Tips on Transmitting Serial Data with HomeVision

HomeVision can transmit serial data to an attached computer or other device. Common uses include:

- Logging data to a computer text file
- Using a text-to-speech program to "speak"
- Playing WAV files
- Issuing commands to a thermostat or security system

In order to do these things, you must be able to transmit the required information. That's quite easy if it's simple ASCII text. However, it's often necessary to transmit variable values or binary data. This article shows how to accomplish this. It also provides some tricks on how you might collect and use certain data.

TEXT TO SPEECH

The most popular use of serial transmissions is for "text to speech". There are several computer programs that can read in serial data, and then "speak" it using the computer's sound card. If you route the audio output into your home's audio distribution system, HomeVision can speak to you anywhere in your house! Home Voice is a popular program that works with HomeVision to do this (and also gives you voice control of your home).

Whether you use Home Voice or another program, the concept is the same: you transmit the text you want to speak to the PC, in the proper format. For example, to have Home Voice say "alarm is armed", you need to transmit this:

```
speak alarm is armed <carriage return>
```

This is done using the following two serial commands:

```
Serial xmit: "speak alarm is armed"
Serial xmit: Carriage return and line feed
```

You'll probably also want to be able to speak variable values (such as the current temperature, thermostat setpoint, etc.). HomeVision has a command to transmit a variable value as ASCII text. Here's an example:

```
Serial xmit: "speak the temperature is "
Serial xmit: Var #0 (Temp) value as 3 ASCII bytes
Serial xmit: " degrees"
Serial xmit: Carriage return and line feed
```

If the variable value is 78, this will transmit the following (the carriage return and line feed are not shown):

```
speak the temperature is 078 degrees
```

Note that when HomeVision transmits a variable value as ASCII, it always transmits three bytes. If the variable is less than 100, it sends a leading zero. For example, a variable value of 78 is transmitted as:

```
078
```

and a value of 5 is transmitted as:

```
005
```

Depending upon the PC program, this may or may not work right because of the leading zero(s). To overcome this problem, HomeVision version 2.6 (currently in Alpha testing) has a new command that will omit the leading zeros. If you need this capability prior to the version 2.6 release, let us know and we can send you an advance version.

TRANSMITTING THE TIME

It’s often useful to transmit the current time or the time of a prior event. HomeVision has a built-in command that transmits the current time and date together on one line. However, you'll probably want to use a different format. For example, you may want to transmit the time like this:

```
11:05 AM
```

Here’s how to do this. First, use two variables to hold the desired time (you could use three variables if you need the seconds value). For example, to store the current time, perform these two commands:

```
Var #1 (Hour) = current hour
Var #2 (Minute) = current minute
```

The hour value will be in 24-hour format (i.e., 0 = 12:00 AM, 1 = 1:00 AM, …. 23 = 11:00 PM), so you’ll need to convert it to AM/PM format. We recommend you do this with a macro so you can use it elsewhere in your schedule also. Here’s a macro to do this:

Macro #1 (Convert time to AM/PM):

```
If
  Var #1 (Hour) = 0
Then
  Var  #1 (Hour) = 12
  Set flag #1 (Time is AM)
Else
  If
    Var #1 (Hour) >= 1
    And Var #1 (Hour) <= 11
  Then
    Set flag #1 (Time is AM)
  Else
    If
      Var #1 (Hour) = 12
    Then
      Clear flag #1 (Time is AM)
    Else
      If
        Var #1 (Hour) >= 13
        And Var #1 (Hour) <= 23
      Then
        Var  #1 (Hour) = var #1 (Hour) - 12
        Clear flag #1 (Time is AM)
      Else
        ; ERROR
      End If
    End If
  End If
End If
```

This macro converts the value in variable #1 into AM/PM format. Initially, variable #1 ranges from 0 to 23. When the macro is done, the variable is between 1 and 12 (unless there’s an error), and flag #1 indicates whether the result is AM or PM (Set = AM, Clear = PM).

Once the hour is in AM/PM format, use a second macro to transmit it out the serial port, like this:

<u>Macro #2 (Transmit time):</u>

```
Serial xmit: Var #1 (Hour) value as 3 ASCII bytes
Serial xmit: ":"
Serial xmit: Var #2 (Minute) value as 3 ASCII bytes
If
  Flag #1 (Time is AM) is set
Then
  Serial transmit: " AM"
Else
  Serial transmit: " PM"
End If
```

This macro transmits the hour and minute values contained in the two variables, separated by a colon. It then transmits either “AM” or “PM” based on the state of the flag. Note that if you are sending this to a text-to-speech program, you may need to replace the colon with a blank space. The colon works properly with Home Voice's text-to-speech.

