculate the occupied and occupied standby offset values.

$$Offset_{Occ} = 74.0 - \frac{(74.0 + 71.0)}{2} = 1.5$$

$$Offset_{OccStby} = 78.0 - \frac{(78.0 + 67.0)}{2} = 5.5$$

Then use these offset values to calculate the heating and cooling effective occupied and occupied standby setpoints.

$$CSP_{EffectiveOcc} = 72.0 + 1.5 = 73.5$$

$$HSP_{EffectiveOcc} = 72.0 - 1.5 = 70.5$$

$$CSP_{EffectiveOccStby} = 72.0 + 5.5 = 77.5$$

$$HSP_{EffectiveOccStby} = 72.0 - 5.5 = 66.5$$

Table 17 displays the effective setpoint values when the space temperature setpoint is set to 72.0°F

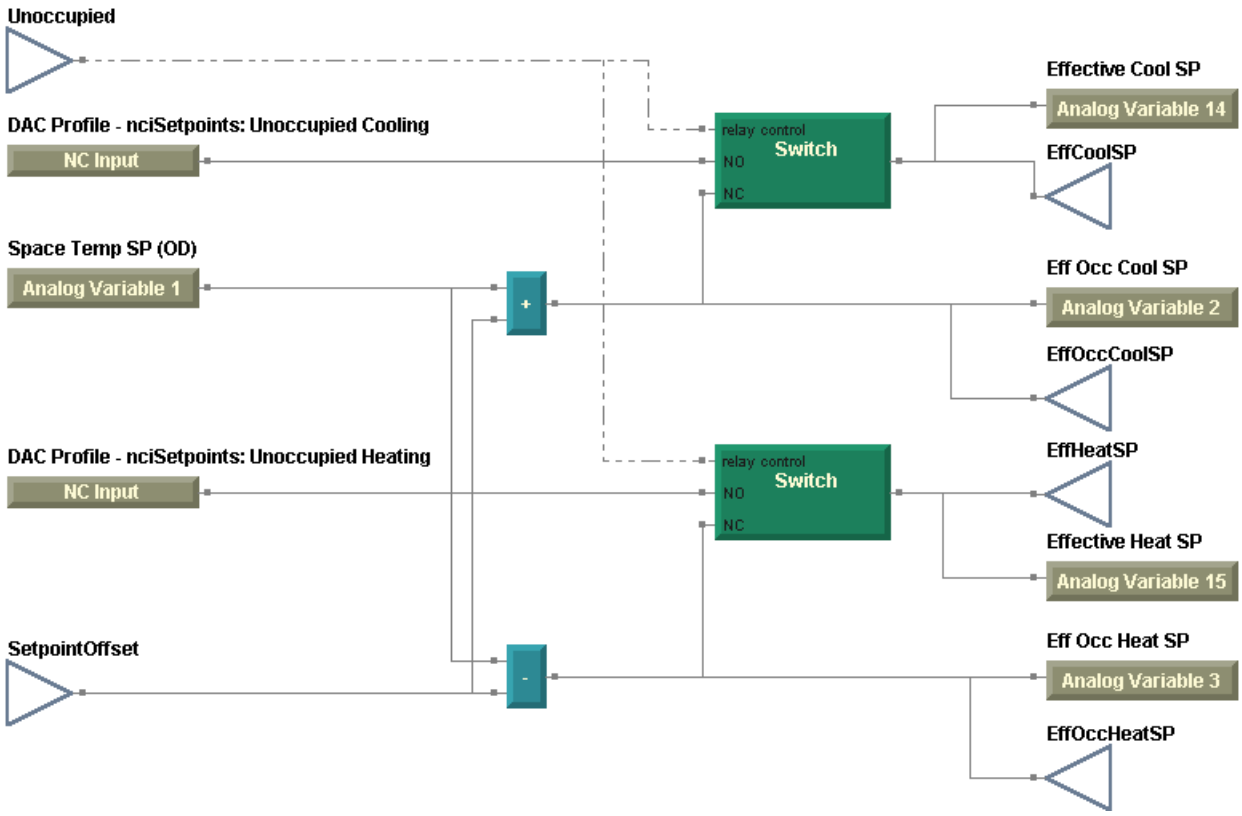
Table 17: Effective setpoint values

Setpoints			Effective setpoints
Space temperature setpoint		72.0°F	
Default setpoints	Unoccupied cooling	85.0°F	85.0°F
	Occupied standby cooling	78.0°F	77.5°F
	Occupied cooling	74.0°F	73.5°F
	Occupied heating	71.0°F	70.5°F
	Occupied standby heating	67.0°F	66.5°F
	Unoccupied heating	60.0°F	60.0°F

Use the module in Figure 126 on page 140 to complete the second part of the program that calculates the effective setpoint values.

- Use the Add block to add the offset to the space temperature setpoint to obtain the effective cooling setpoint.
- Use the Subtract block to subtract the offset from the space temperature setpoint to obtain the effective heating setpoint.
- Use two Switch blocks to determine the effective setpoints based on the occupancy mode. In unoccupied mode, set the effective setpoints to the default unoccupied setpoints. Otherwise, calculate the effective setpoints based on the effective offset and the space temperature setpoint.
- Use Network Configuration Input (nci) blocks to access the unoccupied default setpoints associated with the DAC profile.
- Use wireless connections to transfer the effective setpoints to other parts of the program.

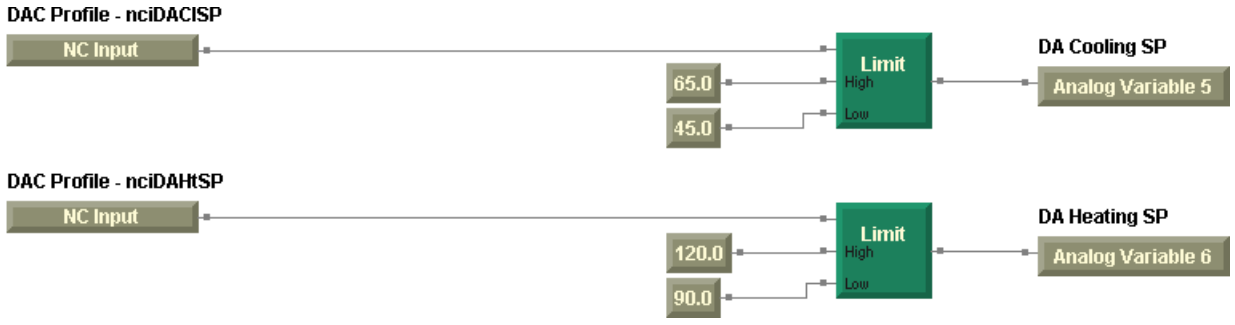
Figure 126: Effective setpoint calculation in TGP



Validating the discharge air setpoints

The effective setpoints provide a basis for the heat/cool mode decision, but they do not provide a basis for discharge air control. The DAC profile provides configured discharge air cooling and heating setpoints in the form of two network variables: nciDAClSP, and nciDAHtSP, respectively. Use the TGP module in Figure 127 to validate the discharge air setpoints.

Figure 127: Validating discharge air setpoints



The Limit block applies constants as high and low limits to the discharge air cooling and heating setpoints. Use variables to store the resultant effective setpoints for use in another program.

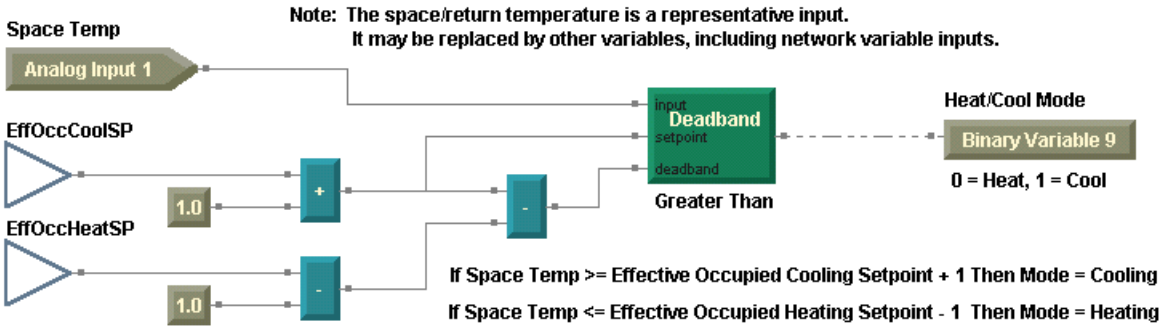
Determining the heat/cool mode

Now that you have all the setpoints you need, start to use them. What do you know from the sequence of operation about determining the active mode, heating or cooling?

- In any occupancy mode, the heat/cool decision is based on the effective occupied or occupied standby cooling and heating setpoints.
- The air handler changes to cooling if the space temperature exceeds the effective cooling setpoint plus 1°F.
- The air handler changes to heating if the space temperature falls below the effective heating setpoint minus 1°F.

Use the TGP module in Figure 128 to make the heat/cool decision. Use a Deadband block to switch between heating and cooling modes and wireless connections to provide the effective cooling and heating setpoints calculated in another part of the program. Store the heat/cool mode in a binary variable for use in another program.

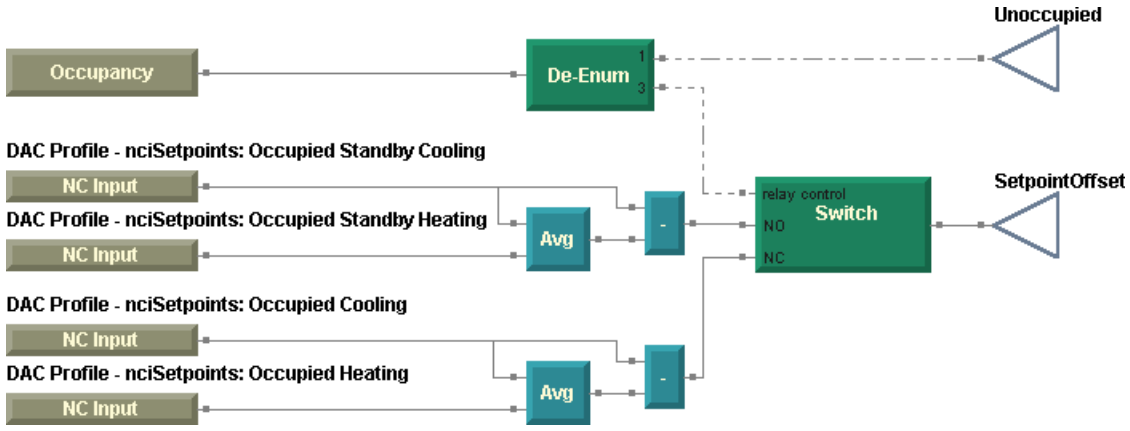
Figure 128: Heat/Cool mode decision



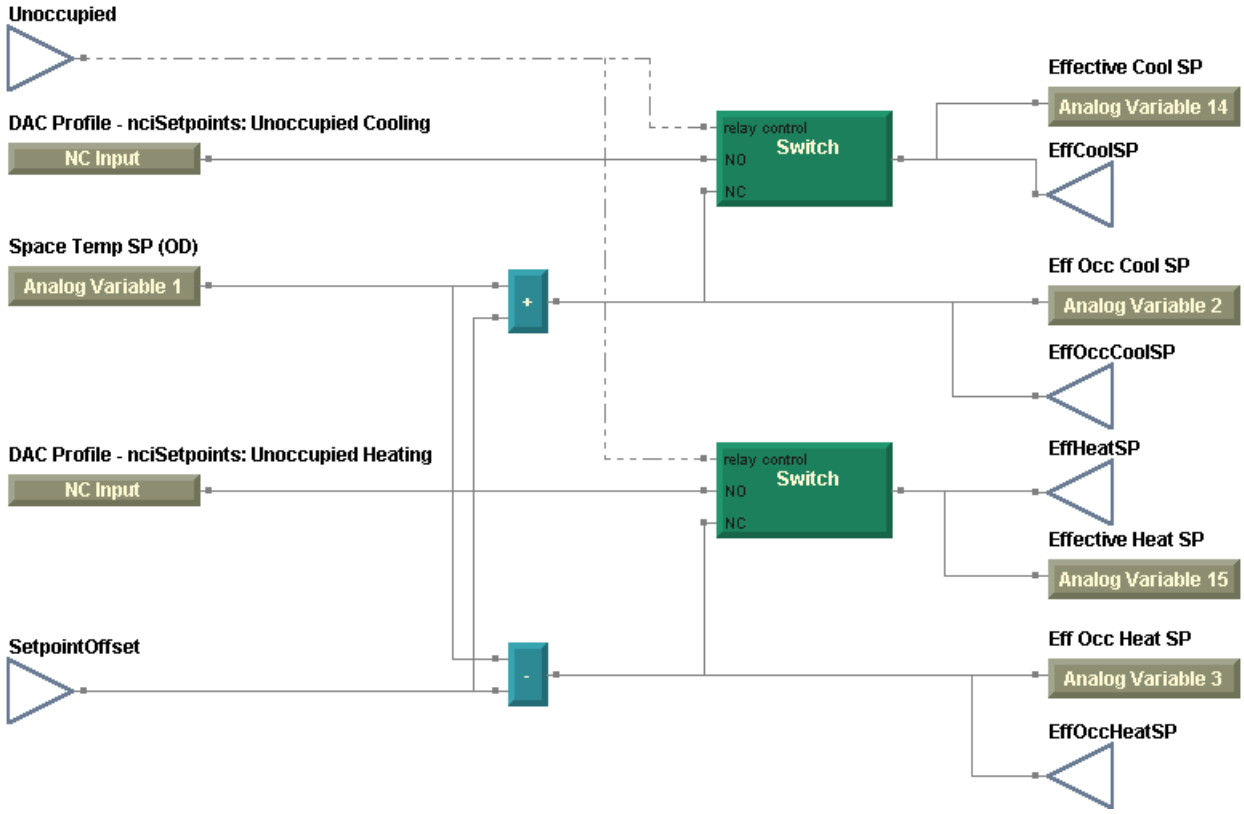
The mode and setpoints program completes the VAV air-handling unit. Compile and download this final program (Figure 129) to the controller.

Figure 129: Completed modes and setpoints program

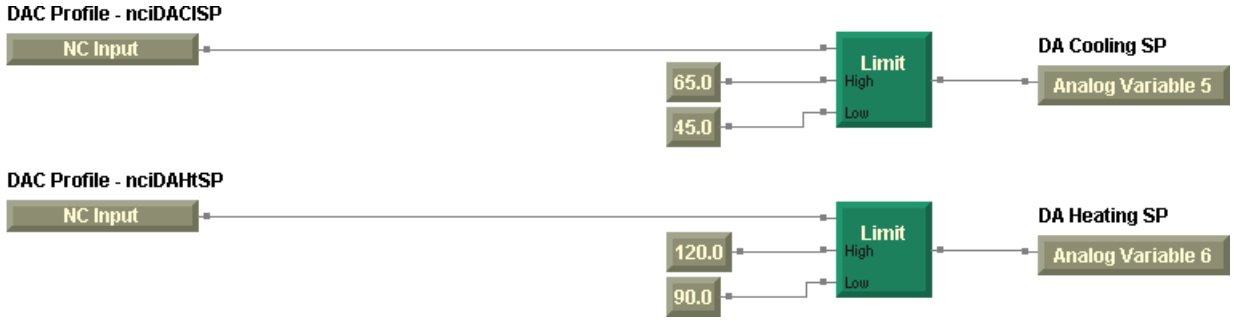
Offset calculation



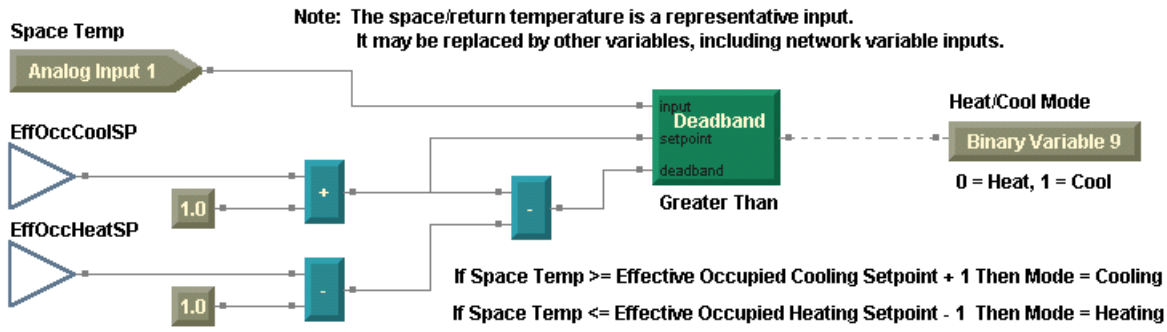
Effective setpoint calculation



Validating discharge air setpoints



Heat/Cool mode decision



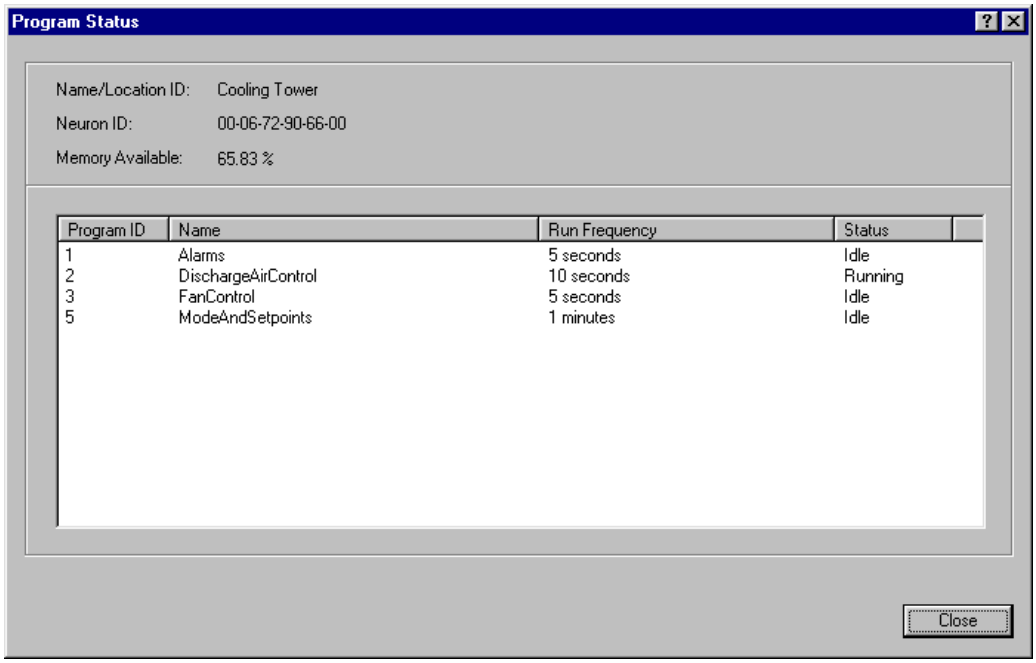
Viewing program status

You can view the status of each program downloaded to the Tracer MP580/581 controller. The program status includes the program name, run frequency, and state. It also displays the remaining memory available for programs.

To view program status:

- From the Tools menu, choose Program Status. The Program Summary dialog box appears (Figure 130).

Figure 130: Program Summary dialog box



Summary questions

Answer the following questions to review the skills, concepts, and definitions you learned in this chapter. The answers to these questions are on page 237.

1. How would you use the outdoor air enthalpy to determine whether to economize?

2. Use the calculation application in combination with a program to create a fan maintenance timer. Every 4,000 hours, turn on a binary variable to indicate that fan maintenance is required. Also, use another binary variable to reset the maintenance timer.

3. Implement a binary variable to enable the operator to switch between a space temperature setpoint source of operator display or a local, wired zone sensor with a thumbwheel setpoint adjustment knob. Be sure to apply limits to the resulting setpoint. Modify the modes and setpoints program to accommodate this new feature.

4. What modifications are necessary to the Mode and Setpoints program to allow the operator to adjust the discharge air cooling and heating setpoints?

Chapter 7

Constant-volume AHU example

In this chapter, you will expand your programming skills. You will use what you learned in Chapters 1 through 6 to program a constant-volume air handler with space temperature control.

Note:
Many of the chapters in this book build on previous chapters, so be sure to complete the chapters in the order presented. See “About this book” on page 1 for additional instructions.

What you will learn

In this chapter, you will learn a variety of skills, concepts, and definitions.

Skills

You will learn how to interpret increasingly complex sequences of operation, with a focus on constant-volume air-handler applications.

Concepts and definitions

You will understand the following concepts and definitions:

- Writing program modules to meet a sequence of operation
- Reusing and modifying program modules from previous programs
- Incorporating peer-to-peer binding with another Comm5 device

Block

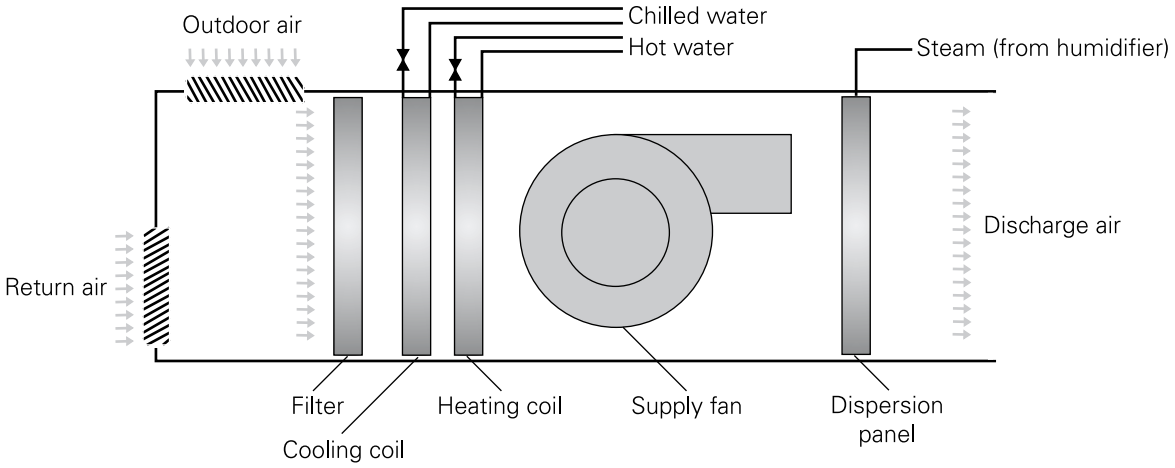
You will learn how to use the following blocks:

- Reset
- Between

Reviewing the sequence of operation

In this scenario a constant-volume air handler provides comfort heating and cooling to a space. The air handler contains a variable-speed fan, both cooling and heating coils, and an outdoor air damper. The air handler is a stand-alone unit (Figure 131).

Figure 131: Constant-volume air handler



Modes and setpoints

The following parameters for the sequence of operation are presented according to the occupancy mode, the heat/cool mode, or the setpoint.

Occupied mode (including occupied bypass)

When the air handler is in occupied mode, operate the supply fan continuously and modulate the cooling valve, heating valve, and outside air damper to maintain space comfort conditions. During unoccupied periods, when the local timed override button (the ON button on the space temperature sensor) is pressed, transition the air handler to occupied bypass mode for the configured occupied bypass time period. When the local CANCEL button is pressed, terminate the timed override request and revert the air handler to the scheduled mode.

Occupied standby mode

When the air handler is in occupied standby mode, operate the supply fan continuously and modulate the cooling valve, heating valve, and outside air damper to maintain space comfort conditions. Use occupied standby mode to save energy by using lower heating and higher cooling temperature setpoints and reduced ventilation requirements during intermittent unoccupied periods.

An occupancy sensor determines the presence of people in the space served by the air-handling unit. When the space is occupied, operate the air handler in occupied mode. When the space is unoccupied during a

scheduled occupancy period, set the space temperature control to follow the occupied standby setpoints.

Unoccupied mode (including night heat/cool)

When the air handler is in unoccupied mode, turn the supply fan off and fully close the outside air damper and the cooling and heating valves. Open the heating valve completely if the outdoor air temperature falls below the freeze avoidance setpoint, 35°F (adjustable).

If the space temperature falls below the unoccupied heating setpoint, start the supply fan. Transition the air handler to heating and keep the outdoor air damper closed to maintain the unoccupied heating setpoint. When the space temperature rises above the unoccupied heating setpoint, shut down the air handler.

If the space temperature rises above the unoccupied cooling setpoint, start the supply fan and transition the air handler to cooling to maintain the unoccupied cooling setpoint. Modulate the outside air damper if the outside air temperature is less than the economizer changeover setpoint. When the space temperature falls below the unoccupied cooling setpoint, shut down the air handler.

Space setpoints

For occupied, occupied standby, and occupied bypass modes, calculate effective cooling and heating setpoints based on a single space temperature setpoint and the configured default occupied and occupied standby setpoints. The space temperature setpoint source may be the operator display or a wired thumbwheel setpoint adjustment knob.

Note:
Calculate the effective cooling and heating setpoints as in other Comm5 devices. See “Calculating the effective space setpoints” on page 135 for more information.

Discharge air setpoints

Compare the space temperature and space temperature setpoint to determine the discharge air temperature setpoint. Reset the discharge air temperature setpoint according to heating or cooling demand.

Heat/Cool arbitration

Determine the heat/cool mode of the air handler based on the effective occupied or occupied standby cooling and heating setpoints. Transition the air handler to cooling if the space temperature exceeds the cooling setpoint plus 1°F. Transition the air handler to heating if the space temperature falls below the heating setpoint minus 1°F.

Control

The following parameters for the sequence of operation are presented according to the equipment that must be controlled.

Supply fan

Operate the supply fan continuously whenever the air handler is in occupied or night heat/cool mode. Turn the supply fan off whenever one of the following occurs:

- The air handler is unoccupied.
- The run/stop interlock is open.
- The mixed air temperature is too cold, and the low limit detection is closed.
- The supply fan status indicates a fan failure after a 1-minute delay.

Note:
A failure due to low mixed air temperature or fan failure requires a manual reset. Also, the low limit switch may require a manual, hardware reset.

Outdoor air damper

When the economizer function is enabled and the outdoor air temperature is less than the economizer changeover setpoint, modulate the outdoor air damper between the adjustable minimum position and fully open to maintain the discharge air cooling setpoint. Modulate the outdoor air damper closed, overriding the minimum position, to maintain the mixed air temperature at or above the mixed air setpoint.

If the economizer function is disabled or the air handler is in the heat mode, control the outdoor air damper to its minimum position. If the outdoor air temperature falls below a low outdoor air temperature limit, control the outdoor air damper to its closed position. In unoccupied mode, control the outdoor air damper to its closed position. Also, if the supply fan is off or the mixed air temperature sensor is failed, control the outdoor air damper to its closed position.

Exhaust fan

Coordinate exhaust fan operation with the unit supply fan and outdoor air damper position. Turn on the exhaust fan whenever the supply fan is on and the outdoor air damper is open beyond 30%. Keep the exhaust fan on until the outdoor air damper closes to below 20% open or the supply fan is turned off.

Cooling valve

Modulate the cooling valve to maintain the discharge air temperature at the discharge air setpoint. If the economizer function is enabled and the outdoor air damper is not open at least 90%, control the cooling valve to its closed position. Also, close the cooling valve if the air handler is in heat

mode, the supply fan is off, or the discharge air temperature sensor is failed.

Heating valve

Modulate the heating valve to maintain the discharge air temperature at the discharge air setpoint. Close the heating valve if the air handler is in cool mode, the supply fan is off, or the discharge air temperature sensor is failed. Open the heating valve if the supply fan is off and the outdoor air temperature falls below the adjustable freeze avoidance setpoint.

Humidification

Enable humidification mode when the air handler is in occupied mode, the supply fan is on, and the outdoor air temperature falls below 55°F. Modulate the humidifier to maintain the space relative humidity setpoint. If the duct relative humidity exceeds 85%, turn off the humidifier to prevent condensation in the duct work and indicate an alarm.

Dehumidification

Enable dehumidification mode when the air handler is occupied, the supply fan is on, the outdoor air temperature is above 55°F, and the space relative humidity is above the space dehumidification setpoint. Modulate the cooling valve to maintain space relative humidity and the heating valve to maintain discharge air temperature.

Alarms

In addition to the alarm requirements mentioned above, indicate an alarm at the operator display and turn on the alarm output when any sensor fails. Alarms must be resettable at the operator display. Diagnostic conditions include the following:

- Dirty filter
- Duct humidity high limit
- Exhaust fan failure
- Low outdoor air temperature
- Low (mixed air) temperature detection
- Sensor failure (including discharge air temperature, mixed air temperature, outdoor air temperature, space temperature, and duct static pressure)
- Supply fan failure

The following failures shutdown the air handler and require a manual reset:

- Discharge or mixed air temperature sensor failure
- Fan failure
- Low mixed air temperature

The corresponding data definition is presented in Table 18 on page 150 and Table 19 on page 151, and a wiring diagram is presented in Figure 132 on page 153.

Table 18: Constant-volume AHU inputs and outputs data definition

Inputs and outputs		Type	Name	Notes
Inputs	1	Analog	Space Temp	Universal input configured as thermistor or RTD
	2	Analog	Thumbwheel SP	Universal input configured as thermistor or RTD
	3	Analog	Mixed Air Temp	Universal input configured as thermistor or RTD
	4	Analog	Discharge Air Temp	Universal input configured as thermistor or RTD
	5	Analog	Outdoor Air Temp	Universal input configured as thermistor
	6	Analog	Duct Humidity	Universal input configured as current (0–20 mA)
	7	Binary	Low Temp Detect	
	8	Binary	Run/Stop Interlock	
	9	Binary	Occupancy/Generic	
	10	Binary	Supply Fan Status	
	11	Binary	Filter Status	
	12	Binary	Exhaust Fan Status	
	Pressure		Duct Static Pressure	
Binary outputs	1		Supply Fan Start/Stop	
	2		Exhaust Fan Start/Stop	
	3		Not Used	
	4		Not Used	
	5		Not Used	
	6		Alarm Output	Status/custom alarm indicator
Analog outputs	1		Supply Fan Speed*	Analog output configured as voltage, 0–10 V
	2		Cooling Valve Position	Analog output configured as voltage, 0–10 V
	3		Heating Valve Position	Analog output configured as voltage, 0–10 V
	4		Face and Bypass Damper*	Analog output configured as voltage, 0–10 V
	5		OA Damper Position	Analog output configured as voltage, 0–10 V
	6		Humidifier	Analog output configured as voltage, 0–10 V

* These inputs and outputs are not used in this program but are often used in air-handler programs.

Note:The inputs and outputs in this table are configured so that a Tracer AH540 main logic board could be replaced with a Tracer MP580/581 board, except for the Mixed Air Temp, which is input 6 on the Tracer AH540.

Table 19: Constant-volume AHU variables data definition

Variable	Name	Notes
Tracer Summit binary variables	Not Used	
Local binary variables	Supply Fan Failure	Sourced from a program
	Exhaust Fan Failure	Sourced from a program
	Sensor Failure	Sourced from a program
	Duct Humidity Limit	Sourced from a program
	Low OA Temp Alarm	Sourced from a program
	Fan Failure Reset	Sourced from operator display and a program
	Alarm Reset	Sourced from operator display and a program
	Manual Reset Alarm	Sourced from a program
	Auto Reset Alarm	Sourced from a program
	Thumbwheel SP Enable	Sourced from operator display
	Economizer Enable	Sourced from operator display
	Heat/Cool Mode	Sourced from a program
	OK to Economize	Sourced from a program
	Humidify	Sourced from a program
	Dehumidify	Sourced from a program
	Night Heat/Cool	Sourced from a program
Tracer Summit analog variables	Not Used	
Local analog variables	Space Temp SP (OD)	Setpoint sourced from the operator display
	Eff Occ Cool SP	Sourced from a program
	Eff Occ Heat SP	Sourced from a program
	Effective Cool SP	Sourced from a program
	Effective Heat SP	Sourced from a program
	Economizer Changeover SP	Sourced from the operator display
	Discharge Air SP	Sourced from a program
	Low OA Temp SP	Setpoint sourced from the operator display
	Mixed Air Temp SP	Setpoint sourced from the operator display
	Dehumidification SP	Setpoint sourced from the operator display
	Humidification SP	Setpoint sourced from the operator display
	Ex Fan OA Damper SP	Setpoint sourced from the operator display
	Ex Fan OA Damper Dbd	Setpoint sourced from the operator display

Before you write the program, configure these inputs, outputs, and variables in your Tracer MP580/581 controller.

Also, because you are programming a constant-volume air-handling unit, set the controller to use the SCC profile. For more information about configuring the controller, including setting it to use the SCC profile, see the *Tracer MP580/581 Programmable Controller Programming* guide

(CNT-SVP01A-EN). The SCC profile provides the configuration data listed in Table 20.

Table 20: Network configuration inputs for the SCC profile

nci	SNVT	Notes
nciBypassTime	SNVT_time_min	Occupied bypass time
nciOAMinPos	SNVT_lev_percent	Outdoor air damper minimum position
nciSpaceCO2Lim	SNVT_ppm	Space CO_2 limit
nciSpaceRHSetpt	SNVT_lev_percent	Space relative humidity setpoint
nciSetpoints	SNVT_temp_setpt	Provides default unoccupied, occupied, and occupied standby cooling and heating setpoints

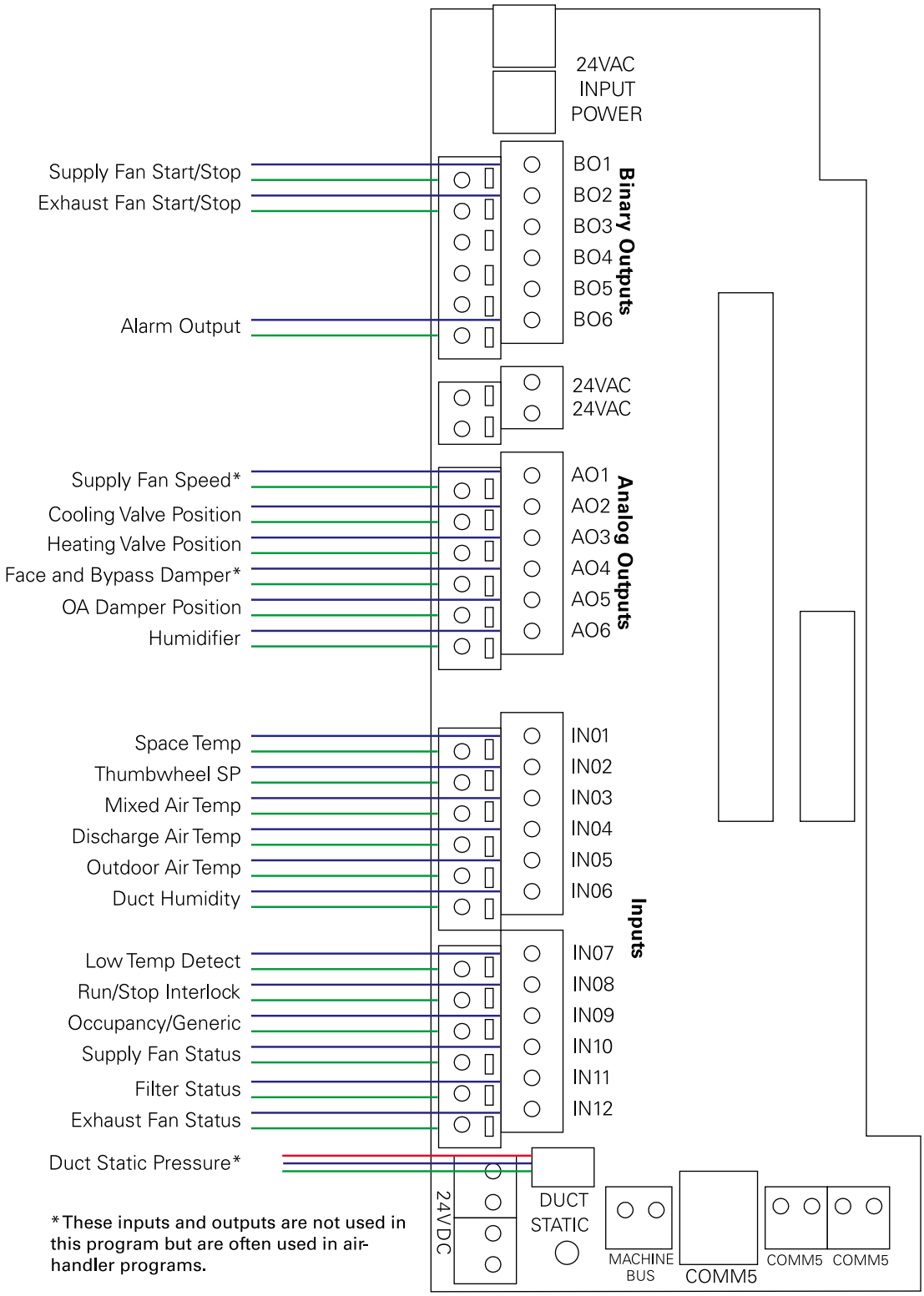
The air-handling unit receives the space relative humidity from a peer Comm5 device, a Tracer MP503 I/O module in this scenario. The space relative humidity is sent to the Tracer MP580/581 controller through a network variable input nviCurrent_mA01. See Table 21 for more information.

Table 21: Network variable inputs

nvi	SNVT	Notes
nviCurrent_mA01	SNVT_amp_mil	Space relative humidity, measured in current (mA)
Note:The Tracer MP503 module communicates this value that must be converted to units of humidity at the Tracer MP580/581 controller using TGP.		

Then make sure that the TGP editor is open and ready to go.

Figure 132: Constant-volume AHU wiring diagram



Determining a programming approach

As in the previous chapter, the programming instructions in this chapter are in a modular form. Remember that this sequence of operation could result in a large number of variations in program content and programming technique.

From the sequence of operation, determine the basic control functions required. The required control functions are very similar to those for the Chapter 6, "VAV AHU example." Address the following control functions:

- Effective space setpoint calculation
- Mode determination, including night heat/cool
- Supply fan control
- Exhaust fan control
- Mixed air and outdoor air damper control
- Discharge air setpoint calculation
- Cooling valve control, including dehumidification
- Heating valve control, including dehumidification
- Alarm management

Not every control function fits into one program. Determine the number of programs required by grouping control functions together that require the same information, and/or the same run frequency as shown in Table 22.

