Changes to Control Basic in BACnet controllers

Applicable versions of software

This technical bulletin applies to the following applications:

- BACstage 2.3 and later
- TotalControl Design Studio 1.5 and later

New versions of Control Basic

With the release of BACstage 2.3 and TotalControl Design Studio 1.5, KMC Controls introduced several changes to Control Basic.

- Controllers with Standard Control Basic are programmed with little change. However, when existing programs are loaded from a controller you will see some changes to the keywords and references to remote points. See Table 2 on page 8 for a listing of controllers that support Standard Control Basic.
- Controllers with Next Generation Control Basic have several new keywords available. In addition there are other changes to the language. As of the date of this bulletin, only the BAC-A1616, the building controller, supported Next Generation Control Basic.

Review the following topics to become familiar with the new features of Control Basic:

In addition to the topics in this bulletin, each of the new features are documented in the help system installed with new versions of BACstage and TotalControl Design Studio. Review also those topics before programming with the new features.

Deprecated keywords

The use of the keywords in the table “Control Basic deprecated keywords” on page 1 change in BACstage version 2.3. Only the keywords are changed; the functions and statements they represent remain the same.

- When writing programs for controllers with Standard Control Basic (see *Versions of Control Basic* on page 8), BACstage will accept and compile *either* the deprecated keywords or the replacement versions of the keywords. For example either DEW-POINT or DEWPOINT may be used when writing a program.
- When BACstage retrieves a Control Basic program from a controller and decompiles it, the deprecated keywords are replaced with the new keywords. For example DEW-POINT becomes DEWPOINT and TIME-ON becomes TIMEON.
- For controllers with Next Generation Control Basic, BACstage will not accept or compile the deprecated keywords in the Table 1.

Table 1 Control Basic deprecated keywords

Deprecated keyword	Replacement keyword
COS-1	ARCCOS
DEW-POINT	DEWPOINT
DEW-POINT-SI	DEWPOINTSI
ENTHALPY SI	ENTHALPYSI
LN-1	INVLN
MODEL-NUMBER	MODELNUMBER
NETSENSOR-STATUS	NETSENSORSTATUS
ON-ERROR	ONERROR
OUTPUT-OVERRIDE	OUTPUTOVERRIDE
PANEL-ADDRESS	PANELADDRESS
SCHED-ON	SCHEDON
SCHED-OFF	SCHEDOFF
SENSOR-OFF	SENSOROFF
SENSOR-ON	SENSORON
SIN-1	ARCSIN
TAN-1	ARCTAN
TIME-ON	TIMEON
TIME-OFF	TIMEOFF

Keywords for new functions and statements

The following keywords are added to Control Basic:

ALIAS, BIND, CONST, FLUSH, HALT, ISNAN, LOCALS, NAN

A brief description of each of the new keywords follow. The help systems and reference manuals for BACstage and TotalControl Design Studio cover using the new keywords in greater detail. Review the keywords in those publications before using them in a program.

Unless noted otherwise, the new keywords can only be used when writing programs for controllers with Next Generation Control Basic.

ALIAS

ALIAS(device, object, property, local, read interval, write interval)

Declares a local variable and dynamically binds the value of a property to the variable. It also sets two intervals at which Control Basic will read from or write to the property bound to the variable.

```
ALIAS(1212, BO1, PV@4, Lights, 1:00:00, 60)
ALIAS("VAV", "Temp_204", PV, OAT, 100, NONE)
```

BIND

BIND (device, network, mac, option)

Binds a device instance to a physical network address. This is typically used to bind an MS/TP slave to a master device. *BIND* is required in only one program within the device.

```
BIND (550013,1,13)
BIND(123456, 678, 0x24)
```

CONST

CONST, variable, variable, ...

Use to declare one or more variables and assign to them a fixed value. Do not use with variables that change with subsequent steps in the program.

- Constant variables must be declared before they are used in a program.
- *CONST* may declare variables anywhere in the program but typically it is at the beginning.
- Must start with a letter A-Z, a-z, or an underscore (_). They are not case sensitive.
- Can be any combination of letters (A-Z or a-z), numbers (0-9) or the underscore (_).
- A constant may be used only within the program in which it is declared.

```
CONST Freeze = 32
CONST Boiling = 212
```

FLUSH

Flush (LocalAlias)

When a *FLUSH* statement runs, Control Basic immediately reads from or writes to the property bound to the local variable declared by *ALIAS*.

