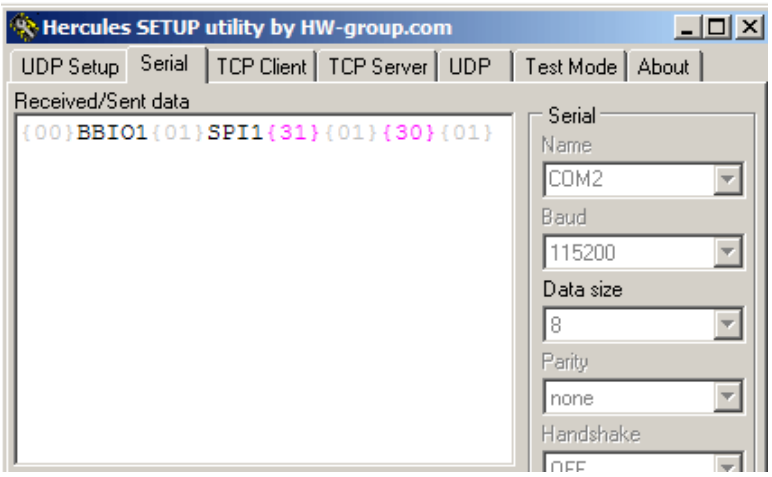
SPI (binary)



Raw SPI is a new mode that provides high-speed access to the Bus Pirate SPI hardware. It was developed in conjunction with Michal Ludvig, so that AVRDude can use the Bus Pirate to program AVR microcontrollers and EEPROMs.

Firmware v2.3 includes two new raw I/O modes that give computer software and scripts direct access to the Bus Pirate hardware. Hopefully this opens the door to a whole new class of Bus Pirate applications, like chip programmers. In this post we describe the raw SPI access mode. We'll describe the raw bitbang mode in a few days. The protocol is documented below the break.

Contents [hide]

Overview

Commands are a single byte, except bulk SPI transfers. The Bus Pirate responds to SPI write commands with the data read from the SPI bus during the write. Most other commands return 0x01 for success, or 0x00 for failure/unknown command.

Last update: firmware v5.8

Key points

Send 0x00 to the user terminal (max.) **20 times** to enter the raw binary bitbang mode. Pause briefly after sending each 0x00 to check if **BBIO1** is returned. Example binary mode entry functions.
Enter 0x01 in bitbang mode to enter raw SPI mode.
Return to raw bitbang mode from raw SPI mode by sending 0x00 **one time**.
Operations that write a byte to the SPI bus also return a byte read from the SPI bus.
Hex values shown here, like 0x00, represent actual byte values; not typed ASCII entered into a terminal.
Other values are shown as 8bit binary numbers. Here's a binary->decimal->hex converter.

Commands

00000000 - Enter raw bitbang mode, reset to raw bitbang mode

This command has two purposes. First, send it to the command line interface 20 times to enter the raw bitbang binary mode. It's also used to exit the raw SPI mode and return to raw bitbang mode.

Send the value 0x00 to the Bus Pirate command line interface **20 times** to enter raw bitbang mode. The Bus Pirate replies 'BBIOx', where x is the raw bitbang version number (currently 1).

Once in raw SPI mode (see command 00000001), the 0x00 command returns to raw bitbang mode. Send 0x00 **once** to return to raw bitbang mode.

In raw bitbang mode, send 0x0F to exit raw bitbang mode and reset the Bus Pirate.

00000001 - Enter raw SPI mode, display version string

Once in raw bitbang mode, send 0x01 to enter raw SPI mode. The Bus Pirate responds 'SPIx', where x is the raw SPI protocol version (currently 1). Get the version string at any time by sending 0x01 again.

0000001x - CS high (1) or low (0)

Toggle the Bus Pirate chip select pin, follows HiZ configuration setting. CS high is pin output at 3.3volts, or HiZ. CS low is pin output at ground. Bus Pirate responds 0x01.

000011XX - Sniff SPI traffic when CS low(10)/all(01)

(updated in v5.1)

The SPI sniffer is implemented in hardware and should work up to 10MHz. It follows the configuration settings you entered for SPI mode. The sniffer can read all traffic, or filter by the state of the CS pin.

> [/] - CS enable/disable
> \xy - escape character (\) precedes two byte values X (MOSI pin) and Y (MISO pin) **(updated in v5.1)**

Sniffed traffic is encoded according to the table above. The two data bytes are escaped with the '\' character to help locate data in the stream.