Table 22: Control functions within each program

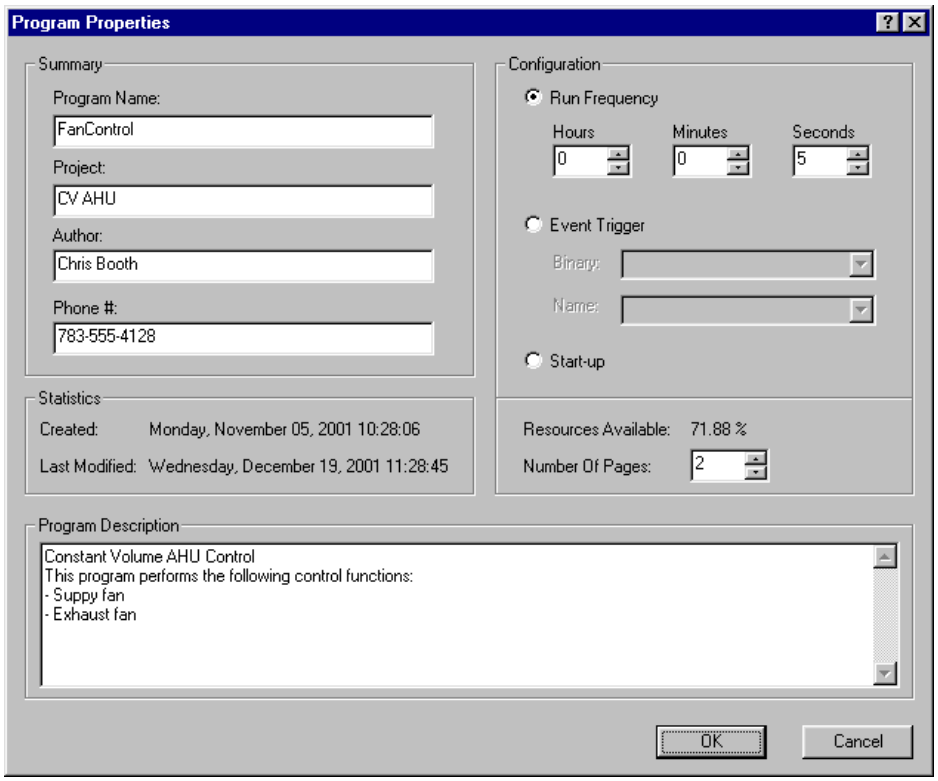
Program name	Control functions
FanControl	Supply fan control
	Exhaust fan control
MixedAirControl	Mixed air/outdoor air damper control
DischargeAirControl	Discharge air setpoint calculation
	Cooling valve control
	Heating valve control
	Dehumidification
	Humidification
Alarms	Alarm management
ModeAndSetpoints	Effective space setpoint calculation
	Setpoint source determination
	Mode determination

The functionality of many components of the constant-volume air handler is similar to that of the VAV air handler. Keep in mind that you can modify those programs and re-use them for the constant-volume air handler.

Writing the fan control program

Begin constructing the fan control program for the constant-volume air-handling unit by creating a new program and setting its properties as shown in Figure 133.

Figure 133: Fan control program properties



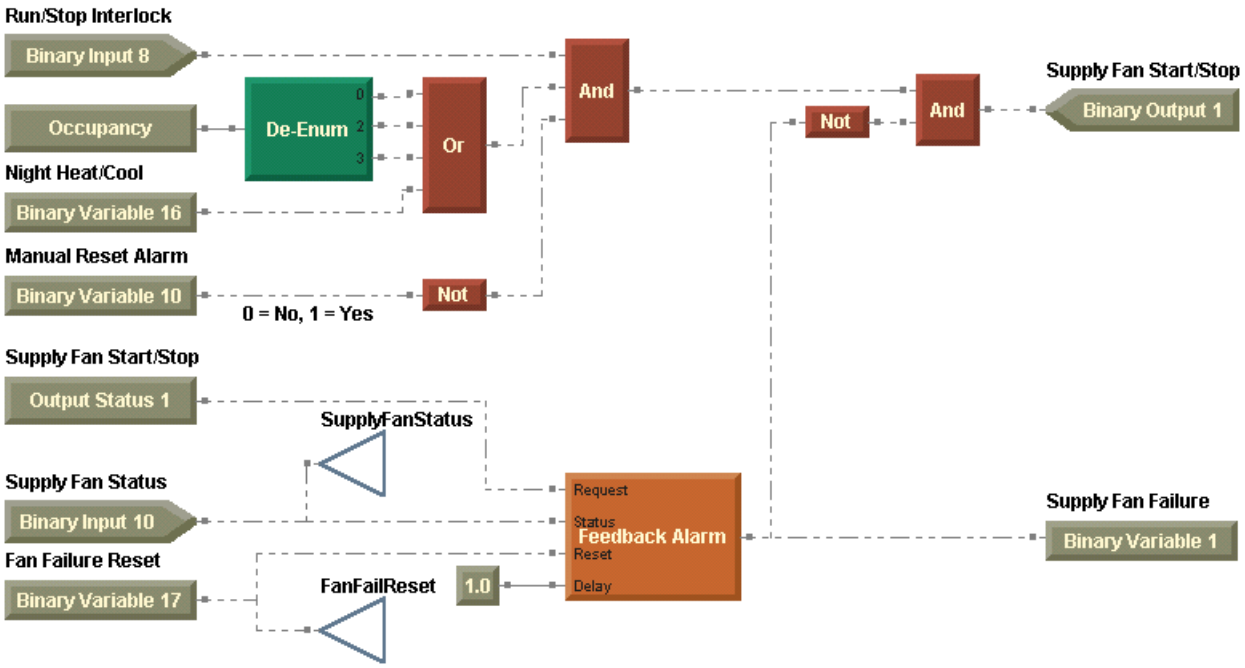
Controlling the supply fan

Control the supply fan on and off according to the sequence of operation. Note that this sequence is the same as that for the VAV air-handling unit with the omission of duct static pressure considerations. Turn the supply fan on whenever the unit is occupied. Turn the supply fan off when *any* of the following is true:

- The unit is unoccupied, with the exception of night heat/cool.
- The run/stop interlock is open.
- The low temperature detection is closed.
- The supply fan status indicates a fan failure after a 1-minute delay.

The module shown in Figure 134 on page 156 is identical to that used in the fan control program for the VAV air handler.

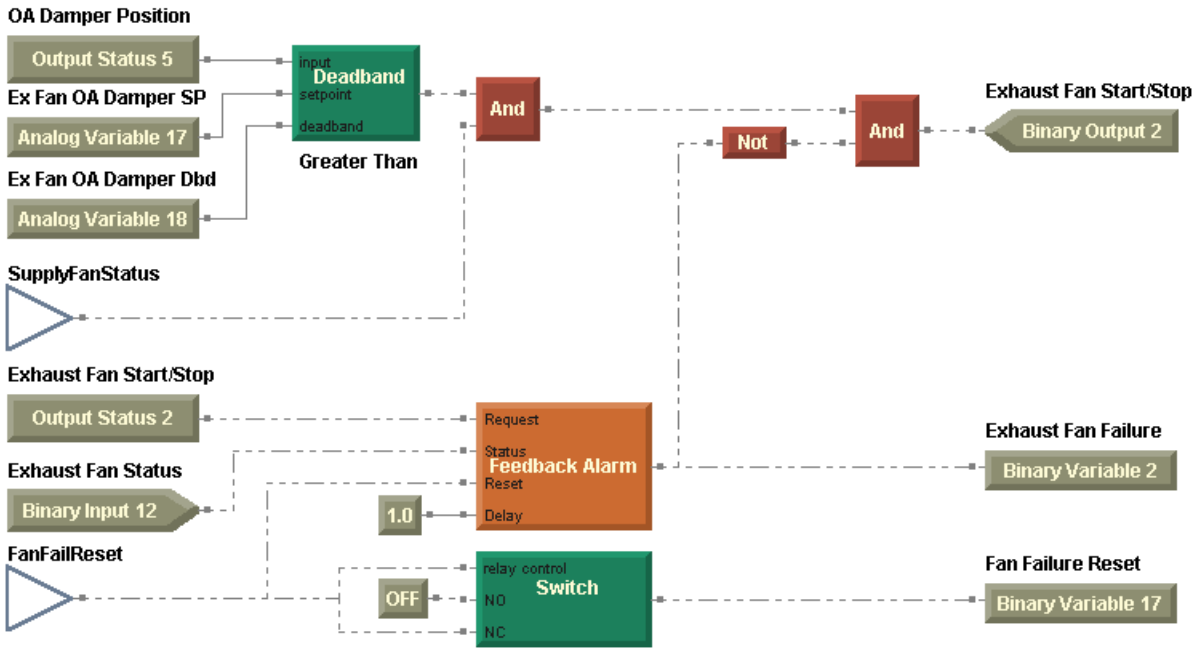
Figure 134: Supply fan control



Controlling the exhaust fan

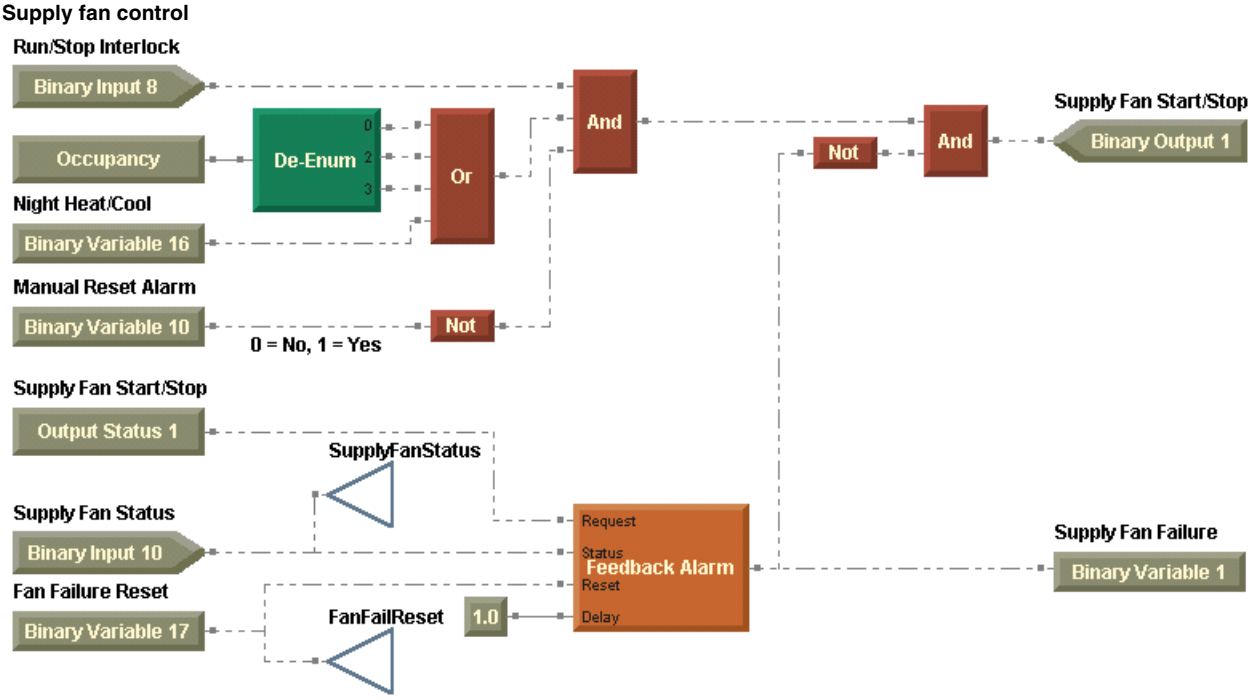
The control module for the exhaust fan remains functionally the same as that for the VAV air-handling unit. Except, use variables for the outdoor air damper setpoint and deadband instead of constants. Variables allow adjustment of the outdoor air damper position, as shown in Figure 135 on page 157.

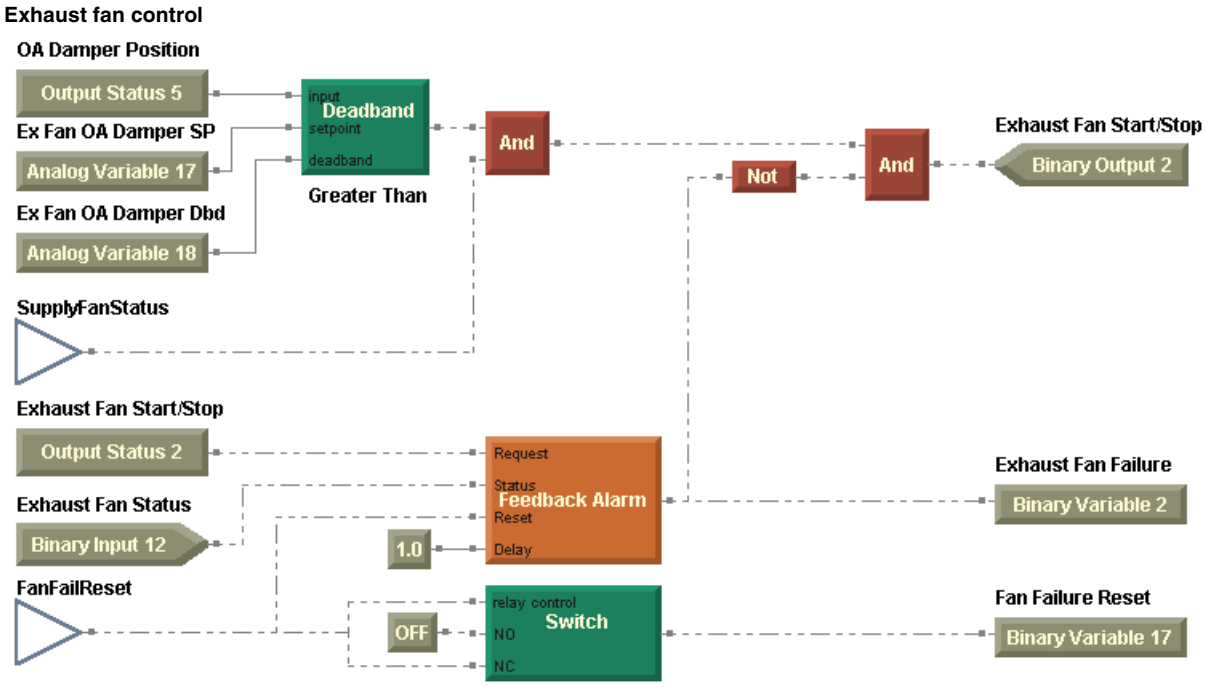
Figure 135: Exhaust fan control



After completing the program (Figure 136), compile and download it the controller.

Figure 136: Completed fan control program



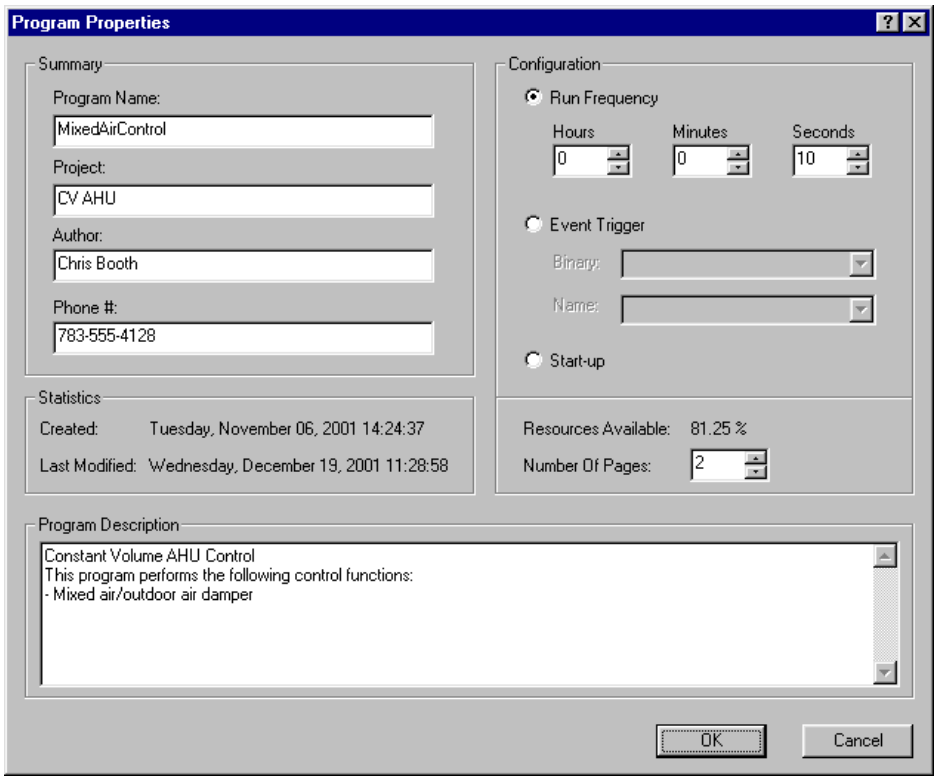


Writing the mixed air control program

To control discharge air temperature, the mixed air control program in your constant-volume air-handling unit must control the mixed air damper and the outdoor air damper. In the VAV air-handling programs, control of the mixed air and the discharge air reside in the same program. However, due to the potential increase in complexity of discharge air control in the constant-volume air handler, the mixed air control resides in a separate program. Functionally, mixed air control remains identical to the VAV air handler.

Create a new program and set its properties as shown in Figure 137 on page 159.

Figure 137: Mixed air control program properties



Operation of the outdoor air damper and subsequent mixed air temperature control must satisfy a number of scenarios according to the sequence of operation.

Modulate the outdoor air damper between the adjustable minimum position and fully open to maintain the discharge air cooling setpoint when *all* of the following are true:

- The economizer is enabled.
- The outdoor air temperature is less than the economizer changeover (or outdoor air temperature) setpoint.
- The air handler is in cooling mode, including when it is in night heat/cool mode.

Modulate the outdoor air damper closed, overriding the minimum position, to maintain the mixed air temperature at or above the mixed air setpoint.

Control the outdoor air damper to its minimum position when *either* of the following is true:

- The economizer function is disabled.
- The air handler is in heating mode.

Close the outdoor air damper completely when any of the following is true:

- The unit is in unoccupied mode.
- The unit is in night heat/cool mode and heating.
- The unit is in night heat/cool mode and cooling, and economizing is not allowed.
- The outdoor air temperature falls below the low outdoor air temperature setpoint.
- The supply fan is off.
- The mixed air temperature sensor is failed.

Simplify the program by separating the decision to economize from the actual control of the damper. Add a second stage of decision making to close the outdoor air damper in night heat/cool mode. Economizing is only permitted in night heat/cool mode when the unit is cooling, the economizer is enabled, and the outdoor air temperature is less than the economizer changeover setpoint.

Add the OK to Economize binary variable to the same program for the VAV air handler, as shown in Figure 138. Use this Variable block to pass the result of the economize decision to other programs, such as the discharge air control program.

Figure 138: Determining whether to economize

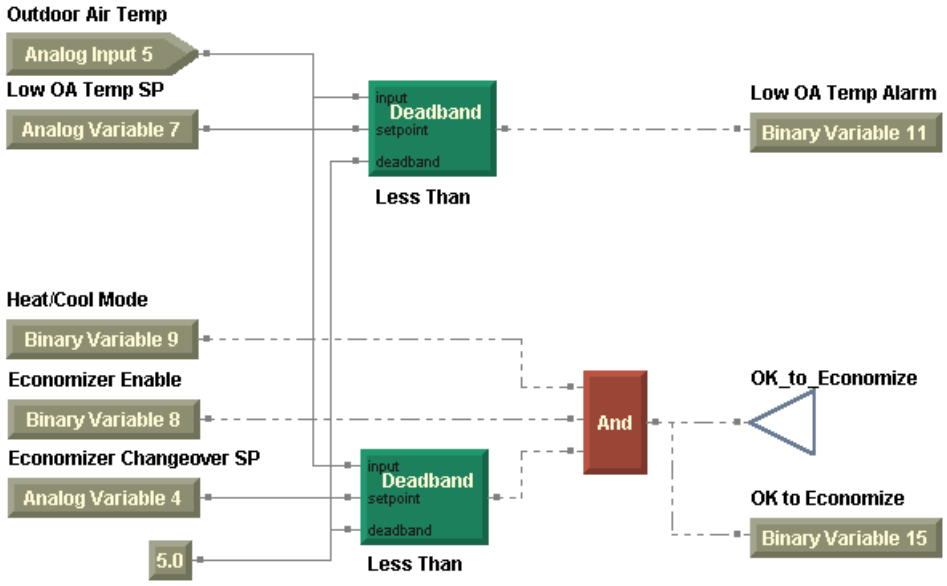
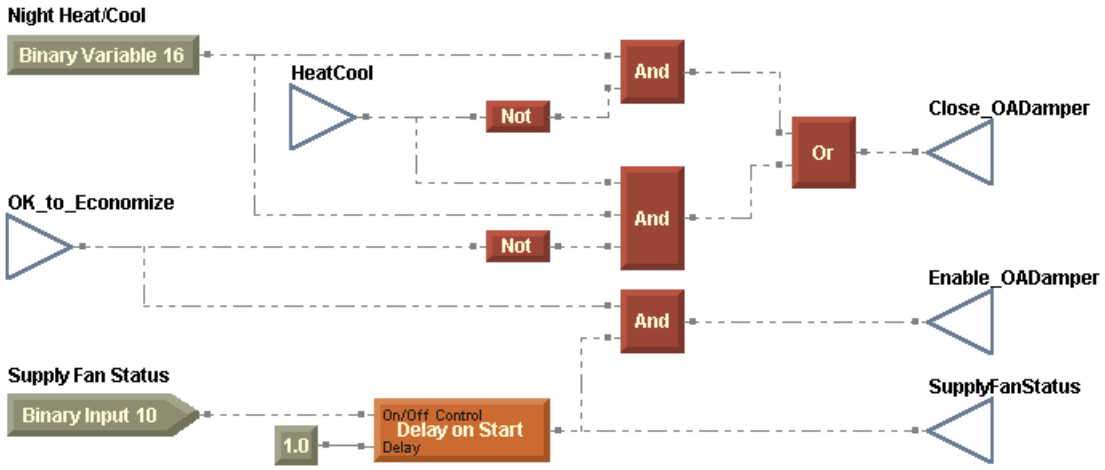


Figure 139 on page 161 shows the modifying decision based on night heat/cool mode.

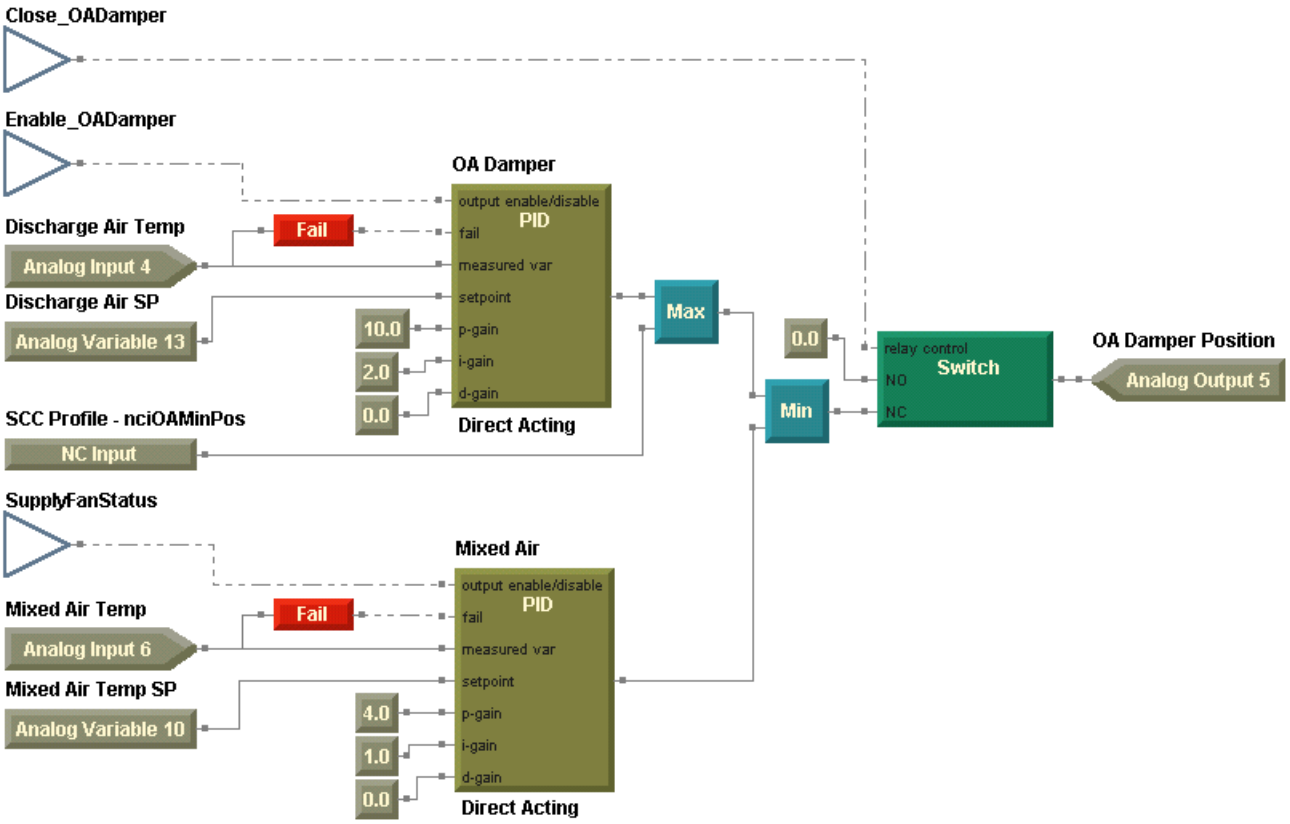
Figure 139: Night heat/cool incorporated into outside air damper control



Control of the outdoor air damper position remains the same, but with the following exceptions (as shown in Figure 140 on page 162):

- The constant-volume air handler employs a single discharge air temperature setpoint, instead of both cooling and heating discharge air temperature setpoints.
- Unlike the DAC profile, the SCC profile provides the outdoor air damper minimum position, but does not provide the mixed air low limit setpoint. Use an NC Input (network configuration input) block to supply the outdoor air damper minimum position, and a Variable block to supply the mixed air temperature setpoint.
- Use the Wireless block, Enable_OADamper to enable and disable the primary PID loop.
- Use the Close_OADamper wireless block to communicate the result of the decision in Figure 140 on page 162, based on night heat/cool mode.

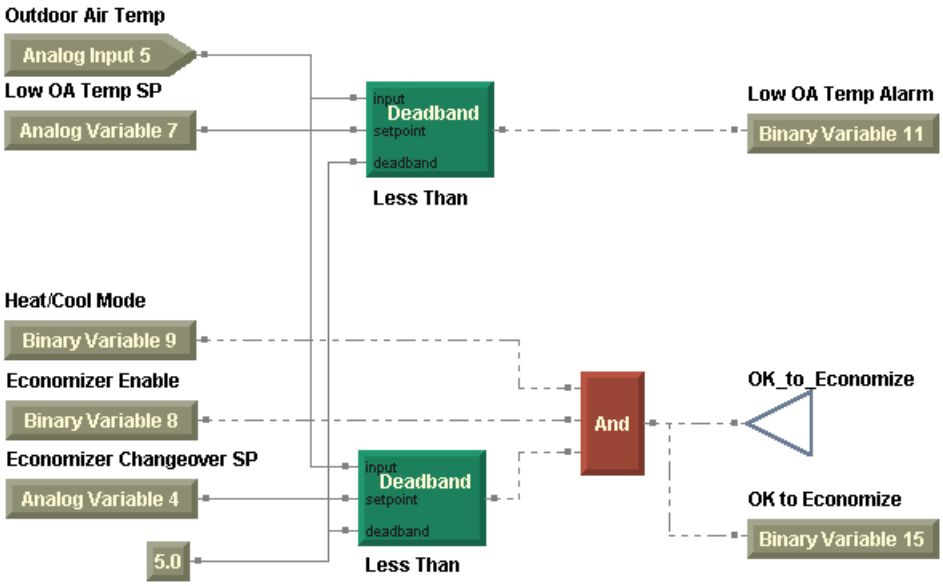
Figure 140: Controlling the outdoor air damper position



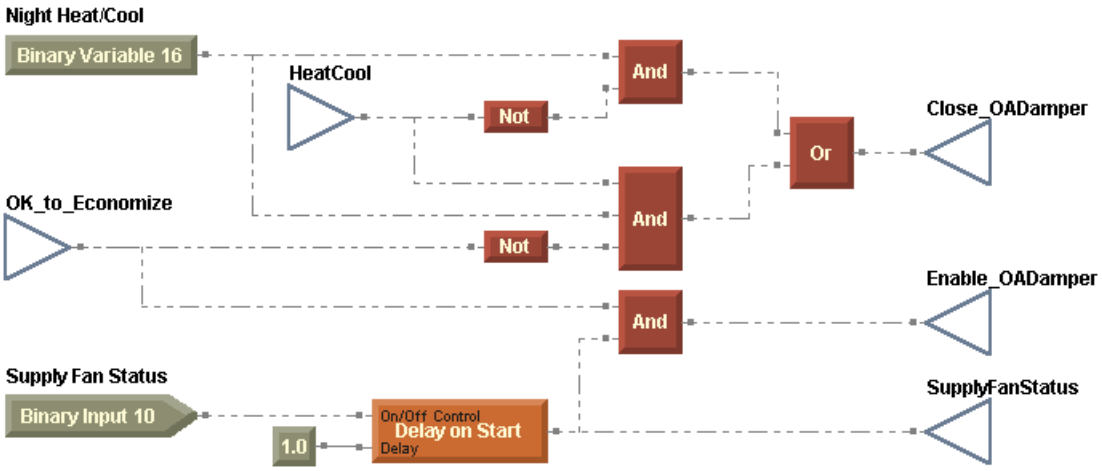
Compile and download the completed mixed air control program (Figure 137 on page 159).

Figure 141: Completed mixed air control program

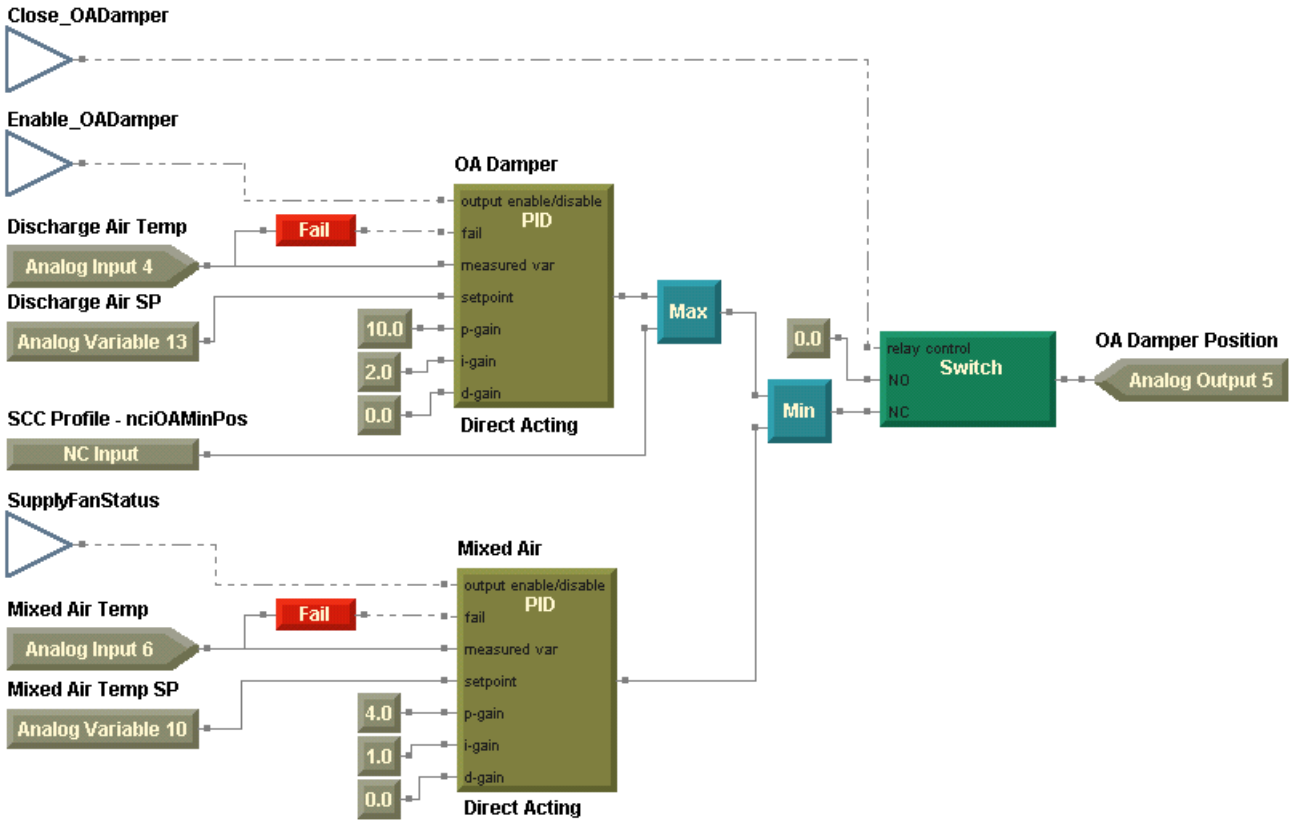
Determining whether to economize



Night heat/cool incorporated into outside air damper control



Controlling the outdoor air damper position



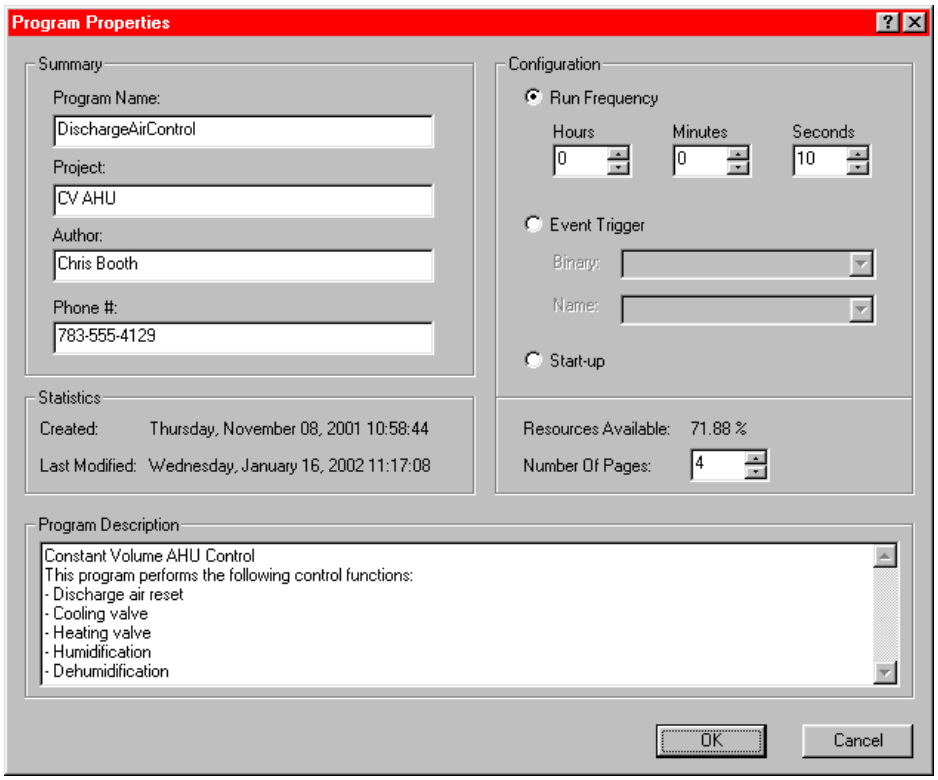
Writing the discharge air control program

The discharge air control program differs considerably from its VAV air handler counterpart. This program in your constant-volume air-handling unit must perform the following tasks:

- Discharge air setpoint calculation
- Cooling valve control
- Heating valve control
- Dehumidification
- Humidification

Create a new program and set its properties as shown in Figure 142 on page 165.

Figure 142: Discharge air control program properties



Calculating the discharge air setpoint

Unlike the VAV air handler, the constant-volume air handler does not have defined discharge air cooling and heating temperature setpoints. The sequence of operation states the following:

> Compare the space temperature and space temperature setpoint to determine the discharge air temperature setpoint. Reset the discharge air temperature setpoint according to heating or cooling demand.

Calculate the discharge air temperature setpoint based on the difference between the space temperature and the effective cooling or heating setpoint. Use a method commonly known as discharge air reset, or cascade control. Use a PID loop to calculate the discharge air setpoint. Then apply the resulting discharge air setpoint in combination with additional PID loops to control the cooling and heating valves (Figure 143 on page 166).

- Use a Switch block to select the appropriate setpoint based on the heat/cool mode.
- Set the properties for the discharge air reset PID block differently than the PID loops seen in previous chapters. Although the program executes every 10 seconds, this PID loop executes every 60 seconds because the space temperature and setpoint do not change rapidly. The output is a temperature instead of a percentage. Be sure to set

the output minimum and maximum values, as well as the disable and fail safe positions accordingly. See the PID properties in Figure 144.

- Use wireless connections to pass the supply fan status and the calculated discharge air setpoint to other parts of the program.
- Assign the discharge air setpoint to a variable for display purposes.