See also <u>*ALIAS* on page 3</u>.

```
ALIAS(1212, BO1, PV@4, Lights, 1:00:00, 60)
FLUSH(Lights)
```

HALT

HALT "Message"

Stops the program from running and sets the *Program State* property in the program object to *Halted*. The string *Message* is displayed in the property *Description of Halt*. The program can be restarted by doing any of the following:

- Performing a warm start or cold start
- Cycling controller power
- Setting the *Program Change* property to *Run*.

```
HALT "Shutting down the program"
```

ISNAN

ISNAN(_expression_)

ISNAN tests the value of *expression* to determine if it is a valid number. If the value of *expression* is equal to *NAN* (Not A Number), then ISNAN returns *true*. A typical use of *ISNAN* is to test the present value property of an object in a remote device.

In the following example the program tests the present value of analog input AI4 in device instance 4410 once every minute. If the value is a usable number then the remote value is stored in value object AV503. If the remote value is not valid—which would happen if the device were not responding—the value object is set equal to *55*, the default value.

```
If interval 00:01:00 then
  Rem Verify that the value is good
  If ISNAN( 4410.AI4 ) then
    REM Set a default value
    AV503 = 55
    else
    REM Use the received value
    AV503 = 4410.AI4
  EndIF
EndIf
```

LOCALS

LOCALS variable[, variable, ...]

Use to declare local variables. A local variable may be used only within the program in which it is declared.

- Local variables must be declared before they are used in a program.
- *LOCALS* may declare variables anywhere in the program but typically variables are declared at the beginning of the program.
- Must start with a letter A-Z, a-z, or an underscore (_). They are not case sensitive.
- Can be any combination of letters (A-Z or a-z), numbers (0-9) or the underscore (_).
- In programs that use *LOCALS*, all single-letter variables in use will also have to be declared. Single-letter variables (A-Z or a-z) are automatically declared unless LOCALS declares other variables.

```
Locals ChilledWaterSetpoint, a, b
```

NAN

Use *NAN* to set a variable or property to a *Not A Number* constant or to test if the variable or property is equal to *Not A Number*. *NAN* can be used in both both Standard and Next Generation Control Basic.

Standard Control Basic example

```
10 IF A <> NAN THEN GOTO 30
20 B = 55
30 B = A
```

Next Generation Control Basic example

```
    IF A <> NAN THEN GOTO CONTINUE
    B=55
CONTINUE:
    B=A
```

Labels in Next Generation Control Basic

In Next Generation Control Basic, labels are used instead of line numbers when program flow is redirected with any of the following statements.

- GOSUB
- GOTO
- ONERROR
- ON GOSUB
- ON GOTO

In the following program example, *CoolMode* and *HeatMode* destinations of a program redirection.

```
     IF T > 55 THEN GOTO CoolMode
     IF T <= 55 THEN GOTO HeatMode
     END
CoolMode:
     REM Cooling sequence runs here
     END
HeatMode:
     REM Heating sequence runs here
     END
```

Declare labels by typing a name followed immediately by a colon (:).

- A label can be any combination of letters (A-Z or a-z), numbers (0-9) or the underscore (_).
- Labels are not case sensitive.
- Labels are unique to the program in which they are declared.
- Labels cannot be a Control Basic keyword.

Line numbers

Line numbers are not used in Next Generation Control Basic programs. However, a line number is displayed in the Control Basic editors for both BACstage and TotalControl Design Studio. The line numbers displayed are only for identification of problems when the program is compiled.

Line numbers continue to be used in controllers with Standard Control Basic.

- In BACstage the programmers must enter the line number as the programs are written.
- In TotalControl Design Studio, the line number is generated automatically.