Send the SPI sniffer command to start the sniffer, the Bus Pirate responds 0x01 then sniffed data starts to flow. Send any byte to exit. ~~Bus Pirate responds 0x01 on exit.~~ (0x01 reply location was changed in v5.8)

If the sniffer can't keep with the SPI data, the MODE LED turns off and the sniff is aborted. **(new in v5.1)**

The sniffer follows the output clock edge and output polarity settings of the SPI mode, but not the input sample phase.

More detailed notes on the SPI sniffer in the SPI user terminal documentation.

```
There's a utility to access the binary SPI sniffer.
```

0001xxxx - Bulk SPI transfer, send/read 1-16 bytes (0=1byte!)

Bulk SPI allows direct byte reads and writes. The Bus Pirate expects xxxx+1 data bytes. Up to 16 data bytes can be sent at once, each returns a byte read from the SPI bus during the write.

Note that 0000 indicates 1 byte because there's no reason to send 0. BP replies 0x01 to the bulk SPI command, and returns the value read from SPI after each data byte write.

The way it goes together:

> The upper 4 bit of the command byte are the bulk read command (0001xxxx)
> xxxx = the number of bytes to read. 0000=1, 0001=2, etc, up to 1111=16

If we want to read (0001) four bytes (0011=3=read 4) the full command is 00010011 (0001 + 0011). Convert from binary to hex and it is 0x13

0100wxyz - Configure peripherals w=power, x=pull-ups, y=AUX, z=CS

Enable (1) and disable (0) Bus Pirate peripherals and pins. Bit w enables the power supplies, bit x toggles the on-board pull-up resistors, y sets the state of the auxiliary pin, and z sets the chip select pin. Features not present in a specific hardware version are ignored. Bus Pirate responds 0x01 on success.

Note: CS pin always follows the current HiZ pin configuration. AUX is always a normal pin output (0=GND, 1=3.3volts).

01100xxx - SPI speed

000=30kHz, 001=125kHz, 010=250kHz, 011=1MHz, 100=2MHz, 101=2.6MHz, 110=4MHz, 111=8MHz

This command sets the SPI bus speed according to the values shown. Default startup speed is 000 (30kHz).

1000wxyz - SPI config, w=HiZ/3.3v, x=CKP idle, y=CKE edge, z=SMP sample

This command configures the SPI settings. Options and start-up defaults are the same as the user terminal SPI mode. w= pin output HiZ(0)/3.3v(1), x=CKP clock idle phase (low=0), y=CKE clock edge (active to idle=1), z=SMP sample time (middle=0). The Bus Pirate responds 0x01 on success.

Default raw SPI startup condition is 0010. HiZ mode configuration applies to the SPI pins and the CS pin, but not the AUX pin. See the PIC24FJ64GA002 datasheet and the SPI section[PDF] of the PIC24 family manual for more about the SPI configuration settings.

00000100 - Write then read

This command was developed to help speed ROM programming with Flashrom. It might be helpful for a lot of common SPI operations. It enables chip select, writes 0-4096 bytes, reads 0-4096 bytes, then disables chip select.

All data for this command can be sent at once, and it will be buffered in the Bus Pirate. The write and read operations happen all at once, and the read data is buffered. At the end of the operation, the read data is returned from the buffer. The goal is to meet the stringent timing requirements of some ROM chips by buffering everything instead of letting the serial port delay things.

Write then read command format

command (1byte)	number of write bytes (2bytes)	number of read bytes (2bytes)	bytes to write (0-4096bytes)

Return data format

success/0x01 (1byte)	bytes read from SPI (0-4096bytes)

1. First send the *write then read* command (00000100)
2. The next two bytes (High8/Low8) set the number of bytes to write (0 to 4096)
3. The next two bytes (h/l) set the number of bytes to read (0 to 4096)
4. **If the number of bytes to read or write are out of bounds, the Bus Pirate will return 0x00 now**
5. Now send the bytes to write. Bytes are buffered in the Bus Pirate, there is no acknowledgment that a byte is received.
6. Now the Bus Pirate will write the bytes to SPI and read/return the requsted number of read bytes
7. CS goes low, all write bytes are sent at once
8. Read starts immediately, all bytes are put into a buffer at max SPI speed (no waiting for UART)
9. At the end of the read, CS goes high
10. The Bus Pirate now returns 0x01, success
11. Finally, the buffered read bytes are returned via the serial port

Except as described above, there is no acknowledgment that a byte is received.

00000101 - Write then read, no CS

Same as the previous command, but CS transitions are NOT automated/included.
Added for AVRDude

AVR Extended Commands

00000110 - AVR Extended Commands
00000000 - Null operation - verifies extended commands are available.
00000001 - Return version (2 bytes)
00000010 - Bulk Memory Read from Flash