Figure 143: Discharge air reset

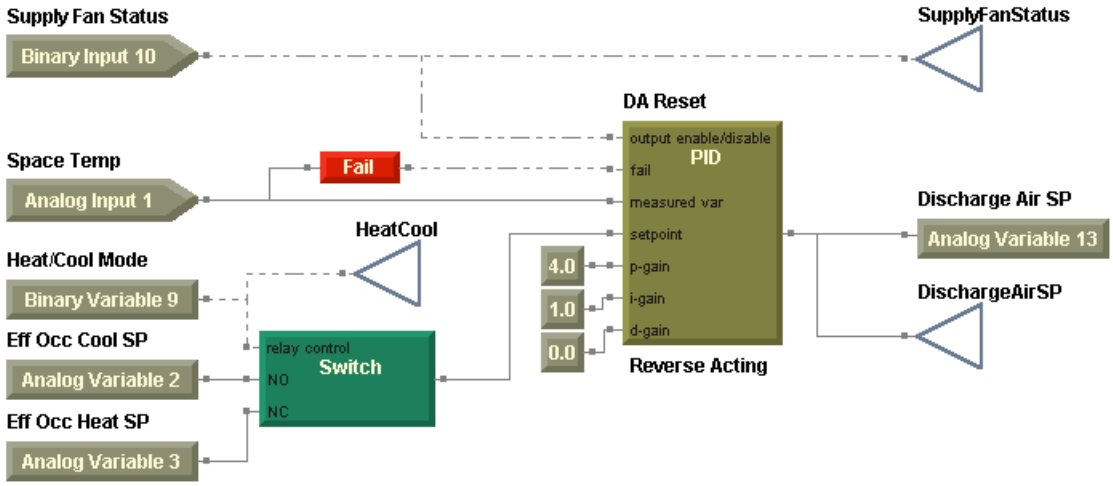
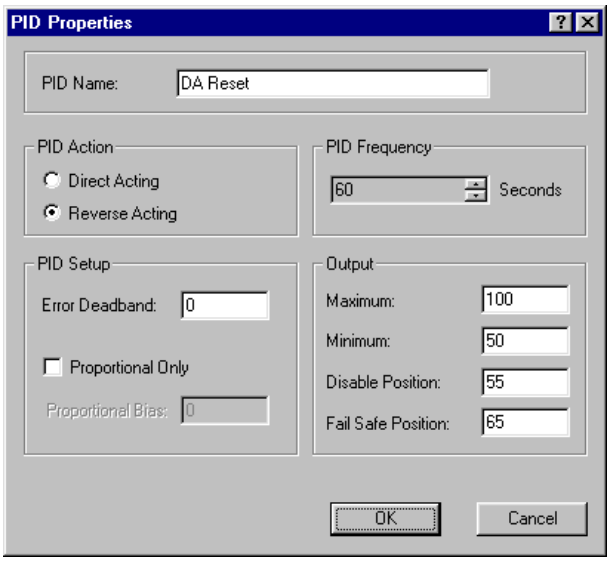


Figure 144: Discharge air reset PID properties



Implementing humidification and dehumidification

You will use the discharge air setpoint, later, in the program to control the heating and cooling valves. Now, determine whether to humidify or to dehumidify. Use this decision to control the cooling and heating valves. The sequence of operation contains the following information about humidification and dehumidification.

Enable humidification and modulate the humidifier to maintain the space relative humidity setpoint when *all* of the following conditions are true:

- The air handler is in occupied mode.
- The supply fan is on.
- The outdoor air temperature is less than 55°F.

If the duct relative humidity exceeds 85%, turn off the humidifier to prevent condensation in the duct work and indicate an alarm.

Enable dehumidification, modulate the cooling valve to maintain space relative humidity, and modulate the heating valve to maintain the discharge air temperature when *all* of the following conditions are true:

- The air handler is in occupied mode.
- The supply fan is on.
- The outdoor air temperature is greater than 55°F.
- The space relative humidity is greater than the space dehumidification setpoint.

Use the module in Figure 146 on page 169 to implement the decision but not the control actions.

- Use the Deadband blocks to provide the on/off mechanism to activate and deactivate humidification or dehumidification.
- Use a network variable input to access the space relative humidity, which is communicated from another Comm5 device. The Comm5 device in this scenario is a Tracer MP503 I/O module that is bound to the Tracer MP580/581 controller. For more information about network variable bindings with the Tracer MP580/581, see the *Tracer MP580/581 Programmable Controller Programming* guide (CNT-SVP01A-EN).
- Because the Tracer MP503 provides the space relative humidity in units of current (mA), use a Reset block to convert it to units of percent relative humidity.

➤ *Using a Reset block*

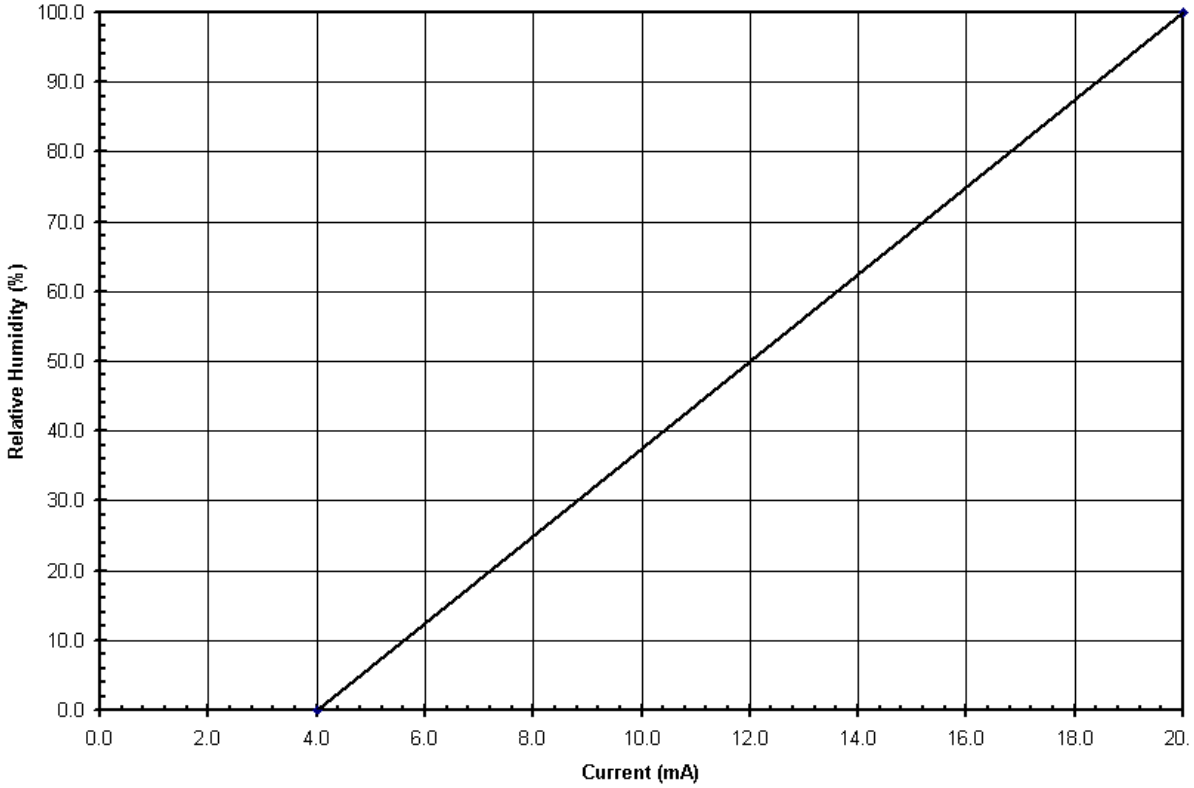
The Reset block calculates a reset schedule or a linear equation. In this program, use the Reset block to determine the linear equation necessary to calculate the relative humidity based on the network variable nviCurrent_mA01 that supplies current.

The inputs to the block include the following:
x1 = 4.0 mA, y1 = 0.0%
x2 = 20.0 mA, y2 = 100.0%

The resulting linear equation is as follows:
Relative humidity (%) = 6.25 × Current (mA) – 25.00

The Reset block uses this equation internally to calculate the relative humidity output with the current input. For example, when the network variable input nviCurrent_mA01 communicates the value of 14.0 mA, the Reset block calculates a relative humidity of 62.5%. See Figure 145 on page 168 for a plot of this equation.

Figure 145: Reset block linear equation plot



- SNVT_amp_mil is not a network variable input (nvi) type that includes the flag for the Fail block, so use a Between block to check that the value from the nvi is in a valid range.

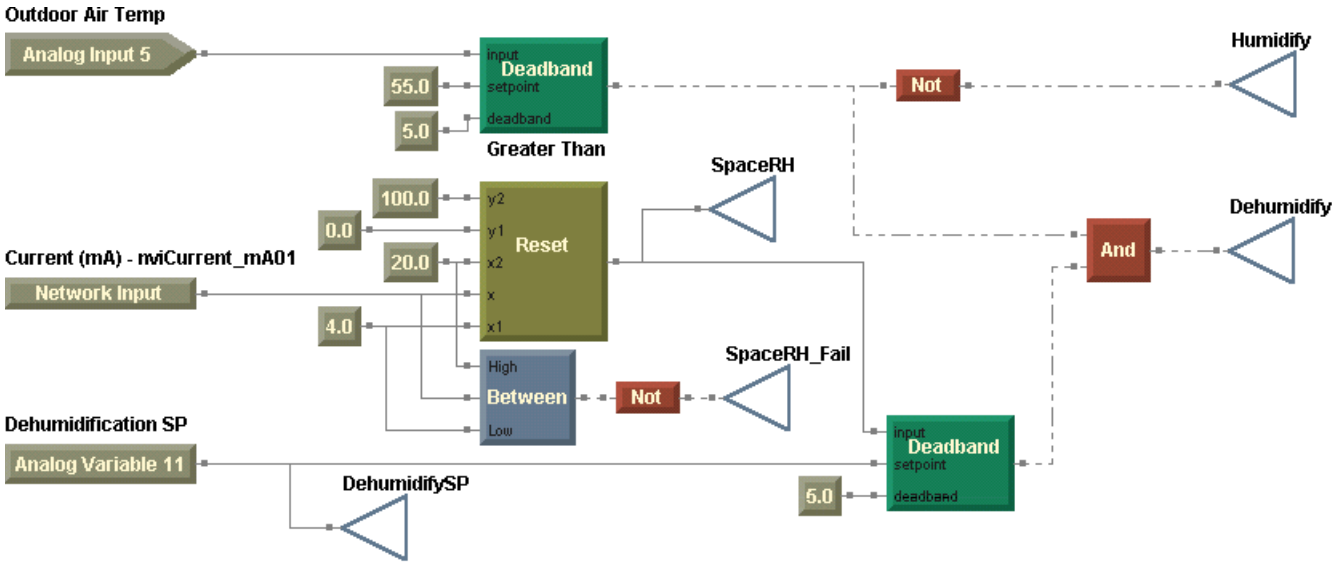
➤ ***Using a Between block***

Use a Between block to check an analog value against a range and then output a binary value accordingly. If the analog value is between the upper and lower limits, the output is true.

- Use wireless connections to carry the resultant data to the portions of the program controlling the cooling and heating valves and controlling the humidifier.

Note:
This module enables or disables only the ability to humidify or dehumidify. Control of the cooling and heating valves to accomplish dehumidification is incorporated into the appropriate modules. Control of the humidifier itself requires a separate module.

Figure 146: Whether to humidify/dehumidify



Controlling the cooling valve

Take a closer look at the sequence of operation of the cooling valve. Note the dehumidification requirement.

Modulate the cooling valve to maintain the discharge air temperature at the discharge air cooling setpoint when *all* of the following are true:

- The supply fan is on.
- The air handler is in cooling mode.
- The economizer function is not enabled.

Or when *all* of the following are true:

- The supply fan is on.
- The air handler is in cooling mode.
- The economizer function is enabled.
- The outdoor air damper is open at least 90%.

Modulate the cooling valve to maintain the space relative humidity at the space relative humidity setpoint when *all* of the following are true:

- The supply fan is on.
- The air handler is in cooling mode.
- Dehumidification is enabled.

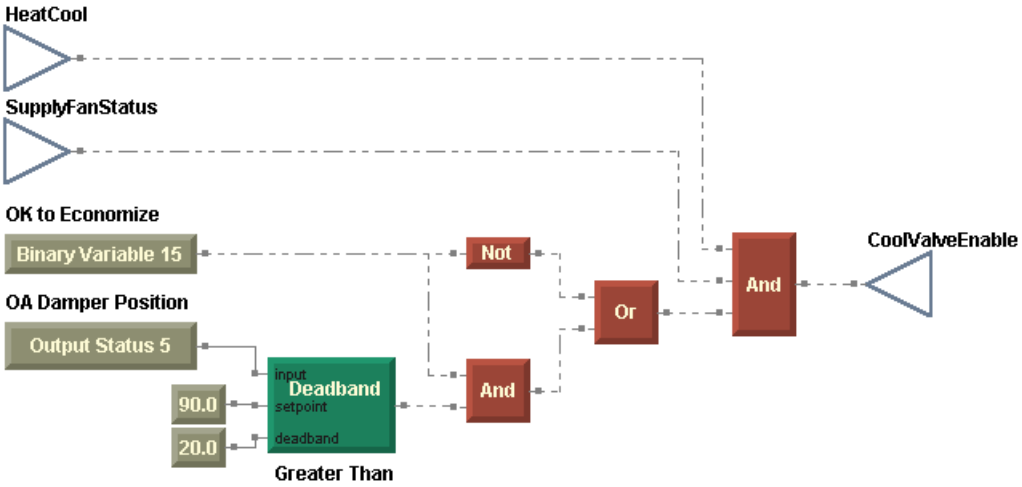
Close the cooling valve when *any* of the following is true:

- The air handler is in heating mode.
- The supply fan is off.
- The discharge air temperature sensor is failed.

Think of the cooling valve control in two parts.

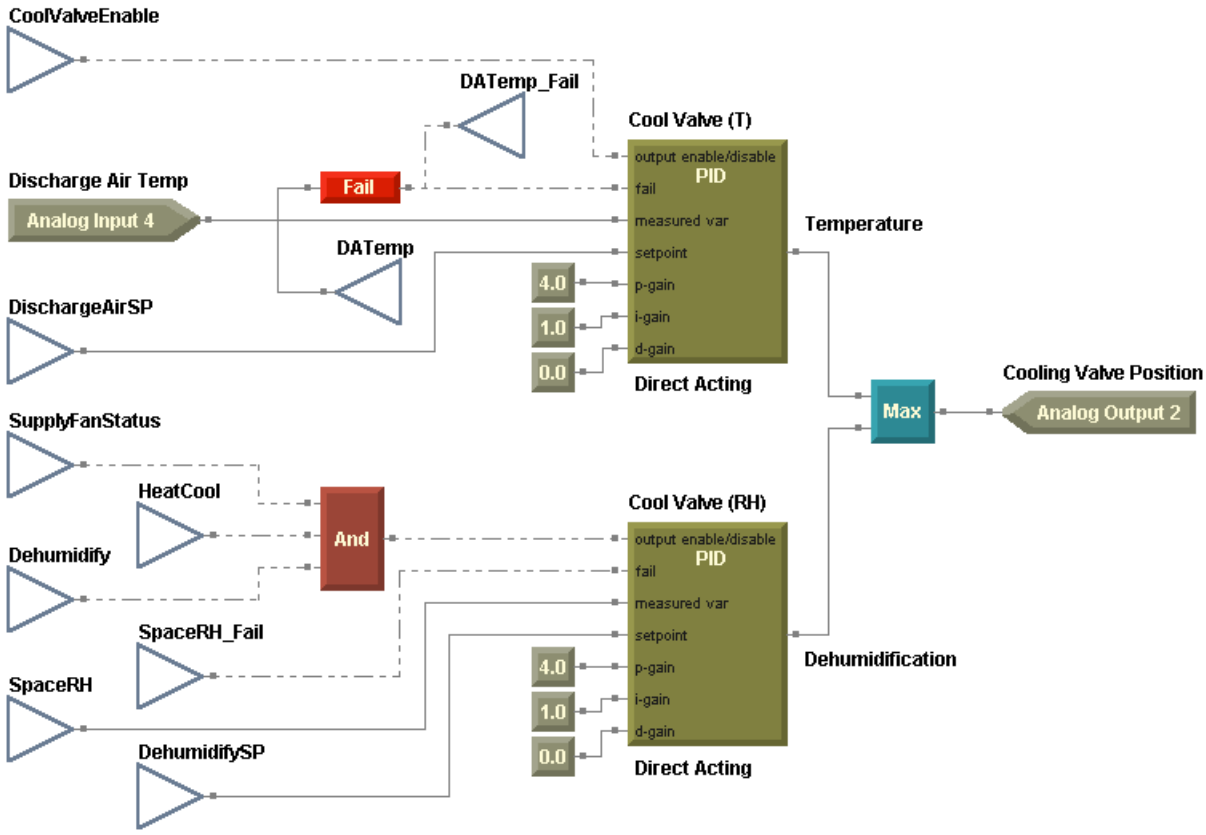
- Use a combination of logic blocks to determine if the cooling valve operates.
- Apply the output of the economizer decision here.
- Use a Deadband block to consider the position of the outdoor air damper as shown in Figure 147.

Figure 147: Operate the cooling valve?



When the cooling valve is enabled, use a PID loop to perform discharge air temperature control. However, if dehumidification is also active, you need a second PID loop to calculate the cooling valve position based on the space relative humidity. Control the valve to the greater of the two calculated valve positions (Figure 148 on page 171).

Figure 148: Controlling the cooling valve



Controlling the heating valve

The following may be derived from the sequence of operation for the heating valve. Note the additional requirement regarding dehumidification.

Modulate the heating valve to maintain the discharge air temperature at the discharge air heating setpoint when *all* of the following are true:

- The supply fan is on.
- The air handler is in heating mode.

Or when *all* of the following are true:

- The supply fan is on.
- Dehumidification is enabled.

Open the heating valve completely when *both* of the following are true:

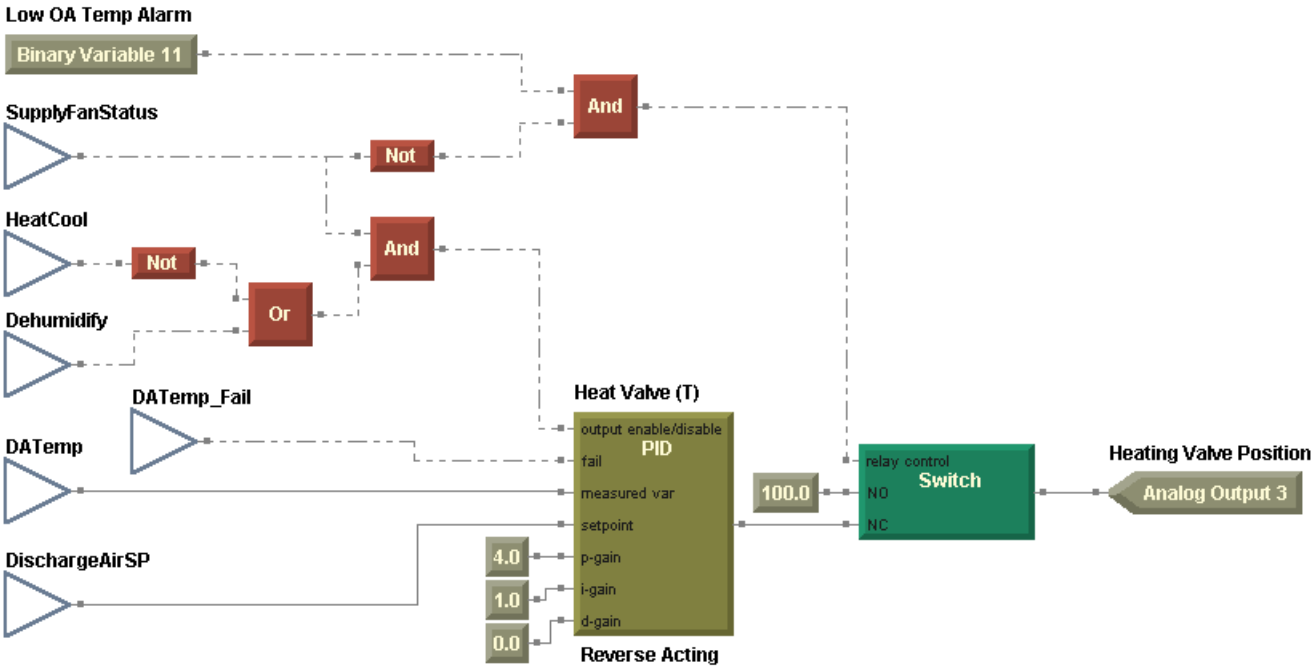
- The supply fan is off.
- The outdoor air temperature falls below the adjustable freeze avoidance setpoint.

Close the heating valve when *any* of the following is true:

- The air handler is in cooling mode, and dehumidification is disabled.
- The supply fan is off.
- The discharge air temperature sensor is failed.

The logic that enables and disables the heating valve PID loop is slightly more complex than the logic for the heating valve in Chapter 6, due to the addition of dehumidification. However, unlike the cooling valve, only one PID loop is required (Figure 149).

Figure 149: Controlling the heating valve



Controlling the humidifier

Control of the humidifier completes the discharge air program. In the decision to humidify or dehumidify, the program covered part of the sequence of operation regarding humidification, but it did not account for the following items:

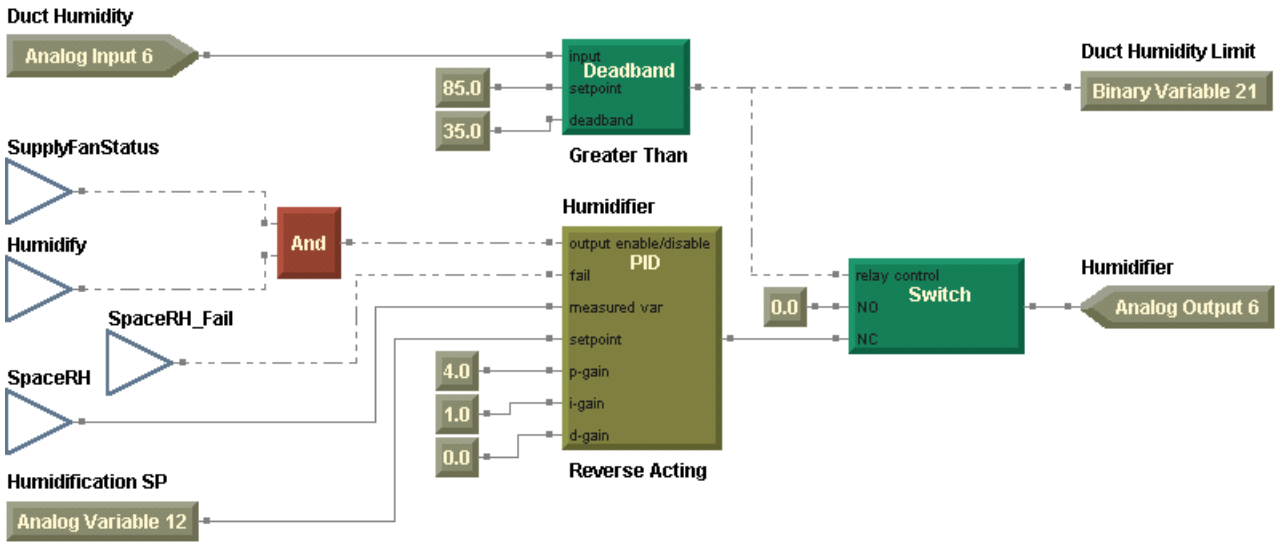
When humidification is enabled, complete the following:

- Modulate the humidifier to maintain the space relative humidity set-point.
- If the duct relative humidity exceeds 85%, turn off the humidifier and indicate an alarm to prevent condensation in the air-handler duct-work.

Complete the module shown in Figure 150 on page 173 to control the humidifier.

- Modulate the humidifier output with a PID loop.
- Use a Deadband block to control the humidifier off when the duct humidity exceeds 85%.

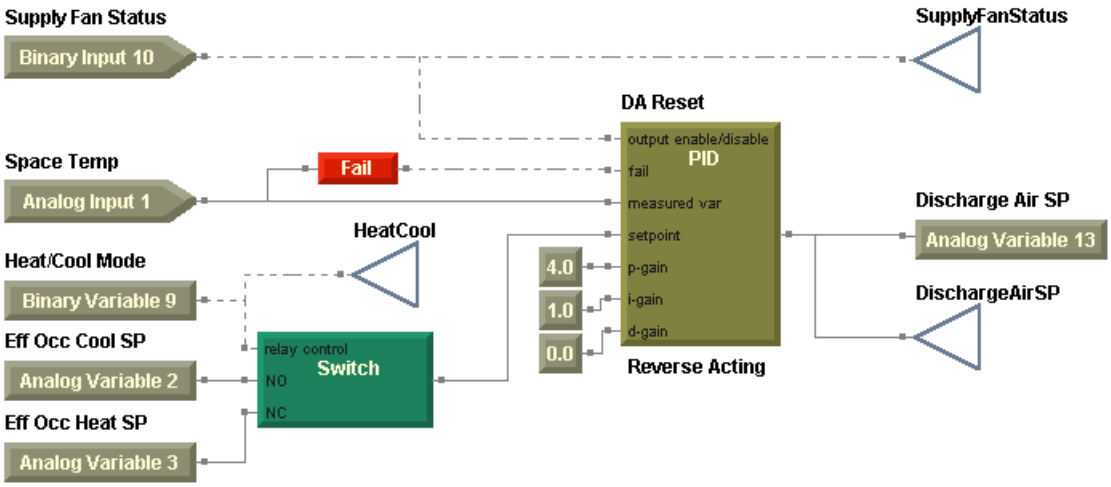
Figure 150: Controlling the humidifier



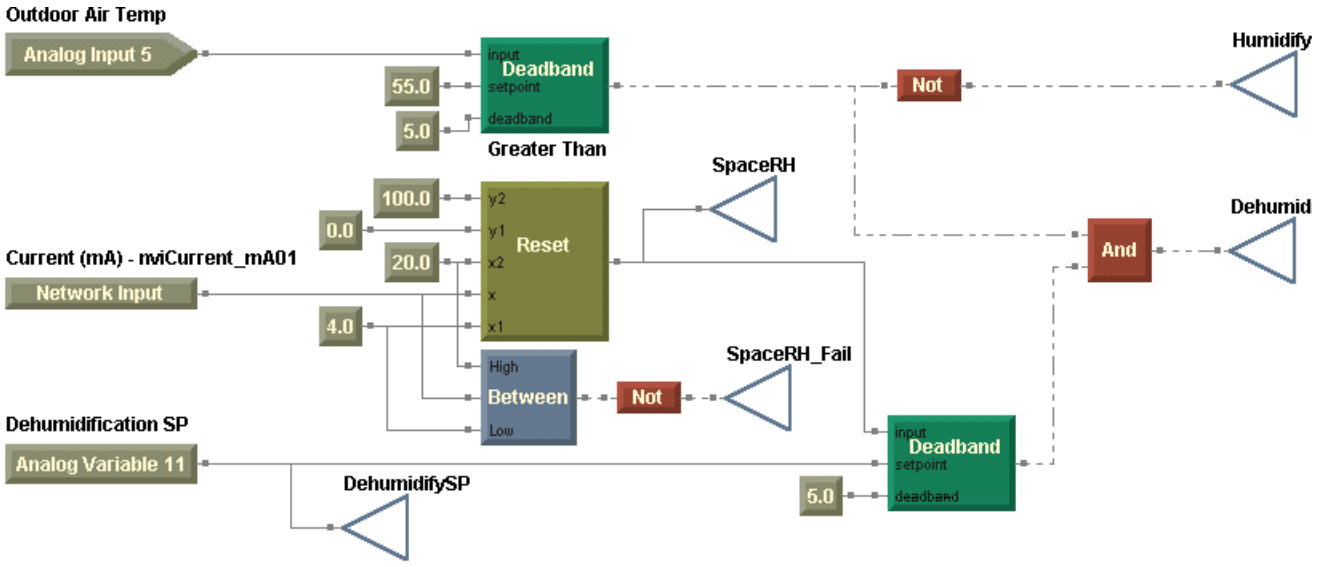
When the humidifier module is complete, compile and download this program (Figure 151) to the controller.

Figure 151: Completed discharge air control program

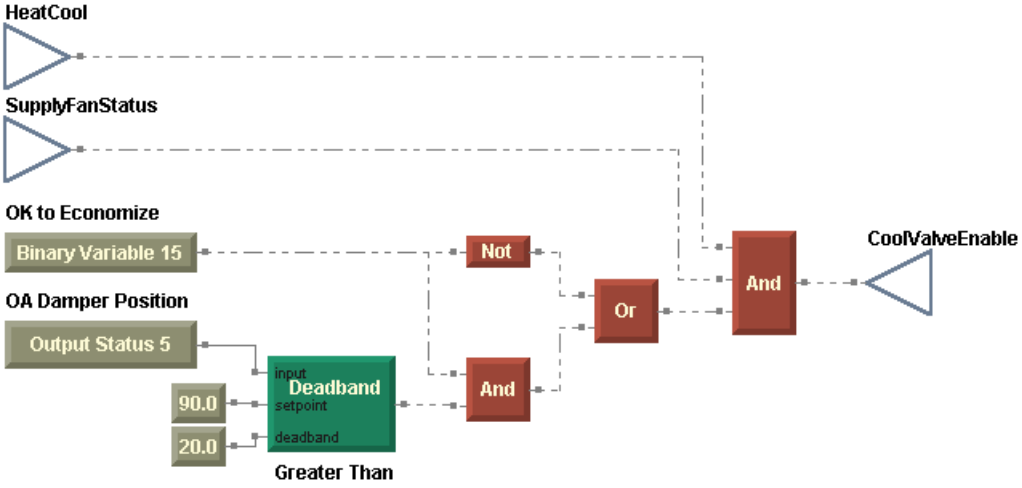
Discharge air reset



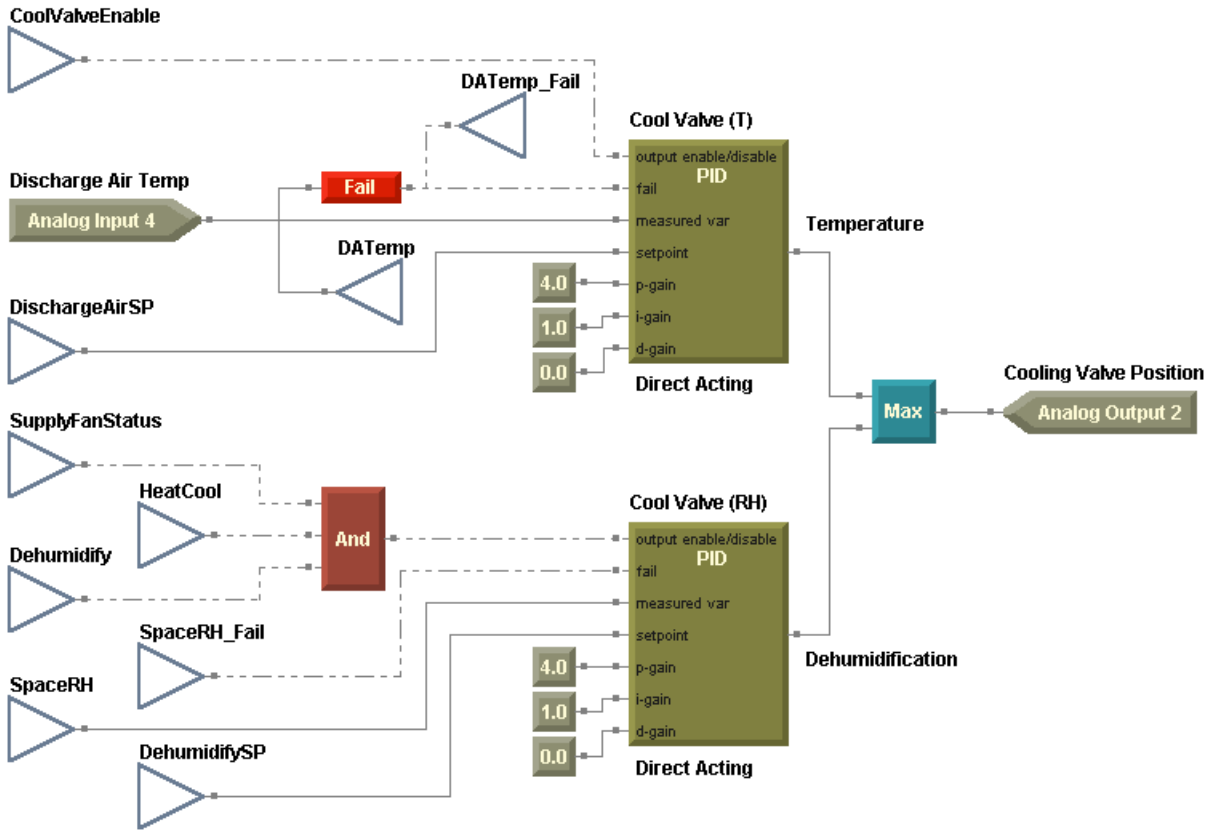
Whether to humidify/dehumidify



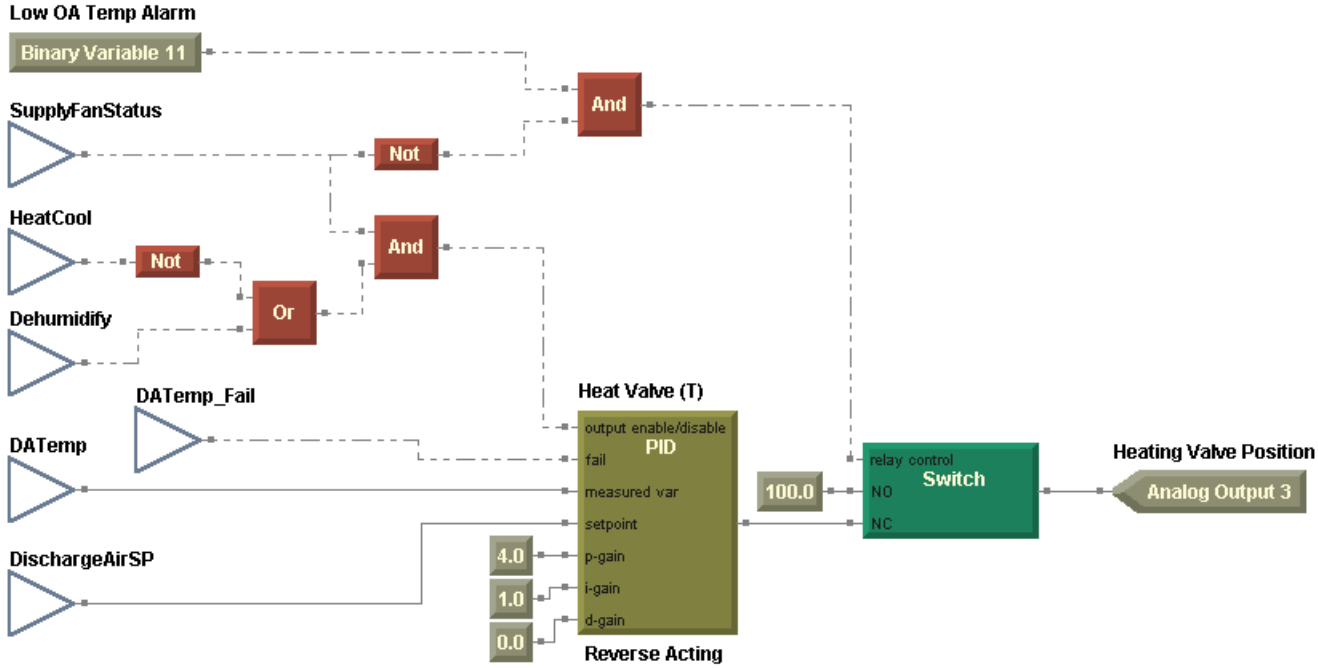
Operate the cooling valve?



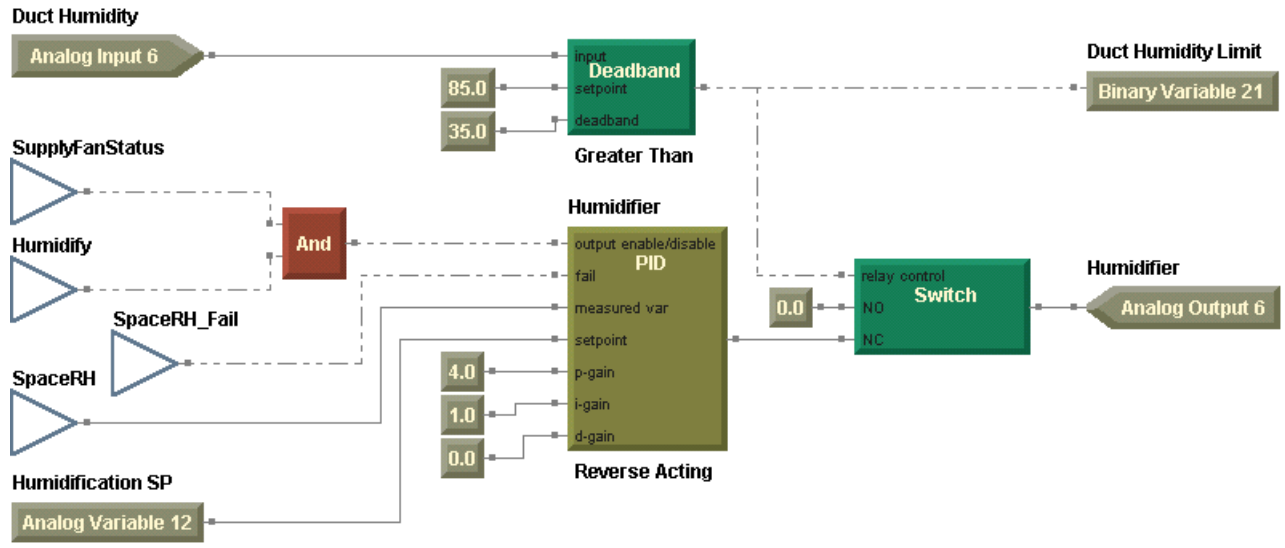
Controlling the cooling valve



Controlling the heating valve



Controlling the humidifier



Writing the alarms program

The alarms program indicates diagnostic conditions to the operator and protects the equipment from potential harm when such a condition exists. According to the sequence of operations, the following diagnostic conditions require an alarm indication:

- Dirty filter
- Duct humidity high limit
- Exhaust fan failure
- Low outdoor air temperature
- Low (mixed air) temperature detection
- Sensor failure (including discharge air temperature, mixed air temperature, outdoor air temperature, and space temperature)
- Supply fan failure

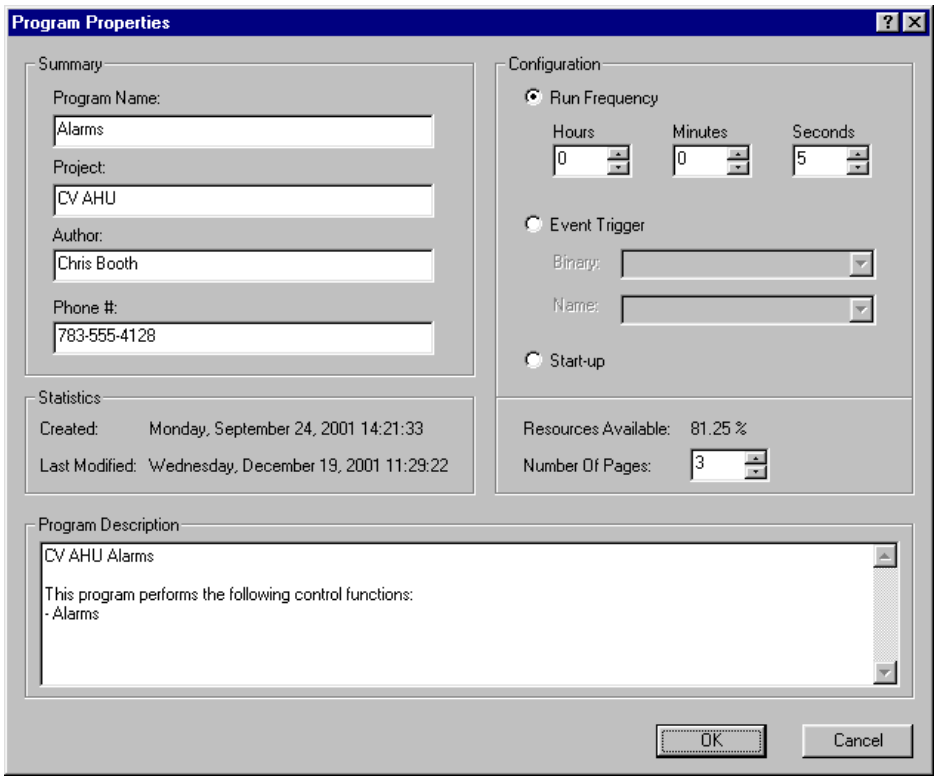
The following failures shutdown the air handler and require a manual reset:

- Discharge air or mixed air temperature sensor failure
- Fan failure
- Low mixed air temperature

All alarms must be resettable at the operator display.