To summarize, first set variables #1 and #2 to the desired time, then run macro #1 to convert it to AM/PM format, then run macro #2 to transmit it. Here’s how you would transmit the current time:

```
Var #1 (Hour) = current hour
Var #2 (Minute) = current minute
Do macro #1 (Convert time to AM/PM) once
Do macro #2 (Transmit time) once
```

STORING AN EVENT TIME

It’s often useful to store the time an event occurs. For example, you could determine the time that your mail arrives by placing a switch on your mailbox door. To store this time, first create variables to hold it (one for the hour, one for the minute, and, if desired, one for the second). When the event occurs, put the current time into the variables, like this:

```
Var #3 (Mail hour) = current hour
Var #4 (Mail minute) = current minute
```

Now let’s assume that sometime later you need to transmit this time out the serial port. You would use these commands:

```
Var #1 (Hour) = Var  #3 (Mail hour)
Var #2 (Minute) = Var  #4 (Mail minute)
Do macro #1 (Convert time to AM/PM) once
Do macro #2 (Transmit time) once
```

This puts the mail arrival time into variables #1 and #2, then runs the macros described previously. This is quite easy since we’re reusing macros written previously. The only trick is to remember that the macros use the values in variables #1 and #2, so we need to set the variables prior to running the macros.

SUNRISE AND SUNSET TIMES

HomeVision calculates sunrise and sunset times each day for your latitude and longitude, and you can easily perform events based on these times. It could also be useful to transmit these times. For example, in the morning you could have your computer speak to you and tell you what time sunset is today. HomeVision doesn’t have a command to transmit the sunset time, but here's a way you can do it. Create a scheduled event that runs at sunset and include these two commands in it:

```
Var #5 (Sunset hour) = current hour
Var #6 (Sunset minute) = current minute
```

Since the event runs only at sunset, the current time will be the sunset time. You simply store the current time in two variables. Later, you can transmit these variable values as described previously.

TRANSMITTING BINARY VALUES

HomeVision can transmit binary data in addition to ASCII text. Binary data is transmitted in a two step process, as shown in this example:

```
Var #7 (Binary #) = 13
Serial xmit: Var #7 (Binary #) value as 1 binary byte
```

This example sets the variable to 13, which is the binary value for a carriage return. It then transmits the variable value as a single binary byte. Since you can set variables to any value from 0 to 255, you can transmit any binary value. If you need to do this multiple times in your schedule, you can create a macro and then run it whenever you need to.

Note that HomeVision already has a command to transmit a carriage return and line feed. You'll usually want to do this at the end of each line of text you send. However, there may be times you want to transmit only a carriage return, as shown in the above example.

ACCESSING OTHER PC SERIAL PORTS

HomeVision has the unique capability to allow the controller to access multiple serial ports through a PC. To do this, the HomeVision software must be running using the serial port the controller is connected to. The controller can then send commands to the PC to open another serial port, send data to it, and close it. While the port is open, any data coming into it is forwarded to the HomeVision controller.

The desired comm port is opened by sending ASCII text to the PC, like this:

```
Open comm port "2,19200,N,8,1,1"
```

In your schedule, you transmit it with this serial command:

```
Serial xmit: "Open comm port "2,19200,N,8,1,1""
```

Refer to the owner's manual for details on the command parameters. After opening the port, you can send data to it like this:

```
Serial xmit: "To comm port "Text to send""
```

When the PC software receives the text:

```
To comm port "Text to send"
```

it takes the text between the double quotes and sends it to the open serial port. It also automatically puts a carriage return and line feed on the end (remember this, it's important!).

Now let’s assume you want to transmit a variable value to the serial port. For example, you may have a thermostat connected that you want to change to a temperature setpoint of 78 degrees. Let’s assume the thermostat must receive the command in this format:

```
Setpoint=078<carriage return>
```

The easiest way to do this is with this command:

```
Serial xmit: "To comm port "Setpoint=078""
```

Since HomeVision automatically puts a carriage return on the end, this command successfully changes the thermostat setpoint.

In this example, the 78 degree number is a fixed value, not a variable as we’d like. So how do we send a variable value? Well, here’s the first thing most users would try, but it won't work:

```
Serial xmit: "To comm port "Setpoint=""
Serial xmit: Var #0 (Temp) value as 3 ASCII bytes
Serial xmit: Carriage return and line feed
```

The problem with this is that the HomeVision PC software puts a carriage return after the first command when sending it to the thermostat. Here’s what actually gets transmitted to the thermostat (assuming variable #0 is 78):

```
Setpoint=<carriage return>078<carriage return>
```

To leave off the carriage return after the "Setpoint=" text, we use a trick. We leave off the ending double quote from the first command, send the variable value, then send a final double quote. Here are the commands to do that:

```
Serial xmit: "To comm port "Setpoint="
Serial xmit: Var #0 (Temp) value as 3 ASCII bytes
Serial xmit: """
```

Here’s what these commands actually transmit to the PC:

```
To comm port "Setpoint=
078
"
```

Note that there are no carriage returns on any line. As far as the PC is concerned, all the data comes in together, like this:

```
To comm port "Setpoint=078"
```

This is exactly what we want. When the PC receives this, it takes the text between the double quotes and sends it to the thermostat.

You can use this same trick for any commands where the PC is looking for text between double quotes. These commands are:

- Open comm port "2,19200,N,8,1,1"
- To comm port "Text to send"
- Write to file filename.txt "Text to write"
- Send keys "Keystrokes"
- Activate window "Window name"
- Display message "Message to display"

CONCLUSION

HomeVision can transmit any combination of fixed text, variable values, and binary data out the serial port. Since variables can hold just about any information you want (temperature, time, date, etc.), there are very few limits to what you can do.