References to objects in remote devices

When referring to an object in a remote device, the device name or instance is now separated from the object reference by a period(.). In previous versions, the instance and name were separated with a dash (-).

```
10 A = 1214.AI1
```

In BACstage the name of the device or object can be used in place of an instance number.

```
20 A = MechanicalRoom.TempMechRoom
30 A = 1214.TempMechRoom
40 A = MechanicalRoom.AI1
```

Changes to IF THEN

Next Generation Basic now supports block and nested IF THEN statements.

```
Locals ChilledWaterSetpoint

AV24 = ChilledWaterSetpoint
IF BV258 THEN
    ChilledWaterSetpoint = 52
        ELSE
    Chilledwatersetpoint = 48
ENDIF

IF TIME > 7:00 THEN
    IF TIME < 9:00 THEN
        START BO1
    ENDIF
ENDIF
```

Local variables

The single-letter local variables a-z and A-Z may still be used without program modification. In addition to single letters, more descriptive variables may be used by declaring a variable with the statement LOCALS. However, once a variable is declared, all single letter variables used in the program must also be declared. For details on declaring local variables, see *LOCALS* on page 4.

File compatibility

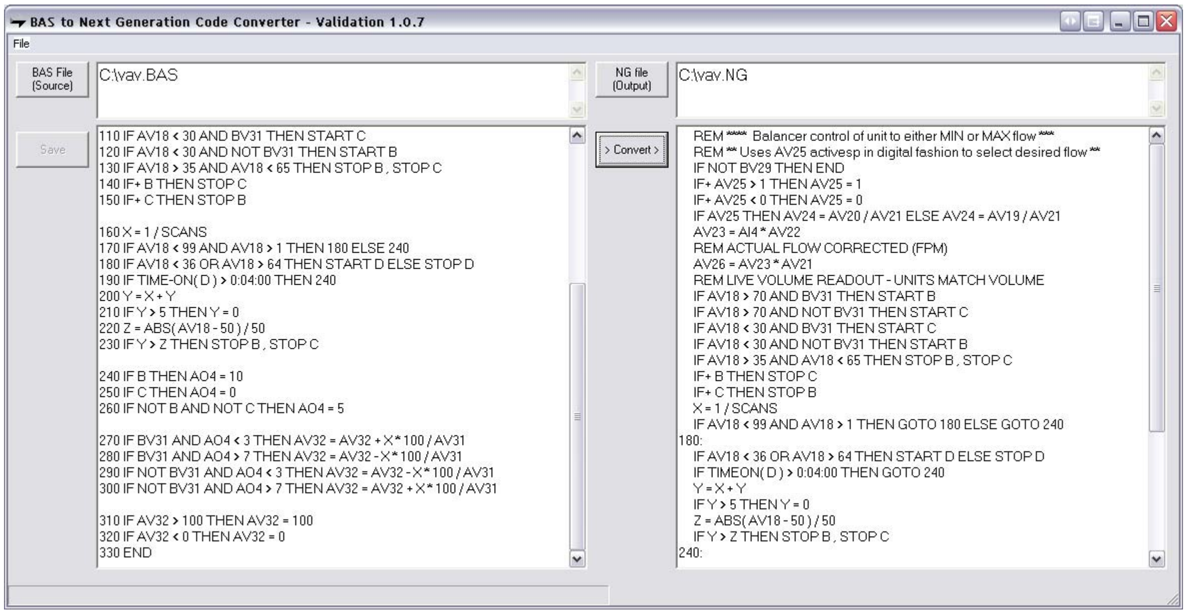
When saving and opening files with versions of BACstage other than BACstage 2.3, be aware of the following compatibility issues.

- Programs sent to a controller with BACstage 2.3 can be loaded from a controller with earlier versions of BACstage. BACstage 2.2 and earlier will list the programs using the deprecated keywords and original syntax for remote points.
- If a .BAS file includes any of the new or deprecated keywords, BACstage versions earlier than 2.3 will open but not compile the program. The new keywords and syntax must be changed to the original format.
- When transferring a .BAS file to a controller with Next Generation basic, the .BAS files can be converted to Next Generation format (.NG extension) with the conversion tool. See *Using the conversion tool* on page 7. Controllers that support Next Generation Control basic are listed in *Control Basic versions in controllers* on page 8.
- Files created with the *Backup Device* (.BAC files) in BACstage 2.3 are backwards compatible with BACstage 2.2 and earlier.