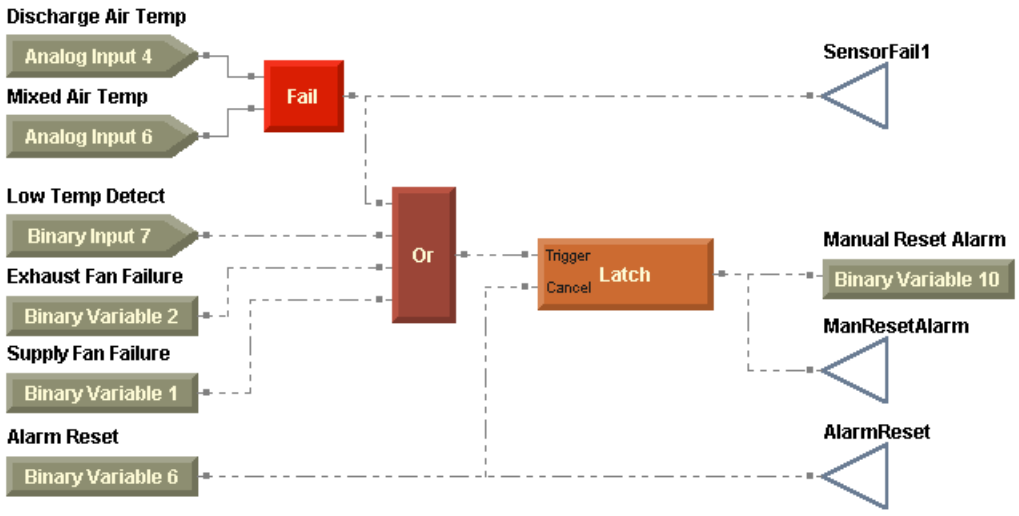
The requirements for the Alarms program are nearly identical to the same program in the VAV air handler with one exception: The duct static pressure high limit is no longer needed. The duct humidity high limit was added. The high limit for the duct static pressure is not applicable to a constant-volume AHU. The use of humidification requires the addition of a duct humidity high limit. Set the program properties as shown in Figure 152 on page 177.

Figure 152: Alarms program properties



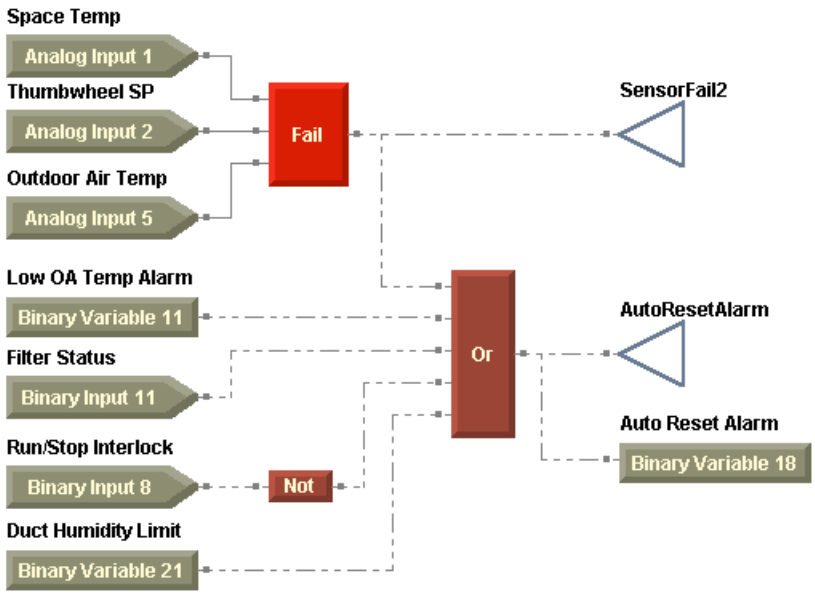
For the constant-volume air-handling unit, remove the duct static pressure limit from the manual reset alarms module as shown in Figure 153.

Figure 153: Manual reset alarms



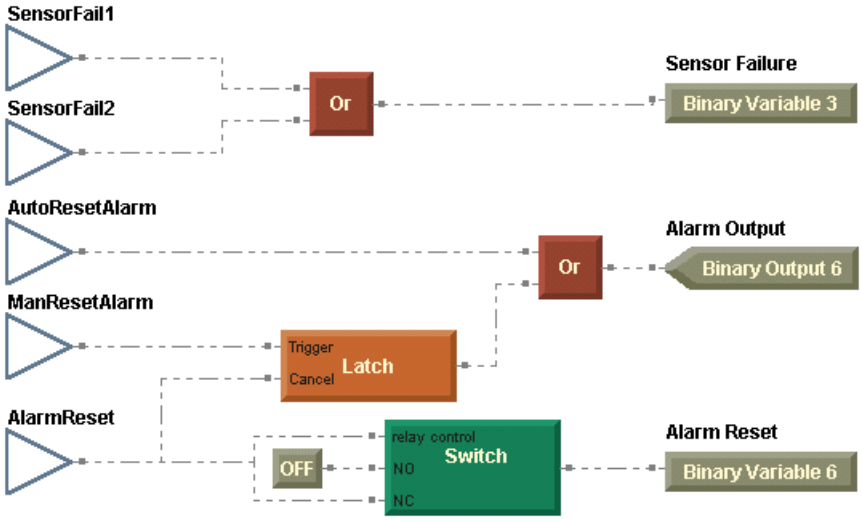
Add the duct humidity high limit to the auto-reset alarms module as shown in Figure 154.

Figure 154: Auto reset alarms



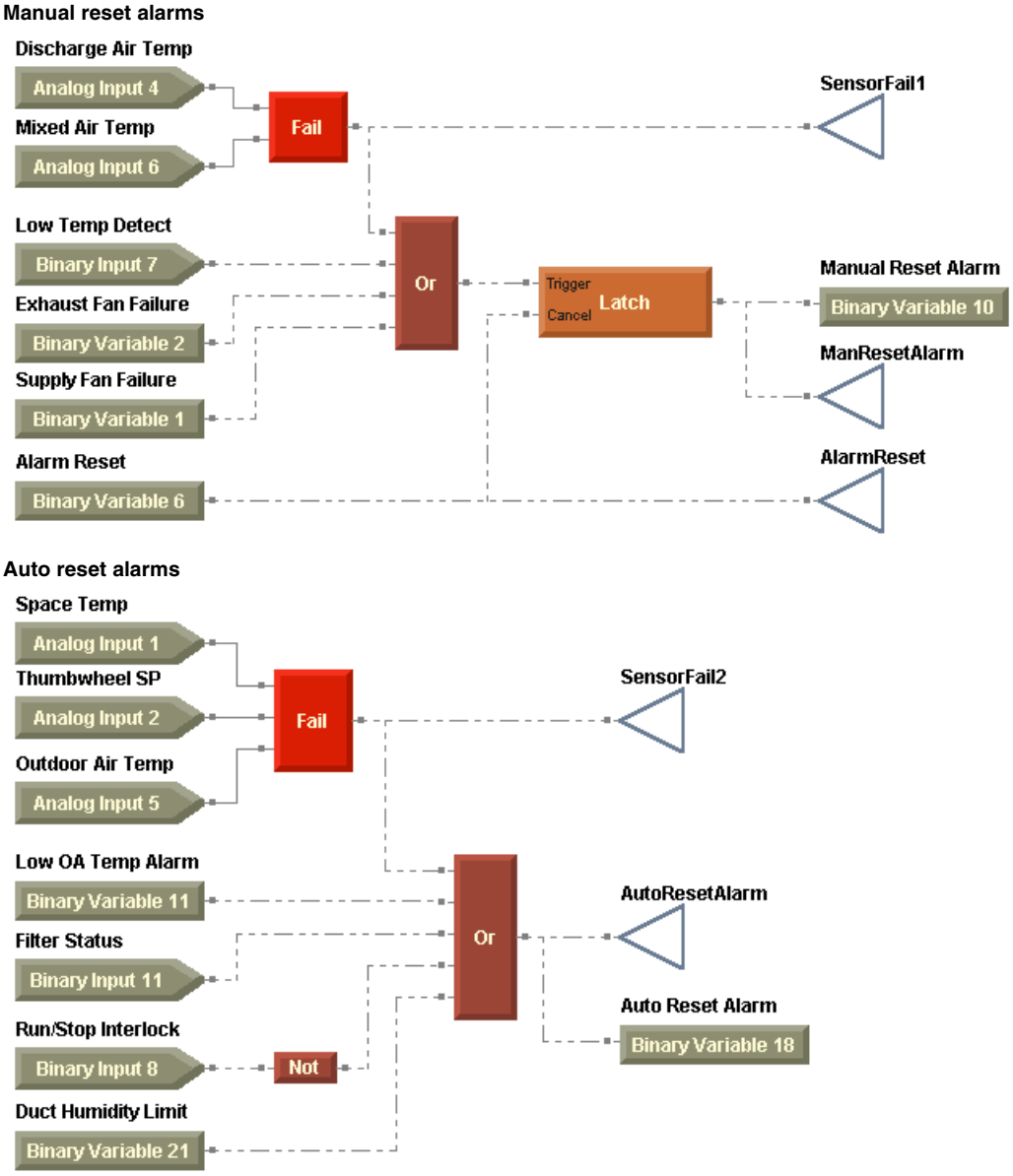
The alarm indication and reset module remains the same as the one you used in Chapter 6 (Figure 155).

Figure 155: Alarm indication and reset

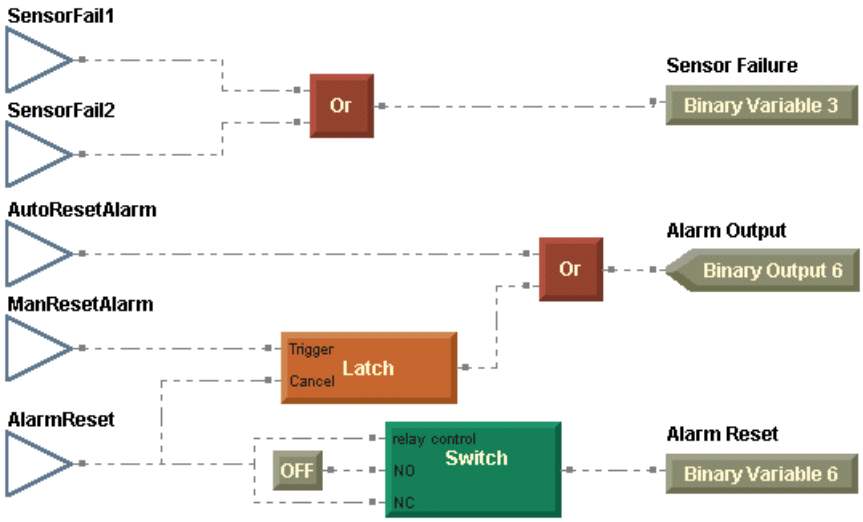


When your alarms program (Figure 156 on page 179) is complete, compile and download it to the controller.

Figure 156: Completed alarms program



Alarm indication and reset



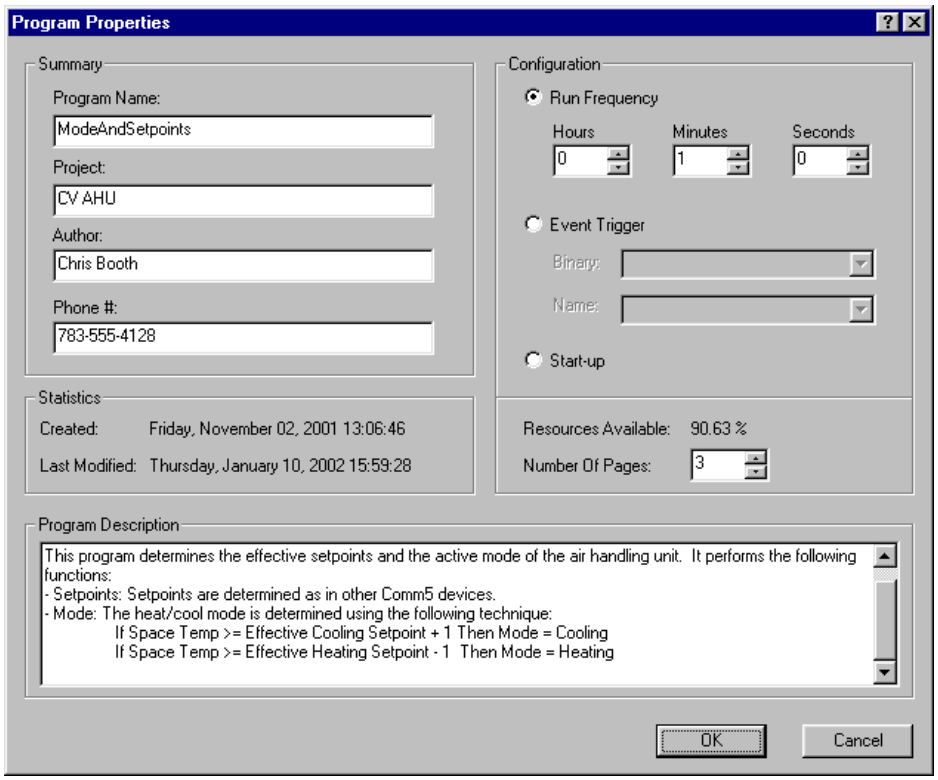
Writing the mode and setpoints program

The mode and setpoints program for this constant-volume air-handling unit calculates setpoints and determines the heat/cool mode. This program is nearly identical to that used for the VAV air-handling unit, with the following exceptions:

- Determination of the space temperature setpoint source is required by the sequence of operation.
- Discharge air setpoint validation is not required for a constant-volume air-handling unit.
- For this constant-volume air handler, the program implements a night heat/cool decision.

Set the program properties as shown in Figure 157 on page 181.

Figure 157: Mode and setpoints program properties



When complete, this program will perform the following tasks:

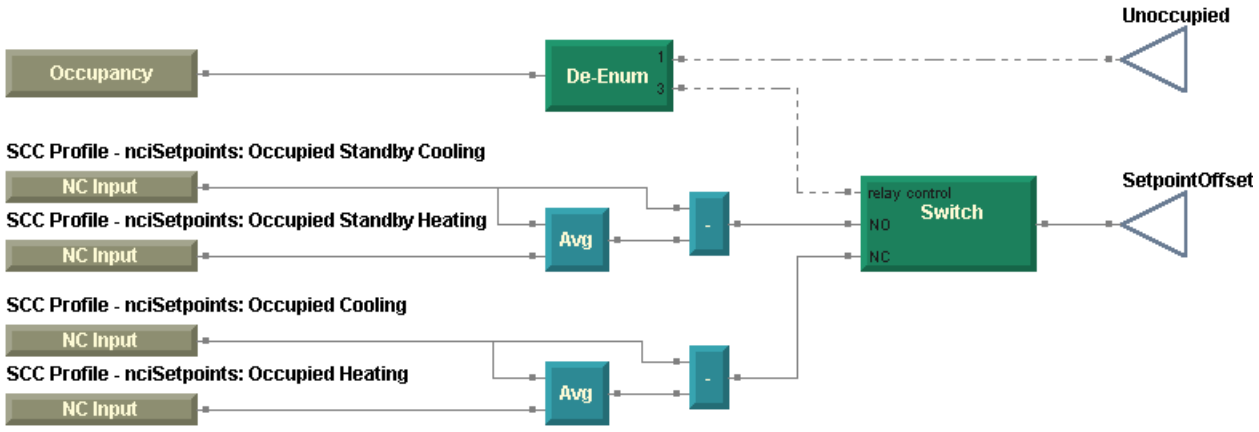
- Setpoint source determination
- Effective space setpoint calculation
- Heat/Cool mode determination, including night heat/cool

Implementing effective space setpoint calculation

The effective space setpoint calculation for the constant-volume air-handling unit is exactly the same as that used for the VAV air-handling unit. First, determine the setpoint based on the occupancy mode and the default setpoints as shown in Figure 158 on page 182.

One important distinction exists between the setpoint calculation module here and the one you used for the VAV air-handling unit. Instead of using default setpoints associated with the DAC profile, use the default setpoints associated with the SCC profile.

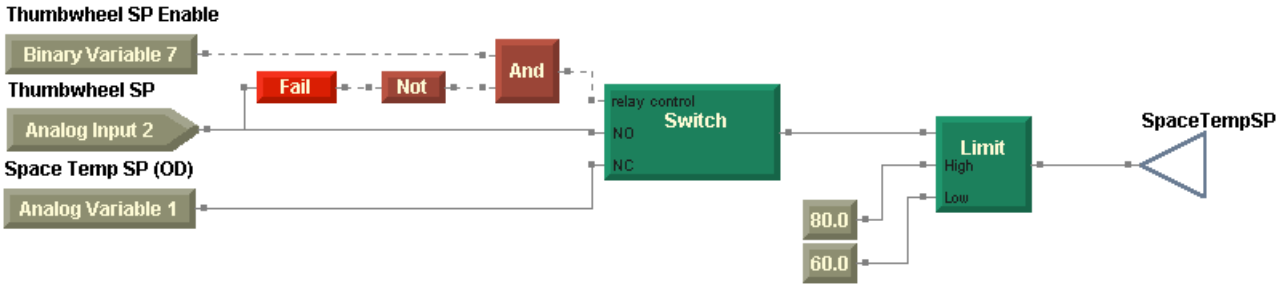
Figure 158: Offset calculation



Determine the space temperature setpoint source before calculating the effective setpoints using the resultant offset as shown in Figure 159. According to the sequence of operation, the space temperature setpoint may originate from the operator display or from a setpoint adjustment knob on a local wired thumbwheel.

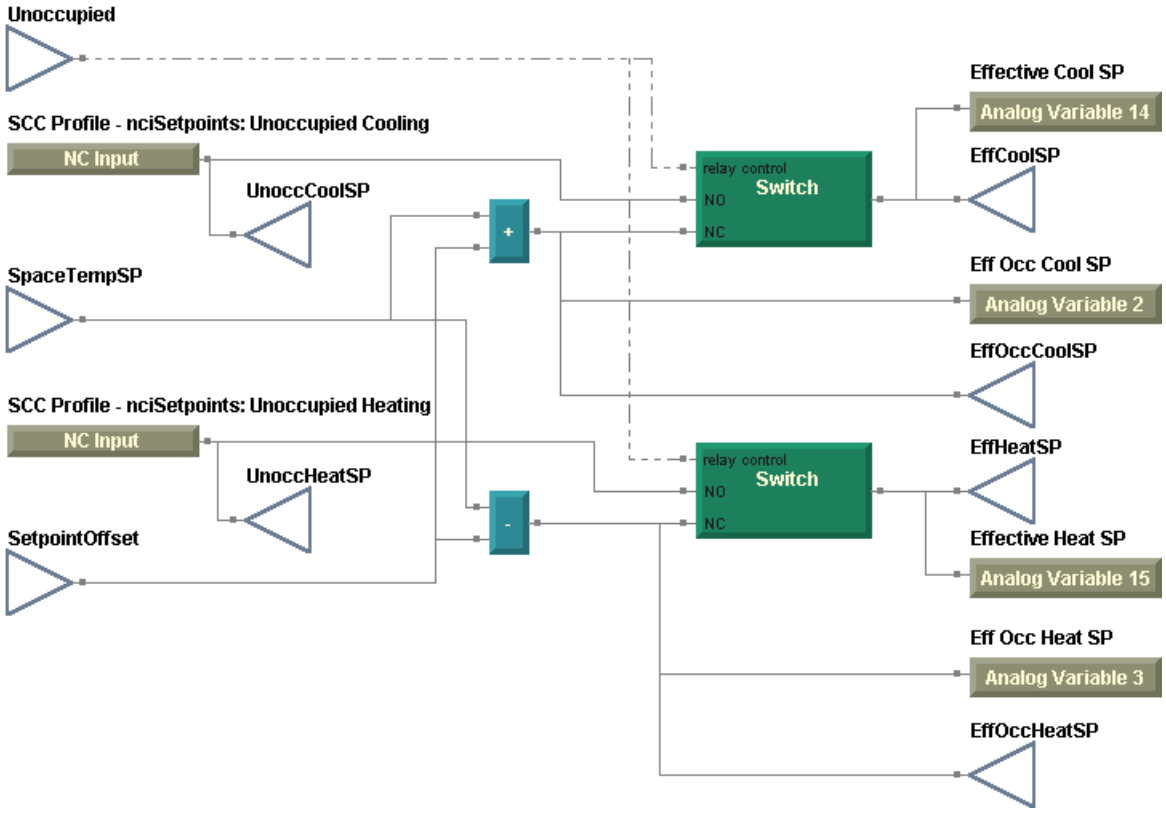
- Use the binary variable, Thumbwheel SP Enable, to activate the thumbwheel setpoint.
- If the thumbwheel setpoint fails, use the space temperature setpoint from the operator display.
- Use the Limit block to apply appropriate limits to the setpoint.
- Use a wireless connection to pass the resulting space temperature setpoint to the effective setpoint calculation.

Figure 159: Space temperature setpoint source determination



Use the resultant space temperature setpoint and the calculated offset in the effective setpoint calculation (Figure 160 on page 183), as in the VAV air-handling unit.

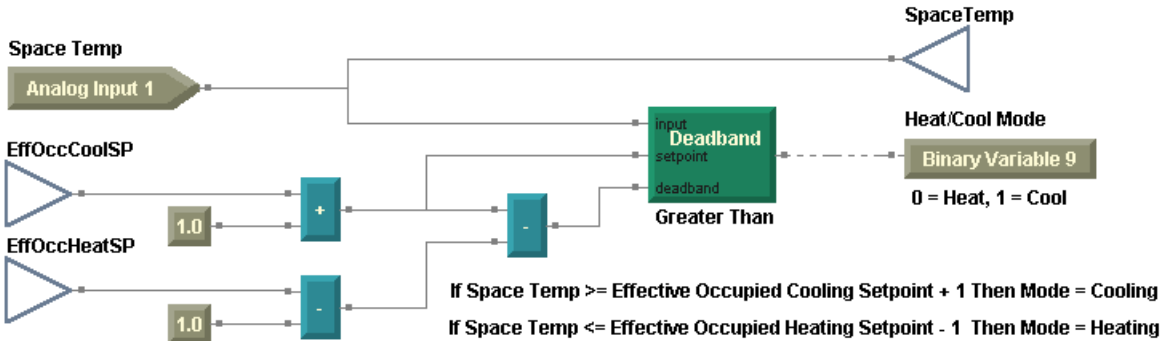
Figure 160: Effective setpoint calculation



Determining the heat/cool mode

Use the same module to determine the heat/cool mode as you did in the VAV air-handling unit (Figure 161).

Figure 161: Heat/Cool mode decision



Adding night heat/cool

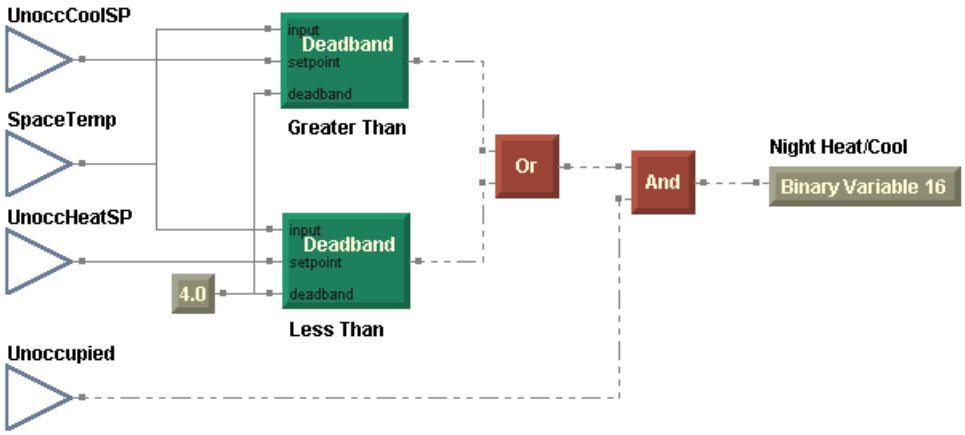
Consider the following important points about night heat/cool mode from the sequence of operation.

- The air-handling unit must be unoccupied.
- The supply fan starts if the space temperature falls below the unoccupied heating setpoint or the space temperature rises above the unoccupied cooling setpoint.
- The supply fan stops if the space temperature rises above the unoccupied heating setpoint or the space temperature falls below the unoccupied cooling setpoint.
- When in night heat/cool mode and heating, the outdoor air damper remains closed.
- When in night heat/cool mode and cooling, the outdoor air damper modulates if the outside air temperature is less than the economizer changeover setpoint and the economizer is enabled.

Implement the night heat/cool mode decision as shown in Figure 162.

- Use the Deadband blocks to turn the night heat/cool mode on and off. (A deadband value of 4°F is used in this example.)
- Use the Night Heat/Cool variable to allow for implementation of night heat/cool mode in other programs.
- Use an And block to make sure that the air-handling unit is unoccupied before entering night heat/cool mode.

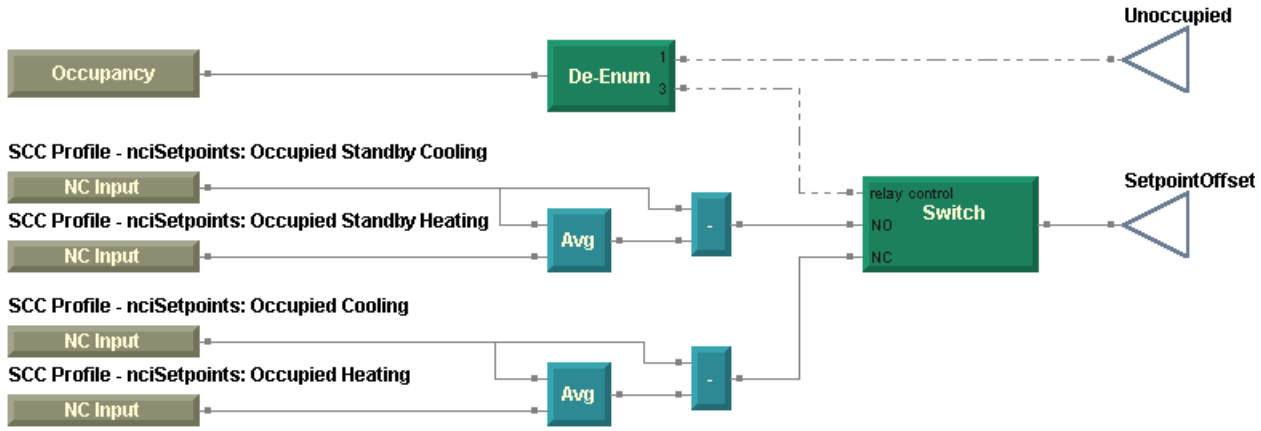
Figure 162: Night heat/cool decision



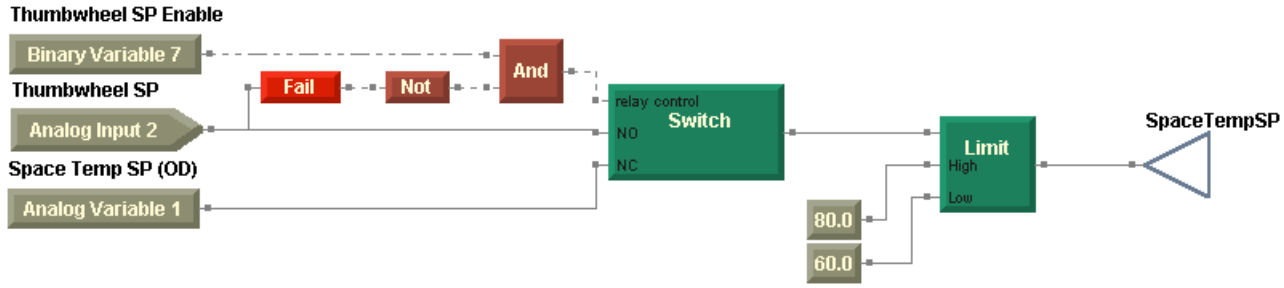
The mode and setpoints program (Figure 163 on page 185) completes the constant volume air-handling unit. Compile and download the program.

Figure 163: Completed modes and setpoints program

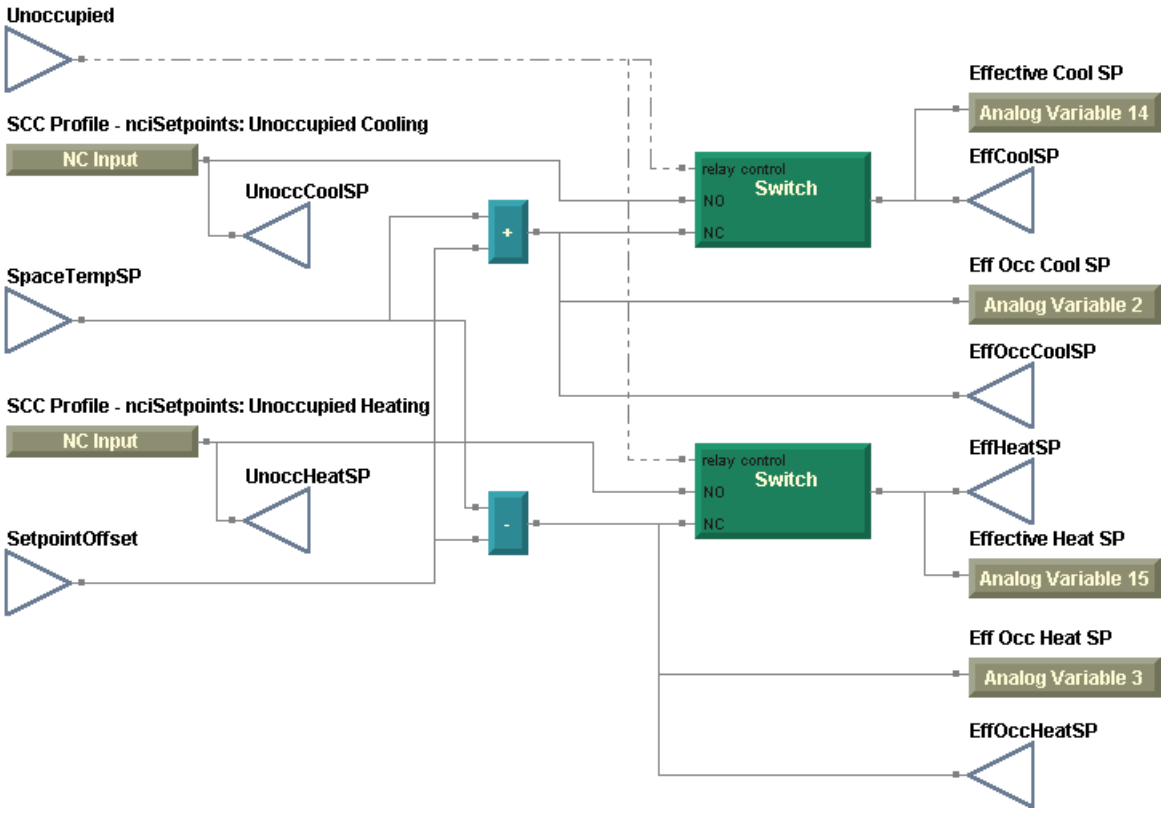
Offset calculation



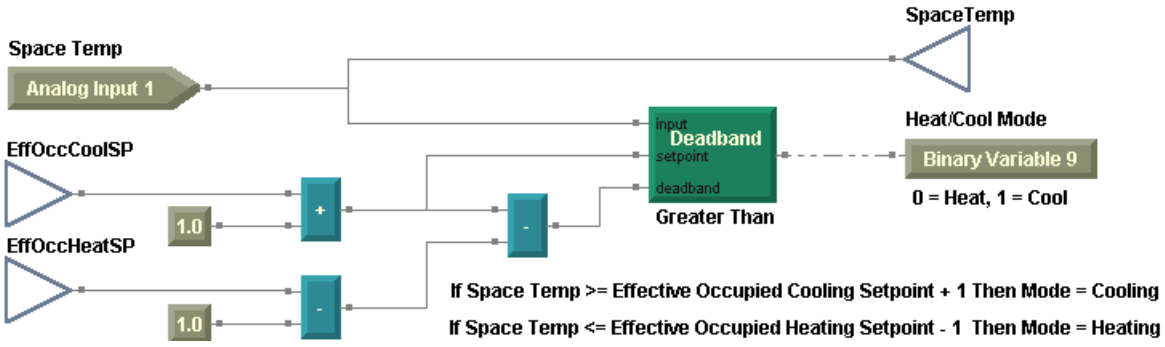
Space temperature setpoint source determination

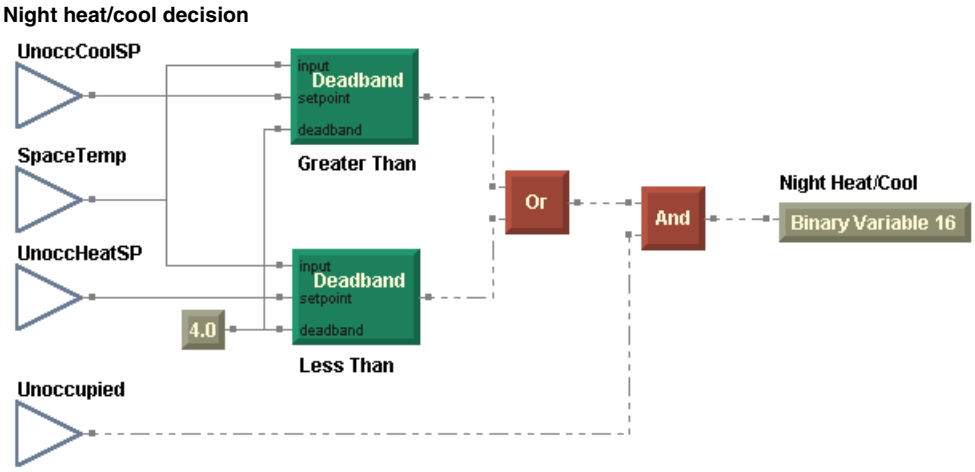


Effective setpoint calculation



Heat/Cool mode decision





Summary questions

Answer the following questions to review the skills, concepts, and definitions you learned in this chapter. The answers to these questions are on page 241.

1. What configuration property of the Tracer MP580/581 controller must be set for the unit to enter occupied standby mode?

2. Occupied bypass mode, previously known as timed override, is mentioned in the sequence of operation. How is occupied bypass mode accounted for in the configuration and programming?

3. Data communicated from other devices is not always reliable. For instance, a temperature sensed by another Comm5 device may be invalid. This may be caused by a communications failure. How would you account for the possibility of an invalid temperature value?

Chapter 8

Constant-volume AHU with warm-up, pre-cool, and communications

In this chapter, you will continue to hone your programming skills. You will modify the constant-volume air-handler programs you created in Chapter 7 to accommodate additional modes and communications with a Tracer Summit building automation system.

Note:
Many of the chapters in this book build on previous chapters, so be sure to complete the chapters in the order presented. See “About this book” on page 1 for additional instructions.

What you will learn

In this chapter, you will learn a variety of skills, concepts, and definitions.

Concepts and definitions

You will understand the following concepts and definitions:

- How to use a LonMark profile for an air-handling application
- How to write a program to accommodate communications with a building automation system (BAS)

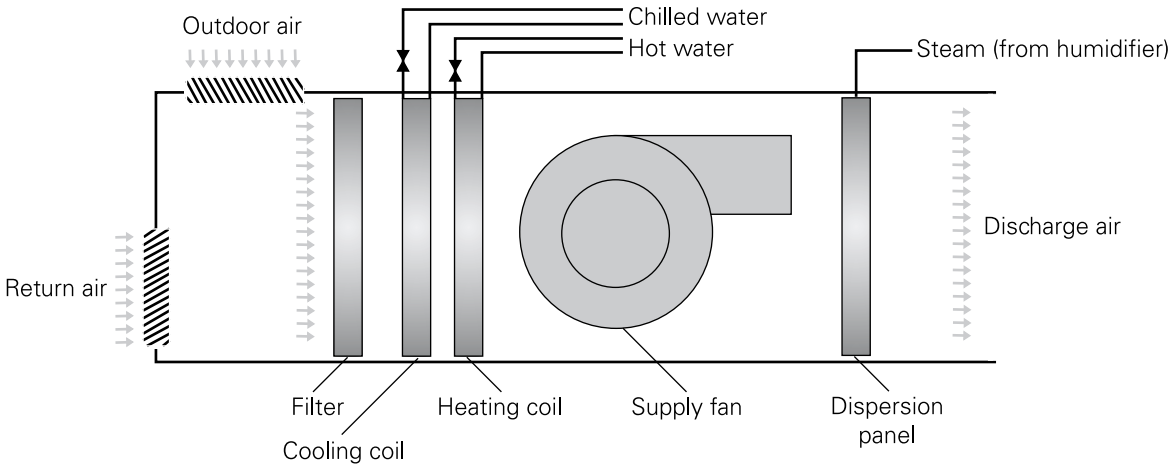
Block

You will learn how to use the Network Variable Output block.

Reviewing the sequence of operation

In this scenario a constant-volume air handler provides comfort heating and cooling to a space. The air handler contains a fan, both cooling and heating coils, and an outdoor air damper (Figure 164).

Figure 164: Constant-volume air handler



Modes, setpoints, and communications

The following parameters for the sequence of operation are presented according to the occupancy mode, the heat/cool mode, or the setpoint.

