Control Basic Programs Reference

(reflects firmware version R1.3.0.4 in a BAC-1xx63C)

General Information

This document provides a general reference to the programs in the FlexStat.

Because of firmware upgrades and differences between models, the programming in any particular FlexStat may be different from the programming shown here (firmware version 1.3.0.4 in a BAC-11163C)!

Before attempting to custom program a FlexStat, read ALL of the Custom Programming section in the BAC-10000 Series Application Guide (P/N 913-019-03)!

Using BACstage or TotalControl, a program (1 through 5) can be copied, pasted into a new program code object (6 through 10), edited, and run in place of the original. (Programs 1–5 cannot be edited, but they can be taken out of service.)

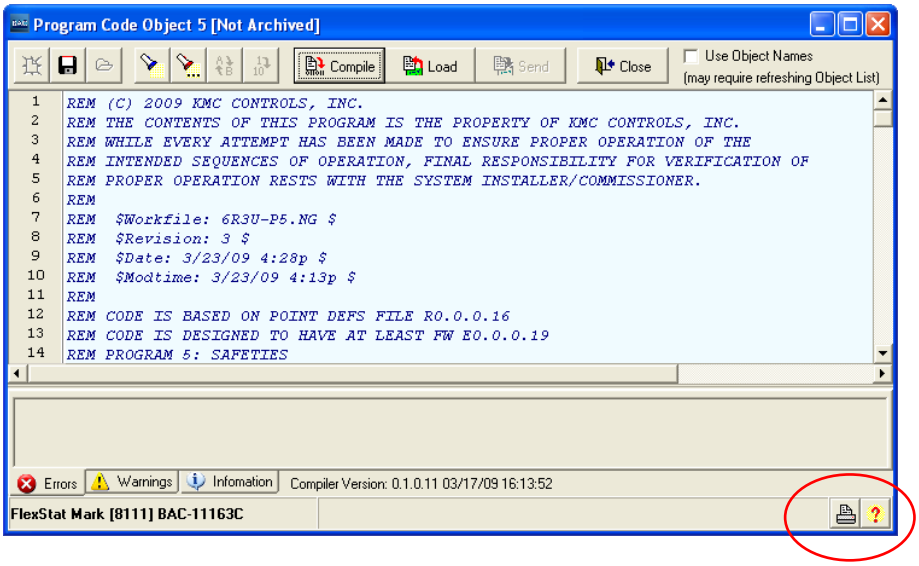
Control Basic Programs 1 through 5 are used for built-in applications and can NOT be modified directly. Programs 6 through 10 are empty and can be used for additional programming. Using BACstage or TotalControl, a program (1 through 5) can be copied, pasted into a new program code object (6 through 10), edited, and run in place of the original. (Although Programs 1–5 cannot be edited, they can be halted and set to not autorun after restart.)

Because programs are executed sequentially, if a program is copied and modified, ALL of the following programs must also be copied (even if they are not modified)! For example, if program 2 is copied into program 6 and modified, then programs 3 through 5 must also be copied into programs 7 through 9. Then programs 2 through 5 are set to NOT autorun, programs 6 through 9 are set to autorun, and the FlexStat is restarted.

See the Help system in TotalControl or BACstage for command definitions and uses.

Use the Search feature in TotalControl or BACstage to find a particular command, function, section, etc.

To more easily examine the code in a particular FlexStat, copy the code and paste it into a text editor or print it by selecting the printer icon (which, in BACstage, automatically prints to the default printer).



In Program 1 below, the bold text represents the "common code" and the italicized text immediately following that is the "branch" code (for a BAC-1xx63). Program 1 interacts with the user interface (display and menus). If Program 1 is copied and replaced, ensure that the common code (up to the application branch code section) in Program 1 is running. Failure to do so may disable one or more user interface functions.

