SPI Port

This embedded C function can be used to add additional SPI ports to the Micromite.

It has the following features:

- It is always the master and can run at speeds up to 2MHz.
- The operating mode (mode 0 to 3) and the number of bits to send/receive can be specified.
- It can use any three I/O pins and these I/O pins can be changed from call to call so an unlimited number of SPI interfaces can be created.

Adding the Function to MMBasic

To add the SPIPort function to MMBasic you must insert the following code somewhere in your BASIC program (you can use copy and paste from this document). The exact spot is not important.

```
CFunction SPIPort(integer, integer, integer, integer) integer
    00000008
    40024800 00442021 40024800 0044102B 1440FFFD 00000000 03E00008 00000000
    27BDFFB0 AFBF004C AFBE0048 AFB70044 AFB60040 AFB5003C AFB40038 AFB30034
    AFB20030 AFB1002C AFB00028 00808821 00A09021 00C0A021 00E0A821 10800005
    8FB30068 10A00004 3C029D00 14C00008 3C109D00 3C029D00 8C420010 00002021
    24050002 0040F809 00003021 3C109D00 8E020010 8E240000 24050002 0040F809
    00003021 8E020024 8E240000 0040F809 00002821 AFA20018 8E020028 0040F809
    8E240000 AFA2001C 8E020024 8E440000 0040F809 24050006 AFA20020 8E020024
    8E440000 0040F809 24050005 AFA20024 8E020028 0040F809 8E440000 24170001
    0057B804 8FA20064 10400008 3C109D00 8C420000 50400006 8E020024 24030003
    5443000D 8E020024 3C109D00 8E020024 8E840000 0040F809 24050005 0040B021
    8E020024 8E840000 0040F809 24050006 1000000A 0040F021 8E840000 0040F809
    24050006 0040B021 8E020024 8E840000 0040F809 24050005 0040F021 3C029D00
    8C420028 0040F809 8E840000 24140001 0054A004 12600002 24110008 8E710000
    2631FFFF 32220020 24030001 02238804 02208021 0002800A 0002880B 2402FFFF
    2403FFFF AFA20010 12A00005 AFA30014 8EA20000 8EA30004 AFA20010 AFA30014
    8FA30060 10600002 0000A821 8C750000 02301025 00009021 10400039 00009821
    8FA40010 02241824 8FA60014 02061024 00621025 10400004 8FA30024 8FA20020
    10000002 AC570000 AC770000 AED40000 00000000 00000000 00000000 00000000
    00000000 00000000 52A00013 00121FC2 02A02021 0411FF6A 00000000 00121FC2
    00131040 00122840 8FA60018 8CC40000 8FA6001C 00C42006 30840001 00A49025
    00629825 AFD40000 0411FF5D 02A02021 1000000C 00101FC0 00131040 00122840
    8FA60018 8CC40000 8FA6001C 00C42006 30840001 00A49025 00629825 AFD40000
    00101FC0 00111042 00621025 00101842 00408821 00431025 1440FFC9 00608021
    00000000 00000000 00000000 00000000 8FA20064 10400006 00000000 8C420000
    10400003 24030002 14430003 02401021 AED40000 02401021 02601821 8FBF004C
    8FBE0048 8FB70044 8FB60040 8FB5003C 8FB40038 8FB30034 8FB20030 8FB1002C
    8FB00028 03E00008 27BD0050
End CFunction
```

Usage

```
rd = SPIPort( rx, tx, clk, data_to_send, speed, mode, bits )
```

Where:

- 'rd' is the value returned by the function and is the data received during the transaction.
- ‘rx’ is the pin number for the data input (MISO)
- ‘tx’ is the pin number for the data output (MOSI)
- ‘clk’ is the pin number for the clock generated by this function (CLK)

The following parameters are optional. If not required they can be left off the end of the list.

- ‘data_to_send’ is optional and is an integer representing the data to send over the output pin. If it is not specified the ‘tx’ pin will be held low.
- ‘speed’ is the speed of the clock (see below for an explanation). The default is 0.
- 'mode' is a single numeric digit representing the transmission mode (see below for an explanation). It is optional and if not specified the default is 0.
- 'bits' is the number of bits to send/receive. Range is 1 to 64. The default is 8.

All parameters must be integers (**not** floating point numbers or variables).

The SPI function will return the data received during the transaction as an integer. Note that a single SPI transaction will send data while simultaneously receiving data from the slave.

Clock Speed

The 'speed' parameter is a number which can range from zero to some arbitrary number, the greater the number the slower the resultant SPI clock speed. The following table lists the various SPI clock speeds that result from typical CPU speeds and values used for the 'speed' parameter:

CPU speed	'speed' parameter and the resultant SPI clock speed		
	0	4	24
100 MHz	2.5 MHz	1.2 MHz	600 KHz
80 MHz	2.0 MHz	1.0 MHz	500 KHz
40 MHz	1.0 MHz	500 KHz	250 KHz
20 MHz	500 KHz	250 KHz	125 KHz

Transmission Format

The most significant bit is sent and received first. The format of the transmission can be specified by the 'mode' parameter as follows:

Mode	Description	CPOL	CPHA
0	Clock is active high, data is captured on the rising edge and output on the falling edge	0	0
1	Clock is active high, data is captured on the falling edge and output on the rising edge	0	1
2	Clock is active low, data is captured on the falling edge and output on the rising edge	1	0
3	Clock is active low, data is captured on the rising edge and output on the falling edge	1	1

For a more complete explanation see: http://en.wikipedia.org/wiki/Serial_Peripheral_Interface_Bus

I/O Pins

Before invoking this function ‘tx’ and ‘clk’ pins must be configured as outputs (either normal or open collector) using the SETPIN command. The output value of these pins should be set (using the PIN function) before the SETPIN command so that they starts with the correct polarity. The 'rx' pin does not have to be set as an input as that is done by the CFunction.

The SPI enable signal is often used to select a slave and “prime” it for data transfer. This signal is not generated by this function and if required should be generated using the PIN function.

The SPIPort function does not “take control” of the I/O pins and the PIN command will continue to operate as normal on them. Also, because the I/O pins can be changed between function calls it is possible to communicate with many different SPI slaves on different I/O pins.

Example

The following example uses pin 21 for 'rx', 22 for 'tx', 23 for the clock and 24 as the enable. It will send the command 80 (hex) and receive two bytes from the slave SPI device. Because the mode, speed and number of bits are not specified the defaults are used.

```
PIN(22) = 0 : SETPIN 22, 8       ' set tx pin low then set it as an output
PIN(23) = 0 : SETPIN 23, 8       ' set clk pin low then set it as an output
PIN(24) = 1 : SETPIN 24, 8       ' pin 24 will be used as the enable signal

PIN(24) = 0                      ' assert the enable line (active low)
junk = SPIPort(21, 22, 23, &H80) ' send the command and ignore the return
byte1 = SPIPort (21, 22, 23)     ' get the first byte from the slave
byte2 = SPIPort (21, 22, 23)     ' get the second byte from the slave
PIN(24) = 1                      ' deselect the slave
```