Occupied mode (including occupied bypass)

When the air handler is in occupied mode, operate the supply fan continuously and modulate the cooling valve, heating valve, and outside air damper to maintain space comfort conditions. During unoccupied periods, when the local timed override button (the ON button on the space temperature sensor) is pressed, transition the air handler to occupied bypass mode for the configured occupied bypass time period. When the local CANCEL button is pressed, terminate the timed override request and revert the air handler to the scheduled mode.

Occupied standby mode

When the air handler is in occupied standby mode, operate the supply fan continuously and modulate the cooling valve, heating valve, and outside air damper to maintain space comfort conditions. Use occupied standby mode to save energy by using lower heating and higher cooling temperature setpoints and reduced ventilation requirements during intermittent unoccupied periods.

An occupancy sensor determines the presence of people in the space served by the air-handling unit. When the space is occupied, operate the air handler in occupied mode. When the space is unoccupied during a scheduled occupancy period, control the space temperature to follow the

occupied standby setpoints and maintain the outdoor air damper to its minimum position.

Unoccupied mode (including night heat/cool)

When the air handler is in unoccupied mode, turn the supply fan off, close the outside air damper fully, and close the cooling and heating valves fully. Open the heating valve completely if the outdoor air temperature falls below the freeze avoidance setpoint, 35°F (adjustable).

If the space temperature falls below the unoccupied heating setpoint, start the supply fan and transition the air handler to heating and keep the outdoor air damper closed to maintain the unoccupied heating setpoint. When the space temperature rises above the unoccupied heating setpoint, shut down the air handler.

If the space temperature rises above the unoccupied cooling setpoint, start the supply fan and transition the air handler to cooling to maintain the unoccupied cooling setpoint. Modulate the outside air damper if the outside air temperature is less than the economizer changeover setpoint. When the space temperature falls below the unoccupied cooling setpoint, shut down the air handler.

Warm-up and pre-cool modes

During occupied and occupied standby modes, when there is a call for heating and the space temperature is 3°F or more below the effective occupied heating setpoint, initiate a warm-up sequence. During warm-up, operate the supply fan continuously, close the outdoor air damper, and open the heating valve to 100%. When the space temperature comes within 2°F of the heating setpoint, open the outside air damper to the occupied minimum position and transition the air handler to occupied mode.

During occupied and occupied standby modes, when there is a call for cooling and the space temperature is 3°F or more above the effective occupied cooling setpoint, initiate a morning cool-down sequence. Operate the supply fan continuously and the air handler in the economizing mode, if possible. If economizing is not possible, open the cooling valve to 100% and keep the outside air damper closed. When the space temperature reaches the cooling setpoint, open the outside air damper to the occupied minimum position and transition the air handler to occupied mode.

Setpoints

For occupied, occupied standby, and occupied bypass modes, calculate effective cooling and heating setpoints based on a single space temperature setpoint and the configured default occupied and occupied standby setpoints. The space temperature setpoint source may be the building automation system, the operator display, or a wired thumbwheel setpoint

adjustment knob, with the building automation system setpoint being given first consideration.

Note:
Calculate the effective cooling and heating setpoints as in other Comm5 devices. See "Calculating the effective space setpoints" on page 135 for more information.

Determine the discharge air temperature setpoint through a comparison of the space temperature and the space temperature setpoint. Reset the discharge air temperature setpoint according to heating or cooling demand.

Heat/Cool arbitration

Determine the heat/cool decision of the air handler based on the effective occupied or occupied standby cooling and heating setpoints. Transition the air handler to cooling if the space temperature exceeds the cooling setpoint plus 1°F. Transition the air handler to heating if the space temperature falls below the heating setpoint minus 1°F.

Communications

The air handler is part of a Tracer Summit building automation system. The following items are communicated to the air handler from the building automation system:

- Occupancy
- Space temperature setpoint
- Outdoor air temperature
- Outdoor air humidity
- Space relative humidity
- Economizer enable

If communication with the BAS is lost, program the air handler to use its predetermined default setpoints and operate in the occupied mode.

The air handler must communicate the following items to the building automation system so that the building operator can view the status of the air-handling unit:

- Space temperature
- Discharge air temperature
- Mixed air temperature
- Effective setpoint
- Effective occupancy
- Supply fan status
- Cooling valve status
- Heating valve status
- Outdoor air damper status
- Alarm status
- Application mode

Control

The following parameters for the sequence of operation are presented according to the equipment that must be controlled.

Supply fan

Operate the supply fan continuously whenever the air handler is in occupied or night heat/cool modes. Turn the supply fan off whenever one of the following occurs:

- The air handler is unoccupied.
- The run/stop interlock is open.
- The mixed air temperature is too cold, and the low limit detection is closed.
- The supply fan status indicates a fan failure after a 1-minute delay.

Note:
A failure due to low mixed air temperature or fan failure requires a manual reset. Also, the low limit switch may require a manual, hardware reset.

Outdoor air damper

When the economizer function is enabled and the outdoor air temperature is less than the economizer changeover setpoint, modulate the outdoor air damper between the adjustable minimum position and fully open to maintain the discharge air cooling setpoint. Modulate the outdoor air damper closed, overriding the minimum position, to maintain the mixed air temperature at or above the mixed air setpoint.

If the economizer function is disabled or the air handler is in the heat mode, control the outdoor air damper to its minimum position. If the outdoor air temperature falls below a low ambient damper lockout setpoint, control the outdoor air damper to its closed position. In unoccupied mode, control the outdoor air damper to its closed position. Also, if the supply fan is off or the mixed air temperature sensor is failed, control the outdoor air damper to its closed position.

Exhaust fan

Coordinate exhaust fan operation with the unit supply fan and outdoor air damper position. Energize the exhaust fan whenever the supply fan is on and the outdoor air damper is open beyond 30%. Keep the exhaust fan on until the outdoor air damper closes to below 20% open or the supply fan is turned off.

Cooling valve

Modulate the cooling valve to maintain the discharge air temperature at the discharge air cooling setpoint. If the economizer function is enabled and the outdoor air damper is not open at least 90%, control the cooling valve to its closed position. Also, close the cooling valve if the air handler

is in heat mode, the supply fan is off, or the discharge air temperature sensor is failed.

Heating valve

Modulate the heating valve to maintain the discharge air temperature at the discharge air heating setpoint. Close the heating valve if the air handler is in cool mode, the supply fan is off, or the discharge air temperature sensor is failed. Open the heating valve if the supply fan is off and the outdoor air temperature falls below the adjustable freeze avoidance setpoint.

Humidification

Enable humidification when the air handler is in occupied mode, the supply fan is on, and the outdoor air temperature falls below 55°F. Modulate the humidifier to maintain the space relative humidity setpoint. If the duct relative humidity exceeds 85%, turn off the humidifier to prevent condensation in the duct work and indicate an alarm.

Dehumidification

Enable dehumidification mode when the air handler is occupied, the supply fan is on, the outdoor air temperature is above 55°F, and the space relative humidity is above the space dehumidification setpoint. Modulate the cooling valve to maintain space relative humidity and the heating valve to maintain discharge air temperature.

Alarms

In addition to the alarm requirements mentioned above, indicate an alarm at the operator display and turn on the alarm output when any sensor fails. Alarms must be resettable at the operator display. Diagnostics conditions include the following:

- Dirty filter
- Duct humidity high limit
- Exhaust fan failure
- Low outdoor air temperature
- Low (mixed air) temperature detection
- Sensor failure (including discharge air temperature, mixed air temperature, outdoor air temperature, space temperature, and duct static pressure)
- Supply fan failure

The following failures shutdown the air handler and require a manual reset:

- Discharge or mixed air temperature sensor failure
- Fan failure
- Low mixed air temperature

The corresponding data definition is presented in Table 23 and Table 24, and a wiring diagram is presented in Figure 165 on page 198.

Table 23: Modified constant-volume AHU inputs and outputs data definition

Inputs and outputs		Type	Name	Notes
Inputs	1	Analog	Space Temp	Universal input configured as thermistor or RTD
	2	Analog	Thumbwheel SP	Universal input configured as thermistor or RTD
	3	Analog	Mixed Air Temp	Universal input configured as thermistor or RTD
	4	Analog	Discharge Air Temp	Universal input configured as thermistor or RTD
	5	Analog	Outdoor Air Temp	Universal input configured as thermistor
	6	Analog	Duct Humidity	
	7	Binary	Low Temp Detect	
	8	Binary	Run/Stop Interlock	
	9	Binary	Occupancy/Generic	
	10	Binary	Supply Fan Status	
	11	Binary	Filter Status	
	12	Binary	Exhaust Fan Status	
	Pressure		Duct Static Pressure*	
Binary outputs	1		Supply Fan Start/Stop	
	2		Exhaust Fan Start/Stop	
	3		Not Used	
	4		Not Used	
	5		Not Used	
	6		Alarm Output	Status/custom alarm indicator
Analog outputs	1		Supply Fan Speed*	Analog output configured as voltage, 0–10 V
	2		Cooling Valve Position	Analog output configured as voltage, 0–10 V
	3		Heating Valve Position	Analog output configured as voltage, 0–10 V
	4		Face and Bypass Damper*	Analog output configured as voltage, 0–10 V
	5		OA Damper Position	Analog output configured as voltage, 0–10 V
	6		Humidifier	Analog output configured as voltage, 0–10 V

* These inputs and outputs are not used in this program but are often used in air-handler programs.

Note: The inputs and outputs in this table are configured so that a Tracer AH540 main logic board could be replaced with a Tracer MP580/581 board, except for the Mixed Air Temp, which is input 6 on the Tracer AH540.

Table 24: Modified constant-volume AHU variables data definition

Variable	Name	Notes
Tracer Summit binary variables	Economizer Enable	
	BAS Setpoint Enable	
	Alarm Reset (BAS)	

Table 24: Modified constant-volume AHU variables data definition (continued)

Variable	Name	Notes
Local binary variables	Supply Fan Failure	Sourced from a program
	Exhaust Fan Failure	Sourced from a program
	Sensor Failure	Sourced from a program
	Duct Humidity Limit	Sourced from a program
	Low OA Temp Alarm	Sourced from a program
	Fan Failure Reset	Sourced from operator display and a program
	Alarm Reset	Sourced from operator display and a program
	Manual Reset Alarm	Sourced from a program
	Auto Reset Alarm	Sourced from a program
	Thumbwheel SP Enable	Sourced from operator display
	Heat/Cool Mode	Sourced from a program
	OK to Economize	Sourced from a program
	Humidify	Sourced from a program
	Dehumidify	Sourced from a program
	Night Heat/Cool	Sourced from a program
	Warm-Up	Sourced from a program
	Pre-Cool	Sourced from a program
Tracer Summit analog variables	Not Used	
Local analog variables	Space Temp SP (OD)	Setpoint sourced from the operator display
	Eff Occ Cool SP	Sourced from a program
	Eff Occ Heat SP	Sourced from a program
	Effective Cool SP	Sourced from a program
	Effective Heat SP	Sourced from a program
	Economizer Changeover SP	Sourced from the operator display
	Discharge Air SP	Sourced from a program
	Low OA Temp SP	Setpoint sourced from the operator display
	Mixed Air Temp SP	Setpoint sourced from the operator display
	Dehumidification SP	Setpoint sourced from the operator display
	Humidification SP	Setpoint sourced from the operator display
	Ex Fan OA Damper SP	Setpoint sourced from the operator display
	Ex Fan OA Damper Dbd	Setpoint sourced from the operator display

Before you write the program, configure these inputs, outputs, and variables in your Tracer MP580/581 controller. Because you are programming a constant-volume air-handling unit, set the controller to use the Space Comfort Controller (SCC) profile. For more information about configuring the controller, including setting it to use the SCC profile, see the *Tracer MP580/581 Programmable Controller Programming* guide (CNT-SVP01A-EN). The SCC profile provides the configuration data listed in Table 25 on page 197, and the network variable inputs and out-

puts listed in Table 26 and Table 27. Then make sure that the TGP editor is open and ready to go.

Table 25: Network configuration inputs for the SCC profile

nci	SNVT	Notes
nciBypassTime	SNVT_time_min	Occupied bypass time
nciOAMinPos	SNVT_lev_percent	Outdoor air damper minimum position
nciSpaceCO2Lim	SNVT_ppm	Space CO_2 limit
nciSpaceRHSetpt	SNVT_lev_percent	Space relative humidity setpoint
nciSetpoints	SNVT_temp_setpt	Provides default unoccupied, occupied, and occupied standby cooling and heating setpoints

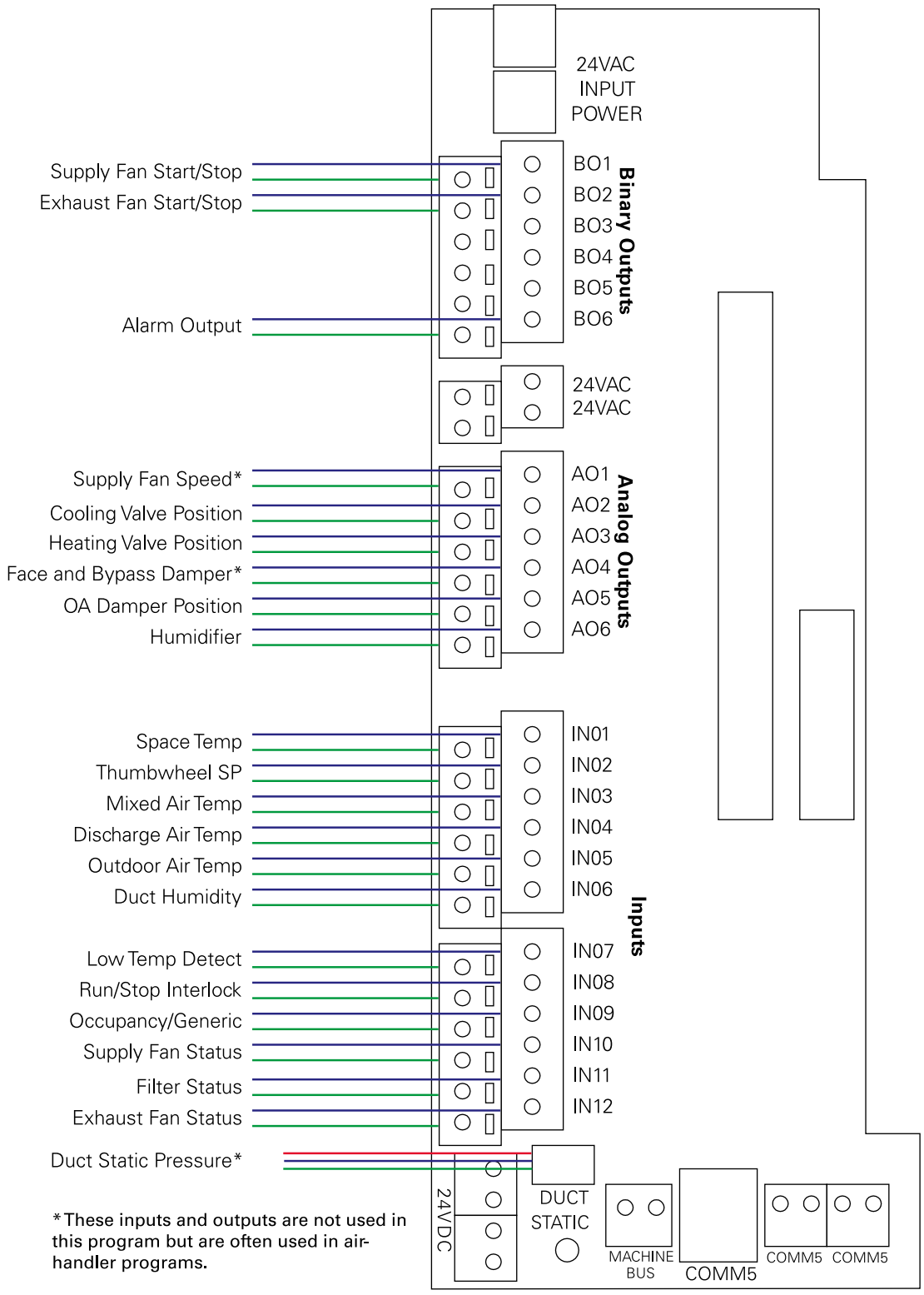
Table 26: Network variable inputs for the SCC profile

nvi	SNVT	Notes
nviSpaceTemp	SNVT_temp_p	Space temperature
nviSetpoint	SNVT_temp_p	Space temperature setpoint
nviOccSchedule	SNVT_tod_event	Schedule input
nviOccManCmd	SNVT_occupancy	Unit override (enumerated)
nviOccSensor	SNVT_occupancy	Occupancy sensor input (enumerated)
nviApplicMode	SNVT_hvac_mode	Application mode (enumerated)
nviValveOverride	SNVT_hvac_overid	Cooling and heating valve override (structure)
nviFlowOverride	SNVT_hvac_overid	Air flow override (structure)
nviEmergOverride	SNVT_hvac_emerg	Emergency override
nviOutdoorTemp	SNVT_temp_p	Outdoor air temperature
nviSpaceRH	SNVT_lev_percent	Space relative humidity
nviSpaceCO2	SNVT_ppm	Space CO_2 level

Table 27: Network variable outputs for the SCC profile

nvo	SNVT	Notes
nvoSpaceTemp	SNVT_temp_p	Space temperature
nvoUnitStatus	SNVT_hvac_status	Unit status (structure)
nvoEffectSetpt	SNVT_temp_p	Effective setpoint
nvoEffectOccup	SNVT_occupancy	Effective occupancy (enumerated)
nvoDischAirTemp	SNVT_temp_p	Discharge air temperature
nvoSpaceRH	SNVT_lev_percent	Space relative humidity
nvoSpaceCO2	SNVT_ppm	Space CO_2 level
nvoMATemp	SNVT_temp_p	Mixed air temperature

Figure 165: Modified constant-volume AHU wiring diagram



Determining a programming approach

The programs created in Chapter 7 serve as the starting point for all of the programs in this chapter. Modify the programs to apply the new warm-up and pre-cool modes and write a new program to incorporate some of the communications functions with the Tracer Summit building automation system.

From the sequence of operation, determine the basic control functions required. The required control functions are very similar to those in the Chapter 7, "Constant-volume AHU example." In this case, the following control functions must be addressed.

- Effective space setpoint calculation
- Mode determination, including night heat/cool, pre-cool, and warm-up
- Supply fan control
- Exhaust fan control
- Mixed air/outdoor air damper control
- Discharge air setpoint calculation
- Cooling valve control, including dehumidification
- Heating valve control, including dehumidification
- Humidification
- Alarm management
- Communications

Not every control function fits into one program. Determine the number of programs required by grouping control functions that require the same information and/or the same run frequency as shown in Table 28.

Table 28: Control functions within each program

Program name	Control functions
FanControl	Supply fan control
	Exhaust fan control
MixedAirControl	Mixed air/outdoor air damper control
DischargeAirControl	Discharge air setpoint calculation
	Cooling valve control
	Heating valve control
	Dehumidification
	Humidification
Alarms	Alarm management
ModeAndSetpoints	Effective space setpoint calculation
	Setpoint source determination
	Mode determination
Communications	Communications to BAS

Writing the fan control program

Start with the fan control program from Chapter 7. Does the sequence of operation require any changes to this program? No. This entire program remains exactly the same as it appeared in Chapter 7. See “Writing the fan control program” on page 155 for more information.

Writing the mixed air control program

Operation of the outdoor air damper and subsequent mixed air temperature control must satisfy a number of scenarios according to the sequence of operation. The changes for this chapter are shown in *italics*.

Modulate the outdoor air damper between the adjustable minimum position and fully open to maintain the discharge air cooling setpoint when *all* of the following are true:

- The economizer is enabled.
- The outdoor air temperature is less than the economizer changeover (or outdoor air temperature) setpoint.
- *The air handler is in cooling mode, including night heat/cool and pre-cool modes.*

Modulate the outdoor air damper closed, overriding the minimum position, to maintain the mixed air temperature at or above the mixed air setpoint.

Control the outdoor air damper to its minimum position when *either* of the following is true:

- The economizer function is disabled.
- The air handler is in heating mode.

Close the outdoor air damper completely when *any* of the following is true:

- The unit is in unoccupied mode.
- The unit is in night heat/cool mode and heating.
- The unit is in night heat/cool mode and cooling, and economizing is not allowed.
- *The unit is in warm-up mode.*
- *The unit is in pre-cool mode, and economizing is not allowed.*
- The outdoor air temperature falls below the low outdoor air temperature setpoint.
- The supply fan is off.
- The mixed air temperature sensor is failed.

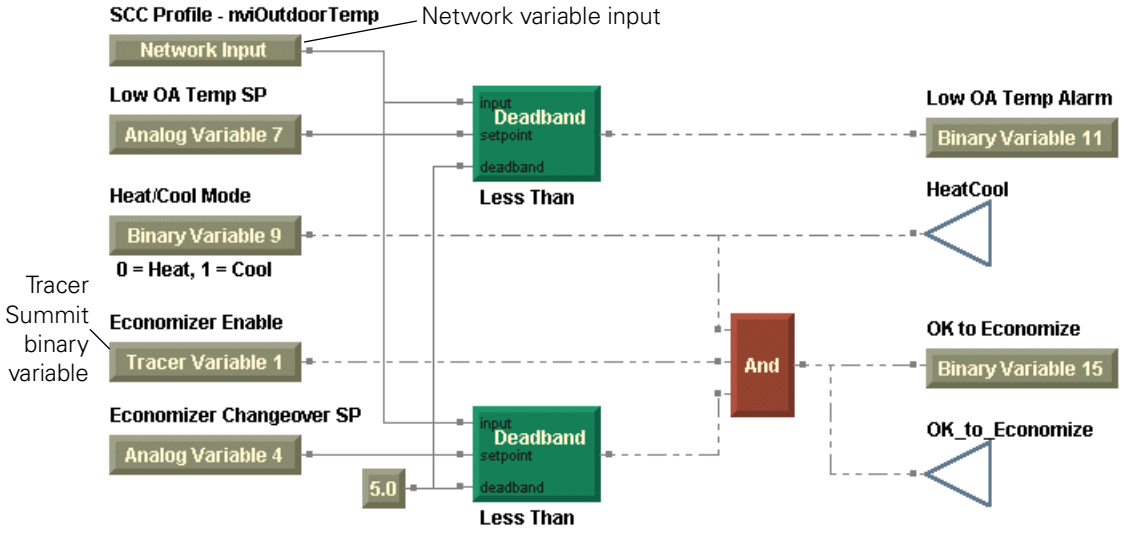
The original program was simplified by separating the decision to economize from the actual control of the damper. This first decision to economize requires changes specified by the communications portion of the

sequence of operation. Modify the first part of the decision to accommodate the following communicated values:

- Outdoor air temperature
- Economizer enable

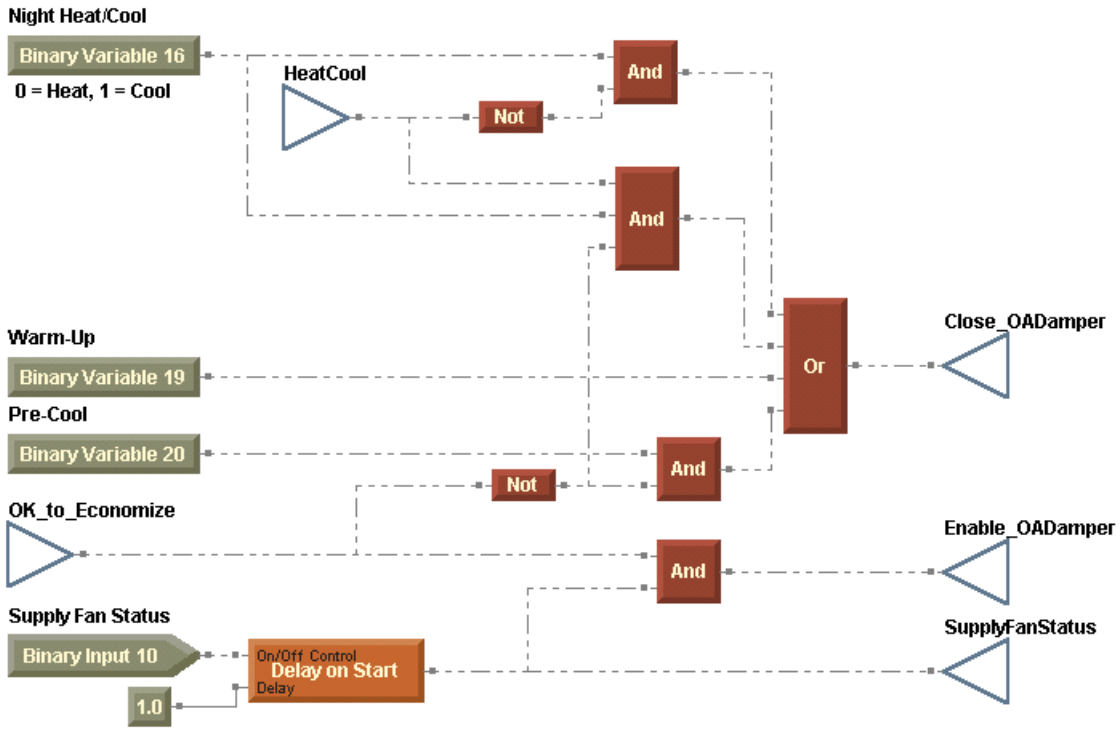
The SCC profile contains a variable for the outdoor air temperature, nviOutdoorTemp. However, there is no such variable to enable and disable the economizer. Use a Tracer Summit binary variable to satisfy this need (Figure 166).

Figure 166: Economize decision with communicated values



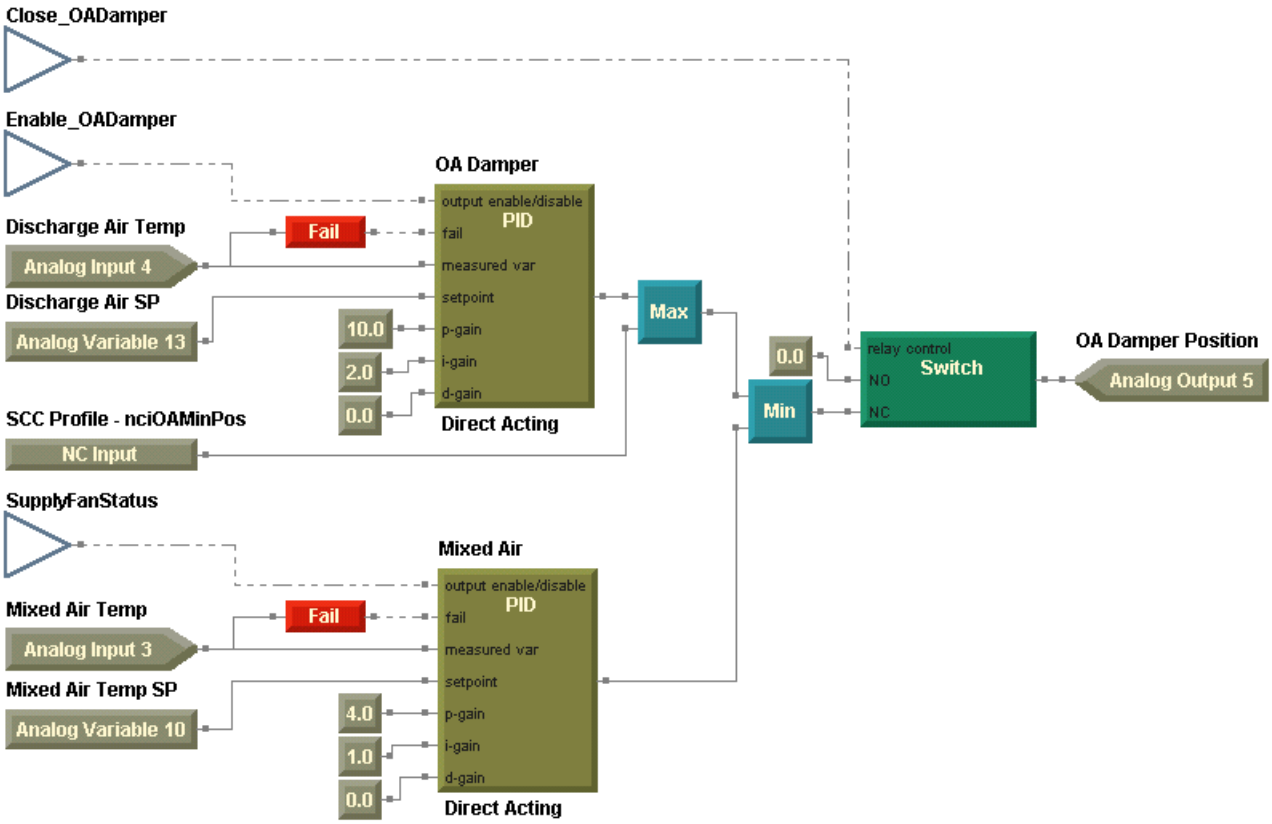
In Chapter 7, a second stage of decision making was added to accommodate night heat/cool mode. Figure 167 on page 202 shows how consideration of pre-cool and warm-up modes changes the second stage of decision making. Add the Warm-Up and Pre-Cool binary variable blocks and the logic to consider them in the decision whether to close the outdoor air damper.

Figure 167: Decision incorporating pre-cool and warm-up modes



Change the control of the outdoor air damper to incorporate the ability to close the damper based on the logic in Figure 167 as shown in Figure 168 on page 203. Note that the delay, which prevents the damper from opening for 1 minute after the supply fan starts, is also included in the logic Figure 167.

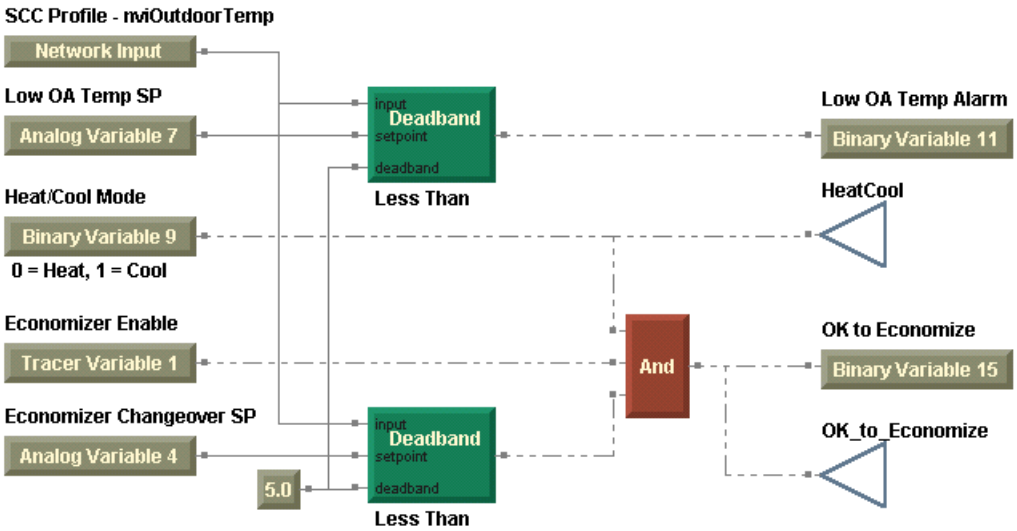
Figure 168: Outdoor air damper control



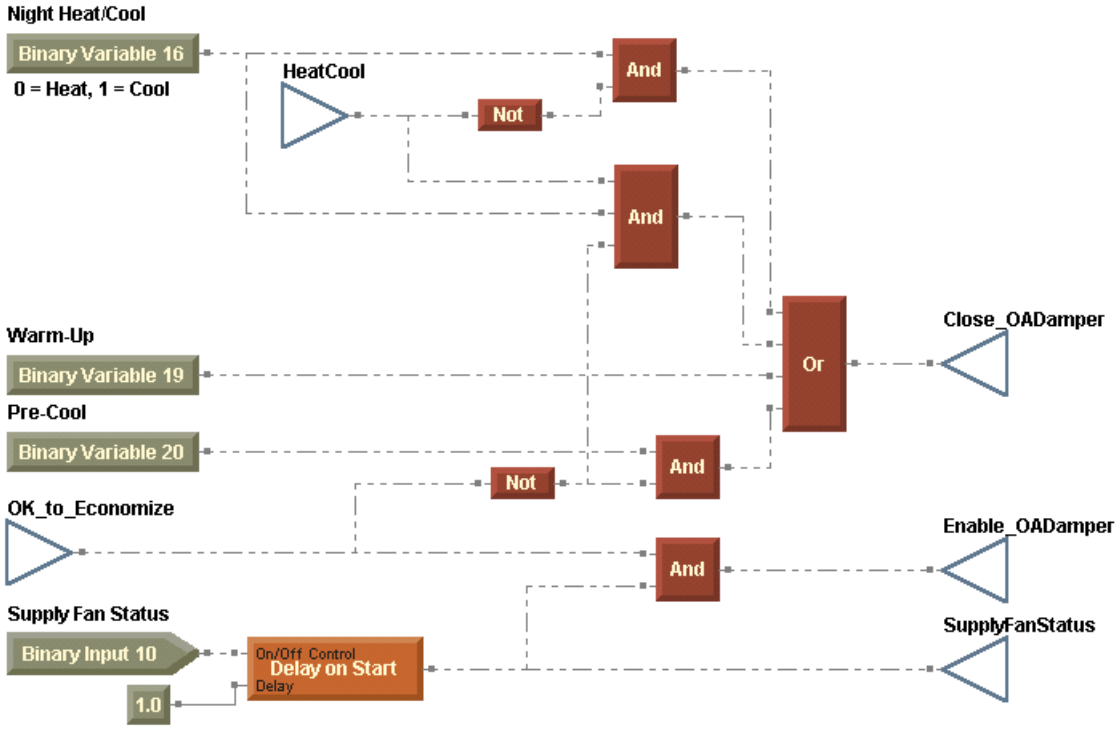
After making the changes, compile the modified program (Figure 169) and download it to the controller.

Figure 169: Completed mixed air control program

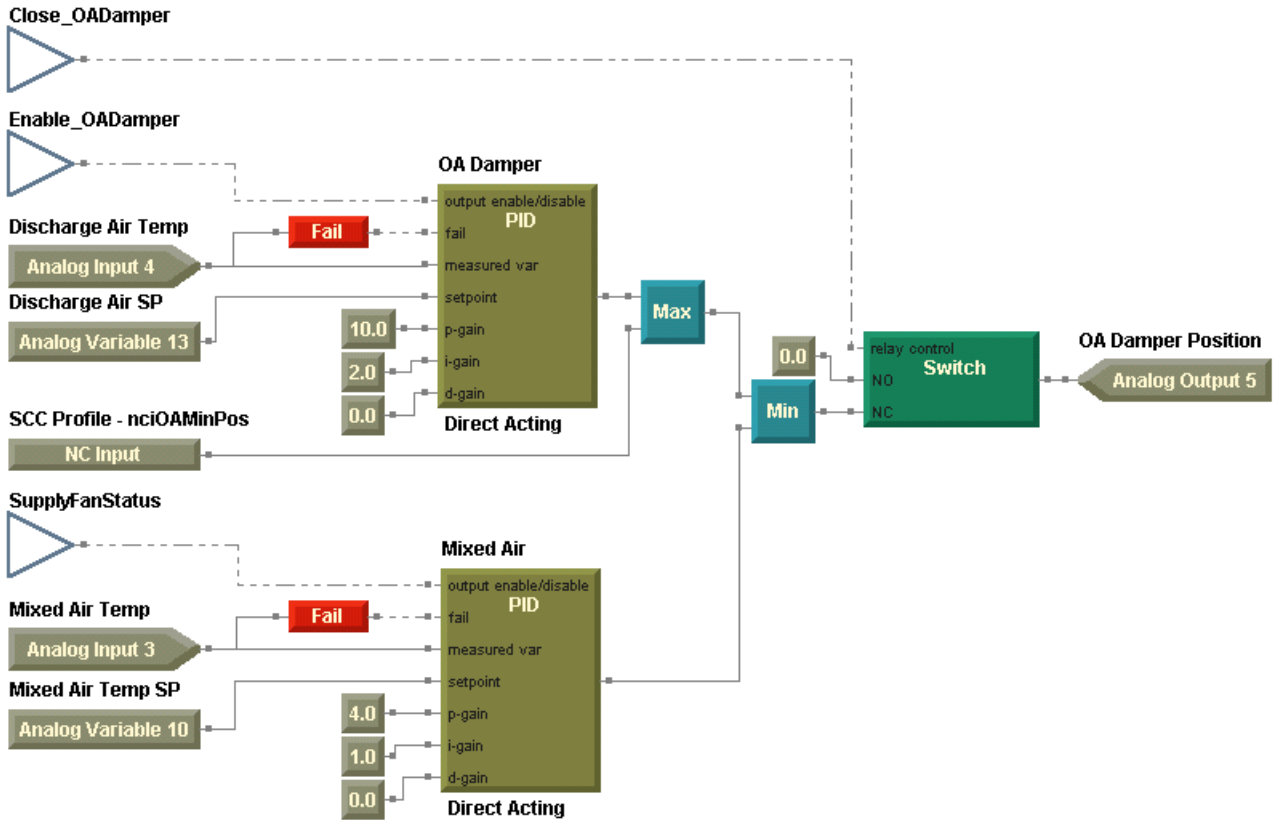
Economize decision with communicated values



Decision incorporating pre-cool and warm-up modes



Outdoor air damper control



Writing the discharge air control program

The discharge air control program must accomplish the same tasks as it did in Chapter 7. Those tasks include the following:

- Discharge air setpoint calculation
- Cooling valve control
- Heating valve control
- Dehumidification
- Humidification

Much of the programming remains the same; however, the addition of pre-cool and warm-up modes, as well as consideration of the SCC profile, require some changes to the programming. Start with the program you created in Chapter 7.