Program 1: Setpoints and Modes

```
REM (C) 2009 KMC CONTROLS, INC.
REM THE CONTENTS OF THIS PROGRAM IS THE PROPERTY OF KMC CONTROLS, INC.
REM WHILE EVERY ATTEMPT HAS BEEN MADE TO ENSURE PROPER OPERATION OF THE
REM INTENDED SEQUENCES OF OPERATION, FINAL RESPONSIBILITY FOR VERIFICATION OF
REM PROPER OPERATION RESTS WITH THE SYSTEM INSTALLER/COMMISSIONER.
REM
REM  $Workfile: 6R3U-P1.NG $
REM  $Revision: 16 $
REM  $Date: 1/14/10 10:37a $
REM  $Modtime: 1/14/10 10:23a $
REM
REM CODE IS BASED ON POINT DEFS FILE R0.0.0.22
REM CODE IS DESIGNED TO HAVE AT LEAST FW E1.3.0.2
REM PROGRAM 1: SETPOINTS/MODES

REM MAP HUM SENSOR EVEN IF NOT CONFIGURED, THIS IS FOR FACTORY TESTING
IF BV20 OR MODELNUMBER = 10163 OR MODELNUMBER = 11163 THEN AV22 = AI5

REM EXECUTE NO CODE IF IN TEST MODE
IF MSV1 = 1 THEN END

REM MAP OAT & SPACE CO2
REM AV'S ARE USED SO THESE POINTS CAN BE DISPLAYED ON U/I NO MATTER WHERE THE
REM POINT IS PHYSICALLY CONNECTED
IF MSV1 = 2 OR MSV1 = 3 OR MSV1 = 5 THEN AV23 = AI4
REM AV24 = AI7


IF POWERLOSS THEN
REM INITIALIZE OPERATING AND DISPLAY SETPOINTS
AV27 = AV26
REM CLEAR ALARM FLAGS AFTER A WARM START
RLQ BV4
RLQ BV5
REM MUST DO AN END HERE SO ALARM VALUE CHANGES CAN PROPAGATE THROUGH TO EVENT OBJECTS
END
```

```
ENDIF


REM DEFINE PROGRAM LOCAL VARIABLES/CONSTANTS
CONST NOT_CONFIG = 1
CONST AHU = 2
CONST RTU = 3
CONST FCU = 4
CONST HP = 5
CONST PIPE2 = 1
CONST PIPE4 = 2
CONST ONE_FAN_SPEED = 1
CONST TWO_FAN_SPEEDS = 2
CONST THREE_FAN_SPEEDS = 3
CONST NO_ECON = 1
CONST MOD_ECON = 2
CONST ECON_EN_DIS = 3
CONST COMP_LOCKOUT = 2
CONST FAN_STATUS_PRESENT = 2
CONST MINUTES = 60
CONST HOURS = 3600
CONST TWO_HP_COMP = 2
CONST ZERO_DEADBAND = 1
CONST HALF_DEADBAND = 50
CONST FULL_DEADBAND = 99
CONST AUTO = 1
CONST HTG = 2
CONST CLG = 3
CONST OFF = 4
CONST EMER_HT = 5
CONST AUTO_FAN = 0
CONST ON_FAN = 1
CONST CLG_HTG_RESET = 4
CONST OAD_RESET = 6
REM THIS SECTION ADJUSTS FIXES VALUES TO EITHER DEG F OR DEG C
CONST DEG_F = 0
CONST DEG_C = 1
LOCALS OAD_PROP
LOCALS MAT_MIN
LOCALS MAT_MAX
LOCALS CHW
LOCALS HW
LOCALS WATER_DB
LOCALS COMP_LOCKOUT_DB
LOCALS ECON_ENBL_DB
LOCALS HALF_MAT_SPAN
LOCALS FC
FC = BV14
IF FC = DEG_C THEN
OAD_PROP = 8.33
MAT_MIN = 11.7
MAT_MAX = 20.0
CHW = 17.2
HW = 32.2
WATER_DB = 1.11
COMP_LOCKOUT_DB = 1.11
ECON_ENBL_DB = 1.11
HALF_MAT_SPAN = 4.17
ELSE IF FC = DEG_F THEN
OAD_PROP = 15
MAT_MIN = 53
MAT_MAX = 68
CHW = 63
HW = 90
WATER_DB = 2
COMP_LOCKOUT_DB = 2
ECON_ENBL_DB = 2
```

```
HALF_MAT_SPAN = 7.5
ENDIF
LOCALS OPPOSITE_WATER_AVLBL
LOCALS STPT_OVRD_OCC_STATUS
LOCALS OVRD_BY_STPT_ADJUST
LOCALS SPACE_TEMP
SPACE_TEMP = AI1
LOCALS FAN
FAN = BO1
LOCALS FAN2
FAN2 = BO2
LOCALS FAN3
FAN3 = BO3
LOCALS MIN_DEADBAND
MIN_DEADBAND = AV1
LOCALS ACT_CLG_STPT
ACT_CLG_STPT = AV2
LOCALS ACT_HTG_STPT
ACT_HTG_STPT = AV3
LOCALS UNOCC_CLG_STPT
UNOCC_CLG_STPT = AV4
LOCALS UNOCC_HTG_STPT
UNOCC_HTG_STPT = AV5
LOCALS DEADBAND
DEADBAND = AV6
LOCALS MIN_CLG_STPT
MIN_CLG_STPT = AV7
LOCALS MAX_HTG_STPT
MAX_HTG_STPT = AV8
LOCALS ECON_ENABLE_TEMP
ECON_ENABLE_TEMP = AV11
LOCALS DEHUM_STPT
DEHUM_STPT = AV15
LOCALS OVRD_TIMER
OVRD_TIMER = AV16
LOCALS COMP_OAT_LL
COMP_OAT_LL = AV17
LOCALS OCC_CLG_STPT
OCC_CLG_STPT = AV20
LOCALS OCC_HTG_STPT
OCC_HTG_STPT = AV21
LOCALS SPACE_HUM
SPACE_HUM = AV22
LOCALS OAT
OAT = AV23
LOCALS CURRENT_MODE_STPT
CURRENT_MODE_STPT = AV26
LOCALS UI_STPT
UI_STPT = AV27
LOCALS DEHUM_DEADBAND
DEHUM_DEADBAND = AV28
LOCALS MIN_STPT_DIFF
MIN_STPT_DIFF = AV33
LOCALS WATER_EVAL_TIMER
WATER_EVAL_TIMER = AV37
LOCALS MIN_EVAL_INTERVAL
MIN_EVAL_INTERVAL = AV38
LOCALS OPP_WATER_TIMER
OPP_WATER_TIMER = AV39
LOCALS OCC_MODE
LOCALS STPT_OVRD
STPT_OVRD = BV2
LOCALS FAN_ALARM
FAN_ALARM = BV5
LOCALS HTG_CLG_MODE
HTG_CLG_MODE = BV7
LOCALS OCC_FAN_MODE
```

```
OCC_FAN_MODE = BV12
LOCALS UNOCC_FAN_MODE
UNOCC_FAN_MODE = BV13
LOCALS DEHUM_ENABLE
DEHUM_ENABLE = BV20
LOCALS DEHUM_MODE
DEHUM_MODE = BV21
LOCALS OCC_SNSR_OVRD_ENBL
OCC_SNSR_OVRD_ENBL = BV24
LOCALS ALLOW_HTG_DEHUM
ALLOW_HTG_DEHUM = BV25
LOCALS APP_MAIN_TYPE
APP_MAIN_TYPE = MSV1
LOCALS APP_SUB_TYPE
APP_SUB_TYPE = MSV2
LOCALS FAN_CONTROL_OPTION
FAN_CONTROL_OPTION = MSV3
LOCALS OAD_OPTION
OAD_OPTION = MSV4
LOCALS FAN_STATUS_OPTION
FAN_STATUS_OPTION = MSV5
LOCALS AUX_HEAT_OPTION
AUX_HEAT_OPTION = MSV6
LOCALS SYSTEM_MODE
SYSTEM_MODE = MSV7
LOCALS CLG_LOOP
CLG_LOOP = LOOP1
LOCALS HTG_LOOP
HTG_LOOP = LOOP2
LOCALS DEHUM_HTG_LOOP
DEHUM_HTG_LOOP = LOOP4
REM CONDITIONAL LOCALS STATEMENTS
IF MODELNUMBER > 11000 THEN
LOCALS OCC_SNSR
OCC_SNSR = BI6
ENDIF
IF APP_MAIN_TYPE = FCU AND APP_SUB_TYPE = PIPE2 THEN
LOCALS WATER_TEMP
WATER_TEMP = AI3
LOCALS CLG_VALVE_NEED
CLG_VALVE_NEED = AV29
LOCALS HTG_VALVE_NEED
HTG_VALVE_NEED = AV30
LOCALS WATER_EVAL_NEED
WATER_EVAL_NEED = BV9
LOCALS CLG_AVLBL
CLG_AVLBL = BV10
LOCALS HTG_AVLBL
HTG_AVLBL = BV11
ENDIF

REM USER JUST MODIFIED THE SPACE SETPOINT - DO THIS CODE ONCE
IF AV26 <> AV27 THEN
REM TELL PROGRAM THAT OVERRIDE MODE WAS STARTED BY ADJUSTING SETPOINT
START OVRD_BY_STPT_ADJUST
REM SETPOINT LIMITS WON'T ALLOW CONTROLLING TO ONE SETPOINT
REM VALID HTG/CLG SETPOINT WILL BE ALLOWED AND OPPOSITE STPT
REM WILL BE THE LIMIT FOR THAT H/C MODE.
REM THE FOLLOWING LINE TRANSFERS THE VALUE FROM FIRMWARE-WRITTEN
REM PRIORITY 8 TO CONTROL BASIC PRIORITY LEVEL 9
AV27 = AV27
AV26 = AV27
START BV2
REM GRAB THE PRESENT OCC MODE SO CODE KNOWS WHEN TO END OVERRIDE MODE IF TIMER IS SET TO ZERO
IF+ BV2 THEN STPT_OVRD_OCC_STATUS = SCHED1
IF BV7 THEN
AV2 = AV27
```

```
ACT_CLG_STPT = AV2
AV3 = MIN(AV27 - MIN_STPT_DIFF , MAX_HTG_STPT)
ACT_HTG_STPT = AV3
ELSE
AV3 = AV27
ACT_HTG_STPT = AV3
AV2 = MAX(AV27 + MIN_STPT_DIFF , MIN_CLG_STPT)
ACT_CLG_STPT = AV2
ENDIF
RLQ AV27@8
ENDIF

REM CHECK FOR OCC SENSOR OVERRIDE, IF OCC SENSOR IS PRESENT AND THE OPTION IS ENABLED
IF MODELNUMBER > 11000 AND OCC_SNSR_OVRD_ENBL THEN
IF OCC_SNSR AND NOT OCC_MODE THEN START BV2
ENDIF
REM IF SPACE SETPT IS MODIFIED THEN OVERRIDE THE OCC MODE TO ON
IF BV2 THEN START BV1

REM THE 'SYSTEM OFF' FUNCTION STOPS OCCUPIED AND OVERRIDE MODES
IF SYSTEM_MODE = OFF THEN
STOP BV1
STOP BV2
ELSE
REM IF SYSTEM_MODE <> OFF AND THERE IS NO SETPT OVERRIDE, LET OCC_MODE GO BACK TO SCHEDULE
CONTROL
IF NOT BV2 THEN RLQ BV1@9
ENDIF

REM WAIT UNTIL THE VALUE OF BV1 IS DEFINITE TO SET OCC-MODE
OCC_MODE = BV1

REM FLEXSTAT IS IN SETPOINT OVERRIDE MODE
IF BV2 THEN
REM CHECK TO SEE IF HTG/CLG STPTS NEED TO BE SWAPPED - ONLY IF OVRD WAS STARTED BY ADJUSTING
STPT
IF OVRD_BY_STPT_ADJUST THEN
IF SYSTEM_MODE = AUTO THEN
IF HTG_CLG_MODE THEN
IF SPACE_TEMP < AV3 THEN
STOP BV7
AV3 = MIN(AV26 , MAX_HTG_STPT)
ACT_HTG_STPT = AV3
AV2 = AV3 + MIN_STPT_DIFF
ACT_CLG_STPT = AV2
ENDIF
ELSE IF NOT HTG_CLG_MODE THEN
IF SPACE_TEMP > AV2 THEN
START BV7
AV2 = MAX(AV26 , MIN_CLG_STPT)
ACT_CLG_STPT = AV2
AV3 = AV2 - MIN_STPT_DIFF
ACT_HTG_STPT = AV3
ENDIF
ENDIF
ENDIF
ELSE
REM OVRD WAS STARTED BY MOTION OR U/I START (NO STPT ADJUST)
REM SO GET OCC STPTS
AV2 = OCC_CLG_STPT
AV3 = OCC_HTG_STPT
REM UPDATE HGT/CLG MODE IF SPACE TEMP PASSES ACT SETPOINTS
IF SPACE_TEMP > AV2 THEN START BV7
IF SPACE_TEMP < AV3 THEN STOP BV7
ENDIF
REM CURRENT MODE SETPOINT IS SETPOINT FROM CURRENT HTG/CLG MODE
IF BV7 THEN
```

```
AV26 = AV2
ELSE
AV26 = AV3
ENDIF
AV27 = AV26
REM CHECK TO SEE IF SETPT OVRD MODE NEEDS TO STOP
IF OVRD_TIMER > 0 THEN
REM STOP OVERRIDE MODE IF THERE'S AN EXPIRED TIMER
IF TIMEON(BV2) > (OVRD_TIMER * HOURS) THEN
STOP BV2
REM ENSURE STPT ADJUST FLAG IS RESET
STOP OVRD_BY_STPT_ADJUST
ENDIF
ENDIF
REM END OVRD MODE IF SCHEDULE CHANGE
IF+ SCHED1 <> STPT_OVRD_OCC_STATUS THEN STOP BV2
ENDIF

REM FLEXSTAT IS IN NORMAL OPERATING MODE (NOT OVERRIDE)
IF NOT BV2 THEN
REM RESET OVRD BY STPT ADJUST FLAG, IF SET
STOP OVRD_BY_STPT_ADJUST
REM GRAB SETPOINTS UPON SCHEDULE CHANGE TO OCCUPIED
IF BV1 THEN
AV2 = OCC_CLG_STPT
AV3 = OCC_HTG_STPT
REM CURRENT MODE SETPOINT IS SETPOINT FROM CURRENT HTG/CLG MODE
IF BV7 THEN
AV26 = AV2
ELSE
AV26 = AV3
ENDIF
AV27 = AV26
ENDIF
REM GRAB SETPOINTS UPON SCHEDULE CHANGE TO UNOCCUPIED
IF NOT BV1 THEN
AV2 = UNOCC_CLG_STPT
AV3 = UNOCC_HTG_STPT
REM CURRENT MODE SETPOINT IS SETPOINT FROM CURRENT HTG/CLG MODE
IF BV7 THEN
AV26 = AV2
ELSE
AV26 = AV3
ENDIF
AV27 = AV26
ENDIF

REM UPDATE HGT/CLG MODE IF SPACE TEMP PASSES ACT SETPOINTS
IF SPACE_TEMP > AV2 THEN START BV7
IF SPACE_TEMP < AV3 THEN STOP BV7
ENDIF

REM SET UP SPACE CLG/HTG PID LOOPS WITH PROPER VALUES
REM PLUG DEADBAND INTO PROPORTIONAL VALUE
LOOP1.P = DEADBAND
LOOP2.P = DEADBAND
IF DEHUM_ENABLE THEN LOOP4.P = DEADBAND
REM PREVENT PID WINDUP ON CLG/HTG LOOPS
LOOP1.I = (CLG_HTG_RESET * DEADBAND / 100) * FAN * NOT FAN_ALARM * OCC_MODE
LOOP2.I = (CLG_HTG_RESET * DEADBAND / 100) * FAN * NOT FAN_ALARM * OCC_MODE
IF DEHUM_ENABLE THEN LOOP4.I = (CLG_HTG_RESET * DEADBAND / 100) * FAN * NOT FAN_ALARM *
OCC_MODE * DEHUM_MODE

REM DETERMINE HTG/CLG MODE
IF SYSTEM_MODE = CLG THEN START BV7
IF SYSTEM_MODE = HTG THEN STOP BV7
IF APP_MAIN_TYPE = HP AND SYSTEM_MODE = EMER_HT THEN STOP BV7
```

```
REM SET THE FAN ICON STATUS
REM FAN ICON ON DISPLAY IS CONTROLLED FROM CONTROL BASIC
REM FAN ICON IS TURNED ON WHENEVER DEFINED FAN OUTPUTS ARE ACTIVE
