NetMedia

Basic Express BX-35
Application Note

Six Day 8 Channel ADC Data Logger

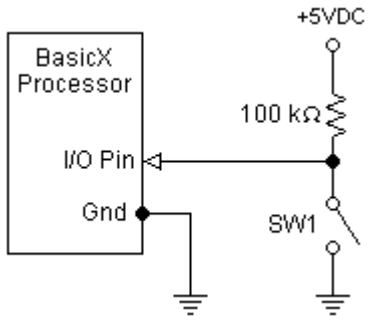
Introduction

This application note illustrates how you can use a BasicX system as an 8 channel analog data logger. Once every 5 minutes the system reads and logs data from each of the 8 analog to digital converter (ADC) inputs. The 10 bit integer data is recorded to nonvolatile EEPROM memory.

The logged data can be retrieved at any time by placing the logger in upload mode. In this mode, all logged data is transmitted out the serial port in the form of comma-delimited ASCII text strings.

Program overview

The source code file that accompanies this application note is called Datalogger.bas. The program has 2 modes of operation -- (1) data logging, and (2) upload. The mode is determined at power-up based on the position of the upload mode switch SW1 (below). At power up, if SW1 is in its open position (logic high), the program will run in data logging mode. In this mode the BasicX will sample and log the value of each ADC channel at a rate of once every five minutes.



Upload Mode Switch

The switch SW1 is connected to an I/O pin on the BasicX processor. The pin number depends on the processor type. With SW1 in its closed position, the I/O pin is at 0 V and the program will run it upload mode. With the switch in its open position, the pin is at +5 V, and the program will run in data logging mode.

If SW1 is in its closed position (logic low) at power up, the program will run in upload mode. In this mode, the program retrieves the logged ADC data from EEPROM, one sample at a time. Each block of 8 ADC values is then converted to an ASCII string and transmitted out the Com1 serial port.

Note -- with SW1 in the log position, if you press the reset button, or if power is disrupted, the program will start over. Once restarted, the program will overwrite previously stored information.

You can optionally modify the program to automatically clear the block of EEPROM memory where the ADC data is stored. Procedure ClearMemoryBlock can be called if the FirstTime function returns true, which means this is the first time the function was called since the program was downloaded. The initialization process may take several minutes, and can be skipped if you don't care whether the memory is initialized.

Low level code

To store and retrieve data in EEPROM, the system calls PutEEPROM and GetEEPROM are used. Example syntax for PutEEPROM:

```
Dim CurrentAddress As Long
Dim ADCdata(1 To 8) As Byte
Dim Channel As Integer
Const ElementSize As Integer = 2

Call PutEEPROM(CurrentAddress, ADCdata(Channel), ElementSize)
```

Keeping track of time

The data logger program uses the internal real time clock (RTC) to determine when a sample should be taken. The program does this by initializing the RTC to midnight, then using the Timer function to keep track of the number of seconds past midnight.

Whenever the elapsed time matches the desired sample period, which in our case is 300 seconds (5 minutes), a new sample is taken. At this point, the RTC is reset to midnight again, and the process starts over.

Although the BasicX doesn't log the time at which each group of ADC samples is taken, sample times are still easily obtainable. Since data samples are taken and logged at 5 minute intervals, the timestamp for each sample can be approximated by recording the time of day at which the logger was started, then adding 5 minutes per sample. For example, if the logger was started at 1:00 PM, the first sample would be recorded at 1:05 PM and the 10th sample would be recorded at about 1:50 PM.

Sizing the EEPROM memory block

Each ADC sample requires 2 bytes of memory, and storing all 8 channels requires 16 bytes. At a sample rate of 16 bytes every 300 s (5 min), and assuming about 28 KB of EEPROM is available, that translates to just over 6 days of data logging.

Since the program shares EEPROM space with the logged data, you need to be careful to avoid memory conflicts. The MPP map file can be used to determine the memory available for storing data. The MPP file is generated whenever you compile a program.

As an alternative, you can use block data objects for storing data in EEPROM. The advantage is that block data objects are semantically similar to arrays, which means you can avoid making low level calls to GetEEPROM and PutEEPROM. In addition, the compiler automatically allocates memory to the objects, and will warn you if there are memory conflicts.

Another advantage is that the downloader can initialize EEPROM memory much faster than the data logger program -- several seconds for the downloader vs. several minutes for the data logger.

Refer to the language reference manual and block data application notes for more information about block data classes.

Uploading logged data

The BasicX compiler/downloader can be run in file-capture mode in order to record data received from the BasicX system. In the downloader, go to the File/Capture-to-File menu choice and specify a filename. The data is recorded as a plain ASCII text file. Alternatively, you can use a terminal program such as HyperTerminal to record the data.

Once the data is stored in a file, you can use Excel or similar spreadsheet programs to convert the raw ADC values into the desired data formats.

Once you have your PC properly configured and ready to receive the upload, all you need to do is this:

1. Disconnect the logger's power source.
2. Connect the logger's serial port (Com1) to a PC running a capture program.
3. Place the logger's mode switch in the upload position.
4. Reconnect the logger's power source.

If everything is working properly, the incoming data should be in a format similar to this:

```
1023, 865, 743, 685, 620, 571, 516, 463
1023, 907, 785, 710, 640, 586, 536, 493
1023, 915, 795, 718, 646, 590, 540, 501
   [...]
```

A total of 1750 lines of text are uploaded for the entire 6-plus days of sampling.

2.00.A