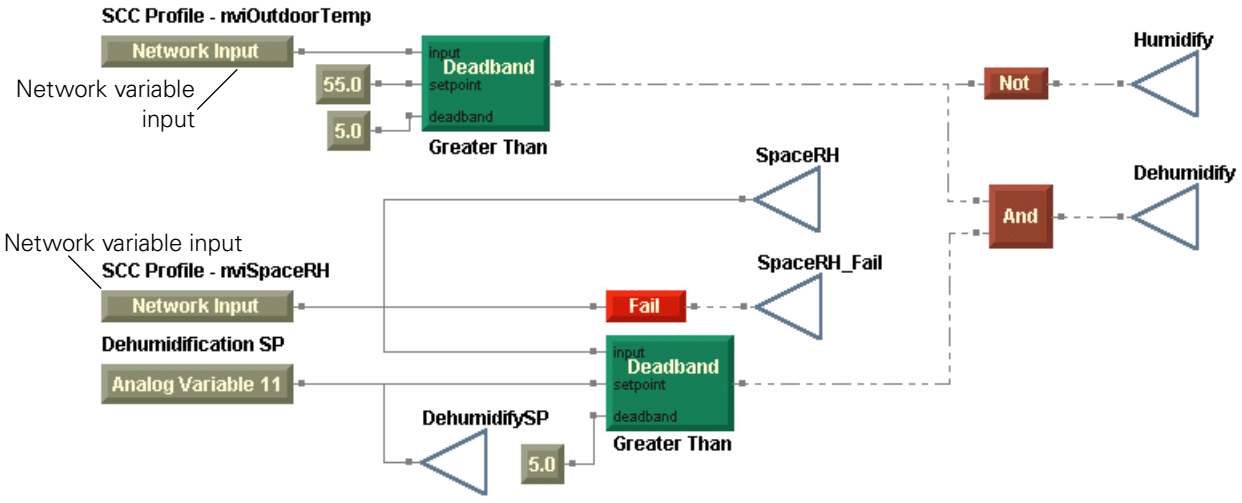
The first module calculates the discharge air setpoint based on the effective space setpoint. See “Calculating the discharge air setpoint” on page 165 for more information. This part of the program is unaffected by the additional modes and the profile considerations.

Whether to humidify or dehumidify

The decision to humidify or dehumidify requires a minor modification. Can you identify the necessary changes? (Hint: Take a look at the SCC profile variables. Where do the outdoor air temperature and the space relative humidity come from?)

Because the outdoor air temperature and the space relative humidity are now communicated from the building automation system, replace the Outdoor Air Temp input block with a Network Variable Input block. And change the Network Variable Input block assigned to nviPercent01 to nviSpaceRH in the SCC profile (Figure 170).

Figure 170: Dehumidify or humidify?



Controlling the cooling valve

How do the additional modes and profile considerations change control of the cooling valve? Examine the sequence of operation with regard to the cooling valve. The changes for this chapter are shown in *italics*.

Modulate the cooling valve to maintain the discharge air temperature at the discharge air cooling setpoint when *all* of the following are true:

- The supply fan is on.
- The air handler is in cooling mode.
- The economizer function is not enabled.

Or when *all* of the following are true:

- The supply fan is on.
- *The air handler is in cooling mode (including pre-cool).*
- The economizer function is enabled.
- The outdoor air damper is open at least 90%.

Modulate the cooling valve to maintain the space relative humidity at the space relative humidity setpoint when *all* of the following are true:

- The supply fan is on.
- The air handler is in cooling mode.
- Dehumidification is enabled.

Open the cooling valve completely when *all* of the following are true:

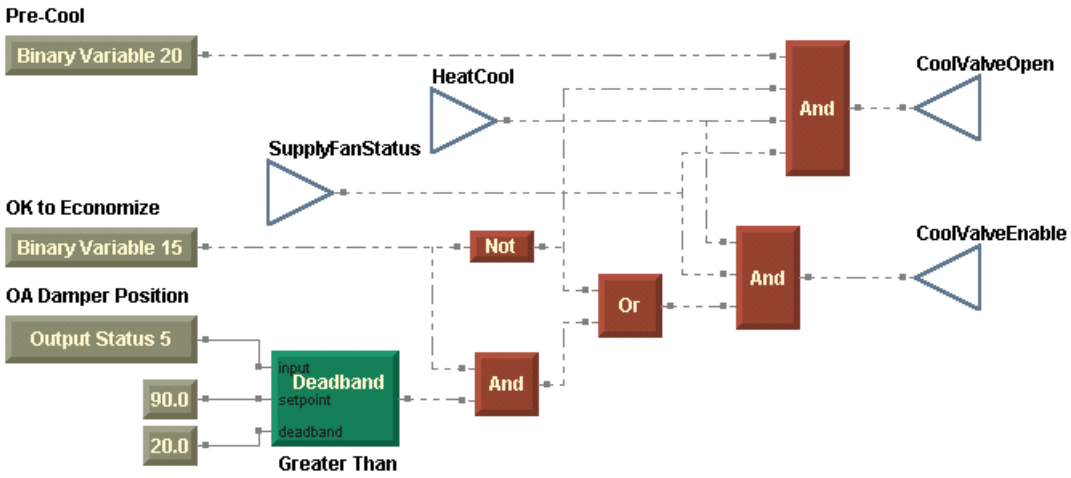
- *The supply fan is on.*
- *The air handler is in cooling mode.*
- *The pre-cool mode is active.*
- *Economizing is not allowed,* or the outdoor air damper is open at least 90%.

Close the cooling valve when *any* of the following is true:

- The air handler is in heating mode.
- The supply fan is off.
- The discharge air temperature sensor is failed.

Incorporate pre-cool mode into the decision to enable the cooling valve. Add another And block and complete the logic to determine whether the cooling valve is enabled or if it should open completely (Figure 171 on page 207).

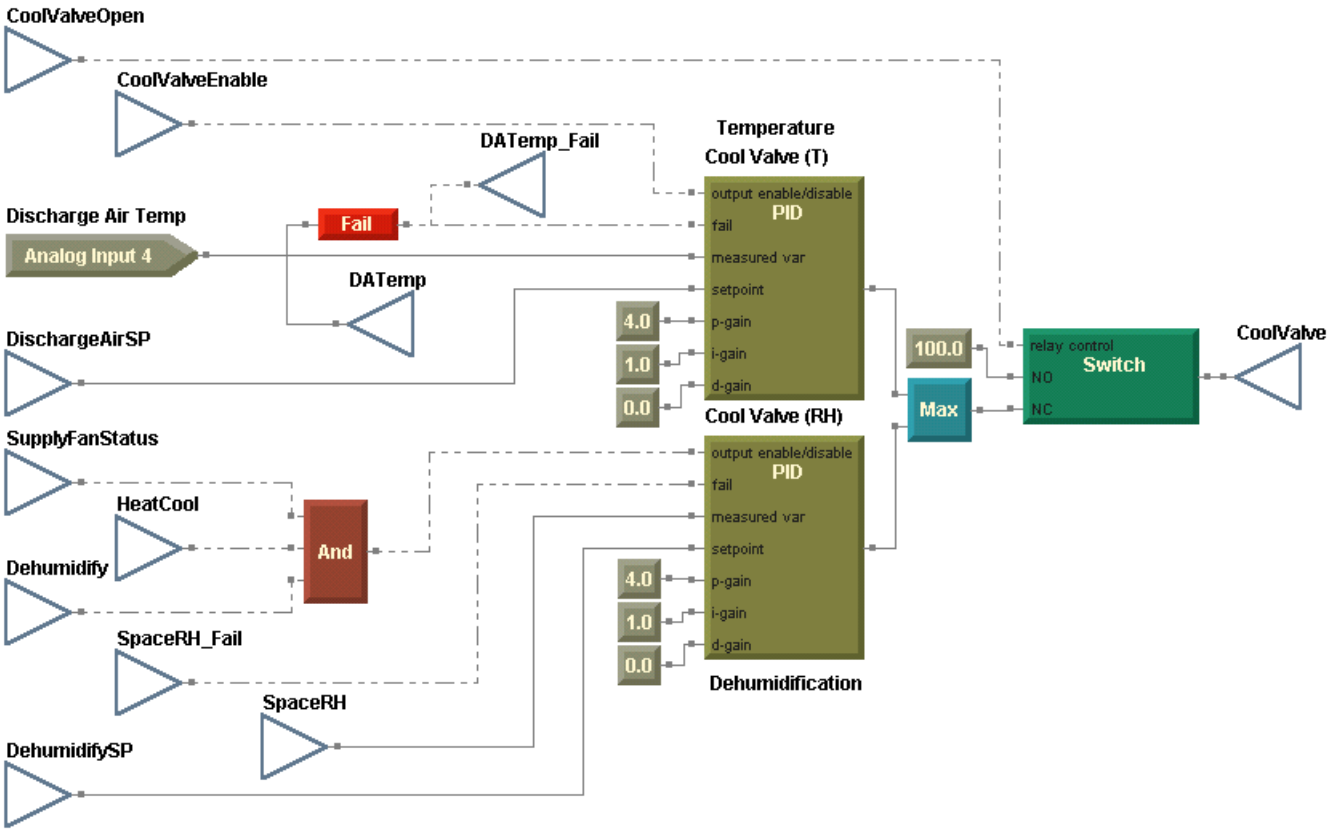
Figure 171: Modulate or open the cooling valve?



As in Chapter 7, when the cooling valve is enabled, a PID loop controls the discharge air temperature. And if dehumidification is also active, a second PID loop calculates the cooling valve position based on the space relative humidity. Control the valve to the greater of the two calculated valve positions.

If pre-cool mode is active and economizing is not allowed, use a Switch block to bypass the PID outputs and open the valve completely (Figure 172 on page 208).

Figure 172: Controlling the cooling valve



Controlling the heating valve

Incorporate warm-up for the heating valve in the same manner you incorporated pre-cool for the cooling valve. Minor changes to the sequence of operation relative to the sequence in Chapter 7 are shown below in *italics*.

Modulate the heating valve to maintain the discharge air temperature at the discharge air heating setpoint when *both* of the following are true:

- The supply fan is on.
- The air handler is in heating mode.

Or when *both* of the following are true:

- The supply fan is on.
- Dehumidification is enabled.

Open the heating valve completely when *both* of the following are true:

- The supply fan is off.
- The outdoor air temperature falls below the adjustable freeze avoidance setpoint.

Or when *all* of the following are true:

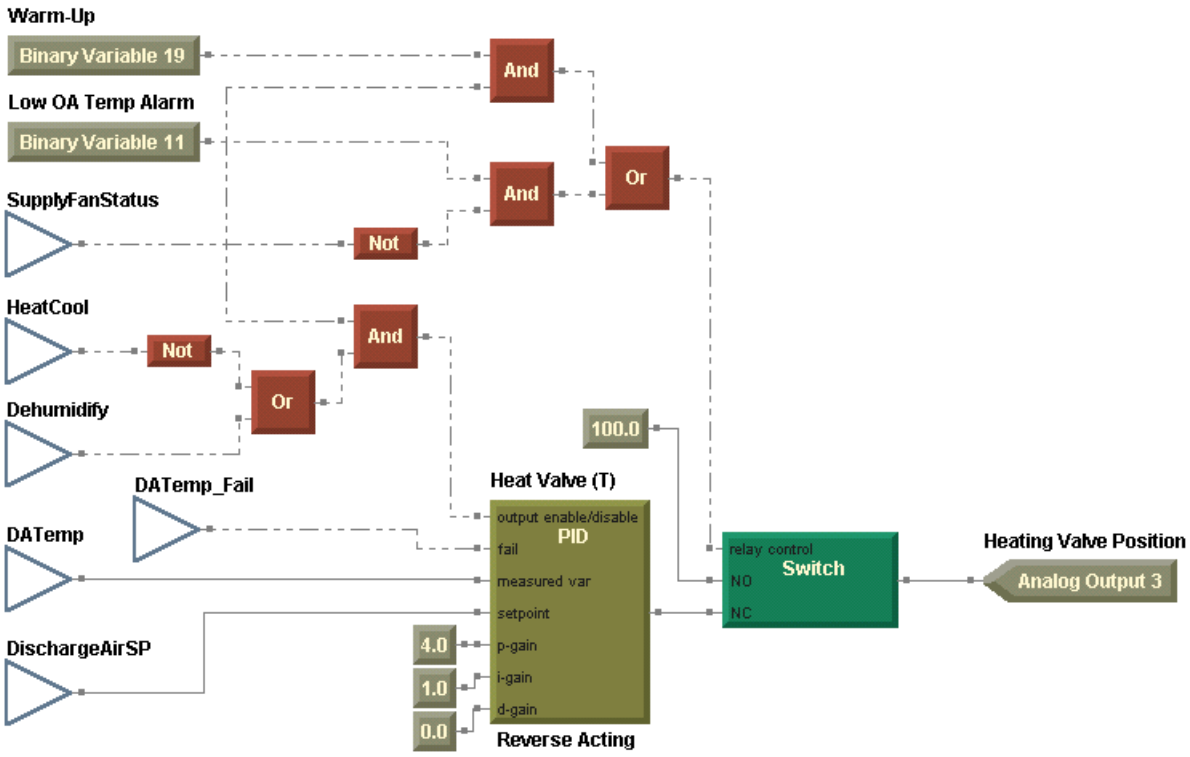
- *The supply fan is on.*
- *The air handler is in heating mode.*
- *The warm-up mode is active.*

Close the heating valve when *any* of the following is true:

- The air handler is in cooling mode, and dehumidification is disabled.
- The supply fan is off.
- The discharge air temperature sensor is failed.

Adding consideration of warm-up mode to the program increases the complexity of the logic. Similar to control of the cooling valve, use a Switch block to bypass the PID output and open the valve completely when warm-up mode is active (Figure 173).

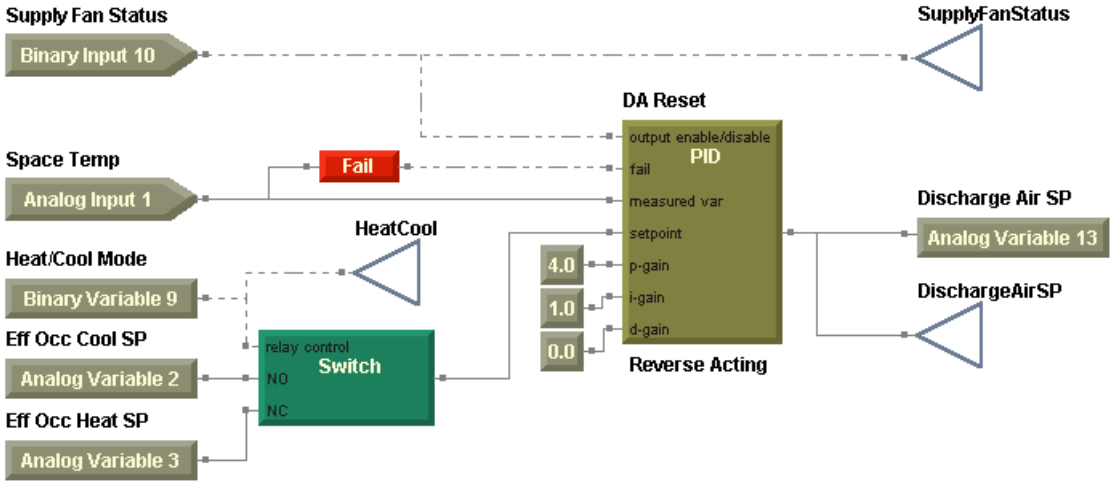
Figure 173: Controlling the heating valve



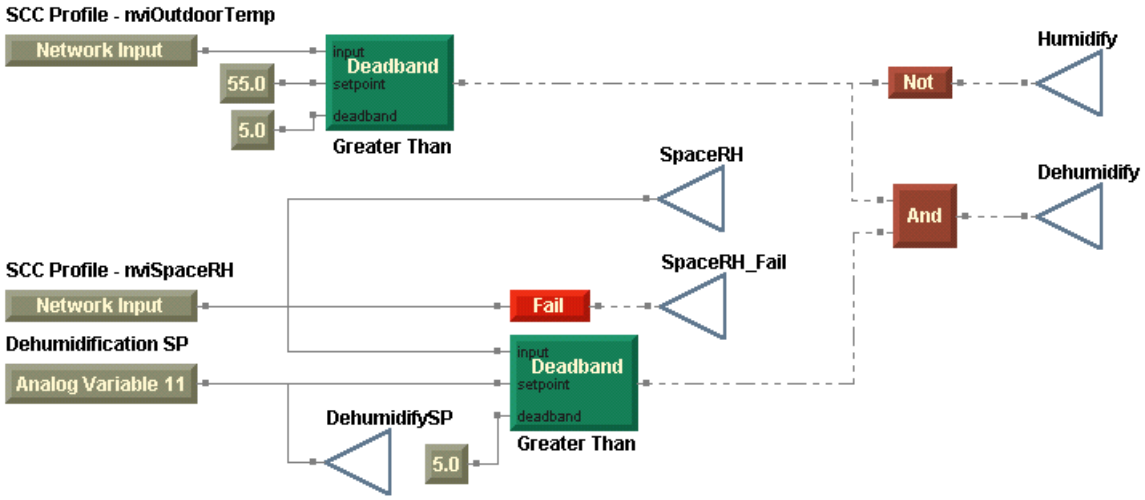
After you complete changes to the discharge air control program (Figure 174 on page 210), compile and download it to the controller.

Figure 174: Completed discharge air control program

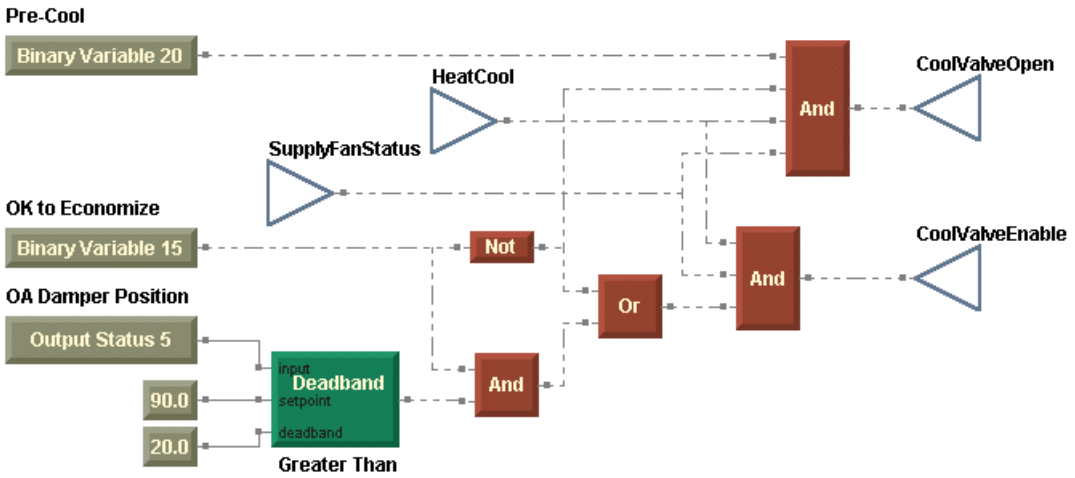
Discharge air reset



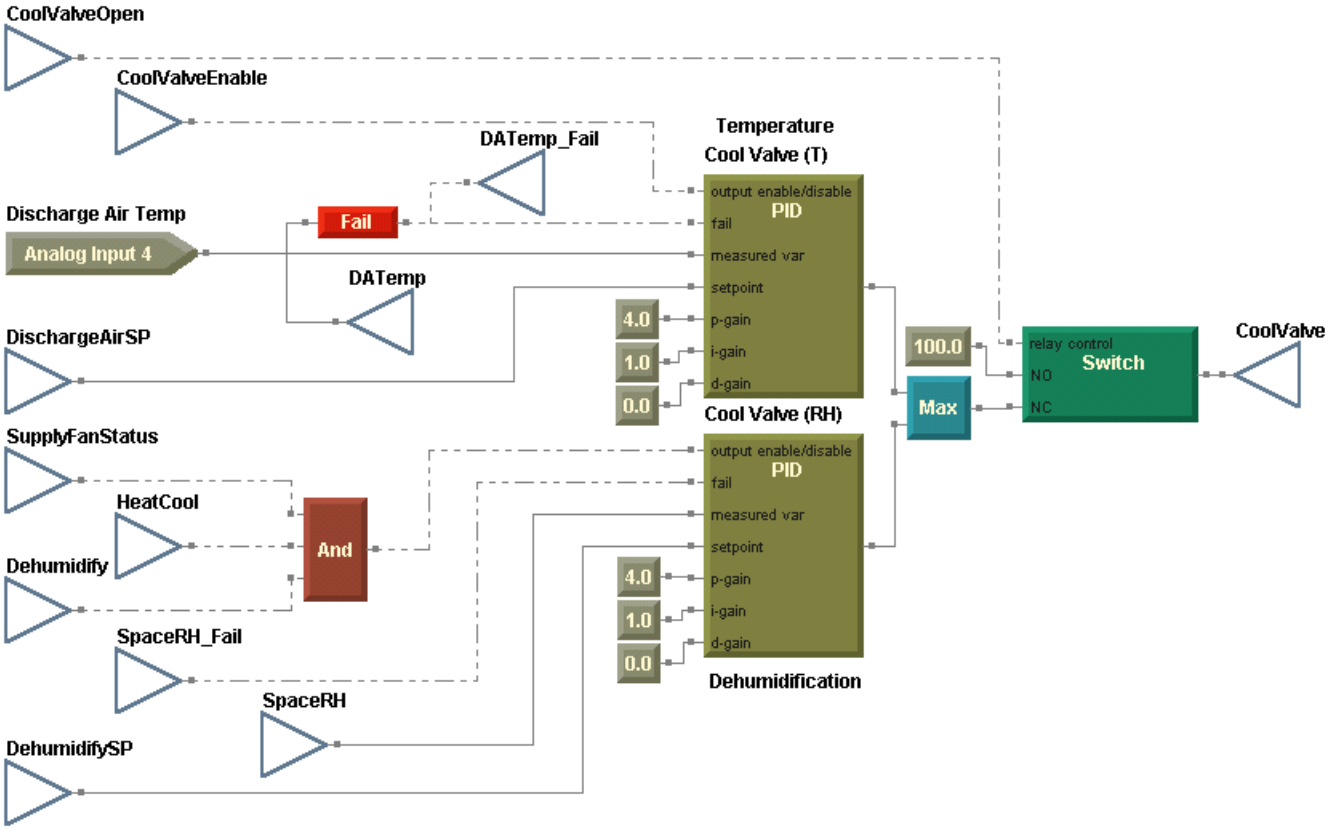
Dehumidify or humidify?



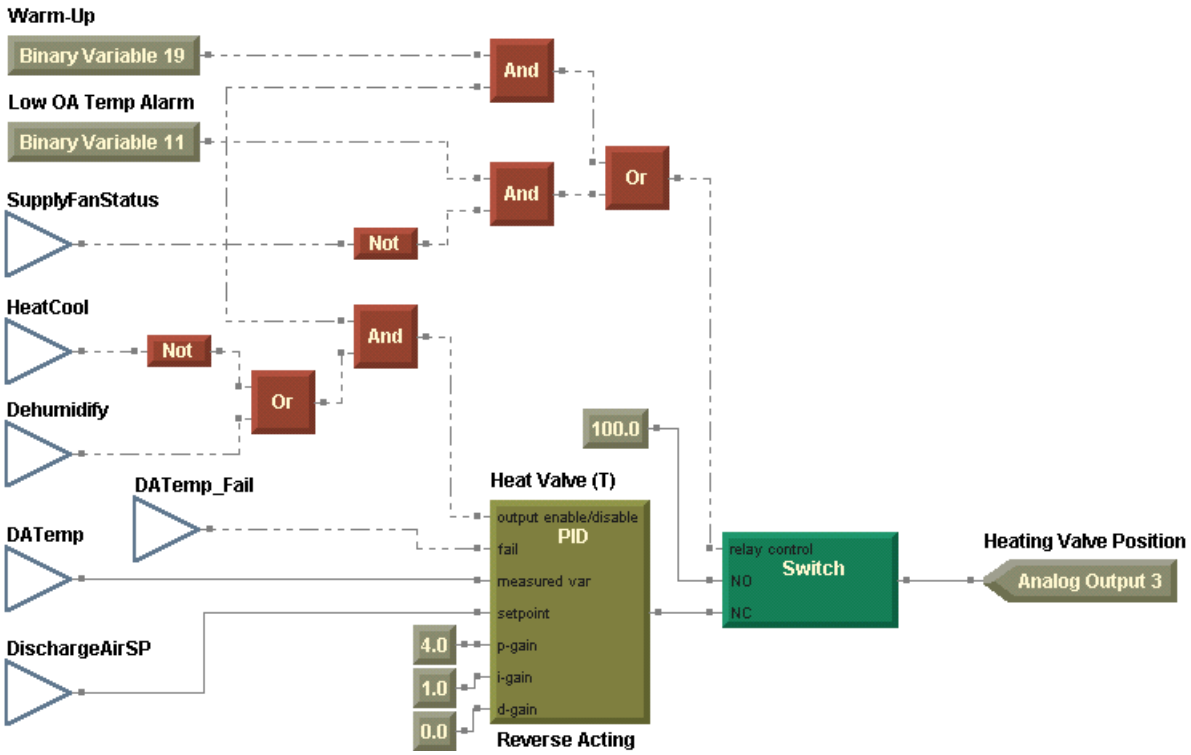
Modulate or open the cooling valve?



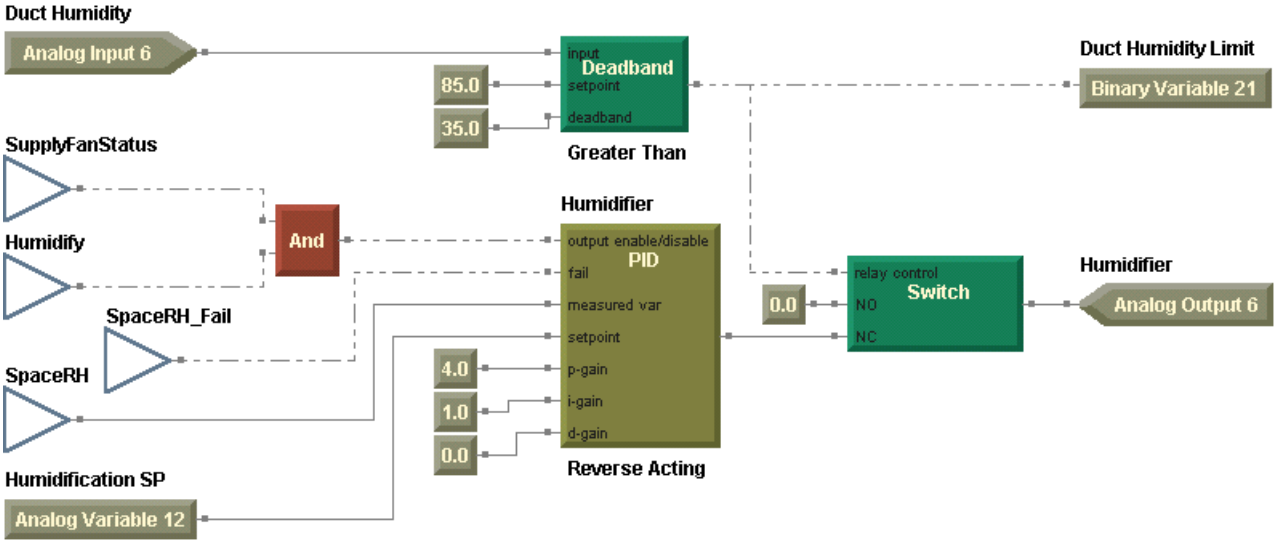
Controlling the cooling valve



Controlling the heating valve



Controlling the humidifier



Writing the alarms program

The sequence of operation did not change with respect to alarm management; therefore, no changes are required to the Alarms program. See “Writing the alarms program” on page 176 for more information.

Writing the mode and setpoints program

The mode and setpoints program in Chapter 7 provides a solid base for the program required by the more complex air handler in this chapter. The following considerations require additions or changes to the program:

- Consider a third source of the space temperature setpoint—the building automation system.
- Implement the pre-cool and warm-up mode decisions.

Modifying the effective setpoint calculation

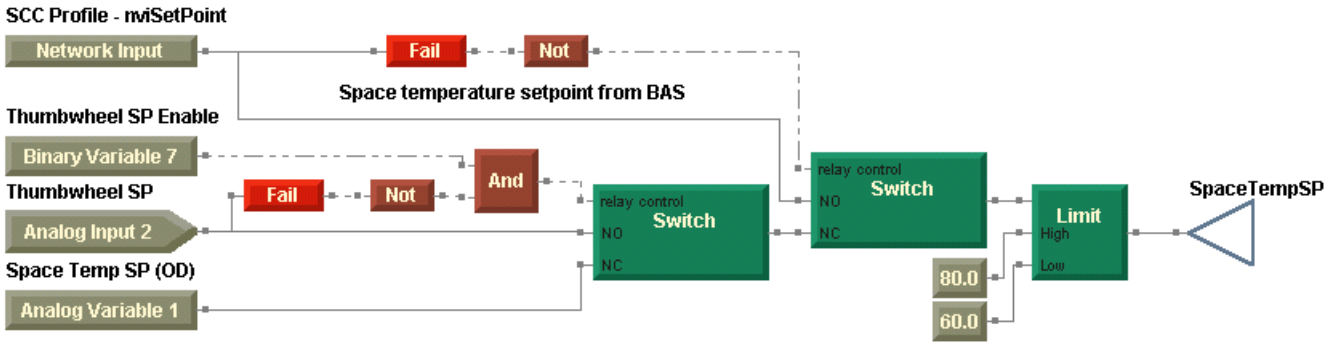
The first module in the program, which calculates the setpoint offset, is unaffected by the modifications. See “Implementing effective space setpoint calculation” on page 181 for more information.

Use additional programming in the space temperature setpoint source determination to include a setpoint from the building automation system. According to the sequence of operation:

> The space temperature setpoint source may be the building automation system, the operator display, or a wired thumbwheel setpoint adjustment knob, with the building automation system setpoint given first consideration.

Use the nviSetpoint network variable input block, associated with the SCC profile, to communicate the building automation setpoint. If a valid setpoint is communicated, apply limits to it and accept the result as the space temperature setpoint. Use a Fail block to switch to the operator display setpoint or the wired setpoint if the communicated setpoint becomes invalid (Figure 175).

Figure 175: Space temperature setpoint source determination



The remaining portions of the program regarding effective setpoint calculations are the same as those in Chapter 7.

Adding the pre-cool and warm-up decisions

The heat/cool mode and night heat/cool decisions are the same as those in Chapter 7. See “Adding night heat/cool” on page 184 for more information.

However, the mode and setpoints program must incorporate additional decisions to accommodate pre-cool and warm-up modes. These decisions are specified in the sequence of operation and summarized here:

Enter pre-cool when *all* of the following are true:

- The unit is occupied (including occupied, occupied standby, and occupied bypass modes).
- The unit is in cooling mode.
- The space temperature is at least 3°F above the effective cooling setpoint.

Exit pre-cool mode when the space temperature falls to within 2°F of the effective cooling setpoint.

Enter warm-up mode when *all* of the following are true:

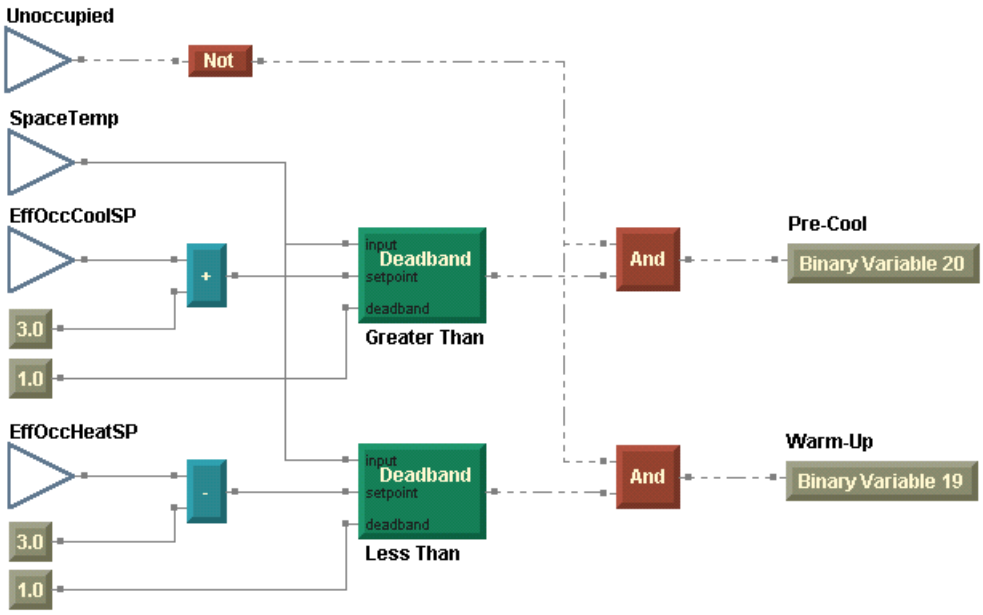
- The unit is occupied (including occupied, occupied standby, and occupied bypass modes).
- The unit is in heating mode.
- The space temperature is at least 3°F below the effective heating setpoint.

Exit warm-up mode when the space temperature rises to within 2°F of the effective heating setpoint.

Implement these decisions in graphical programming using the night heat/cool decision as a model. Note that you must base the decision for each mode on a value that is offset by 3°F from the effective cooling and heating setpoints (Figure 176 on page 215).

- Use Add and Subtract blocks to calculate the setpoints for each Deadband block.
- Configure the Deadband block for pre-cool as greater than, but configure the Deadband block for warm-up as less than.

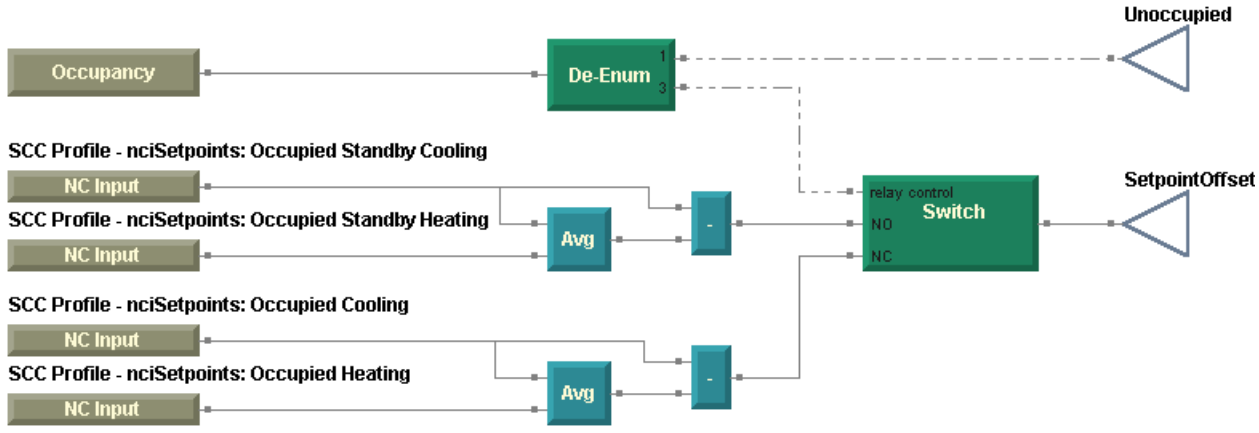
Figure 176: Pre-cool and warm-up decisions



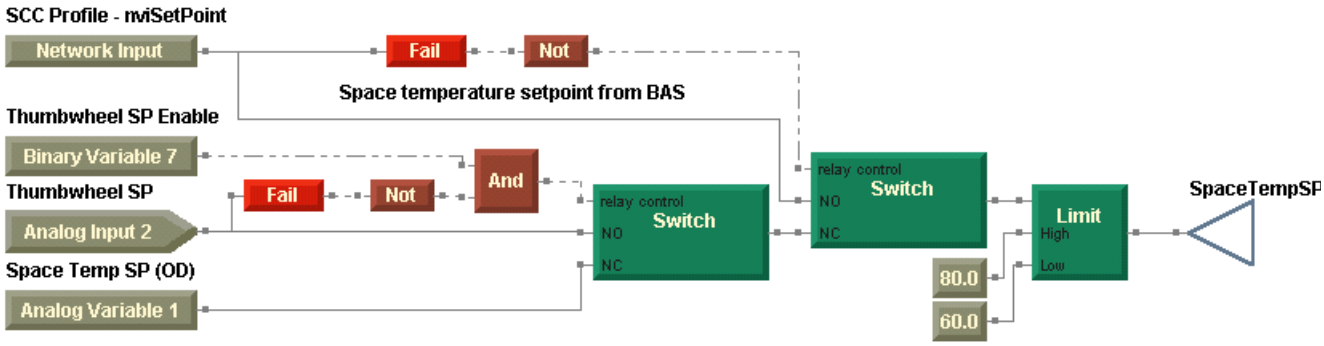
After completing the changes to the mode and setpoints program (Figure 177), compile and download the program to the controller.