IF APP_MAIN_TYPE = AHU OR APP_MAIN_TYPE = RTU OR APP_MAIN_TYPE = HP THEN
BV18 = FAN
ENDIF
IF APP_MAIN_TYPE = FCU THEN
IF FAN_CONTROL_OPTION = ONE_FAN_SPEED THEN
BV18 = FAN
ENDIF
IF FAN_CONTROL_OPTION = TWO_FAN_SPEEDS THEN
BV18 = FAN OR FAN2
ENDIF
IF FAN_CONTROL_OPTION = THREE_FAN_SPEEDS THEN
BV18 = FAN OR FAN2 OR FAN3
ENDIF
ENDIF

REM EVALUATE AND SET FAN STATUS
IF FAN_STATUS_OPTION = FAN_STATUS_PRESENT THEN
IF AI2 < 1 THEN START BV16
IF AI2 > 1 THEN STOP BV16
ENDIF
LOCALS FAN_FEEDBACK
FAN_FEEDBACK = BV16

REM DETERMINE FAN ALARM STATUS BASED ON FAN STATUS AND FAN SPEED OPTIONS
IF FAN_STATUS_OPTION = FAN_STATUS_PRESENT THEN
IF FAN_CONTROL_OPTION = ONE_FAN_SPEED THEN
BV5 = FAN AND NOT FAN_FEEDBACK
ELSE IF FAN_CONTROL_OPTION = TWO_FAN_SPEEDS THEN
BV5 = (FAN OR FAN2) AND NOT FAN_FEEDBACK
ELSE IF FAN_CONTROL_OPTION = THREE_FAN_SPEEDS THEN
BV5 = (FAN OR FAN2 OR FAN3) AND NOT FAN_FEEDBACK
ENDIF
ELSE BV5 = 0

AUTO_FAN_CYCLE_NEED:
REM DETERMINES NEED FOR FAN ON CALL FOR CLG OR HTG IF EITHER FAN MODE IS SET TO AUTO

IF OCC_MODE AND OCC_FAN_MODE = ON_FAN THEN GOTO DEHUM_CONTROL
IF NOT OCC_MODE AND UNOCC_FAN_MODE = ON_FAN THEN GOTO DEHUM_CONTROL

NEED_FAN_CYCLE:
REM IF MAIN APP IS SET TO RTU OR MAIN APP IS SET TO HP AND SUB APP TYPE IS SET TO TWO
COMPRESSORS,
REM TURN ON FAN @ STPT PLUS 1/2 DEADBAND TO ALLOW FAN TO RUN W/ONLY ONE STAGE
REM ELSE FAN CYCLES BASED ON STPT PLUS FULL SPAN OF DEADBAND
IF APP_MAIN_TYPE = RTU OR APP_MAIN_TYPE = HP AND APP_SUB_TYPE = TWO_HP_COMP THEN
IF HTG_CLG_MODE THEN
IF CLG_LOOP > HALF_DEADBAND THEN START BV6
IF CLG_LOOP < ZERO_DEADBAND THEN STOP BV6
ELSE
IF HTG_LOOP > HALF_DEADBAND THEN START BV6
IF HTG_LOOP < ZERO_DEADBAND THEN STOP BV6
ENDIF
ELSE
IF HTG_CLG_MODE THEN
IF CLG_LOOP > FULL_DEADBAND THEN START BV6
IF CLG_LOOP < ZERO_DEADBAND THEN STOP BV6
ELSE
IF HTG_LOOP > FULL_DEADBAND THEN START BV6
IF HTG_LOOP < ZERO_DEADBAND THEN STOP BV6
ENDIF
ENDIF
```

```
DEHUM_CONTROL:

REM DEHUMIDIFICATION MODE
IF DEHUM_ENABLE THEN
REM EVALUATE DEHUM STATUS BASED - HOW IS BASED ON WHETHER DEHUM IS ALLOWED TO OPERATE IN HTG
MODE
IF ALLOW_HTG_DEHUM THEN
IF+ SPACE_HUM > DEHUM_STPT THEN START BV21 , START DEHUM_MODE
ELSE
IF+ SPACE_HUM > DEHUM_STPT AND HTG_CLG_MODE THEN START BV21 , START DEHUM_MODE
ENDIF
IF SPACE_HUM < DEHUM_STPT - DEHUM_DEADBAND THEN STOP BV21 , STOP DEHUM_MODE
REM START FAN IF DEHUM MODE IS ACTIVE
IF DEHUM_MODE THEN START BV6
ENDIF

REM STOP CALL FOR FAN IF SYSTEM MODE IS SET TO 'OFF' AND CURRENT FAN MODE IS 'AUTO'
IF SYSTEM_MODE = OFF THEN
IF OCC_MODE AND OCC_FAN_MODE = AUTO_FAN OR NOT OCC_MODE AND UNOCC_FAN_MODE = AUTO_FAN THEN
STOP BV6
ENDIF
ENDIF

REM PERFORM APPROPRIATE SECTION OF CODE BASED ON FLEXSTAT CONFIGURATION
IF APP_MAIN_TYPE = AHU THEN
GOTO AHU_CODE
ELSE IF APP_MAIN_TYPE = RTU THEN
GOTO RTU_CODE
ELSE IF APP_MAIN_TYPE = FCU THEN
IF APP_SUB_TYPE = PIPE2 THEN GOTO FCU_2_PIPE_CODE
IF APP_SUB_TYPE = PIPE4 THEN GOTO FCU_4_PIPE_CODE
ELSE IF APP_MAIN_TYPE = HP THEN
GOTO HP_CODE
ENDIF

END

AHU_CODE:

REM CHECK FOR AND PERFORM ECON SEQUENCE
IF OAD_OPTION = MOD_ECON THEN
GOSUB MOD_ECON_CODE
ELSE
DISABLE BV3
LOOP3.I = 0
ENDIF

END

RTU_CODE:

REM CHECK FOR AND PERFORM ECON SEQUENCE
IF OAD_OPTION = MOD_ECON THEN
GOSUB MOD_ECON_CODE
ELSE IF OAD_OPTION = ECON_EN_DIS THEN
GOSUB ECON_EN_DIS_CODE
ELSE IF OAD_OPTION = NO_ECON THEN
DISABLE BV3
ENDIF

END

FCU_2_PIPE_CODE:

REM AVOID PID WINDUP FOR SPACE TEMP LOOP BASED ON FAN SPEED OPTION
IF FAN_CONTROL_OPTION=ONE_FAN_SPEED THEN LOOP1.I=(CLG_HTG_RESET*DEADBAND/100)*FAN*NOT
FAN_ALARM*OCC_MODE
```

```
IF FAN_CONTROL_OPTION=TWO_FAN_SPEEDS THEN LOOP1.I=(CLG_HTG_RESET*DEADBAND/100)*(FAN OR
FAN2)*NOT FAN_ALARM*OCC_MODE
IF FAN_CONTROL_OPTION=THREE_FAN_SPEEDS THEN LOOP1.I=(CLG_HTG_RESET*DEADBAND/100)*(FAN OR FAN2
OR FAN3)*NOT FAN_ALARM*OCC_MODE

REM DETERMINE VALVE NEED
IF OCC_MODE AND OCC_FAN_MODE = AUTO_FAN THEN GOTO C1_CYCLE_VALVE_NEED
IF NOT OCC_MODE AND UNOCC_FAN_MODE = AUTO_FAN THEN GOTO C1_CYCLE_VALVE_NEED

C1_MOD_VALVE_NEED:
REM CLG VALVE NEED
AV29 = MAX( 0, MIN( 100, LOOP1 * 1.6667 ) )

REM HTG VALVE NEED
AV30 = MAX( 0, MIN( 100, LOOP2 * 1.6667 ) )

REM IF MOD, SKIP CYCLE CODE
GOTO C1_SYSTEM_MODE_CONTROL:

C1_CYCLE_VALVE_NEED:
REM CLG VALVE NEED
IF SPACE_TEMP > ACT_CLG_STPT + DEADBAND THEN AV29 = 100
IF SPACE_TEMP < ACT_CLG_STPT THEN AV29 = 0

REM HTG VALVE NEED
IF SPACE_TEMP < ACT_HTG_STPT - DEADBAND THEN AV30 = 100
IF SPACE_TEMP > ACT_HTG_STPT THEN AV30 = 0

C1_SYSTEM_MODE_CONTROL:
REM DISABLE APPROPRIATE MODE IF SYSTEM MODE DOESN'T ALLOW THAT MODE
IF SYSTEM_MODE = HTG OR SYSTEM_MODE = OFF THEN AV29 = 0 , CLG_VALVE_NEED = 0
IF SYSTEM_MODE = CLG OR SYSTEM_MODE = OFF THEN AV30 = 0 , HTG_VALVE_NEED = 0

C1_CHECK_FOR_WATER_EVAL:
REM GENERATE NEED FOR WATER EVAL
IF CLG_VALVE_NEED >10 AND NOT CLG_AVLBL THEN START BV9
IF HTG_VALVE_NEED >10 AND NOT HTG_AVLBL THEN START BV9
REM BUT TURN OFF WATER EVAL NEED IF OPPOSITE WATER IS KNOWN AVAILABLE
IF OPPOSITE_WATER_AVLBL THEN STOP BV9
REM EVAL WATER ON GOING FROM UNOCC TO OCC
IF+ OCC_MODE THEN
IF NOT (CLG_AVLBL OR HTG_AVLBL) THEN START BV9
ENDIF

C1_WATER_EVAL_NEED:
REM PERFORM WATER EVAL NEED
IF WATER_EVAL_NEED AND NOT OPPOSITE_WATER_AVLBL THEN
REM ALLOW ENOUGH WATER TO PASS TO SENSE
IF TIMEON(WATER_EVAL_NEED) > WATER_EVAL_TIMER * MINUTES THEN
IF WATER_TEMP <= CHW THEN
REM SET CLG MODE/FLAG
START BV7
START BV10
ENDIF
IF WATER_TEMP >= HW THEN
REM SET HGT MODE/FLAG
STOP BV7
START BV11
ENDIF
REM TURN WATER EVAL MODE BACK OFF
STOP BV9
ENDIF
ENDIF

REM IF YOU NEED ONE KIND OF WATER & OTHER IS AVAILABLE, THEN START A TIMER
REM THAT PREVENTS THE UNIT FROM CHECKING WATER AVAILBALE NO SOONER THAN
REM EVERY FIFTEEN MINUTES
```

```
IF NOT OPPOSITE_WATER_AVLBL THEN
IF CLG_VALVE_NEED > 10 AND BV11 THEN START OPPOSITE_WATER_AVLBL
IF HTG_VALVE_NEED > 10 AND BV10 THEN START OPPOSITE_WATER_AVLBL
ENDIF
IF TIMEON(OPPOSITE_WATER_AVLBL) > OPP_WATER_TIMER * MINUTES THEN STOP OPPOSITE_WATER_AVLBL

REM CLEAR WATER AVAILABLE FLAGS IF VALVE NEED GOES TO ZERO
REM YOU NO LONGER KNOW THE WATER TEMP WITH THE VALVE CLOSED
REM VALVE NEED MUST BE ZERO FOR AT LEAST 10 MINUTES
IF TIMEOFF(CLG_VALVE_NEED) > MIN_EVAL_INTERVAL * MINUTES THEN STOP BV10
IF TIMEOFF(HTG_VALVE_NEED) > MIN_EVAL_INTERVAL * MINUTES THEN STOP BV11

REM CLEAR WATER AVAILABLE FLAGS IF WATER TEMP LEAVES RANGE OF NEEDED WATER
IF CLG_VALVE_NEED > 0 AND CLG_AVLBL AND WATER_TEMP > CHW+WATER_DB THEN STOP BV10
IF HTG_VALVE_NEED > 0 AND HTG_AVLBL AND WATER_TEMP < HW-WATER_DB THEN STOP BV11

REM TURN OFF FAN IF UNIT IS IN OPPOSITE WATER AVAILABLE MODE
IF OPPOSITE_WATER_AVLBL THEN STOP BV6

END

FCU_4_PIPE_CODE:

REM AVOID PID WINDUP FOR SPACE TEMP LOOP BASED ON FAN SPEED OPTION
IF FAN_CONTROL_OPTION=ONE_FAN_SPEED THEN LOOP1.I=(CLG_HTG_RESET*DEADBAND/100)*FAN*NOT
FAN_ALARM*OCC_MODE
IF FAN_CONTROL_OPTION=TWO_FAN_SPEEDS THEN LOOP1.I=(CLG_HTG_RESET*DEADBAND/100)*(FAN OR
FAN2)*NOT FAN_ALARM*OCC_MODE
IF FAN_CONTROL_OPTION=THREE_FAN_SPEEDS THEN LOOP1.I=(CLG_HTG_RESET*DEADBAND/100)*(FAN OR FAN2
OR FAN3)*NOT FAN_ALARM*OCC_MODE

END

HP_CODE:

REM AUX HEAT COMP LOCKOUT MODE
IF AUX_HEAT_OPTION = COMP_LOCKOUT THEN
REM EVALUATE THE COMP LOCKOUT MODE IF SELECTED IN SETUP
REM IF WARM ENOUGH, DISABLE COMP LOCKOUT
IF OAT > COMP_OAT_LL + COMP_LOCKOUT_DB/2 THEN DISABLE BV8
REM IF COLD ENOUGH, ENABLE COMP LOCKOUT
IF OAT < COMP_OAT_LL - COMP_LOCKOUT_DB/2 THEN ENABLE BV8
ELSE
REM DO THIS IF COMP LOCKOUT IS NOT SELECTED IN SETUP
DISABLE BV8
ENDIF

REM CHECK FOR AND PERFORM ECON SEQUENCE
IF OAD_OPTION = MOD_ECON THEN
GOSUB MOD_ECON_CODE
ELSE
DISABLE BV3
LOOP3.I = 0
ENDIF

END

MOD_ECON_CODE:

REM THIS CODE CAN BE RUN FROM ANY APPLICABLE APP TO MODULATE AN OAD OUTPUT
REM ACT BASED ON ECON OPTION STATUS

REM ECON MODE ONLY ENABLES IF CLG MODE IS ACTIVE
IF BV7 THEN
REM DETERMINE ECON MODE
IF FAN AND NOT FAN_ALARM AND OAT < ECON_ENABLE_TEMP - ECON_ENBL_DB/2 THEN ENABLE BV3
IF FAN_ALARM OR NOT FAN OR OAT > ECON_ENABLE_TEMP + ECON_ENBL_DB THEN DISABLE BV3
```

```
ELSE
DISABLE BV3
ENDIF

REM PLUG PROP VALUE INTO OAD LOOP
LOOP3.P = OAD_PROP
REM AVOID PID WINDUP FOR MAT LOOP
LOOP3.I = (OAD_RESET*OAD_PROP/100) * OCC_MODE * BV3

REM DETERMINE MAT SETPT BASED ON COOLING NEED AND OCC MODE
IF OCC_MODE THEN
REM MAT STPT IS RESET BETWEEN THE MAT MIN AND MAX SETTINGS OVER THE RANGE (DEMAND) OF
REM THE SPACE TEMP VARYING BY 2 DEG. F IN OCC MODE
AV9 = MIN( MAT_MAX , MAX( MAT_MIN , MAT_MAX - ( HALF_MAT_SPAN * ( SPACE_TEMP - ACT_CLG_STPT )
) ) )
ELSE
REM MAT STPT IS SET TO FULL CLG IN UNOCC MODE
AV9 = MAT_MIN
ENDIF

REM CHECK FOR MAT LOW LIMIT ALARM
IF AI3 < AV18 THEN START BV4

RETURN

ECON_EN_DIS_CODE:

REM THIS CODE CAN BE RUN FROM ANY APPLICABLE APP TO SIMPLY ENABLE AND DISABLE AN ECON
REM OUTPUT WHERE THE DAMPER IS CONTROLLLED FROM THE HVAC UNIT'S INTEGRAL CONTROLS

REM ECON MODE ONLY ENABLES IF CLG MODE IS ACTIVE
IF BV7 THEN
REM DETERMINE ECON MODE
IF FAN AND NOT FAN_ALARM AND OAT < ECON_ENABLE_TEMP - ECON_ENBL_DB/2 THEN ENABLE BV3
IF FAN_ALARM OR NOT FAN OR OAT > ECON_ENABLE_TEMP + ECON_ENBL_DB THEN DISABLE BV3
ELSE
DISABLE BV3
ENDIF

REM CHECK FOR MAT LOW LIMIT ALARM
IF AI3 < AV18 THEN START BV4

RETURN
```