Using the conversion tool

A conversion tool is available on the KMC Controls web site that converts `.BAS` files to `.NG` files. Open an existing `.BAS` file and then convert it to the Next Generation format. Once converted you can do either of the following:

- Save the file to disk.
- Copy all or part of the text and then paste it into a Control Basic editor.



```
110 IF AV18 < 30 AND BV31 THEN START C
120 IF AV18 < 30 AND NOT BV31 THEN START B
130 IF AV18 > 35 AND AV18 < 65 THEN STOP B , STOP C
140 IF+ B THEN STOP C
150 IF+ C THEN STOP B

160 X = 1 / SCANS
170 IF AV18 < 99 AND AV18 > 1 THEN 180 ELSE 240
180 IF AV18 < 36 OR AV18 > 64 THEN START D ELSE STOP D
190 IF TIME-ON( D ) > 0:04:00 THEN 240
200 Y = X + Y
210 IF Y > 5 THEN Y = 0
220 Z = ABS( AV18 - 50 ) / 50
230 IF Y > Z THEN STOP B , STOP C

240 IF B THEN AO4 = 10
250 IF C THEN AO4 = 0
260 IF NOT B AND NOT C THEN AO4 = 5

270 IF BV31 AND AO4 < 3 THEN AV32 = AV32 + X * 100 / AV31
280 IF BV31 AND AO4 > 7 THEN AV32 = AV32 - X * 100 / AV31
290 IF NOT BV31 AND AO4 < 3 THEN AV32 = AV32 - X * 100 / AV31
300 IF NOT BV31 AND AO4 > 7 THEN AV32 = AV32 + X * 100 / AV31

310 IF AV32 > 100 THEN AV32 = 100
320 IF AV32 < 0 THEN AV32 = 0
330 END
```

```
   REM ****  Balancer control of unit to either MIN or MAX flow ***
   REM ** Uses AV25 activesp in digital fashion to select desired flow **
   IF NOT BV29 THEN END
   IF+ AV25 > 1 THEN AV25 = 1
   IF+ AV25 < 0 THEN AV25 = 0
   IF AV25 THEN AV24 = AV20 / AV21 ELSE AV24 = AV19 / AV21
   AV23 = AI4 * AV22
   REM ACTUAL FLOW CORRECTED (FPM)
   AV26 = AV23 * AV21
   REM LIVE VOLUME READOUT - UNITS MATCH VOLUME
   IF AV18 > 70 AND BV31 THEN START B
   IF AV18 > 70 AND NOT BV31 THEN START C
   IF AV18 < 30 AND BV31 THEN START C
   IF AV18 < 30 AND NOT BV31 THEN START B
   IF AV18 > 35 AND AV18 < 65 THEN STOP B , STOP C
   IF+ B THEN STOP C
   IF+ C THEN STOP B
   X = 1 / SCANS
   IF AV18 < 99 AND AV18 > 1 THEN GOTO 180 ELSE GOTO 240
180:
   IF AV18 < 36 OR AV18 > 64 THEN START D ELSE STOP D
   IF TIMEON( D ) > 0:04:00 THEN GOTO 240
   Y = X + Y
   IF Y > 5 THEN Y = 0
   Z = ABS( AV18 - 50 ) / 50
   IF Y > Z THEN STOP B , STOP C
240:
```

Illustration 1 Control Basic converter tool

Control Basic versions in controllers

The table "Versions of Control Basic" lists the versions of Control Basic that are supported in the BACnet controllers from KMC Controls.

Table 2 Versions of Control Basic

Model number	Control Basic version	Control Basic file format
BAC-A1616	Next Generation	NG
BAC-5801 BAC-5802	Standard	BAS
BAC-5831	Standard	BAS
BAC-7001 BAC-7051	Standard	BAS
BAC-7003 BAC-7053	Standard	BAS
BAC-7301 BAC-7301C	Standard	BAS
BAC-7302 BAC-7302C	Standard	BAS
BAC-7303 BAC-7303C	Standard	BAS
BAC-7401 BAC-7401C	Standard	BAS