Figure 177: Completed modes and setpoints program

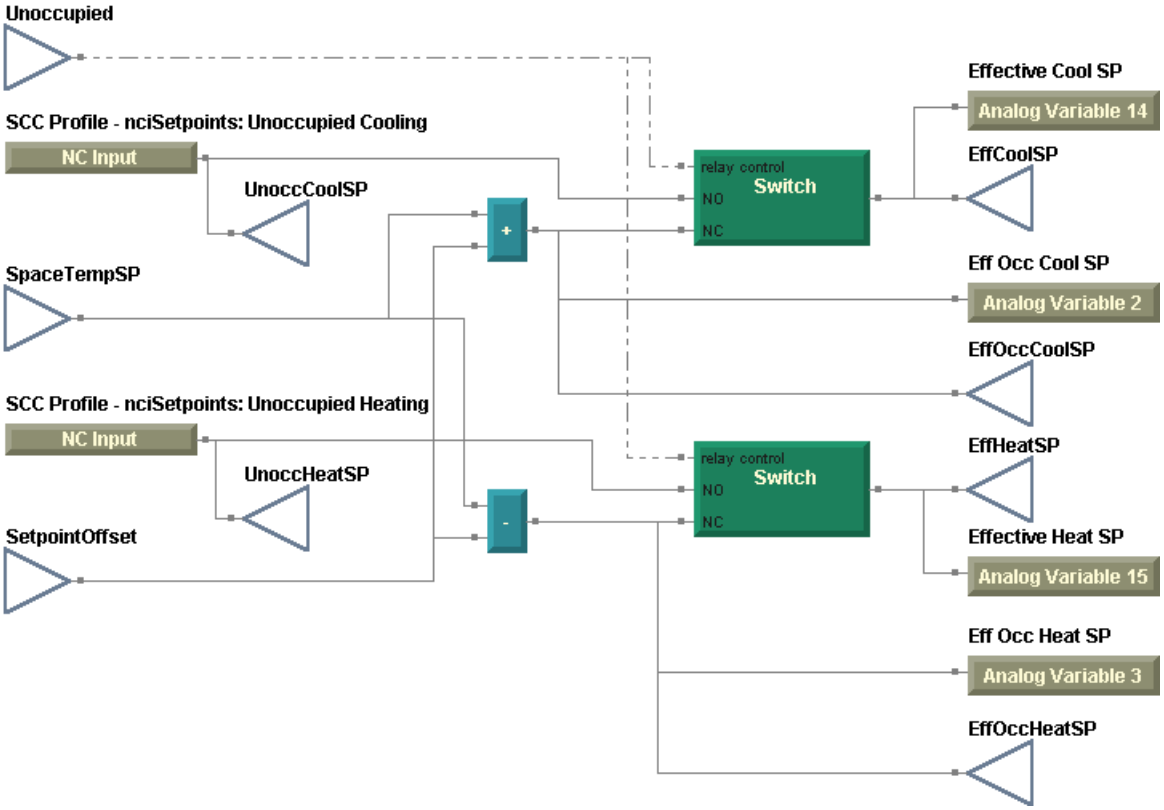
Offset calculation



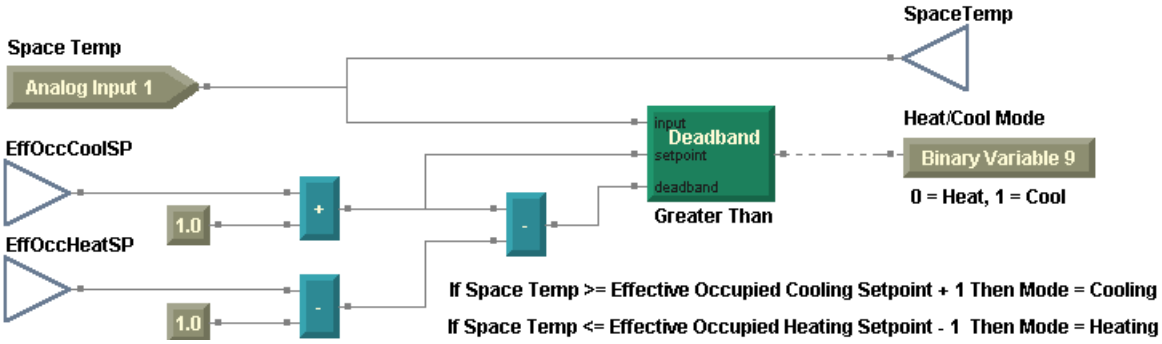
Space temperature setpoint source determination



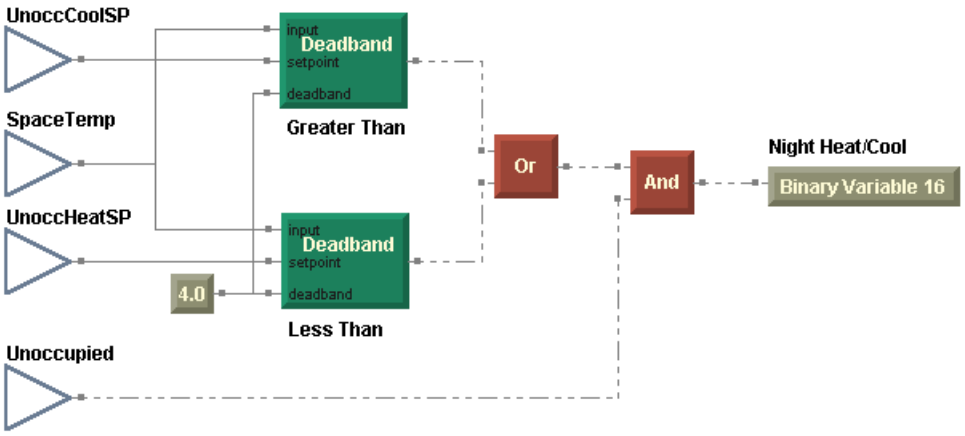
Effective setpoint calculation



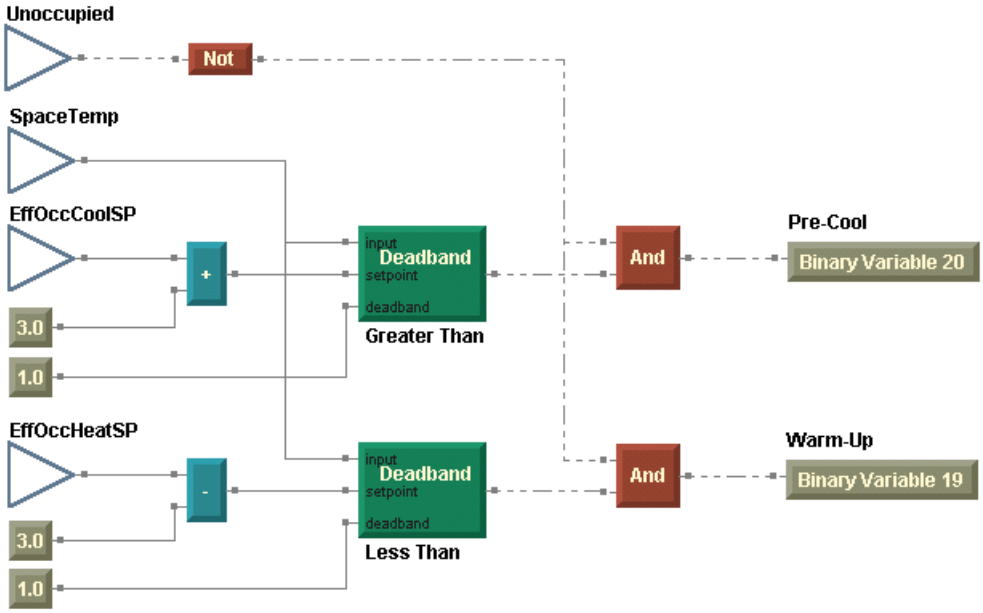
Heat/Cool mode decision



Night heat/cool decision



Pre-cool and warm-up decisions



Writing the communications program

The only new program in this chapter is concerned solely with communications. The sequence of operation requires that the following items be communicated to the air handler:

- Occupancy
- Space temperature setpoint
- Outdoor air temperature
- Space relative humidity
- Economizer enable

If communication with the BAS is lost, the air handler should use its predetermined default setpoints and operate in the occupied mode. The following information is communicated to the air handler in the programs you modified so far in this chapter.

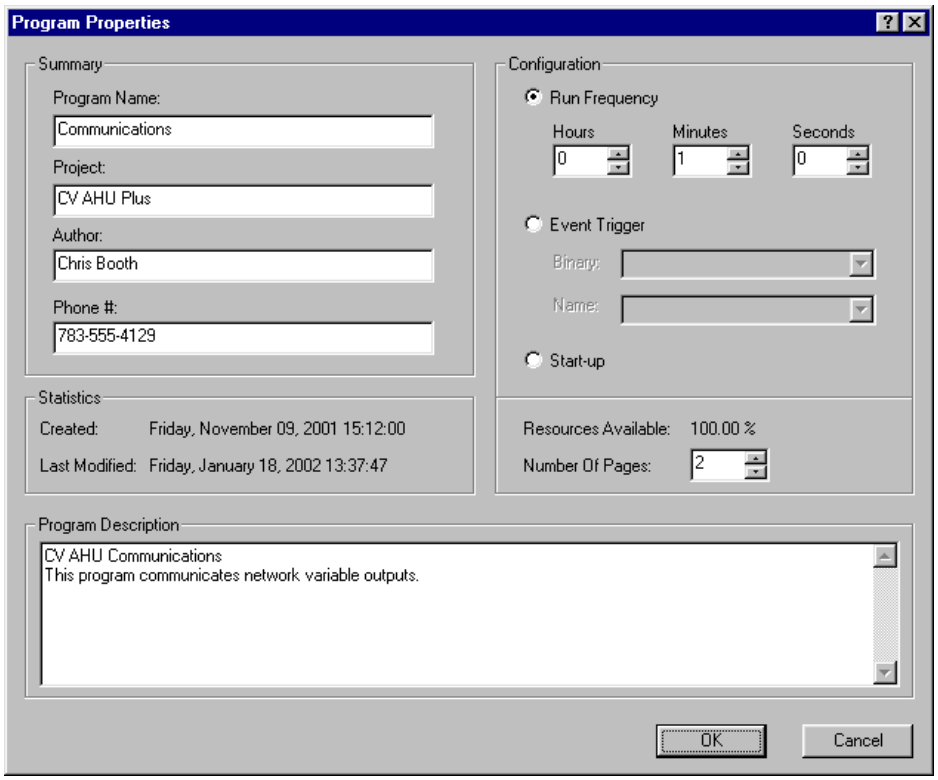
- The occupancy mode, whether it is communicated or based on the local schedule, is accounted for by using the Occupancy block in both the mode and setpoints and the fan control programs. The outdoor air temperature and economizer enable are received in the mixed air control program.
- The space relative humidity is communicated as part of the SCC profile using the nviSpaceRH network variable input in the discharge air control program.
- The space temperature setpoint is communicated as part of the SCC profile using the nviSetpoint network variable input in the mode and setpoints program.

Send some information building automation system (BAS). The air handler must communicate the following items to the building automation system:

- Space temperature
- Discharge air temperature
- Mixed air temperature
- Effective setpoint
- Effective occupancy
- Supply fan status
- Cooling valve status
- Heating valve status
- Outdoor air damper status
- Alarm status
- Application mode

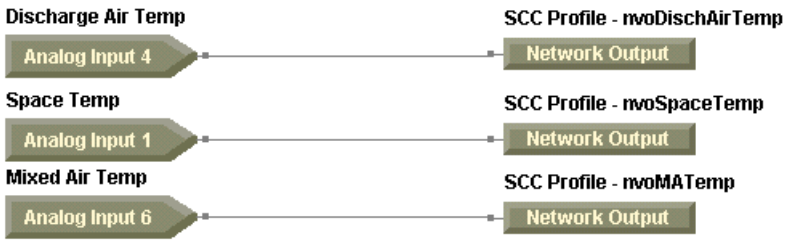
These items are communicated so that the building operator can view the status of the air-handling unit. Because communications to the building automation system are not critical to air-handler operation, this program executes once per minute. Create a new program and set its properties as shown in Figure 178 on page 219.

Figure 178: Communications program properties



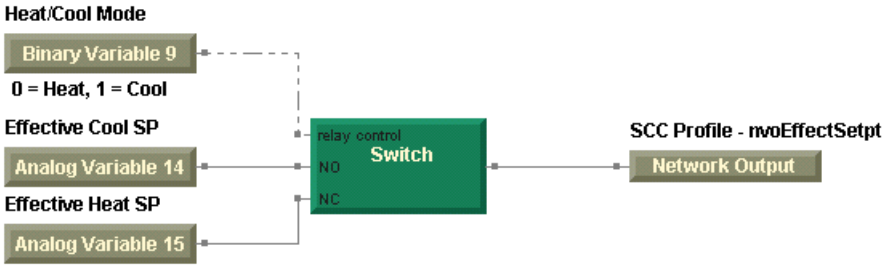
Use Network Variable Output blocks to transmit the space, discharge air, and mixed air temperatures through the SCC Profile (Figure 179). The Network Variable Output block works just like the Network Variable Input block, but it transmits information on the Comm5 network instead of receiving it. Be sure to specify the units in which the Network Variable Output block receives the data from the program using the properties dialog box for the block.

Figure 179: Transmitting temperatures



Transmit the effective setpoint. Remember, the effective setpoint may vary depending on the heat/cool mode and occupancy. So, incorporate the necessary logic to determine the appropriate setpoint to send. The SCC profile provides a variable, nviEffectSetpt. Use a Switch block to select the appropriate effective setpoint, cooling or heating, based on the heat/cool mode (Figure 180 on page 220).

Figure 180: Transmitting the effective setpoint



Use the nvoEffectOccup network variable output to send the effective occupancy to the building automation system (Figure 181).

Figure 181: Transmitting effective occupancy

The remaining items that must be communicated to the building automation system include the following:

- Supply fan status
- Cooling valve status
- Heating valve status
- Outdoor air damper status
- Alarm status
- Application mode

One network variable output exists that may be used to communicate all of the above information. This network variable output, nvoUnitStatus, consists of a variety of information defined by its type, SNVT_hvac_status. This particular standard network variable type (SNVT) is a structure because it contains a variety of elemental variables of different types within one larger variable. Specifically, nvoUnitStatus contains the information shown in Table 29.

Table 29: SNVT_hvac_status structure contents

Element	SNVT	Notes
Mode	hvac_t	Enumerated
Primary heat output	SNVT_lev_percent	From –163.83% to 163.83%
Secondary heat output	SNVT_lev_percent	From –163.83% to 163.83%
Cool output	SNVT_lev_percent	From –163.83% to 163.83%
Econ output	SNVT_lev_percent	From –163.83% to 163.83%
Fan output	SNVT_lev_percent	From –163.83% to 163.83%
In alarm	Boolean	Alarm status, 0 or 1

Consider the mode element of the unit status structure. You know that the mode element is an enumerated value. To determine the appropriate value to assign to this element, you need to fully understand the type definition (Table 30).

Table 30: SNVT_hvac_mode (hvac_t) enumerations

Value	Identifier	Notes
0	HVAC_AUTO	Controller automatically changes between application modes
1	HVAC_HEAT	Heating only
2	HVAC_MRNG_WRMUP	Application-specific morning warm-up
3	HVAC_COOL	Cooling only
4	HVAC_NIGHT_PURGE	Application-specific night purge
5	HVAC_PRE_COOL	Application-specific pre-cool
6	HVAC_OFF	Controller not controlling outputs
7	HVAC_TEST	Equipment being tested
8	HVAC_EMERG_HEAT	Emergency heat mode (heat pump)
9	HVAC_FAN_ONLY	Air not conditioned, fan turned on
10	HVAC_FREE_COOL	Cooling with compressor not running
11	HVAC_ICE	Ice-making mode
12	HVAC_MAX_HEAT	Maximum heating mode
13	HVAC_ECONOMY	Economic heat/cool mode
14	HVAC_DEHUMID	Dehumidification mode
–1	HVAC_NUL	Invalid value

Note: This table was adapted from the LonMark Standard Enumeration Master List.

Programming tip:
Use the properties dialog box for the De-Enumerator block to quickly view enumerations corresponding to the SNVTs listed in Table 31.

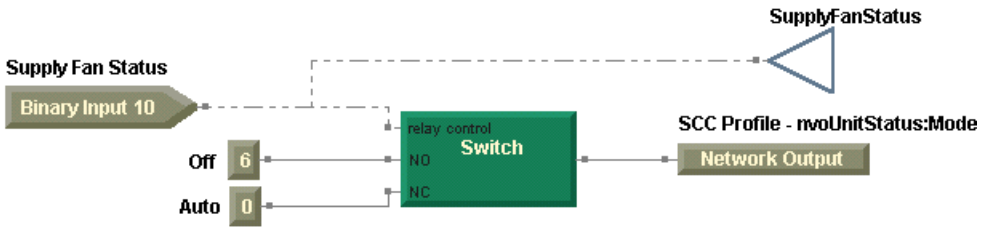
Table 31: Enumerations revealed by the De-Enumerator block

SNVT	Variable available through the De-Enumerator block
SNVT_occupancy (occup_t)	Occupancy
SNVT_hvac_mode (hvac_t)	nviApplicMode
SNVT_hvac_overid (hvac_overrid_t)	nviValveOverride
SNVT_hvac_emerg (emerg_t)	nviEmergOverride

Because the air handler makes its mode decisions internally, simply indicate AUTO when the air handler is running, and OFF when it is not running. Note that when you set the properties of the Network Variable

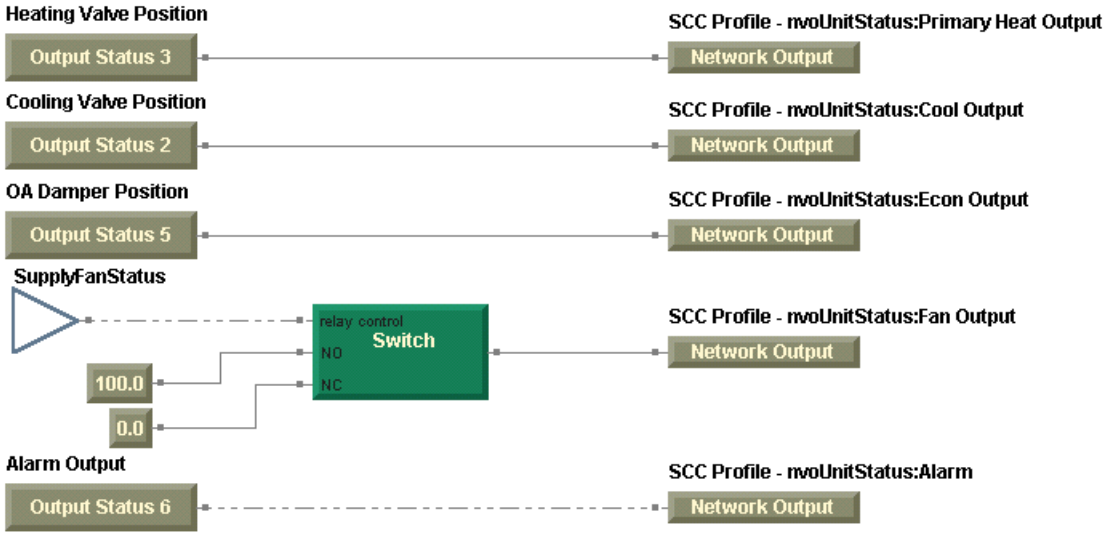
Output block, you must select the appropriate element of nvoUnitStatus (Figure 182).

Figure 182: Indicating mode within nvoUnitStatus



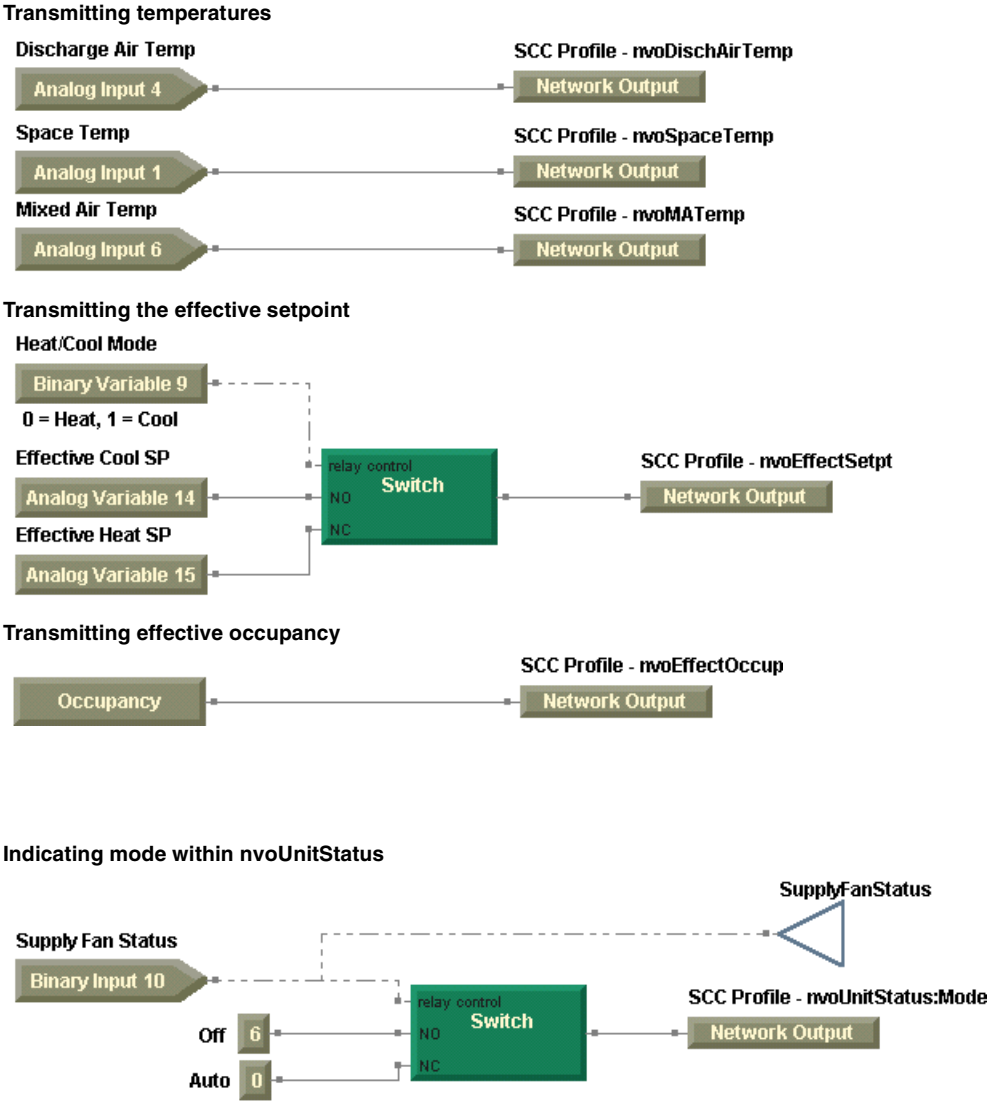
The remaining elements, with the exception of alarm status, are indicated as a percentage because the type of each element is SNVT_lev_percent. In most cases, simply connect the appropriate variable or output status to the corresponding network variable output. Be sure to select the appropriate element of nvoUnitStatus. For the supply fan status, indicate 0% when the unit is off and 100% when the unit is running. Alarm status, which is a Boolean value, uses the output status of the Alarm Output binary output (Figure 183).

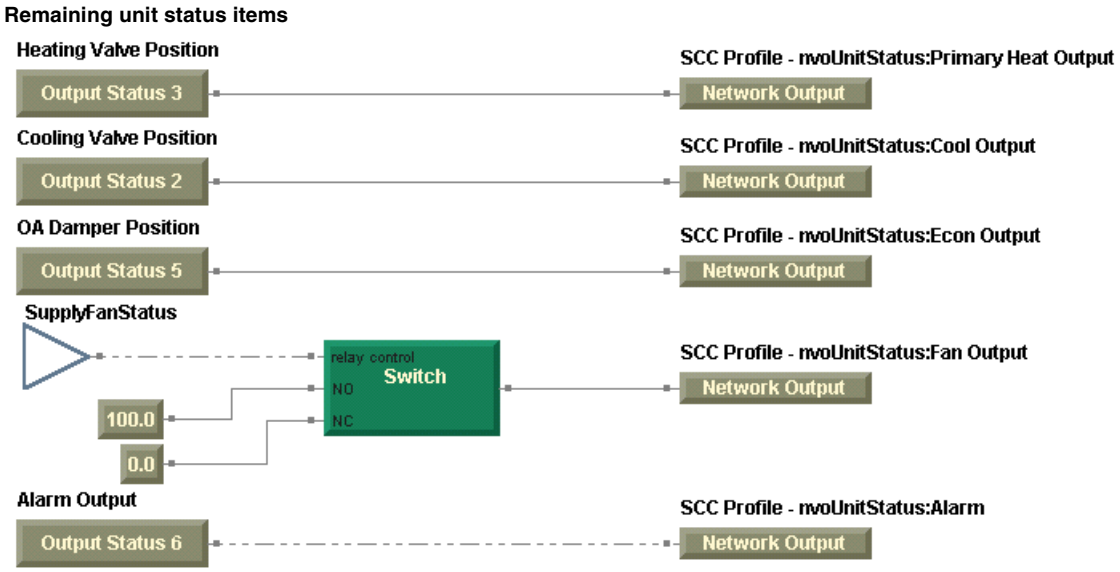
Figure 183: Remaining unit status items



All communications requirements of the sequence of operation are now fulfilled. Compile and download the completed program (Figure 184 on page 223) to the controller.

Figure 184: Completed communications program





Based on the programs in Chapters 6, 7, and 8, you should be able to construct programs to handle most air-handling applications. After completing this tutorial, you may want to try programming some of the common applications that you see in your area. As you develop programs for various jobs, build your own graphical programming library to reuse programs and to help you to become a more efficient programmer.

Summary questions

Answer the following questions to review the skills, concepts, and definitions you learned in this chapter. The answers to these questions are on page 243.

1. The SCC profile contains a variable called nviEmergOverride. This variable contains an enumerated value. It may be used to communicate an emergency shutdown to the Tracer MP580/581 controller. How would you use this variable in the fan control program to control the supply fan?

2. The SCC profile contains a variable called nviValveOverride of type SNVT_hvac_overrid. This variable contains a structure made up of three elements. The first of which is the override mode. The Tracer Summit system is capable of communicating the following values as the override mode to the Tracer MP580/581 controller:

   - 0 (Override Off)
   - 4 (Override Open)
   - 5 (Override Close)

   Incorporate logic in the discharge air control program to accommodate the valve override for both the cooling and heating valves.

3. You constructed a number of programs that incorporated alarm management and reset. In many cases, the alarm reset variable was automatically set to off after the operator turned it on to reset an alarm. How would you accomplish the same functionality from the building automation system (Tracer Summit)? Hint: A CPL program is required.

Programming best practices

During the development of the Tracer Graphical Programming (TGP) editor, we experimented with the editor by creating programs to control a variety of equipment, including pumps, cooling towers, and air handlers. In the process of creating these programs, we came to understand that there were a number of guidelines that, if followed consistently, benefited us in the following ways:

- We became better and more efficient programmers.
- Our programs were easier to troubleshoot.
- Other people found the programs easier to understand.

We have begun sharing these guidelines and refer to them as "TGP Best Practices." Keep these best practices nearby to help you as you start to construct your first graphical programs. They will help you become a better programmer, too.

Keep programs as simple as possible.

Some of the best practices are common sense in disguise. A critical step in programming is to thoroughly define the job before beginning to write programs. When the job is defined, do not make the programs any more complicated than they need to be to get the job done. Keep them as simple as possible.

Configure inputs and outputs prior to programming.

For any Tracer MP580/581 controller, use the software plug-in for the Rover service tool to configure the known inputs, outputs, and variables before writing any programs. Configuring the controller first makes the process of programming more efficient by reducing the amount of time you spend refreshing the controller configuration in the TGP editor.

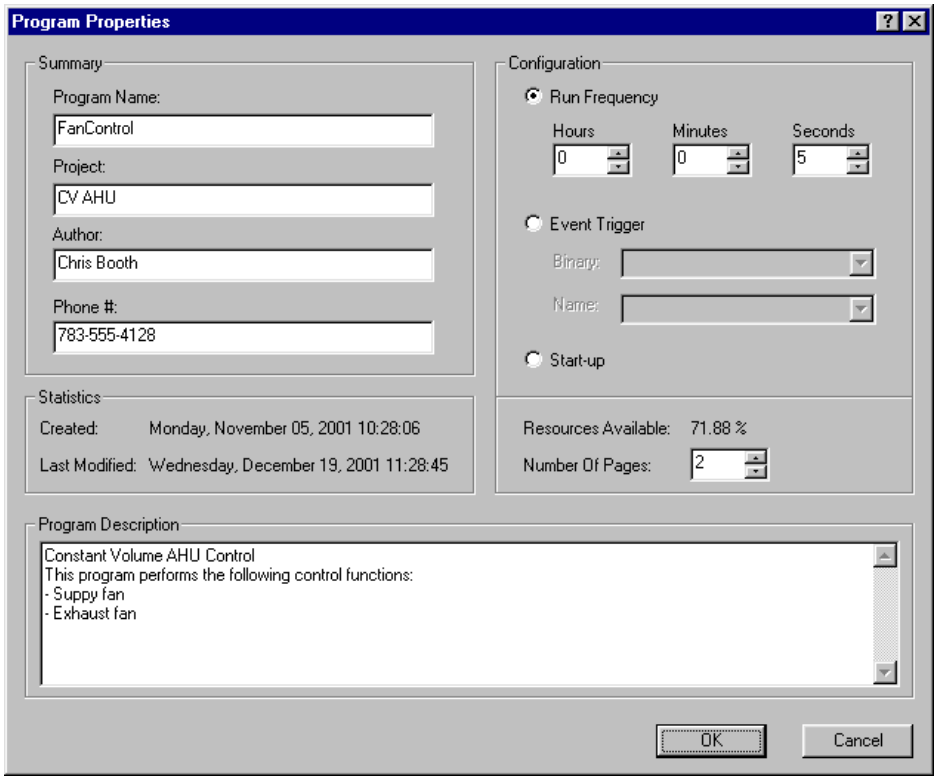
Set the program properties before you begin to write a new program.

Setting the program properties when starting a new program prevents some common mistakes, such as downloading a program with the wrong name or run frequency. In the TGP editor, choose Program Properties from the File menu (or right-click in the design space and select Properties) to access the Program Properties dialog box (Figure 185).

Note:
Program frequency affects the operation of the following blocks:

- PID
- Latch
- Delay on Start
- Delay on Stop
- Feedback Alarm

Figure 185: Program properties

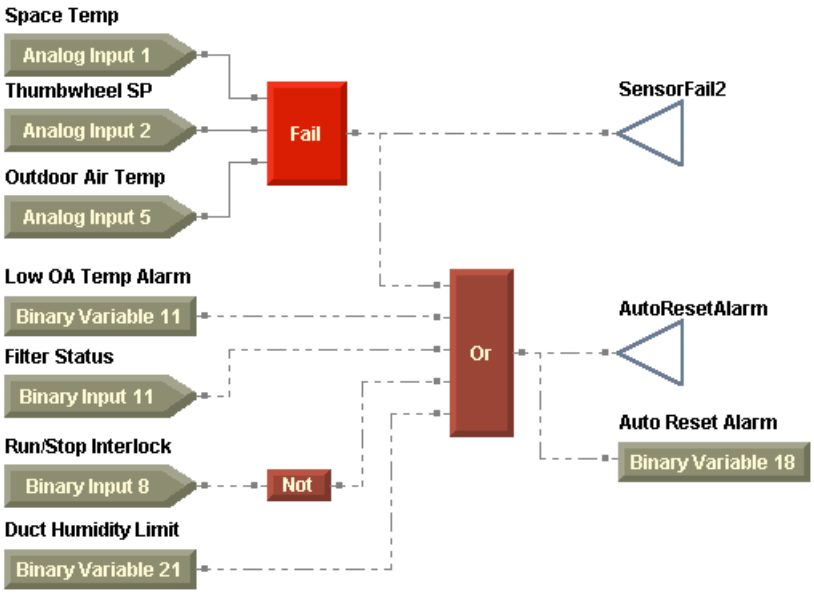


Place input blocks on the left and output blocks on the right so that your program reads from left to right.

Place inputs to the program on the left side of the design space and outputs of the program on the right side to make a consistent program format (Figure 186). When other people open your program, they will know what to expect. The programs will be easier to read because they will all be organized in the same way.

Note:
In this context, inputs and outputs of the program also include local variables, Tracer Summit variables, network variable inputs, and network variable outputs.

Figure 186: Inputs on the left, outputs on the right



Set the properties of each block as you place it in the program.

Many blocks have properties that you can edit to further define the block. To be a more efficient programmer, set the properties for each block as you place it in the program. This practice is especially important for blocks that can be set as analog or binary because the connections between blocks are dependent on the data type: analog or binary.

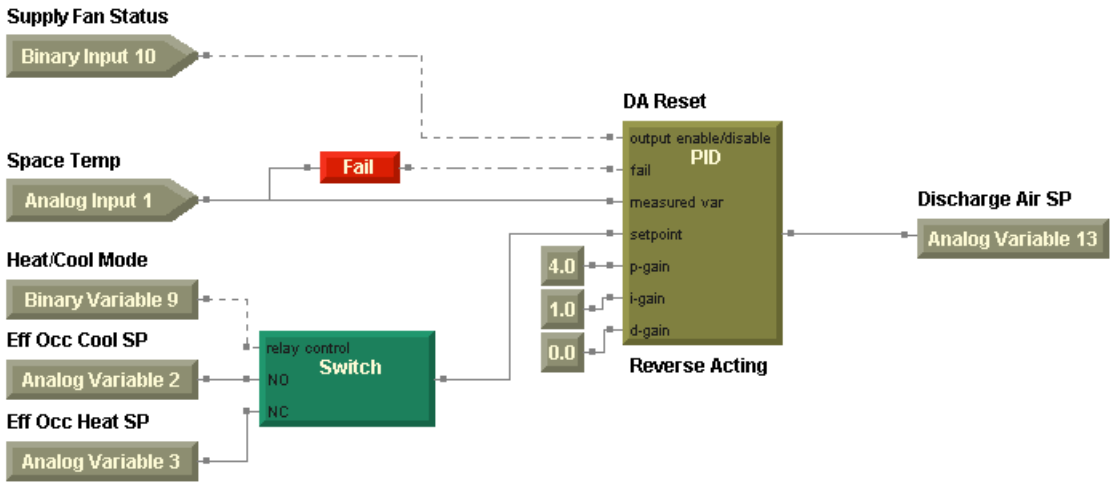
Note:
The properties dialog box for each block varies considerably depending on the options available for that block. A few blocks have no editable properties.

Consider input failures when writing programs.

As in the programmable control module (PCM) and the universal PCM (UPCM), consider failure scenarios in your programs. For example, what should happen if the mixed air temperature sensor fails on an air-handling unit? Checking for failures is easy using the Fail block (Figure 187).

Note:
The Fail block can check either hardware inputs for failure or communicated (network variable) inputs for invalid values.

Figure 187: Checking the space temperature input for failure



Write to variables only once in a program.

Prevent potential conflicts by writing to variables only once in a program. Also, be careful when writing to the same variable from multiple programs. You must consider which program will determine the final value for the variable.

Control outputs only once in a program.

You can prevent potential problems by writing to an output only once per program. Also, be careful when controlling a single output from multiple programs. If you are watching a binary output turn off, then on, then off again, and so on but you know that it should remain on, take a look at the status of the output in the Rover service tool. If you see that the control source is alternating between two different programs, the problem is that the output is controlled in two programs.

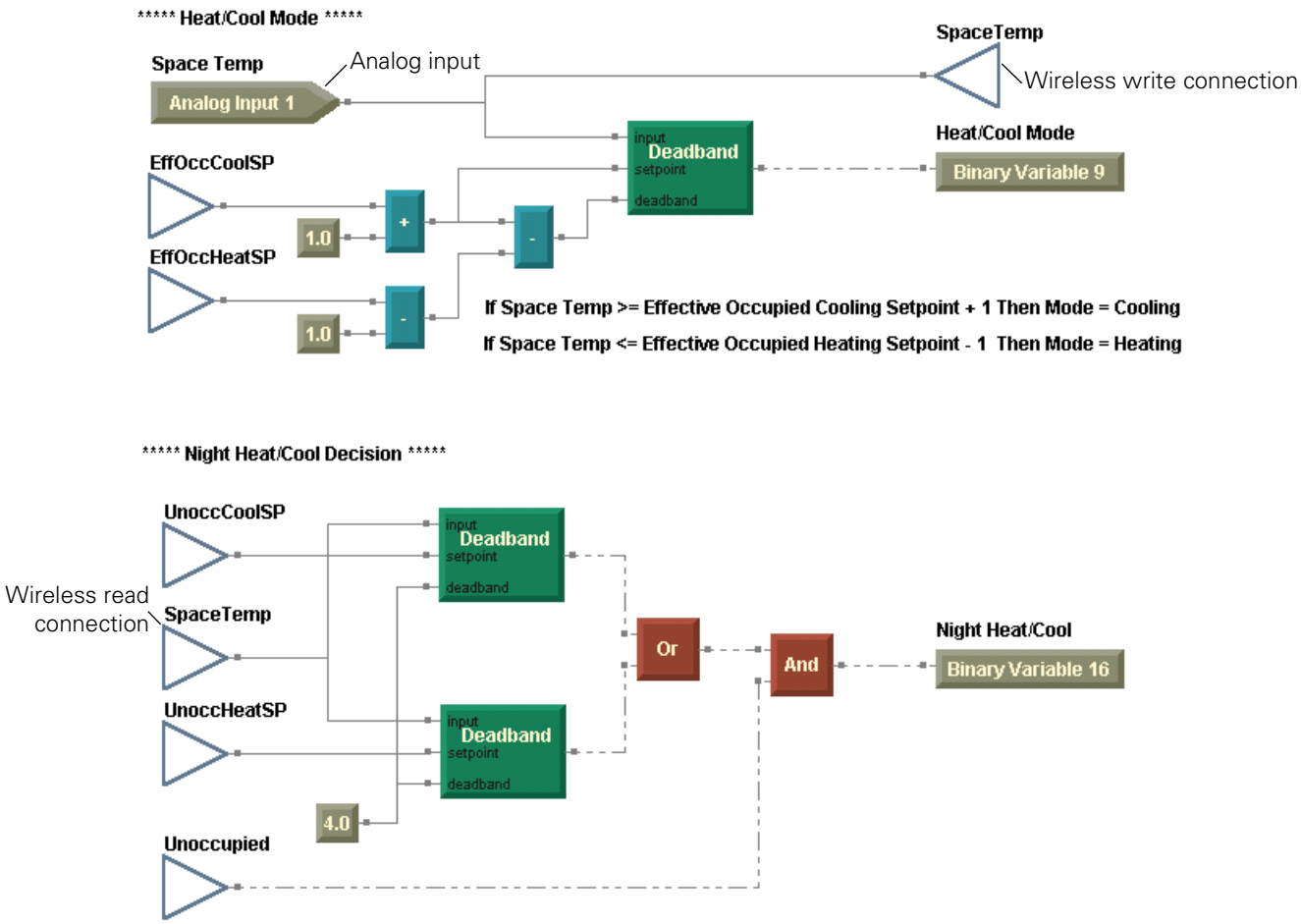
Read input values, including variables and the status of outputs, only once in any program.

There are two reasons to read input values only once in a program:

- The program runs more efficiently. Every time a program reads the value of an input, output, or variable, the program must access the internal database of the Tracer MP580/581 controller. This process consumes time.
- In the future, you will be able to simulate graphical programs. This will look a lot like debug mode; however, you will be able to run the program offline and enter values for each input. The fewer inputs, outputs, and variables that you read, the fewer values that you will be required to supply in simulation mode.

The wireless connection is a tool that makes this practice easy to follow. Use wireless connections to propagate values of inputs throughout the program, as shown in Figure 188 for the space temperature input.

Figure 188: Using wireless connections



Use comments to document your programs.

Comments can provide helpful information, and although they may not seem important to the author of the program, they are especially useful for those who open an unfamiliar program. In Figure 188, comments provide more detail about the heat/cool decision.

When possible, subdivide graphical programs according to the sequence of operation.

Subdividing a program into logical modules makes the program easier to read and understand. For example, a program for a cooling tower that controls a fan, a sump heater, and a pump, would consist of three corresponding modules.

Determine the number of programs required for an application by grouping control functions that require the same information and/or the same run frequency.