Program 2: Fan Control

```
REM (C) 2009 KMC CONTROLS, INC.
REM THE CONTENTS OF THIS PROGRAM IS THE PROPERTY OF KMC CONTROLS, INC.
REM WHILE EVERY ATTEMPT HAS BEEN MADE TO ENSURE PROPER OPERATION OF THE
REM INTENDED SEQUENCES OF OPERATION, FINAL RESPONSIBILITY FOR VERIFICATION OF
REM PROPER OPERATION RESTS WITH THE SYSTEM INSTALLER/COMMISSIONER.
REM
REM  $Workfile: 6R3U-P2.NG $
REM  $Revision: 4 $
REM  $Date: 7/14/09 10:24a $
REM  $Modtime: 7/14/09 8:43a $
REM
REM CODE IS BASED ON POINT DEFS FILE R0.0.0.22
REM CODE IS DESIGNED TO HAVE AT LEAST FW E1.1.0.5
REM PROGRAM 2: FAN CONTROL

REM EXECUTE NO CODE IF IN TEST MODE
IF MSV1 = 1 THEN END

REM DEFINE PROGRAM LOCAL VARIABLES/CONSTANTS
```

```
CONST NOT_CONFIG = 1
CONST AHU = 2
CONST RTU = 3
CONST FCU = 4
CONST HP = 5
CONST PIPE2 = 1
CONST PIPE4 = 2
CONST NO_AUX_HT = 1
CONST AUX_HT_W_OAT_LO = 2
CONST AUX_HT_WO_OAT_LO = 3
CONST ONE_FAN_SPEED = 1
CONST TWO_FAN_SPEEDS = 2
CONST THREE_FAN_SPEEDS = 3
CONST AUTO_FAN = 0
CONST ON_FAN = 1
CONST FAN_SPEED_AUTO = 1
CONST FAN_SPEED_1 = 2
CONST FAN_SPEED_2 = 3
CONST FAN_SPEED_3 = 4
LOCALS FAN_SHUTOFF_DELAY
FAN_SHUTOFF_DELAY = AV13
LOCALS OCC_MODE
OCC_MODE = BV1
LOCALS AUTO_FAN_NEED
AUTO_FAN_NEED = BV6
LOCALS HTG_CLG_MODE
HTG_CLG_MODE = BV7
LOCALS WATER_EVAL_NEED
WATER_EVAL_NEED = BV9
LOCALS OCC_FAN_MODE
OCC_FAN_MODE = BV12
LOCALS UNOCC_FAN_MODE
UNOCC_FAN_MODE = BV13
LOCALS DEHUM_MODE
DEHUM_MODE = BV21
LOCALS APP_MAIN_TYPE
APP_MAIN_TYPE = MSV1
LOCALS APP_SUB_TYPE
APP_SUB_TYPE = MSV2
LOCALS COMP_STAGES
COMP_STAGES = MSV2
LOCALS FAN_CONTROL_OPTION
FAN_CONTROL_OPTION = MSV3
LOCALS AUX_HEAT_OPTION
AUX_HEAT_OPTION = MSV6
LOCALS FAN_SPEED_OUTPUT
FAN_SPEED_OUTPUT = MSV8
LOCALS STAGES
LOCALS FAN_SPEED
IF DEHUM_MODE THEN
REM FAN SPEED CALCULATES ON UPPER 40% OF DEHUM HTG LOOP IF DEHUMIDIFYING
FAN_SPEED = MAX(0 , MIN(100 , ((LOOP4 - 60) * 2.5)))
ELSE
REM OTHERWISE FAN_SPEED CALCULATES BASED ON UPPER 40% OF APPROPRIATE SPACE TEMP LOOP
IF HTG_CLG_MODE THEN
FAN_SPEED = MAX(0 , MIN(100 , ((LOOP1 - 60) * 2.5)))
ELSE
FAN_SPEED = MAX(0 , MIN(100 , ((LOOP2 - 60) * 2.5)))
ENDIF
ENDIF
LOCALS F1
LOCALS F2
LOCALS F3

REM PERFORM APPROPRIATE SECTION OF CODE BASED ON FLEXSTAT CONFIGURATION
IF APP_MAIN_TYPE = AHU THEN
GOTO AHU_CODE
```

```
ELSE IF APP_MAIN_TYPE = RTU THEN
GOTO RTU_CODE
ELSE IF APP_MAIN_TYPE = FCU THEN
GOTO FCU_CODE
ELSE IF APP_MAIN_TYPE = HP THEN
GOTO HP_CODE
ENDIF

END

AHU_CODE:
REM THIS CODE SECTION IS FOR THE AHU APP WITH MODULATING WATER VALVES

REM FAN CYCLES IN UNOCC MODE, RUNS FOR DELAY PERIOD AFTER CLG,HTG OFF
IF OCC_MODE AND OCC_FAN_MODE = AUTO_FAN OR NOT OCC_MODE AND UNOCC_FAN_MODE = AUTO_FAN THEN
IF AUTO_FAN_NEED THEN
START BO1
ELSE
STOP BO1
ENDIF
ENDIF

REM FAN IS CONSTANT IF CURRENT FAN MODE IS SET TO 'ON'
IF OCC_MODE AND OCC_FAN_MODE = ON_FAN OR NOT OCC_MODE AND UNOCC_FAN_MODE = ON_FAN THEN START
BO1

END

RTU_CODE:

REM THESE CONSTANTS ONLY APPLY IN THIS APP
STAGES = BO2 OR BO3 OR BO4 OR BO5

REM FAN CYCLES IN UNOCC MODE, RUNS FOR DELAY PERIOD AFTER CLG,HTG OFF
IF OCC_MODE AND OCC_FAN_MODE = AUTO_FAN OR NOT OCC_MODE AND UNOCC_FAN_MODE = AUTO_FAN THEN
IF AUTO_FAN_NEED THEN START BO1
IF NOT AUTO_FAN_NEED AND TIMEOFF(STAGES) > FAN_SHUTOFF_DELAY THEN STOP BO1
ENDIF

REM FAN IS CONSTANT IF CURRENT FAN MODE IS SET TO 'ON'
IF OCC_MODE AND OCC_FAN_MODE = ON_FAN OR NOT OCC_MODE AND UNOCC_FAN_MODE = ON_FAN THEN START
BO1

END

FCU_CODE:

REM IF FCU IS 2-PIPE AND UNIT IS IN WATER EVAL MODE, THEN STOP FAN & FAN ICON
IF APP_SUB_TYPE = PIPE2 THEN
IF WATER_EVAL_NEED THEN
STOP BO1
STOP BO2
STOP BO3
STOP BV18
END
ENDIF
ENDIF

REM FAN IS CONSTANT IN OCC OR OVRD MODES, CYCLES IN UNOCC

REM DETERMINE HOW MANY FAN SPEEDS
IF FAN_CONTROL_OPTION = ONE_FAN_SPEED THEN GOTO 1_FAN_SPEED
IF FAN_CONTROL_OPTION = TWO_FAN_SPEEDS THEN GOTO 2_FAN_SPEEDS
IF FAN_CONTROL_OPTION = THREE_FAN_SPEEDS THEN GOTO 3_FAN_SPEEDS

1_FAN_SPEED:
REM FAN IS ALWAYS ON IF CURRENT FAN MODE IS SET TO 'ON' OR CYCLES WITH SPACE TEMP DEMAND
```

```
REM IF CURRENT FAN MODE IS SET TO 'AUTO'
BO1 = (OCC_MODE AND OCC_FAN_MODE = ON_FAN OR NOT OCC_MODE AND UNOCC_FAN_MODE = ON_FAN) OR
AUTO_FAN_NEED

END

2_FAN_SPEEDS:
REM FAN IS ALWAYS ON IF CURRENT FAN MODE IS SET TO 'ON'
REM SPEED IS EITHER AUTO CONTROLLED OR SET BY FAN SPEED SELECTION
IF OCC_MODE AND OCC_FAN_MODE = ON_FAN OR NOT OCC_MODE AND UNOCC_FAN_MODE = ON_FAN THEN
IF FAN_SPEED_OUTPUT = FAN_SPEED_AUTO THEN
START BO1
IF FAN_SPEED > 50 THEN START BO2
IF FAN_SPEED < 10 THEN STOP BO2
IF BO2 THEN STOP BO1
ELSE IF FAN_SPEED_OUTPUT = FAN_SPEED_1 THEN
START BO1
STOP BO2
ELSE IF FAN_SPEED_OUTPUT = FAN_SPEED_2 THEN
STOP BO1
START BO2
ENDIF
ENDIF

REM FAN CYCLES WITH DEMAND OF SPACE TEMPERATURE AT EITHER THE PRESELECTED SPEED OR
REM THE HIGHEST AVAILABLE SPEED IF SPEED SELECTION IS SET TO 'AUTO'
IF OCC_MODE AND OCC_FAN_MODE = AUTO_FAN OR NOT OCC_MODE AND UNOCC_FAN_MODE = AUTO_FAN THEN
IF FAN_SPEED_OUTPUT = FAN_SPEED_AUTO THEN
STOP BO1
BO2 = AUTO_FAN_NEED
ELSE IF FAN_SPEED_OUTPUT = FAN_SPEED_1 THEN
BO1 = AUTO_FAN_NEED
STOP BO2
ELSE IF FAN_SPEED_OUTPUT = FAN_SPEED_2 THEN
STOP BO1
BO2 = AUTO_FAN_NEED
ENDIF
ENDIF

END

3_FAN_SPEEDS:
REM FAN IS ALWAYS ON IF CURRENT FAN MODE IS SET TO 'ON'
REM SPEED IS EITHER AUTO CONTROLLED OR SET BY FAN SPEED SELECTION
IF OCC_MODE AND OCC_FAN_MODE = ON_FAN OR NOT OCC_MODE AND UNOCC_FAN_MODE = ON_FAN THEN
IF FAN_SPEED_OUTPUT = FAN_SPEED_AUTO THEN
START F1
IF FAN_SPEED > 35 THEN START F2
IF FAN_SPEED > 70 THEN START F3
IF FAN_SPEED < 35 THEN STOP F3
IF FAN_SPEED < 1 THEN STOP F2
IF F3 THEN START BO3 , STOP BO1 , STOP BO2 , GOTO END_SPEED
IF F2 THEN START BO2 , STOP BO1 , STOP BO3 , GOTO END_SPEED
IF F1 THEN START BO1 , STOP BO2 , STOP BO3
END_SPEED:
ELSE IF FAN_SPEED_OUTPUT = FAN_SPEED_1 THEN
START BO1
STOP BO2
STOP BO3
ELSE IF FAN_SPEED_OUTPUT = FAN_SPEED_2 THEN
STOP BO1
START BO2
STOP BO3
ELSE IF FAN_SPEED_OUTPUT = FAN_SPEED_3 THEN
STOP BO1
STOP BO2
START BO3
```

```
ENDIF
ENDIF

REM FAN CYCLES WITH DEMAND OF SPACE TEMPERATURE AT EITHER THE PRESELECTED SPEED OR
REM THE HIGHEST AVAILABLE SPEED IF SPEED SELECTION IS SET TO 'AUTO'
IF OCC_MODE AND OCC_FAN_MODE = AUTO_FAN OR NOT OCC_MODE AND UNOCC_FAN_MODE = AUTO_FAN THEN
IF FAN_SPEED_OUTPUT = FAN_SPEED_AUTO THEN
STOP BO1
STOP BO2
BO3 = AUTO_FAN_NEED
ELSE IF FAN_SPEED_OUTPUT = FAN_SPEED_1 THEN
BO1 = AUTO_FAN_NEED
STOP BO2
STOP BO3
ELSE IF FAN_SPEED_OUTPUT = FAN_SPEED_2 THEN
STOP BO1
BO2 = AUTO_FAN_NEED
STOP BO3
ELSE IF FAN_SPEED_OUTPUT = FAN_SPEED_3 THEN
STOP BO1
STOP BO2
BO3 = AUTO_FAN_NEED
ENDIF
ENDIF

END

HP_CODE:

REM THESE CONSTANTS ONLY APPLY IN THIS APP
REM STAGES IS ACTIVE BASED ON THE PRESENT # OF COMPRESSORS AND WHETHER AUX HT IS PRESENT
IF COMP_STAGES = 1 THEN
STAGES = BO3
ELSE IF COMP_STAGES = 2 THEN
STAGES = BO3 OR BO4
ENDIF
IF AUX_HEAT_OPTION = AUX_HT_W_OAT_LO OR AUX_HEAT_OPTION = AUX_HT_WO_OAT_LO THEN
STAGES = STAGES OR BO5 OR BO6
ENDIF

REM FAN CYCLES IN UNOCC MODE, RUNS FOR DELAY PERIOD AFTER CLG,HTG OFF
IF OCC_MODE AND OCC_FAN_MODE = AUTO_FAN OR NOT OCC_MODE AND UNOCC_FAN_MODE = AUTO_FAN THEN
IF AUTO_FAN_NEED THEN START BO1
IF NOT AUTO_FAN_NEED AND TIMEOFF(STAGES) > FAN_SHUTOFF_DELAY THEN STOP BO1
ENDIF

REM FAN IS CONSTANT IF CURRENT FAN MODE IS SET TO 'ON'
IF OCC_MODE AND OCC_FAN_MODE = ON_FAN OR NOT OCC_MODE AND UNOCC_FAN_MODE = ON_FAN THEN START
BO1

END
```

