

SPI Board C & Assembly Strategy

Introduction

This document will suggest basic strategies for creating 'C' and Assembly code for the SPI E-Block. Because this E-Block can be used with a variety of upstream boards (e.g. PICmicro multi-programmer board), this document will not provide all of the information required. See the "further reading" section for more complete reference information.

SPI is a relatively simple 3-wire protocol typically used for chip-to-chip communication. One device is the "master" and the other is the "slave". The devices on the SPI E-Block are both slaves and need to be connected to an upstream board capable of operating as a master SPI device.

Although this protocol can be created "from scratch" within firmware, it is recommended that an appropriate code library or a microcontroller with an inbuilt SPI module is used.

The SPI E-Block contains 2 SPI-compatible devices:

- Digital-to-analogue converter (DAC)
- Non-volatile FRAM memory (NVM)

These are accessed using specific SPI commands that are detailed in their respective datasheets.

Implementing a strategy

The following strategy is specific for a PICmicro microcontroller, but should be adaptable to any upstream device.

Initialisation

Initialisation has 2 parts:

- 1) Initialise the SPI module in the microcontroller

The SPI protocol can be used in a variety of "modes", so make sure the mode selected is appropriate for the SPI device. Also, make sure that the SPI clock rate is set appropriately. In a typical PICmicro microcontroller, the "SSP" module implements the SPI protocol. The following registers are used to initialise the SPI module:

- SSPCON1
- SSPSTAT

- 2) Initialise the i/o lines appropriately

The SDI, SDO and SCK pins in the appropriate TRIS register must be set to the correct direction (SDI must be an input, the others must be outputs). Also, the 2 i/o pins connected to the chip enable inputs of the SPI board must be set as outputs - and set these outputs initially to logic '1' (i.e. high).

Writing a value to the DAC

The DAC is the simpler device - essentially, the only command available is one to set the value of its analogue output. This is achieved by sending 2 bytes of information along the SPI bus. See the DAC device datasheet for the format of this command.

On a PICmicro, data is sent on the SPI bus by writing to the SSPBUF register.

- 1) Set the chip enable line of the DAC to logic '0' (low)
- 2) Load the SSPBUF with the first byte

- 3) Wait for about 3 microseconds
- 4) Load the SSPBUF with the second byte
- 5) Wait for another 3 microseconds
- 6) Bring the DAC chip enable line high ('1')

The 3 microsecond delay is approximately long enough for the SPI command to be sent, and this depends on what rate the SPI board has been set to. A better way is probably to poll the BF bit in the SSPBUF register (wait until it is set) as this will indicate exactly when the SPI command has been sent.

Writing a value to the FRAM NVM chip

The FRAM chip is a little more complicated. Here's the basic strategy:

- 1) Clear the NVM enable pin (i.e. set it low)
- 2) Send the "write enable" command
- 3) Set the NVM enable pin (i.e. set it high)
- 4) Clear the NVM enable pin
- 5) Send the "write" command
- 6) Send the high byte of the address
- 7) Send the low byte of the address
- 8) Send the data byte
- 9) Set the NVM enable pin

Each of steps 2, 5, 6, 7 and 8 involve the following 2 steps:

- (a) Load the SSPBUF with the appropriate data
- (b) Wait for the data to be sent by polling the BF bit of the SSPSTAT register

Reading a value from the FRAM NVM chip

- 1) Clear the NVM enable pin
- 2) Send the "read" command
- 3) Send the high byte of the address
- 4) Send the low byte of the address
- 5) Read the data byte
- 6) Set the NVM enable pin

Again, steps 2, 3 and 4 involve loading the SSPBUF register with the appropriate data and then waiting for that data to be sent. Step 5 is a little more complicated and can be broken down to the following:

- (a) Load the SSPBUF with a dummy value
- (b) Wait for the data to be sent by polling the BF bit of the SSPSTAT register
- (c) Read the SSPBUF - it will now contain data sent back by the FRAM chip

Alternative strategies

The SSP module on a PICmicro can also be configured to provide an interrupt to indicate when SPI transmission/reception has completed.

Further reading

More information on the specific SPI commands, including timing diagrams, can be found in the DAC and NVM datasheets.

If using a microcontroller with an in-built SPI module, see the relevant section in that microcontroller's datasheet for specific information regarding that module and how to program it.