In the PCM and the UPCM, process control language (PCL) programs often handled very discrete functions due to their limited size. Graphical programs can be much larger. While an air-handling unit may have required 18 programs in a UPCM, the same air-handling unit functionality might be accomplished in four or five graphical programs. For example, a constant-volume air-handling unit requires the following control tasks:

- Effective space setpoint calculation
- Mode determination (heat/cool)
- Supply fan control
- Exhaust fan control
- Mixed air/outdoor air damper control
- Discharge air setpoint calculation
- Cooling valve control, including dehumidification
- Heating valve control, including dehumidification
- Humidification
- Alarm management

Not every control function will fit into one program. Identify tasks that require the same information, and/or the same run frequency, and group those tasks together to form programs. Table 32 presents a recommended grouping for a constant-volume air handler.

Table 32: Grouping programming tasks into programs

Program name	Control functions
ModeAndSetpoints	Effective space setpoint calculation
	Mode determination
FanControl	Supply fan control
	Exhaust fan control
MixedAirControl	Mixed air/outdoor air damper control

Table 32: Grouping programming tasks into programs (continued)

Program name	Control functions
DischargeAirControl	Discharge air setpoint calculation
	Cooling valve control
	Heating valve control
	Dehumidification
	Humidification
Alarms	Alarm management

Construct your own TGP library and reuse programs

As you program jobs, keep the program files (*.tgp) and the accompanying Rover configuration (*.rcf) files for the jobs. When you program another job that is similar, you can reuse both the configuration files and the programs from your library. You may need to modify the programs to accommodate differences in the sequences of operation, but reusing programs is much more efficient than writing them from scratch.

For more information about wiring, configuring, and programming the Tracer MP580/581 controller, see the following:

- Tracer MP581 Programmable Controller Hardware Installation guide (CNT-SVN01A-EN)
- Tracer MP580/581 Programmable Controller Programming guide (CNT-SVP01A-EN)
- Online Help in the Rover service tool and the TGP editor
- PID Control in Tracer Controllers applications guide (CNT-APG002-EN)

Summary-question answers

The following sections include the answers to the summary questions for each chapter.

Chapter 2, "Writing the exhaust fan program"

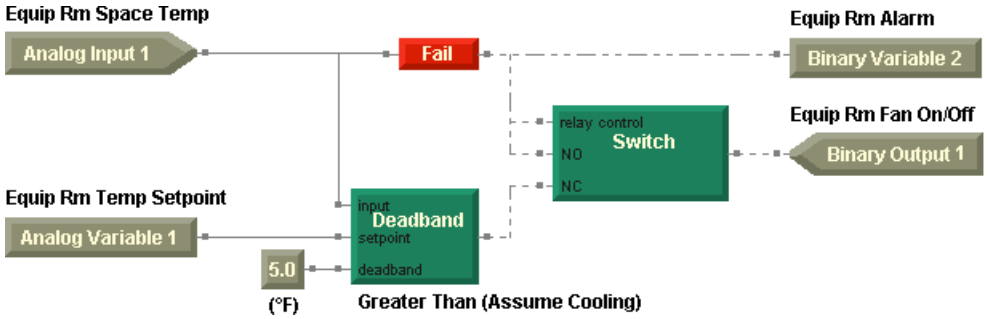
1. Analog (solid wire) and binary (dashed wire)
2. No. Some blocks do not have a properties dialog box. These blocks have only preset properties that cannot be changed.
3. Start wires on the input or output port of a given block, or on another wire. End wires on input and output ports only.

Chapter 3, "Modifying the exhaust fan program"

1. No. The abilities of a single Variable block to read and write to an analog or binary variable are mutually exclusive. It can either read or write but not both.
2. The control source determines the read or write capability of a Variable block. Possible sources include the Tracer Summit system, the operator display or service tool, a program, or the operator display or service tool and a program. For information about which control sources allow read-only, write-only, read/write capabilities, see Table 4 on page 34.
3. The normally open (NO) input value passes to the output of the Switch block when the relay control input is true.
4. The relay control input port is always binary, regardless of the Switch block type (analog or binary).
5. For the Fail block to detect properly an input failure, it must be applied to an analog input configured as thermistor, Balco, Platinum, current, voltage, or resistance. The Fail at End of Range option must also be selected for the input in the Configuration dialog box. The Fail block may also be connected to either a Network Variable Input block or a Network Configuration Input block.
6. The Fail at End of Range check box must be selected in the configuration for the input connected to the Fail block for the block to work properly.

7. Delete the ON constant block. Connect the output of the Fail block to both the Relay Control and Normally Open ports of the Switch block (Figure 189).

Figure 189: Equip Room Exhaust Fan program with a modification



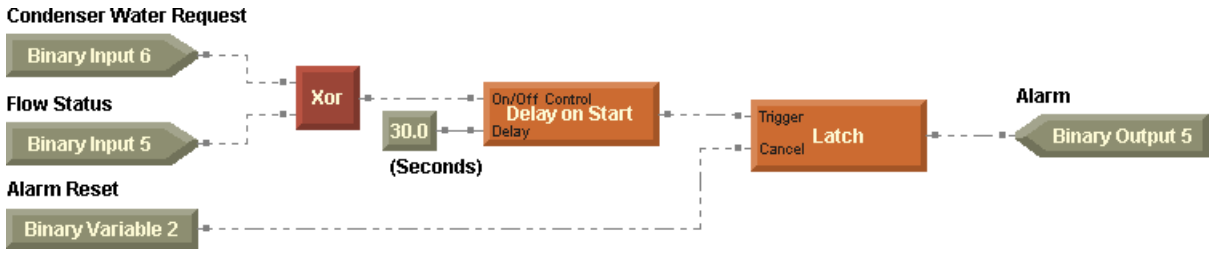
Chapter 4, "Cooling tower with two-speed fan example"

1. The Latch block may be configured in timed or manual mode. In manual mode, it does not provide for a time-interval input.
2. Omit the Delay on Start block and its associated Constant block.

Chapter 5, "Cooling tower with variable-speed fan example"

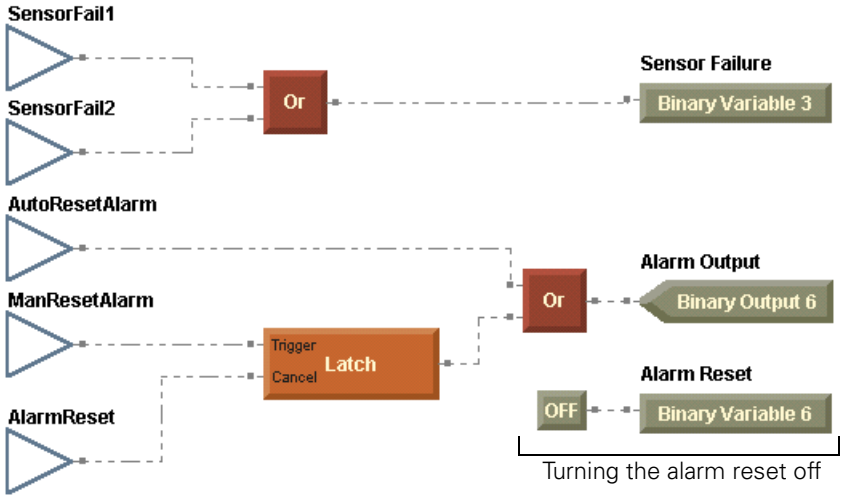
1. The Feedback Alarm block is a combination of the XOR, Delay on Start, and Latch blocks. The sequence of operation might read as follows: Compare Request to Status. If Request does not equal Status for a period of 30 seconds, initiate an alarm. The alarm must be resettable. See Figure 190 for the possible substitution.

Figure 190: Feedback Alarm block substitution



2. See Figure 191 on page 237 for an example of how you could automatically reset the alarm reset without using a Switch block. Each time the program runs, the Alarm Reset is set to off.

Figure 191: Automatically resetting the alarm reset without a Switch block



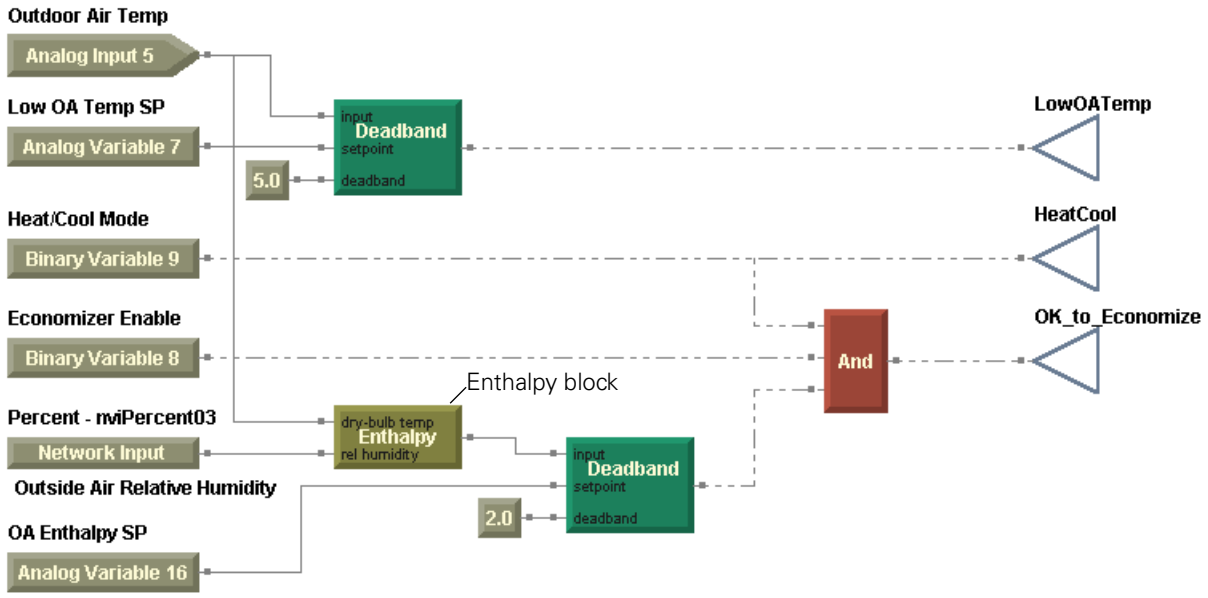
Chapter 6, "VAV AHU example"

1. In the program, Discharge Air Control, use the Enthalpy block to calculate the outside air enthalpy. Note that the relative humidity in this solution is obtained with a nvi and that the Deadband block remains an integral part of the economizer decision. Compare Figure 192 on page 238 with Figure 113 on page 121, where the outdoor air temperature is used.

➤ ***Using the Enthalpy block***

Use the Enthalpy block to calculate the enthalpy of moist air based on dry-bulb temperature (°F or °C) and relative humidity (%). Select the input and output units in the properties dialog box.

Figure 192: Using outdoor air enthalpy



2. First, create a calculation to track fan run hours and starts in the Calculations application (Figure 193). Then construct a small program module within the Alarms program to turn on the variable, Maintenance Required, when the calculated hours exceeds 4,000 hours. Be sure to include a provision to reset the binary variable, Maintenance Timer Reset (Figure 194 on page 239).

Figure 193: Fan Maintenance calculation setup

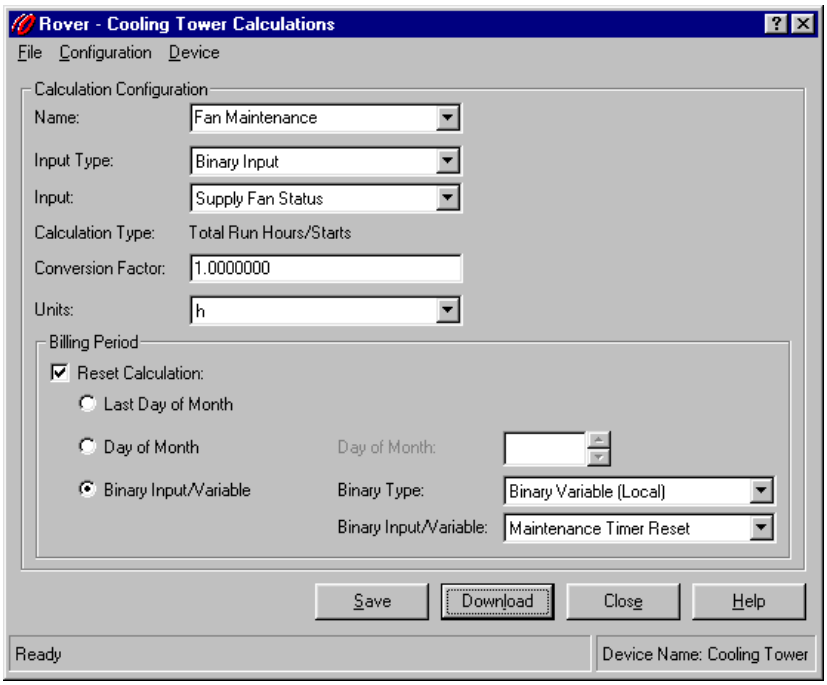
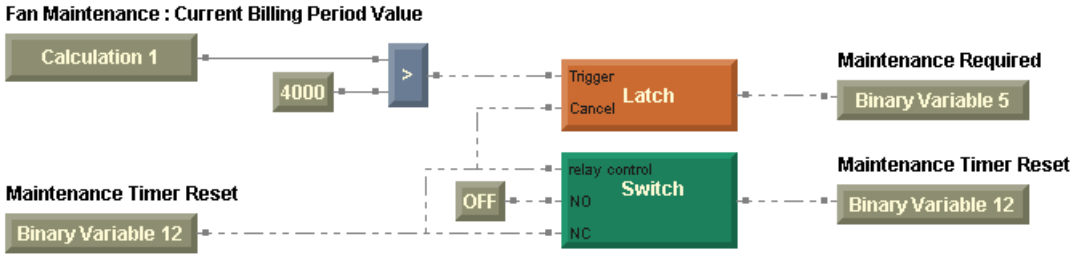


Figure 194: Maintenance timer indication and reset



3. Add the blocks shown in Figure 195 to the effective setpoint calculation (Figure 196 on page 240). The binary variable, Thumbwheel SP Enable, activates the thumbwheel setpoint. If the wired setpoint fails, the program sources the space temperature setpoint from the operator display. The Limit block applies appropriate limits to the setpoint. A wireless connection passes the resulting space temperature setpoint to the effective setpoint calculation.

Figure 195: Adding a setpoint source option

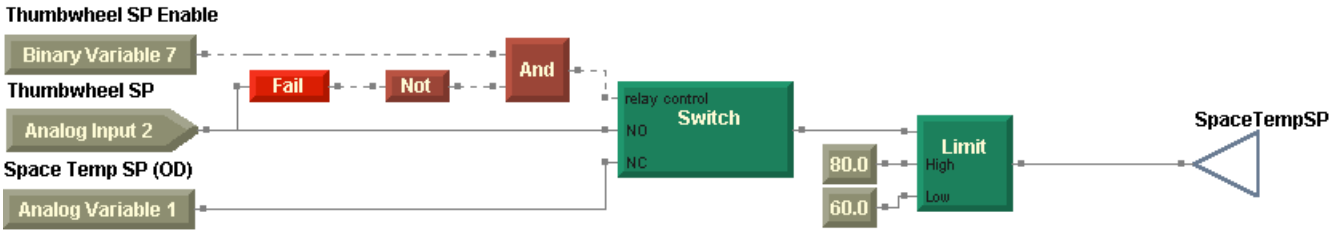
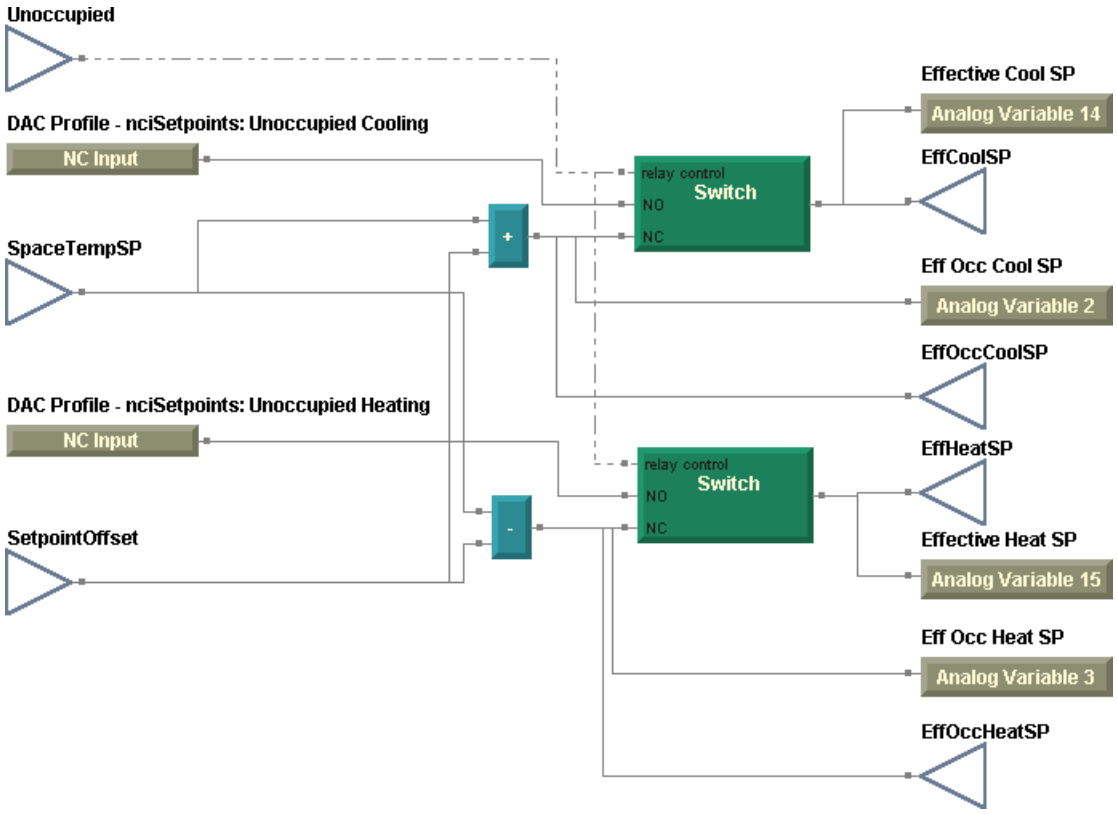


Figure 196: Effective setpoint calculation

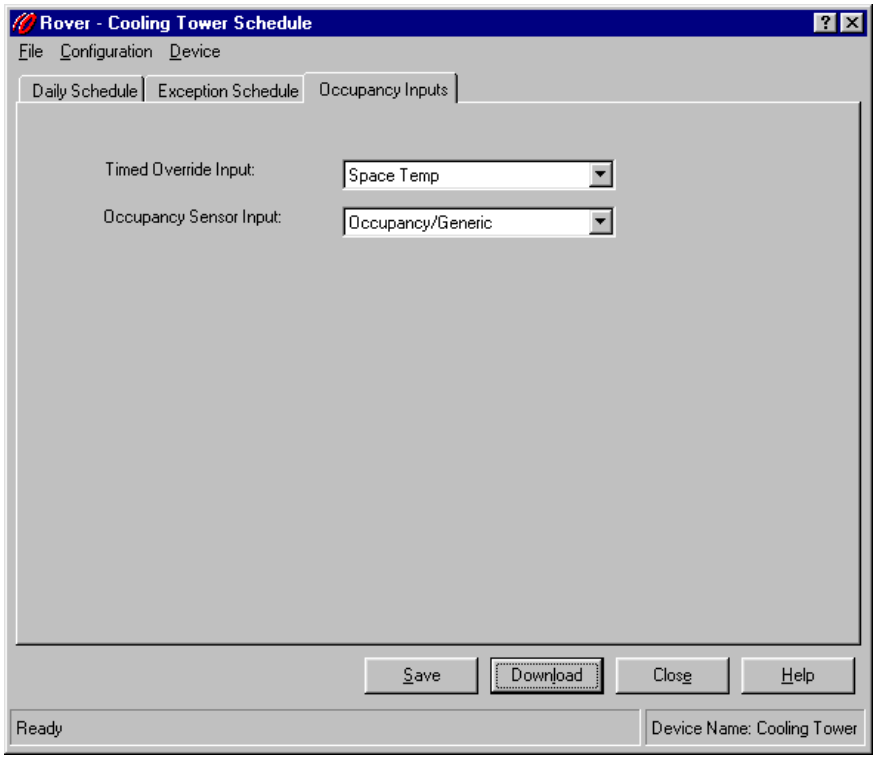


4. Replace the network configuration inputs, nciDAClSP and nciDAHtSP, with analog variables that are adjustable from the operator display.

Chapter 7, "Constant-volume AHU example"

1. A universal input configured as a a binary input must be selected as the occupancy sensor input in the Schedule application. See Figure 197.

Figure 197: Schedule application occupancy inputs



2. The Occupancy block, used in both the modes and setpoints program and the fan control program, provides the occupancy mode of the controller. Remember, the Occupancy block outputs one of four states, which is determined by the Schedule application for the controller:

   0 = Occupied

   1 = Unoccupied

   2 = Occupied Bypass

   3 = Occupied Standby

   When the controller enters occupied bypass mode, the Occupancy block outputs a value of 2. Take a closer look at the application of the Occupancy block in the modes and setpoints program (Figure 158 on page 182). The associated De-Enumerator block is setup to distinguish between occupied standby and unoccupied modes. In occupied and occupied bypass modes, the Switch block remains at its normally

closed position and as a result, the calculated offset value is based on the occupied setpoints.

The Occupancy and De-Enumerator block combination appears again in the fan control program (Figure 134 on page 156). The fan is requested to start in any occupancy mode with the exception of unoccupied mode.

The last piece of the puzzle answers the following question: How does the Occupancy block know to indicate occupied bypass mode? For the controller to sense that it must enter occupied bypass mode, a universal input configured as a thermistor (zone sensor) must be selected as the timed override input in the Schedule application. See Figure 197 on page 241.

3. The network variable input providing a temperature may be checked for an invalid value using the Fail block (Figure 198).

Figure 198: Checking a network variable input for valid data

In some cases, a Fail block may be used to check the network variable inputs for invalid data. Table 33 shows the network variable types (SNVTs) that the Fail block checks for invalid data. You see the hexadecimal invalid values when setting up custom bindings in the Rover service tool, and you see the decimal values when debugging graphical programs in the TGP editor.

Table 33: Network variable input (nvi) and network configuration input (nci) invalid values

SNVT	Invalid state	
	Hexadecimal	Decimal
SNVT_flow	0xFFFF	65535
SNVT_hvac_emerg	0xFF	255
SNVT_hvac_mode	0xFF	255
SNVT_hvac_overid	0xFF	255
SNVT_hvac_status	0xFF	255
SNVT_lev_percent	0x7FFF	32767
SNVT_occupancy	0xFF	255
SNVT_press_p	0x7FFF	32767
SNVT_switch	0xFF	255
SNVT_temp_p	0x7FFF	327.67

Note: The hexadecimal invalid values appear when setting up custom bindings in the Rover service tool. The decimal values appear when debugging graphical programs in the TGP editor.

Table 33: Network variable input (nvi) and network configuration input (nci) invalid values (continued)

SNVT	Invalid state	
	Hexadecimal	**Decimal**
SNVT_temp_setpt	0x7FFF	327.67
SNVT_tod_event	0xFF	255
Note: The hexadecimal invalid values appear when setting up custom bindings in the Rover service tool. The decimal values appear when debugging graphical programs in the TGP editor.		

Chapter 8, "Constant-volume AHU with warm-up, pre-cool, and communications"

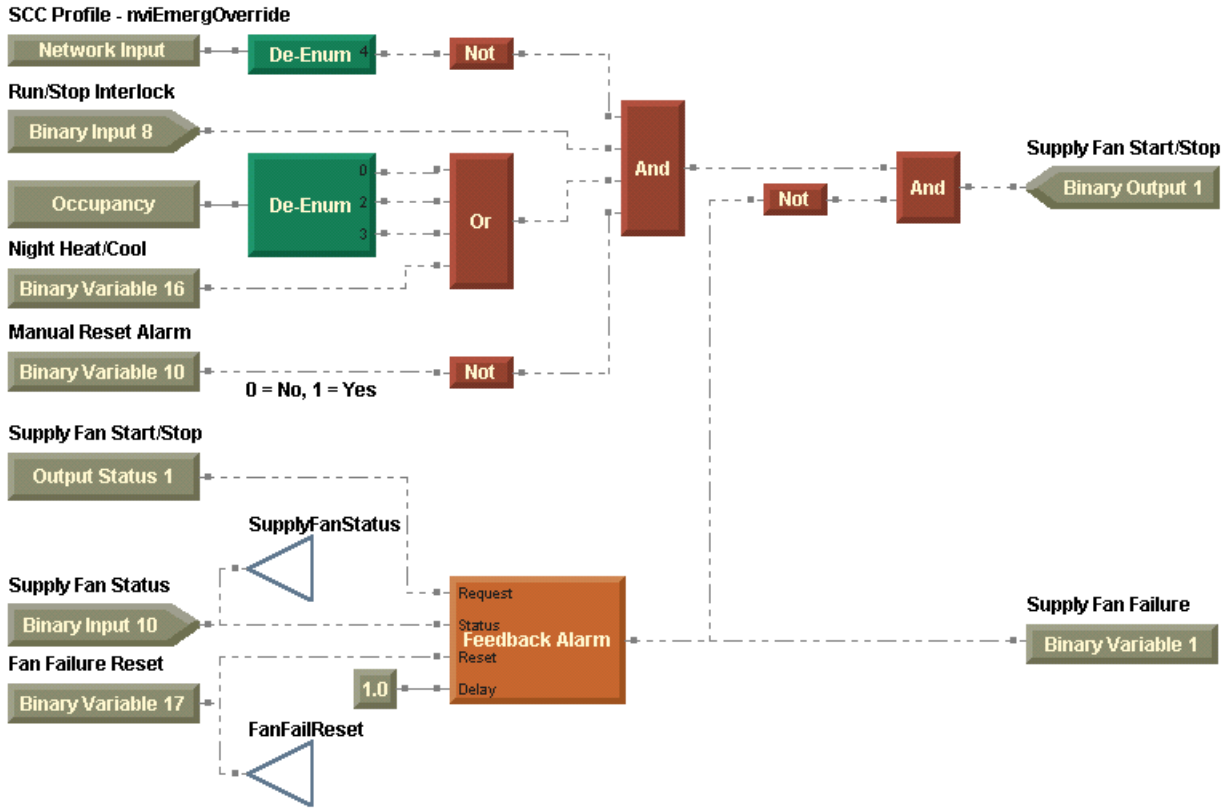
1. Enumerations for the SNVT corresponding to nviEmergOverride are shown in Table 34.

Table 34: SNVT_hvac_emerg (emerg_t) enumerations

Value	Identifier	Notes
0	EMERG_NORMAL	No emergency mode
1	EMERG_PRESSURIZE	Emergency pressurize mode
2	EMERG_DEPRESSURIZE	Emergency depressurize mode
3	EMERG_PURGE	Emergency purge mode
4	EMERG_SHUTDOWN	Emergency shutdown mode
5	EMERG_FIRE	Emergency fire mode
–1	EMERG_NUL	Invalid value
Note: This table was adapted from the LonMark Standard Enumeration Master List.		

Figure 199 on page 244 shows how nviEmergOverride may be incorporated into the supply fan control programming. In the program, the supply fan is allowed to run as long as nviEmergOverride does not present a value of 4, which would require an emergency shutdown.

Figure 199: Adding nviEmergOverride



2. As noted in the question, the valve override may be set to off, open, or closed. Although additional parts of the variable may be used to communicate a percentage or flow value, the Tracer Summit system supports use of only the first element of the structure. Here, values of 0% and 100% are assumed for override modes of closed and open, respectively (Table 35).

Table 35: SNVT_hvac_overrid structure contents

Element	SNVT	Notes
State	hvac_overrid_t	Override mode, enumerated
Percent	SNVT_lev_percent	From –163.83% to 163.83%
Flow	SNVT_lev_percent	From –32,768 to 32,767

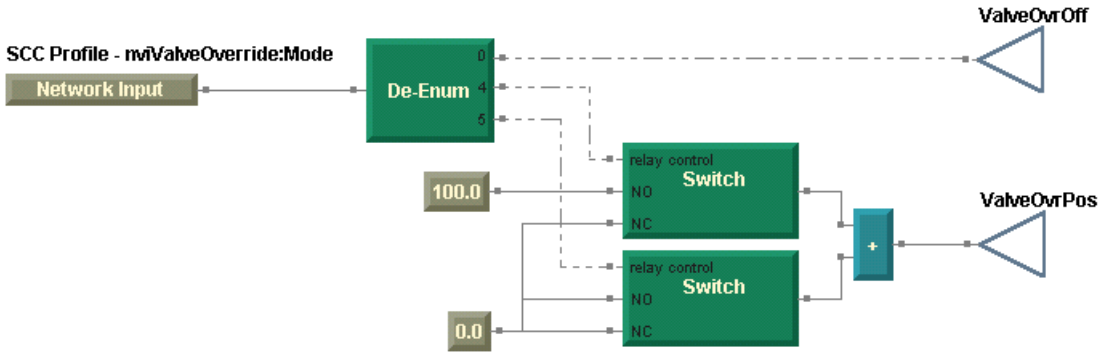
Selected enumerations for the state, or override mode, are shown in Table 36. Remember, you are checking for values of 0, 4, or 5.

Table 36: SNVT_hvac_overrid (hvac_overrid_t) enumerations

Value	Identifier	Notes
0	HVO_OFF	Not overridden
1	HVO_POSITION	Override position—use percent field
2	HVO_FLOW_VALUE	Override flow in liters/second—use flow field
3	HVO_FLOW_PERCENT	Override flow percentage—uses percent field
4	HVO_OPEN	Override to position = 100%
5	HVO_CLOSE	Override to position = 0%
6	HVO_MINIMUM	Override to configured minimum
7	HVO_MAXIMUM	Override to configured maximum
–1	HVO_NUL	Invalid value
Note: This table was adapted from the LonMark Standard Enumeration Master List.		

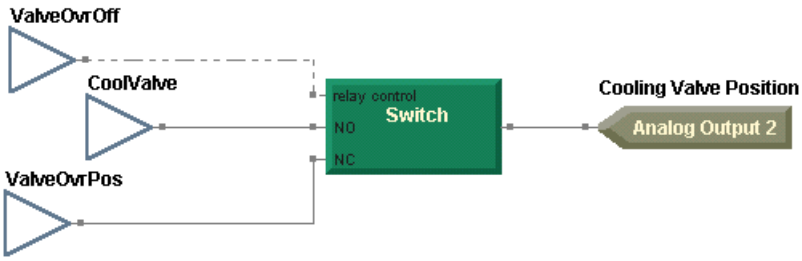
Figure 200 shows the logic used to interpret the valve override. The De-Enumerator block interprets the override mode. The Switch blocks select the appropriate desired valve position based on the override mode. The resulting wireless connections are used to communicate the override values to both the cooling and heating valves as shown in Figure 201 on page 246 and Figure 202 on page 246.

Figure 200: Valve override interpretation



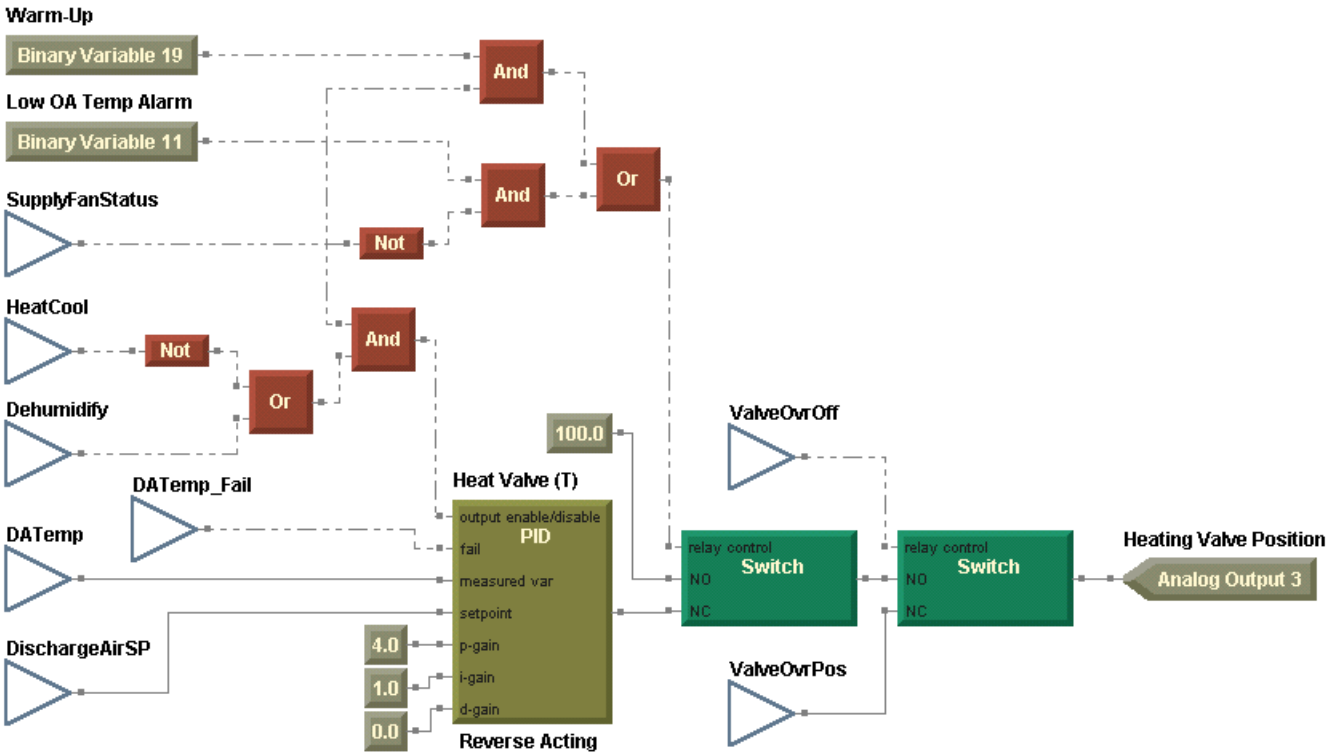
Next, start with the cooling valve module pictured in Figure 172 on page 208. Substitute a wireless connection, CoolValve, for the cooling valve analog output. Then add another Switch block to select the source of the cooling valve control. In normal operation, the calculated cooling valve position is passed to the analog output. When an override exists, the override value is passed to the analog output (Figure 201 on page 246).

Figure 201: Cooling valve override



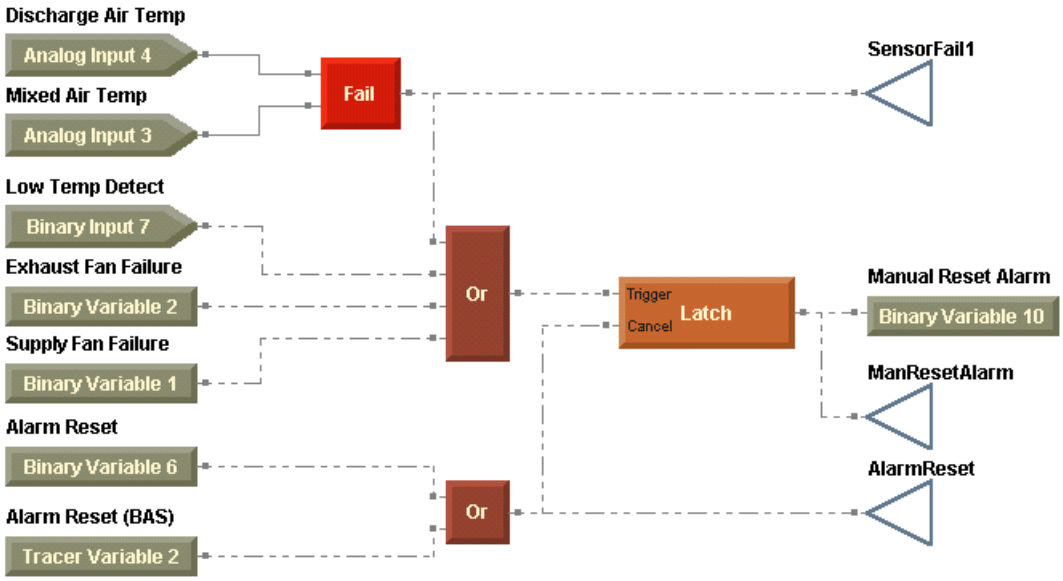
The heating valve works the same way, but there is enough space in the heating valve module to add the override implementation on the same page (Figure 202).

Figure 202: Heating valve control with override



3. To accomplish the same alarm reset functionality, you must do a combination of configuration and programming in the Tracer MP580/581 controller and at the Tracer Summit workstation. First, configure a binary variable to communicate the alarm reset value from the Tracer Summit system. One minor change is required to the alarms program. Add a binary variable to the manual reset alarm logic of the alarms program, as shown in Figure 203 on page 247.

Figure 203: Modified manual reset alarm logic



In Tracer Summit software, create a binary output to represent the alarm reset at the system level. In the Tracer MP580/581 editor, reference the desired binary variable to the newly created binary output. For instance, if the first Tracer Summit binary variable in the Tracer MP580/581 controller is configured as Alarm Reset (BAS) and you created a binary output in Tracer Summit called AHU Alarm Reset, reference the binary variable to the present value of the binary output in Tracer Summit. The operator uses this binary output to turn on the alarm reset.

Then write a CPL program to automatically turn the alarm reset off. The following CPL program has an execution frequency of 10 seconds and turns off the alarm reset after 1 minute, allowing sufficient time for it to be communicated to the Tracer MP580/581 controller.

```
PROGRAM MP_AlarmResetReset

 DEFINT Counter, i

 Counter = Local.{Saved Value}[1]
 IF ({AHU Alarm Reset}.{Present Value} = 1)
 THEN
    // Increment counter
    Counter = Counter + 1
 END IF
 IF (Counter > 5)
 THEN
    FOR i = 5 TO 16
       // Release control at priority levels 5 to 16
       CONTROL ({AHU Alarm Reset}, {Present Value}, 0, i, RELEASE)
    NEXT
    // Reset counter
    Counter = 0
 END IF
 Local.{Saved Value}[1] = Counter

END
```

Index

E

F

G

H

I

K

L

M

N

O

P

R

S

T

U

V

W

TRANE®

Trane
An American Standard Company
www.trane.com

For more information contact
your local district office or
e-mail us at comfort@trane.com

Literature Order Number	CNT-APG001-EN
File Number	SV-ES-BAS-CNT-APG-001-EN-0202
Supersedes	New
Stocking Location	La Crosse