Program 3: Valve and Staging Control

```
REM (C) 2009 KMC CONTROLS, INC.
REM THE CONTENTS OF THIS PROGRAM IS THE PROPERTY OF KMC CONTROLS, INC.
REM WHILE EVERY ATTEMPT HAS BEEN MADE TO ENSURE PROPER OPERATION OF THE
REM INTENDED SEQUENCES OF OPERATION, FINAL RESPONSIBILITY FOR VERIFICATION OF
REM PROPER OPERATION RESTS WITH THE SYSTEM INSTALLER/COMMISSIONER.
REM
REM  $Workfile: 6R3U-P3.NG $
REM  $Revision: 8 $
REM  $Date: 10/28/09 3:05p $
REM  $Modtime: 10/28/09 2:37p $
```

```
REM
REM CODE IS BASED ON POINT DEFS FILE R0.0.0.22
REM CODE IS DESIGNED TO HAVE AT LEAST FW E1.1.0.5
REM PROGRAM 3: VALVE AND STAGING CONTROL

REM EXECUTE NO CODE IF IN TEST MODE
IF MSV1 = 1 THEN END

REM DEFINE PROGRAM LOCAL VARIABLES/CONSTANTS
CONST NOT_CONFIG = 1
CONST AHU = 2
CONST RTU = 3
CONST FCU = 4
CONST HP = 5
CONST PIPE2 = 1
CONST PIPE4 = 2
CONST NO_ECON = 1
CONST MOD_ECON = 2
CONST ECON_EN_DIS = 3
CONST NO_AUX_HEAT = 1
CONST ONE_HP_COMP = 1
CONST TWO_HP_COMP = 2
CONST MODULATING = 1
CONST TWO_POS = 0
CONST ZERO_DEADBAND = 1
CONST HALF_DEADBAND = 50
CONST FULL_DEADBAND = 99
CONST AUTO = 1
CONST HTG = 2
CONST CLG = 3
CONST OFF = 4
CONST EMER_HT = 5
CONST AUTO_FAN = 0
CONST ON_FAN = 1
CONST HT = 0
CONST CL = 1
LOCALS ALLOW_CLG
START ALLOW_CLG : REM INITIALIZE TO ALLOW CLG
LOCALS SPACE_TEMP
SPACE_TEMP = AI1
LOCALS FAN
FAN = BO1
LOCALS ACT_CLG_STPT
ACT_CLG_STPT = AV2
LOCALS ACT_HTG_STPT
ACT_HTG_STPT = AV3
LOCALS CLG_STBK_STPT
CLG_STBK_STPT = AV4
LOCALS HTG_STBK_STPT
HTG_STBK_STPT = AV5
LOCALS DEADBAND
DEADBAND = AV6
LOCALS OCC_MODE
OCC_MODE = BV1
LOCALS ECON_MODE
ECON_MODE = BV3
LOCALS FAN_ALARM
FAN_ALARM = BV5
LOCALS HTG_CLG_MODE
HTG_CLG_MODE = BV7
LOCALS OCC_FAN_MODE
OCC_FAN_MODE = BV12
LOCALS UNOCC_FAN_MODE
UNOCC_FAN_MODE = BV13
LOCALS REV_VALVE_OPER
REV_VALVE_OPER = BV15
LOCALS VALVE_TYPE
```

```
VALVE_TYPE = BV19
LOCALS DEHUM_MODE
DEHUM_MODE = BV21
LOCALS ALLOW_HTG_DEHUM
ALLOW_HTG_DEHUM = BV25
LOCALS APP_MAIN_TYPE
APP_MAIN_TYPE = MSV1
LOCALS APP_SUB_TYPE
APP_SUB_TYPE = MSV2
LOCALS OAD_OPTION
OAD_OPTION = MSV4
LOCALS AUX_HEAT_OPTION
AUX_HEAT_OPTION = MSV6
LOCALS SYSTEM_MODE
SYSTEM_MODE = MSV7
LOCALS CLG_LOOP
CLG_LOOP = LOOP1
LOCALS HTG_LOOP
HTG_LOOP = LOOP2
LOCALS DEHUM_HTG_LOOP
DEHUM_HTG_LOOP = LOOP4
REM CONDITIONAL LOCALS STATEMENTS
IF APP_MAIN_TYPE = RTU THEN
LOCALS CL1
CL1 = BO2
LOCALS CL2
CL2 = BO3
LOCALS HT1
HT1 = BO4
LOCALS HT2
HT2 = BO5
ENDIF
IF APP_MAIN_TYPE = FCU AND APP_SUB_TYPE = PIPE2 THEN
LOCALS CLG_VALVE_NEED
CLG_VALVE_NEED = AV29
LOCALS HTG_VALVE_NEED
HTG_VALVE_NEED = AV30
LOCALS WATER_EVAL_NEED
WATER_EVAL_NEED = BV9
LOCALS CLG_AVLBL
CLG_AVLBL = BV10
LOCALS HTG_AVLBL
HTG_AVLBL = BV11
ENDIF
IF APP_MAIN_TYPE = HP THEN
LOCALS AUX_HEAT_NEED
LOCALS AUX_HEAT_DELAY
AUX_HEAT_DELAY = AV14 * 60
LOCALS COMP_LOCKOUT
COMP_LOCKOUT = BV8
LOCALS COMP1
COMP1 = BO3
LOCALS COMP2
COMP2 = BO4
LOCALS AUX_HT
AUX_HT = BO5
LOCALS EMER_HT
EMER_HT = BO6
REM CMP2 IS A FLAG FOR WHETHER COMPRESSOR 2 IS PRESENT
LOCALS CMP2
IF APP_SUB_TYPE = 1 THEN STOP CMP2
IF APP_SUB_TYPE = 2 THEN START CMP2
ENDIF
IF APP_MAIN_TYPE = RTU OR APP_MAIN_TYPE = HP THEN
LOCALS STAGE_DELAY
STAGE_DELAY = AV12 * 60
ENDIF
```

```
REM DISALLOW MECHANICAL CLG IF MOD ECON OPTION IS SELECTED, ECON MODE IS ACTIVE
REM AND DAMPER IS < 100%
IF OAD_OPTION = NO_ECON THEN
ENABLE ALLOW_CLG
ELSE IF OAD_OPTION = MOD_ECON THEN
IF ECON_MODE THEN
IF AO9 > 99 THEN
START ALLOW_CLG
ELSE
STOP ALLOW_CLG
ENDIF
ELSE
START ALLOW_CLG
ENDIF
ELSE IF OAD_OPTION = ECON_EN_DIS THEN
ENABLE ALLOW_CLG
ENDIF

REM PERFORM APPROPRIATE SECTION OF CODE BASED ON FLEXSTAT CONFIGURATION
IF APP_MAIN_TYPE = AHU THEN
GOTO AHU_CODE
ELSE IF APP_MAIN_TYPE = RTU THEN
GOTO RTU_CODE
ELSE IF APP_MAIN_TYPE = FCU THEN
IF APP_SUB_TYPE = PIPE2 THEN GOTO FCU_2_PIPE_CODE
IF APP_SUB_TYPE = PIPE4 THEN GOTO FCU_4_PIPE_CODE
ELSE IF APP_MAIN_TYPE = HP THEN
GOTO HP_CODE
ENDIF

END

AHU_CODE:
REM THIS CODE SECTION IS FOR THE AHU APP WITH MODULATING WATER VALVES

REM GOTO CORRECT VALVE OPERATING MODE
IF OCC_MODE THEN
IF OCC_FAN_MODE = ON_FAN THEN GOTO AHU_MOD_VALVES
IF OCC_FAN_MODE = AUTO_FAN THEN GOTO AHU_CYCLE_VALVES
ENDIF
IF NOT OCC_MODE THEN
IF UNOCC_FAN_MODE = ON_FAN THEN GOTO AHU_MOD_VALVES
IF UNOCC_FAN_MODE = AUTO_FAN THEN GOTO AHU_CYCLE_VALVES
ENDIF

END

AHU_MOD_VALVES:
REM MODULATES VALVES BASED ON DEMAND

AHU_MOD_CLG_VALVE:
AO7 = MAX( 0, MIN( 100, CLG_LOOP ) )
REM OPEN CLG VALVE 100% IF IN DEHUM MODE, HTG VALVE WILL MAINTAIN TEMP
IF DEHUM_MODE THEN AO7 = 100
REM CLOSE CLG VALVE IF SYSTEM MODE DOESN'T ALLOW IT
IF SYSTEM_MODE = OFF THEN AO7 = 0
IF SYSTEM_MODE = HTG AND NOT DEHUM_MODE THEN AO7 = 0

AHU_MOD_HTG_VALVE:
IF DEHUM_MODE THEN
REM MODULATE HTG VALVE TO MAINTAIN CLG SETPT
AO8 = MAX( 0, MIN( 100, DEHUM_HTG_LOOP ) )
ELSE
REM MODULATE HTG VALVE TO MAINTAIN HTG SETPT
AO8 = MAX( 0, MIN( 100, HTG_LOOP ) )
ENDIF
```

```
REM CLOSE HTG VALVE IF SYSTEM MODE DOESN'T ALLOW IT
IF SYSTEM_MODE = OFF THEN AO8 = 0
IF SYSTEM_MODE = CLG AND NOT DEHUM_MODE THEN AO8 = 0

END

AHU_CYCLE_VALVES:
REM OPENS APPROPRIATE VALVE 100% ON CALL FOR CLG OR HTG

AHU_CLG_VALVE_CYCLE:
IF CLG_LOOP > FULL_DEADBAND THEN AO7 = 100
IF CLG_LOOP < ZERO_DEADBAND THEN AO7 = 0
REM OPEN CLG VALVE 100% IF IN DEHUM MODE, HTG VALVE WILL MAINTAIN TEMP
IF DEHUM_MODE THEN AO7 = 100
REM CLOSE CLG VALVE IF SYSTEM MODE DOESN'T ALLOW IT
IF SYSTEM_MODE = OFF THEN AO7 = 0
IF SYSTEM_MODE = HTG AND NOT DEHUM_MODE THEN AO7 = 0

AHU_HTG_VALVE_CYCLE:
IF DEHUM_MODE THEN
REM CYCLE HTG VALVE TO MAINTAIN CLG SETPT
IF DEHUM_HTG_LOOP > FULL_DEADBAND THEN AO8 = 100
IF DEHUM_HTG_LOOP < ZERO_DEADBAND THEN AO8 = 0
ELSE
REM CYCLE HTG VALVE TO MAINTAIN HTG SETPT
IF HTG_LOOP > FULL_DEADBAND THEN AO8 = 100
IF HTG_LOOP < ZERO_DEADBAND THEN AO8 = 0
ENDIF
REM CLOSE HTG VALVE IF SYSTEM MODE DOESN'T ALLOW IT
IF SYSTEM_MODE = OFF THEN AO8 = 0
IF SYSTEM_MODE = CLG AND NOT DEHUM_MODE THEN AO8 = 0

END

RTU_CODE:
REM STAGING CONTROL

REM IF SYSTEM MODE IS OFF, THEN DON'T HEAT OR COOL
IF SYSTEM_MODE = OFF THEN
GOSUB STOP_STAGES
END
ENDIF

REM SKIP COOLING IF SYSTEM MODE IS HEATING
IF SYSTEM_MODE = HTG AND NOT DEHUM_MODE THEN
STOP BO2
STOP BO3
GOSUB TWO_STAGES_HEATING
END
ENDIF

REM ALLOW CLG IF ECON STATE ALLOWS IT (SEE ALLOW_CLG CODE, ABOVE)
IF ALLOW_CLG THEN
GOSUB TWO_STAGES_COOLING
ELSE
STOP BO2
STOP BO3
ENDIF

REM CALL FOR 100% CLG IF IN DEHUM MODE, HTG WILL STAGE TO MAINTAIN TEMP
IF DEHUM_MODE THEN
START BO2
START BO3
ENDIF

REM SKIP HEATING IF SYSTEM MODE IS COOLING
IF SYSTEM_MODE = CLG AND NOT DEHUM_MODE THEN
```

```
STOP BO4
STOP BO5
END
ENDIF

GOSUB TWO_STAGES_HEATING

END

FCU_2_PIPE_CODE:

REM VALVE CONTROL

REM VALVE CAN OPERATE AS EITHER A 2-POS OR MOD VALVE
IF VALVE_TYPE = MODULATING THEN
GOTO PIPE_2_MOD_VALVE
ELSE IF VALVE_TYPE = TWO_POS THEN
GOTO PIPE_2_TWO_POS_VALVE
ENDIF

PIPE_2_TWO_POS_VALVE:
REM IF WATER EVAL NEED MODE IS ACTIVE, OPEN VALVE
IF WATER_EVAL_NEED THEN
START BO4
END
ENDIF

REM IF THERE IS EITHER A CLG OR HTG VALVE NEED, OPEN VALVE IF THAT WATER IS AVAILABLE
IF CLG_VALVE_NEED * CLG_AVLBL OR HTG_VALVE_NEED * HTG_AVLBL THEN
START BO4
ELSE
STOP BO4
ENDIF


PIPE_2_MOD_VALVE:
REM IF WATER EVAL NEED MODE IS ACTIVE, OPEN VALVE 100%
IF WATER_EVAL_NEED THEN
AO7 = 100
END
ENDIF

REM IF THERE IS EITHER A CLG OR HTG VALVE NEED, OPEN VALVE IF THAT WATER IS AVAILABLE
AO7 = (CLG_VALVE_NEED * CLG_AVLBL) + (HTG_VALVE_NEED * HTG_AVLBL)

END

FCU_4_PIPE_CODE:

REM VALVE CONTROL

REM JUMP TO CODE SECTION FOR EITHER 2-POS. OR MOD VALVES
IF VALVE_TYPE = TWO_POS THEN
GOTO PIPE_4_TWO_POS_VALVES
ELSE IF VALVE_TYPE = MODULATING THEN
GOTO PIPE_4_MOD_VALVES_CODE
ENDIF

END

PIPE_4_TWO_POS_VALVES:

PIPE4_2_POS_CLG_VALVE:
IF CLG_LOOP > FULL_DEADBAND THEN START BO4
IF CLG_LOOP < ZERO_DEADBAND THEN STOP BO4
