spi_butterfly - parport-to-butterfly adapter driver

This is a hardware and software project that includes building and using a parallel port adapter cable, together with an "AVR Butterfly" to run firmware for user interfacing and/or sensors. A Butterfly is a $US20 battery powered card with an AVR microcontroller and lots of goodies: sensors, LCD, flash, toggle stick, and more. You can use AVR-GCC to develop firmware for this, and flash it using this adapter cable.

You can make this adapter from an old printer cable and solder things directly to the Butterfly. Or (if you have the parts and skills) you can come up with something fancier, providing circuit protection to the Butterfly and the printer port, or with a better power supply than two signal pins from the printer port. Or for that matter, you can use similar cables to talk to many AVR boards, even a breadboard.

This is more powerful than "ISP programming" cables since it lets kernel SPI protocol drivers interact with the AVR, and could even let the AVR issue interrupts to them. Later, your protocol driver should work easily with a "real SPI controller", instead of this bitbanger.

The first cable connections will hook Linux up to one SPI bus, with the AVR and a DataFlash chip; and to the AVR reset line. This is all you need to reflash the firmware, and the pins are the standard Atmel "ISP" connector pins (used also on non-Butterfly AVR boards). On the parport side this is like "sp12" programming cables.

Signal	Butterfly	Parport (DB-25)
SCK	J403.PB1/SCK	pin 2/D0
RESET	J403.nRST	pin 3/D1
VCC	J403.VCC_EXT	pin 8/D6
MOSI	J403.PB2/MOSI	pin 9/D7
MISO	J403.PB3/MISO	pin 11/S7,nBUSY
GND	J403.GND	pin 23/GND

Then to let Linux master that bus to talk to the DataFlash chip, you must (a) flash new firmware that disables SPI (set PRR.2, and disable pullups by clearing PORTB.[0-3]); (b) configure the mtd_dataflash driver; and (c) cable in the chipselect.

Signal	Butterfly	Parport (DB-25)
VCC	J400.VCC_EXT	pin 7/D5
SELECT	J400.PB0/nSS	pin 17/C3,nSELECT
GND	J400.GND	pin 24/GND

Or you could flash firmware making the AVR into an SPI slave (keeping the DataFlash in reset) and tweak the spi_butterfly driver to make it bind to the driver for your custom SPI-based protocol.

The "USI" controller, using J405, can also be used for a second SPI bus. That would let you talk to the AVR using custom SPI-with-USI firmware, while letting either Linux or the AVR use the DataFlash. There are plenty of spare parport pins to wire this one up, such as:

Signal	Butterfly	Parport (DB-25)
SCK	J403.PE4/USCK	pin 5/D3
MOSI	J403.PE5/DI	pin 6/D4
MISO	J403.PE6/DO	pin 12/S5,nPAPEROUT
GND	J403.GND	pin 22/GND
IRQ	J402.PF4	pin 10/S6,ACK
GND	J402.GND(P2)	pin 25/GND