REM OPEN CLG VALVE IF IN DEHUM MODE, HTG VALVE WILL MAINTAIN TEMP
IF DEHUM_MODE THEN START BO4
```

```
REM CLOSE CLG VALVE IF SYSTEM MODE DOESN'T ALLOW IT
IF SYSTEM_MODE = OFF THEN STOP BO4
IF SYSTEM_MODE = HTG AND NOT DEHUM_MODE THEN STOP BO4

PIPE4_2_POS_HTG_VALVE:
IF DEHUM_MODE THEN
REM CYCLE HTG VALVE TO MAINTAIN CLG SETPT
IF DEHUM_HTG_LOOP > FULL_DEADBAND THEN START BO5
IF DEHUM_HTG_LOOP < ZERO_DEADBAND THEN STOP BO5
ELSE
REM CYCLE HTG VALVE TO MAINTAIN HTG SETPT
IF HTG_LOOP > FULL_DEADBAND THEN START BO5
IF HTG_LOOP < ZERO_DEADBAND THEN STOP BO5
ENDIF
REM CLOSE HTG VALVE IF SYSTEM MODE DOESN'T ALLOW IT
IF SYSTEM_MODE = OFF THEN STOP BO5
IF SYSTEM_MODE = CLG AND NOT DEHUM_MODE THEN STOP BO5

END

PIPE_4_MOD_VALVES_CODE:
REM GOTO CORRECT MOD VALVE OPERATING MODE
IF OCC_MODE THEN
IF OCC_FAN_MODE = ON_FAN THEN GOTO PIPE4_MODULATING_VALVES
IF OCC_FAN_MODE = AUTO_FAN THEN GOTO PIPE4_CYCLE_MOD_VALVES
ENDIF
IF NOT OCC_MODE THEN
IF UNOCC_FAN_MODE = ON_FAN THEN GOTO PIPE4_MODULATING_VALVES
IF UNOCC_FAN_MODE = AUTO_FAN THEN GOTO PIPE4_CYCLE_MOD_VALVES
ENDIF

END

PIPE4_MODULATING_VALVES:
REM MODULATES VALVES BASED ON DEMAND

PIPE4_MOD_CLG_VALVE:
AO7 = MAX( 0, MIN( 100, CLG_LOOP * 1.6667 ) )
REM OPEN CLG VALVE 100% IF IN DEHUM MODE, HTG VALVE WILL MAINTAIN TEMP
IF DEHUM_MODE THEN AO7 = 100
REM CLOSE CLG VALVE IF SYSTEM MODE DOESN'T ALLOW IT
IF SYSTEM_MODE = OFF THEN AO7 = 0
IF SYSTEM_MODE = HTG AND NOT DEHUM_MODE THEN AO7 = 0

PIPE4_MOD_HTG_VALVE:
IF DEHUM_MODE THEN
REM MODULATE HTG VALVE TO MAINTAIN CLG SETPT
AO8 = MAX( 0, MIN( 100, DEHUM_HTG_LOOP * 1.6667 ) )
ELSE
REM MODULATE HTG VALVE TO MAINTAIN HTG SETPT
AO8 = MAX( 0, MIN( 100, HTG_LOOP * 1.6667 ) )
ENDIF
REM CLOSE HTG VALVE IF SYSTEM MODE DOESN'T ALLOW IT
IF SYSTEM_MODE = OFF THEN AO8 = 0
IF SYSTEM_MODE = CLG AND NOT DEHUM_MODE THEN AO8 = 0

END

PIPE4_CYCLE_MOD_VALVES:
REM OPENS APPROPRIATE VALVE 100% ON CALL FOR CLG OR HTG

PIPE4_MOD_CLG_VALVE_CYCLE:
IF CLG_LOOP > FULL_DEADBAND THEN AO7 = 100
IF CLG_LOOP < ZERO_DEADBAND THEN AO7 = 0
REM OPEN CLG VALVE 100% IF IN DEHUM MODE, HTG VALVE WILL MAINTAIN TEMP
IF DEHUM_MODE THEN AO7 = 100
REM CLOSE CLG VALVE IF SYSTEM MODE DOESN'T ALLOW IT
```

```
IF SYSTEM_MODE = OFF THEN AO7 = 0
IF SYSTEM_MODE = HTG AND NOT DEHUM_MODE THEN AO7 = 0

PIPE4_MOD_HTG_VALVE_CYCLE:
IF DEHUM_MODE THEN
REM CYCLE HTG VALVE TO MAINTAIN CLG SETPT
IF DEHUM_HTG_LOOP > FULL_DEADBAND THEN AO8 = 100
IF DEHUM_HTG_LOOP < ZERO_DEADBAND THEN AO8 = 0
ELSE
REM CYCLE HTG VALVE TO MAINTAIN HTG SETPT
IF HTG_LOOP > FULL_DEADBAND THEN AO8 = 100
IF HTG_LOOP < ZERO_DEADBAND THEN AO8 = 0
ENDIF
REM CLOSE HTG VALVE IF SYSTEM MODE DOESN'T ALLOW IT
IF SYSTEM_MODE = OFF THEN AO8 = 0
IF SYSTEM_MODE = CLG AND NOT DEHUM_MODE THEN AO8 = 0

END

HP_CODE:

REM STAGING CONTROL

REM IF SYSTEM MODE IS OFF, THEN DON'T HEAT OR COOL
IF SYSTEM_MODE = OFF THEN
STOP BO2
STOP BO3
STOP BO4
STOP BO5
STOP BO6
END
ENDIF

REM IF SYSTEM MODE IS SET TO EMER HEAT, THEN SKIP NORMAL CONTROL
IF SYSTEM_MODE = EMER_HT THEN GOTO HP_EMER_HT

REM SKIP COOLING IF SYSTEM MODE IS HEATING
IF NOT HTG_CLG_MODE OR SYSTEM_MODE = HTG THEN GOTO HP_HEATING

HP_COOLING:
REM CLG CODE EXECUTED DEPENDS ON HOW MANY STAGES OF COMPRESSOR
REM # OF COMPRESSORS SPANS THE DEADBAND

REM ONLY ALLOW CLG IF ECON STATE ALLOWS IT (SEE ALLOW_CLG CODE, ABOVE)
IF NOT ALLOW_CLG THEN
IF APP_SUB_TYPE = ONE_HP_COMP THEN STOP BO3
IF APP_SUB_TYPE = TWO_HP_COMP THEN STOP BO3 , STOP BO4
GOTO HP_DEHUM
ENDIF

IF APP_SUB_TYPE = ONE_HP_COMP THEN
IF CLG_LOOP > FULL_DEADBAND THEN START BO3
IF CLG_LOOP < ZERO_DEADBAND THEN STOP BO3
GOTO REVERSING_VALVE
ENDIF

IF APP_SUB_TYPE = TWO_HP_COMP THEN
IF CLG_LOOP > HALF_DEADBAND THEN START BO3
IF CMP2 AND CLG_LOOP > FULL_DEADBAND AND TIMEON(COMP1) > STAGE_DELAY THEN START BO4
REM IF CMP3 AND CLG_LOOP > STAGE_3_ON AND TIMEON(COMP2) > STAGE_DELAY THEN START BO5
REM IF CLG_LOOP < STAGE_3_OFF THEN STOP BO5
IF CLG_LOOP < HALF_DEADBAND THEN STOP BO4
IF CLG_LOOP < ZERO_DEADBAND THEN STOP BO3
GOTO REVERSING_VALVE
ENDIF

HP_HEATING:
```

```
REM HTG CODE EXECUTED DEPENDS ON HOW MANY STAGES OF COMPRESSOR
REM # OF COMPRESSORS SPANS THE DEADBAND

IF APP_SUB_TYPE = ONE_HP_COMP THEN
IF HTG_LOOP > FULL_DEADBAND THEN START BO3
IF HTG_LOOP < ZERO_DEADBAND THEN STOP BO3
ENDIF

IF APP_SUB_TYPE = TWO_HP_COMP THEN
IF HTG_LOOP > HALF_DEADBAND THEN START BO3
IF CMP2 AND HTG_LOOP > FULL_DEADBAND AND TIMEON(COMP1) > STAGE_DELAY THEN START BO4
REM IF CMP3 AND HTG_LOOP > STAGE_3_ON AND TIMEON(COMP2) > STAGE_DELAY THEN START BO5
REM IF HTG_LOOP < STAGE_3_OFF THEN STOP BO5
IF HTG_LOOP < HALF_DEADBAND THEN STOP BO4
IF HTG_LOOP < ZERO_DEADBAND THEN STOP BO3
ENDIF

REVERSING_VALVE:
REM START REV VALVE IF ANY COMPRESSOR IS ON BASED ON THE REV VALVE OPER (BV15)
IF REV_VALVE_OPER = HT THEN
IF HTG_CLG_MODE AND (BO3 OR BO4) THEN START BO2 ELSE STOP BO2
ELSE IF REV_VALVE_OPER = CL THEN
IF NOT HTG_CLG_MODE AND (BO3 OR BO4) THEN START BO2 ELSE STOP BO2
ENDIF

HP_DEHUM:
REM TURN ON ALL APPLICABLE STAGES OF COMPRESSOR AND REVERSING VALVE
REM AUX HEAT WILL CYCLE TO MAINTAIN TEMPERATURE
REM DO NOT ALLOW DEHUM IF THERE IS NO AUX HEAT
IF DEHUM_MODE AND AUX_HEAT_OPTION <> NO_AUX_HEAT THEN
BO2 = NOT REV_VALVE_OPER
START BO3
BO4 = CMP2
IF DEHUM_HTG_LOOP > HALF_DEADBAND THEN START BO5
IF DEHUM_HTG_LOOP > FULL_DEADBAND AND TIMEON(BO5) > STAGE_DELAY THEN START BO6
IF DEHUM_HTG_LOOP < HALF_DEADBAND THEN STOP BO6
IF DEHUM_HTG_LOOP < ZERO_DEADBAND THEN STOP BO5
ENDIF

AUX_HEAT:
IF NOT DEHUM_MODE THEN
IF AUX_HEAT_OPTION <> NO_AUX_HEAT THEN
REM IF COLD ENOUGH, START DEMAND FOR AUX HEAT
IF FAN AND SPACE_TEMP < ACT_HTG_STPT - (1.5 * DEADBAND) THEN START AUX_HEAT_NEED
REM IF NO COMP LOCKOUT, WAIT DELAY TO START AUX HEAT (1ST STAGE OF EMER HT)
IF NOT COMP_LOCKOUT AND TIMEON(AUX_HEAT_NEED) > AUX_HEAT_DELAY THEN START BO5
REM IF COMP LOCKOUT, START AUX HEAT IMMEDIATELY (1ST STAGE EMER HT)
IF COMP_LOCKOUT AND AUX_HEAT_NEED THEN START BO5
REM START 2ND STAGE OF EMER HT, IF NEEDED, AND AUX HT IS ALREADY ON
IF FAN AND SPACE_TEMP < ACT_HTG_STPT - (2 * DEADBAND) AND BO5 THEN START BO6
IF SPACE_TEMP > ACT_HTG_STPT - (1.5 * DEADBAND) THEN STOP BO6
REM IF SPACE WARMS ENOUGH, STOP AUX HEAT NO MATTER WHAT
IF SPACE_TEMP > ACT_HTG_STPT - DEADBAND THEN
STOP BO5
STOP AUX_HEAT_NEED
ENDIF
ELSE
STOP BO5
STOP BO6
ENDIF
ENDIF

COMP_OFF:
REM STOP COMPRESSORS & REV VALVE IF NO FAN OR COMP LOCKOUT IS ACTIVE
IF NOT FAN OR COMP_LOCKOUT THEN
STOP BO2
STOP BO3
```

```
STOP BO4
ENDIF

END

HP_EMER_HT:
REM EMER HT MODE CYCLES THE AUX HT AND EMER HT BASED ON SPACE TEMP
REM IT ALSO LOCKS OUT COMPRESSORS
STOP BO2
STOP BO3
STOP BO4
IF FAN THEN
IF HTG_LOOP > HALF_DEADBAND THEN START BO5
IF HTG_LOOP > FULL_DEADBAND AND TIMEON(HT1) > STAGE_DELAY THEN START BO6
IF HTG_LOOP < HALF_DEADBAND THEN STOP BO6
IF HTG_LOOP < ZERO_DEADBAND THEN STOP BO5
ENDIF

END

STOP_STAGES:
REM SYSTEM MODE IS SET TO 'OFF'
STOP BO2
STOP BO3
STOP BO4
STOP BO5
RETURN

TWO_STAGES_COOLING:
IF CLG_LOOP > HALF_DEADBAND THEN START BO2
IF CLG_LOOP > FULL_DEADBAND AND TIMEON(CL1) > STAGE_DELAY THEN START BO3
IF CLG_LOOP < HALF_DEADBAND THEN STOP BO3
IF CLG_LOOP < ZERO_DEADBAND THEN STOP BO2
RETURN

TWO_STAGES_HEATING:
IF DEHUM_MODE THEN
REM HEATING MODE WHILE IN DEHUM MODE
IF DEHUM_HTG_LOOP > HALF_DEADBAND THEN START BO4
IF DEHUM_HTG_LOOP > FULL_DEADBAND AND TIMEON(HT1) > STAGE_DELAY THEN START BO5
IF DEHUM_HTG_LOOP < HALF_DEADBAND THEN STOP BO5
IF DEHUM_HTG_LOOP < ZERO_DEADBAND THEN STOP BO4
ELSE
REM NORMAL HEATING MODE (WHEN NOT IN DEHUM MODE)
IF HTG_LOOP > HALF_DEADBAND THEN START BO4
IF HTG_LOOP > FULL_DEADBAND AND TIMEON(HT1) > STAGE_DELAY THEN START BO5
IF HTG_LOOP < HALF_DEADBAND THEN STOP BO5
IF HTG_LOOP < ZERO_DEADBAND THEN STOP BO4
ENDIF
RETURN
```

Program 4: Damper Control

```
REM (C) 2009 KMC CONTROLS, INC.
REM THE CONTENTS OF THIS PROGRAM IS THE PROPERTY OF KMC CONTROLS, INC.
REM WHILE EVERY ATTEMPT HAS BEEN MADE TO ENSURE PROPER OPERATION OF THE
REM INTENDED SEQUENCES OF OPERATION, FINAL RESPONSIBILITY FOR VERIFICATION OF
REM PROPER OPERATION RESTS WITH THE SYSTEM INSTALLER/COMMISSIONER.
REM
REM  $Workfile: 6R3U-P4.NG $
REM  $Revision: 4 $
REM  $Date: 4/08/09 1:44p $
REM  $Modtime: 4/08/09 11:00a $
REM
```

```
REM CODE IS BASED ON POINT DEFS FILE R0.0.0.16
REM CODE IS DESIGNED TO HAVE AT LEAST FW E0.0.0.19
REM PROGRAM 4: DAMPER CONTROL

REM EXECUTE NO CODE IF IN TEST MODE
IF MSV1 = 1 THEN END

REM DEFINE PROGRAM LOCAL VARIABLES/CONSTANTS
CONST NOT_CONFIG = 1
CONST AHU = 2
CONST RTU = 3
CONST HP = 5
CONST NO_ECON = 1
CONST MOD_ECON = 2
CONST ECON_EN_DIS = 3
CONST ZERO_DEADBAND = 1
CONST HALF_DEADBAND = 50
CONST FULL_DEADBAND = 99
CONST AUTO = 1
CONST HTG = 2
CONST CLG = 3
CONST OFF = 4
LOCALS UNOCC_ECON
LOCALS SPACE_TEMP
SPACE_TEMP = AI1
LOCALS FAN
FAN = BO1
LOCALS UNOCC_CLG_STPT
UNOCC_CLG_STPT = AV4
LOCALS DEADBAND
DEADBAND = AV6
LOCALS OCC_MODE
OCC_MODE = BV1
LOCALS ECON_MODE
ECON_MODE = BV3
LOCALS APP_MAIN_TYPE
APP_MAIN_TYPE = MSV1
LOCALS APP_SUB_TYPE
APP_SUB_TYPE = MSV2
LOCALS OAD_OPTION
OAD_OPTION = MSV4
LOCALS SYSTEM_MODE
SYSTEM_MODE = MSV7
LOCALS CLG_LOOP
CLG_LOOP = LOOP1
LOCALS HTG_LOOP
HTG_LOOP = LOOP2

REM PERFORM APPROPRIATE SECTION OF CODE BASED ON FLEXSTAT CONFIGURATION
IF APP_MAIN_TYPE <> AHU AND APP_MAIN_TYPE <> RTU AND APP_MAIN_TYPE <> HP THEN
END
ENDIF

IF OAD_OPTION = MOD_ECON THEN
GOTO MOD_ECON_CODE
ELSE IF OAD_OPTION = ECON_EN_DIS THEN
GOTO ECON_EN_DIS_CODE
ENDIF

END

MOD_ECON_CODE:

REM OA DAMPER CONTROL FOR A MODULATING DAMPER

IF OCC_MODE THEN GOTO OCC_MOD_ECON_MODE
IF NOT OCC_MODE THEN GOTO UNOCC_MOD_ECON_MODE
```

```
END

OCC_MOD_ECON_MODE:
REM MODULATES THE OA DAMPER BASED ON MAT & MAT SETPOINT
IF ECON_MODE THEN
AO9 = MAX(AV10 , LOOP3)
ELSE
AO9 = AV10
ENDIF
REM CLOSE OAD IF SYSTEM MODE DOESN'T ALLOW IT
IF SYSTEM_MODE = HTG THEN AO9 = AV10
IF SYSTEM_MODE = OFF THEN AO9 = 0

END

UNOCC_MOD_ECON_MODE:
REM ALLOWS DAMPER TO MODULATE BASED ON UNOCC CALL FOR CLG
IF CLG_LOOP > FULL_DEADBAND AND ECON_MODE THEN START UNOCC_ECON
IF CLG_LOOP < ZERO_DEADBAND OR NOT ECON_MODE THEN STOP UNOCC_ECON
AO9 = LOOP3 * UNOCC_ECON
REM CLOSE OAD IF SYSTEM MODE DOESN'T ALLOW IT
IF SYSTEM_MODE = HTG OR SYSTEM_MODE = OFF THEN AO9 = 0

END

ECON_EN_DIS_CODE:

REM CONTROL FOR A FACTORY-INTEGRATED ECONOMIZER
REM DAMPER POSITION IS CONTROLLED BY FACTORY CONTROLS INTEGRATED INTO UNIT

IF OCC_MODE THEN GOTO OCC_EN_DIS_ECON_MODE
IF NOT OCC_MODE THEN GOTO UNOCC_EN_DIS_ECON_MODE
END

OCC_EN_DIS_ECON_MODE:
REM MODULATES THE OA DAMPER BASED ON MAT & MAT SETPOINT
BO6 = ECON_MODE
REM CLOSE OAD IF SYSTEM MODE DOESN'T ALLOW IT
IF SYSTEM_MODE = HTG OR SYSTEM_MODE = OFF THEN DISABLE BO6

END

UNOCC_EN_DIS_ECON_MODE:
REM ALLOWS DAMPER TO MODULATE BASED ON UNOCC CALL FOR CLG
IF CLG_LOOP > FULL_DEADBAND AND ECON_MODE THEN ENABLE BO6
IF CLG_LOOP < ZERO_DEADBAND OR NOT ECON_MODE THEN DISABLE BO6

REM CLOSE OAD IF SYSTEM MODE DOESN'T ALLOW IT
IF SYSTEM_MODE = HTG OR SYSTEM_MODE = OFF THEN DISABLE BO6

END
```

Program 5: Safeties

```
REM (C) 2010 KMC CONTROLS, INC.
REM THE CONTENTS OF THIS PROGRAM IS THE PROPERTY OF KMC CONTROLS, INC.
REM WHILE EVERY ATTEMPT HAS BEEN MADE TO ENSURE PROPER OPERATION OF THE
REM INTENDED SEQUENCES OF OPERATION, FINAL RESPONSIBILITY FOR VERIFICATION OF
REM PROPER OPERATION RESTS WITH THE SYSTEM INSTALLER/COMMISSIONER.
REM
REM  $Workfile: 6R3U-P5.NG $
REM  $Revision: 4 $
REM  $Date: 3/12/10 2:56p $
REM  $Modtime: 3/12/10 2:45p $
```

```
REM
REM CODE IS BASED ON POINT DEFS FILE R0.0.0.24
REM CODE IS DESIGNED TO HAVE AT LEAST FW E1.3.0.4
REM PROGRAM 5: SAFETIES
REM PROPER OPERATION RESTS WITH THE SYSTEM INSTALLER/COMMISSIONER.

REM EXECUTE NO CODE IF IN TEST MODE
IF MSV1 = 1 THEN END

REM DEFINE PROGRAM LOCAL VARIABLES/CONSTANTS
CONST NOT_CONFIG = 1
CONST AHU = 2
CONST RTU = 3
CONST FCU = 4
CONST HP = 5
CONST NO_ECON = 1
CONST MOD_ECON = 2
CONST ECON_EN_DIS = 3
CONST MOD_HC = 1
CONST PIPE2 = 1
CONST PIPE4 = 2
CONST MODULATING = 1
CONST TWO_POS = 0
CONST ONE_HP_COMP = 1
CONST TWO_HP_COMP = 2
CONST NO_AUX_HEAT = 1
CONST ONE_FAN_SPEED = 1
CONST TWO_FAN_SPEEDS = 2
CONST THREE_FAN_SPEEDS = 3
CONST AUTO = 1
CONST HTG = 2
CONST CLG = 3
CONST OFF = 4
LOCALS FAN
FAN = BO1
LOCALS FAN2
FAN2 = BO2
LOCALS FAN3
FAN3 = BO3
LOCALS FAN_SHUTOFF_DELAY
FAN_SHUTOFF_DELAY = AV13
LOCALS LOW_ALARM
LOW_ALARM = BV4
LOCALS FAN_ALARM
FAN_ALARM = BV5
LOCALS WATER_EVAL_NEED
WATER_EVAL_NEED = BV9
LOCALS VALVE_TYPE
VALVE_TYPE = BV19
LOCALS APP_MAIN_TYPE
APP_MAIN_TYPE = MSV1
LOCALS APP_SUB_TYPE
APP_SUB_TYPE = MSV2
LOCALS FAN_CONTROL_OPTION
FAN_CONTROL_OPTION = MSV3
LOCALS OAD_OPTION
OAD_OPTION = MSV4
LOCALS AUX_HEAT_OPTION
AUX_HEAT_OPTION = MSV6
LOCALS SYSTEM_MODE
SYSTEM_MODE = MSV7
LOCALS STAGES
LOCALS CLG_VLV
LOCALS HTG_VLV
LOCALS OA_DAMPER
LOCALS OAD_ACTIVE
```

```
REM  CONDITIONAL DECLARATIONS
IF APP_MAIN_TYPE = AHU THEN
CLG_VLV = AO7
HTG_VLV = AO8
OA_DAMPER = AO9
ELSE IF APP_MAIN_TYPE = RTU THEN
LOCALS COOL1
COOL1 = BO2
LOCALS COOL2
COOL2 = BO3
LOCALS HEAT1
HEAT1 = BO4
LOCALS HEAT2
HEAT2 = BO5
ELSE IF APP_MAIN_TYPE = FCU THEN
IF APP_SUB_TYPE = PIPE2 THEN
LOCALS VALVE
IF VALVE_TYPE = TWO_POS THEN
VALVE = BO4
ELSE
IF VALVE_TYPE = MODULATING THEN
VALVE = AO7
ENDIF
ENDIF
ELSE IF APP_SUB_TYPE = PIPE4 THEN
IF VALVE_TYPE = TWO_POS THEN
CLG_VLV = BO4
HTG_VLV = BO5
ELSE
IF VALVE_TYPE = MODULATING THEN
CLG_VLV = AO7
HTG_VLV = AO8
ENDIF
ENDIF
ENDIF
ELSE IF APP_MAIN_TYPE = HP THEN
LOCALS COMP1
LOCALS AUX_HT
IF APP_SUB_TYPE = ONE_HP_COMP THEN
COMP1 = BO3
ELSE
IF APP_SUB_TYPE = TWO_HP_COMP THEN
COMP1 = BO3
LOCALS COMP2
COMP2 = BO4
ENDIF
ENDIF
IF AUX_HEAT_OPTION <> NO_AUX_HEAT THEN
AUX_HT = BO5 OR BO6
ENDIF
IF OAD_OPTION = MOD_ECON THEN OA_DAMPER = AO9
ENDIF

REM PERFORM APPROPRIATE SECTION OF CODE BASED ON FLEXSTAT CONFIGURATION
IF APP_MAIN_TYPE = AHU THEN
GOTO AHU_CODE
ELSE IF APP_MAIN_TYPE = RTU OR APP_MAIN_TYPE = HP THEN
GOTO RTU_HP_CODE
ELSE IF APP_MAIN_TYPE = FCU THEN
IF APP_SUB_TYPE = PIPE2 THEN GOTO FCU_2_PIPE_CODE
IF APP_SUB_TYPE = PIPE4 THEN GOTO FCU_4_PIPE_CODE
ENDIF

END

AHU_CODE:
```

```
REM CLOSE EVERYTHING IF FAN IS NOT RUNNING OR THERE IS A FAN ALARM
IF FAN_ALARM OR NOT FAN THEN
AO7 = 0
AO8 = 0
AO9 = 0
ENDIF

REM TURN OFF FAN & FAN ICON, CLOSE CLG VALVE & OA DAMPER & OPEN HTG VALVE IF LOW LIMIT
IF LOW_ALARM THEN
STOP BO1
STOP BV18
AO7 = 0
AO8 = 100
AO9 = 0
ENDIF

REM EVALUATE AND SET SYSTEM MODE (HTG/CLG/EMER) ICON AFTER ALL OUTPUTS ARE FINALIZED
GOSUB SYSTEM_MODE_ICON

END

RTU_HP_CODE:
REM STOP EVERYTHING IF FAN IS NOT RUNNING OR IF THERE IS A FAN ALARM
IF FAN_ALARM OR NOT FAN THEN
STOP BV18
STOP BO2
STOP BO3
STOP BO4
STOP BO5
STOP BO6
STOP BO2@5
STOP BO3@5
STOP BO4@5
STOP BO5@5
STOP BO6@5
AO9 = 0
ENDIF

REM TURN OFF FAN & FAN ICON, EVERYTHING ELSE, CLOSE OA DAMPER IF LOW LIMIT
REM IT'S DONE TWICE - ONCE FOR RESETTING CONTROL BASIC CONTROL OF THE OUTPUTS
REM AND ONCE TO OVERRIDE ANY MINIMUM ONS THAT MAY BE THERE
IF LOW_ALARM THEN
STOP BO1
STOP BV18
STOP BO2
STOP BO3
STOP BO4
STOP BO5
STOP BO6
STOP BO2@5
STOP BO3@5
STOP BO4@5
STOP BO5@5
STOP BO6@5
AO9 = 0
ENDIF

REM TURN OFF EVERYTHING BUT FAN, CLOSE OA DAMPER IF SYSTEM OFF IS SET
STAGES = BO2 OR BO3 OR BO4 OR BO5 OR BO6
IF SYSTEM_MODE = OFF THEN
IF TIMEOFF(STAGES) > FAN_SHUTOFF_DELAY THEN
STOP BO1
STOP BV18
ENDIF
STOP BO2
STOP BO3
STOP BO4
```

```
STOP BO5
STOP BO6
AO9 = 0
ENDIF

REM IF EVERYTHING IS CLEAR THEN STOP OVERRIDING THE MIN ON/OFF
IF NOT(FAN_ALARM OR LOW_ALARM) AND FAN THEN
RLQ BO2@5
RLQ BO3@5
RLQ BO4@5
RLQ BO5@5
RLQ BO6@5
ENDIF

REM EVALUATE AND SET SYSTEM MODE (HTG/CLG/EMER) ICON AFTER ALL OUTPUTS ARE FINALIZED
GOSUB SYSTEM_MODE_ICON

END

FCU_2_PIPE_CODE:

REM PERFORM SAFETY CHECKS AND OPERATIONS

REM IF THE UNIT IS IN WATER_EVAL_NEED MODE, THEN SKIP THE FOLLOWING CHECK
IF WATER_EVAL_NEED THEN GOTO C1_SYSTEM_OFF_CHECK

REM SAFETIES ARE BASED ON VALVE TYPE
IF VALVE_TYPE = TWO_POS THEN
GOTO PIPE_2_TWO_POS
ELSE IF VALVE_TYPE = MODULATING THEN
GOTO PIPE_2_MOD
ENDIF

PIPE_2_TWO_POS:
REM CLOSE EVERYTHING IF FAN IS NOT RUNNING OR IF THERE IS A FAN ALARM
IF FAN_CONTROL_OPTION =1 AND (NOT FAN OR FAN_ALARM) THEN STOP BO4
IF FAN_CONTROL_OPTION =2 AND (NOT (FAN OR FAN2) OR FAN_ALARM) THEN STOP BO4
IF FAN_CONTROL_OPTION =3 AND (NOT (FAN OR FAN2 OR FAN3) OR FAN_ALARM) THEN STOP BO4
GOTO C1_SYSTEM_OFF_CHECK

PIPE_2_MOD:
REM CLOSE EVERYTHING IF FAN IS NOT RUNNING OR IF THERE IS A FAN ALARM
IF FAN_CONTROL_OPTION =1 AND (NOT FAN OR FAN_ALARM) THEN AO7 = 0
IF FAN_CONTROL_OPTION =2 AND (NOT (FAN OR FAN2) OR FAN_ALARM) THEN AO7 = 0
IF FAN_CONTROL_OPTION =3 AND (NOT (FAN OR FAN2 OR FAN3) OR FAN_ALARM) THEN AO7 = 0

C1_SYSTEM_OFF_CHECK:
IF SYSTEM_MODE = OFF THEN
IF VALVE_TYPE = TWO_POS THEN
STOP BO4
ELSE IF VALVE_TYPE = MODULATING THEN
AO7 = 0
ENDIF
ENDIF

REM EVALUATE AND SET SYSTEM MODE (HTG/CLG/EMER) ICON AFTER ALL OUTPUTS ARE FINALIZED
GOSUB SYSTEM_MODE_ICON

END

FCU_4_PIPE_CODE:
REM CLOSE EVERYTHING IF FAN IS NOT RUNNING OR IF THERE IS A FAN ALARM
IF FAN_CONTROL_OPTION = ONE_FAN_SPEED AND (NOT FAN OR FAN_ALARM) THEN AO7 = 0 , AO8 = 0
IF FAN_CONTROL_OPTION = TWO_FAN_SPEEDS AND (NOT (FAN OR FAN2) OR FAN_ALARM) THEN AO7 = 0 ,
AO8 = 0
IF FAN_CONTROL_OPTION = THREE_FAN_SPEEDS AND (NOT (FAN OR FAN2 OR FAN3) OR FAN_ALARM) THEN
AO7 = 0 , AO8 = 0
```

```
IF SYSTEM_MODE = OFF THEN
AO7 = 0
AO8 = 0
ENDIF

REM EVALUATE AND SET SYSTEM MODE (HTG/CLG/EMER) ICON AFTER ALL OUTPUTS ARE FINALIZED
GOSUB SYSTEM_MODE_ICON

END

SYSTEM_MODE_ICON:
REM THIS SECTION CONTROLS WHETHER THE SYSTEM MODE (CLG/HTG/EMER HT) ICON IS ANIMATED OR NOT
VIA BV27
REM ICON IS ACTIVE WHEN ANY APPROPRIATE OUTPUTS ARE ACTIVE
IF APP_MAIN_TYPE = AHU THEN
IF CLG_VLV > 5 OR HTG_VLV > 5 OR OA_DAMPER > 5 THEN START BV27
IF CLG_VLV = 0 AND HTG_VLV = 0 AND OA_DAMPER = 0 THEN STOP BV27
ELSE IF APP_MAIN_TYPE = RTU THEN
IF OAD_OPTION = NO_ECON THEN
STOP OAD_ACTIVE
ELSE IF OAD_OPTION = MOD_ECON THEN
IF OA_DAMPER > 5 THEN START OAD_ACTIVE
IF OA_DAMPER = 0 THEN STOP OAD_ACTIVE
ELSE IF OAD_OPTION = ECON_EN_DIS THEN
OAD_ACTIVE = BO6
ENDIF
BV27 = COOL1 OR COOL2 OR HEAT1 OR HEAT2 OR OAD_ACTIVE
ELSE IF APP_MAIN_TYPE = FCU THEN
IF APP_SUB_TYPE = PIPE2 THEN
IF VALVE_TYPE = TWO_POS THEN
BV27 = VALVE
ELSE IF VALVE_TYPE = MODULATING THEN
IF VALVE > 5 THEN START BV27
IF VALVE = 0 THEN STOP BV27
ENDIF
ELSE IF APP_SUB_TYPE = PIPE4 THEN
IF VALVE_TYPE = TWO_POS THEN
BV27 = CLG_VLV OR HTG_VLV
ELSE IF VALVE_TYPE = MODULATING THEN
IF CLG_VLV > 5 OR HTG_VLV > 5 THEN START BV27
IF CLG_VLV = 0 AND HTG_VLV = 0 THEN STOP BV27
ENDIF
ENDIF
ELSE IF APP_MAIN_TYPE = HP THEN
IF OA_DAMPER > 5 THEN START OAD_ACTIVE
IF OA_DAMPER = 0 THEN STOP OAD_ACTIVE
IF APP_SUB_TYPE = ONE_HP_COMP THEN
BV27 = COMP1 OR AUX_HT OR OAD_ACTIVE
ELSE IF APP_SUB_TYPE = TWO_HP_COMP THEN
BV27 = COMP1 OR COMP2 OR AUX_HT OR OAD_ACTIVE
ENDIF
ENDIF
```

Important Notices

The material in this document is for information purposes only. The contents and the product it describes are subject to change without notice. KMC Controls makes no representations or warranties with respect to this manual. In no event shall KMC be liable for any damages, direct or incidental, arising out of or related to the use of this manual.

KMC Controls
P.O. Box 497
19476 Industrial Drive
New Paris, IN 46553
U.S.A.
TEL: 1.574.831.5250
FAX: 1.574.831.8108
EMAIL: info@kmccontrols.com