Serial Peripheral Interface, SPI

Note Supplement - Brian Willis 03/20/2018

- This document supplements Professor Todd Morton's SPI notes, *SPI9S12.pdf*
  All information presented in this document uses *SPI9S12.pdf* as a starting point. SPI fundamentals and examples for the Freescale S12 University Board can be found in *SPI9S12.pdf*.
- This document covers SPI initialization for the Freescale K65 Tower Board with example use for SPI power driver IC present on the S12 board (Figure 1).

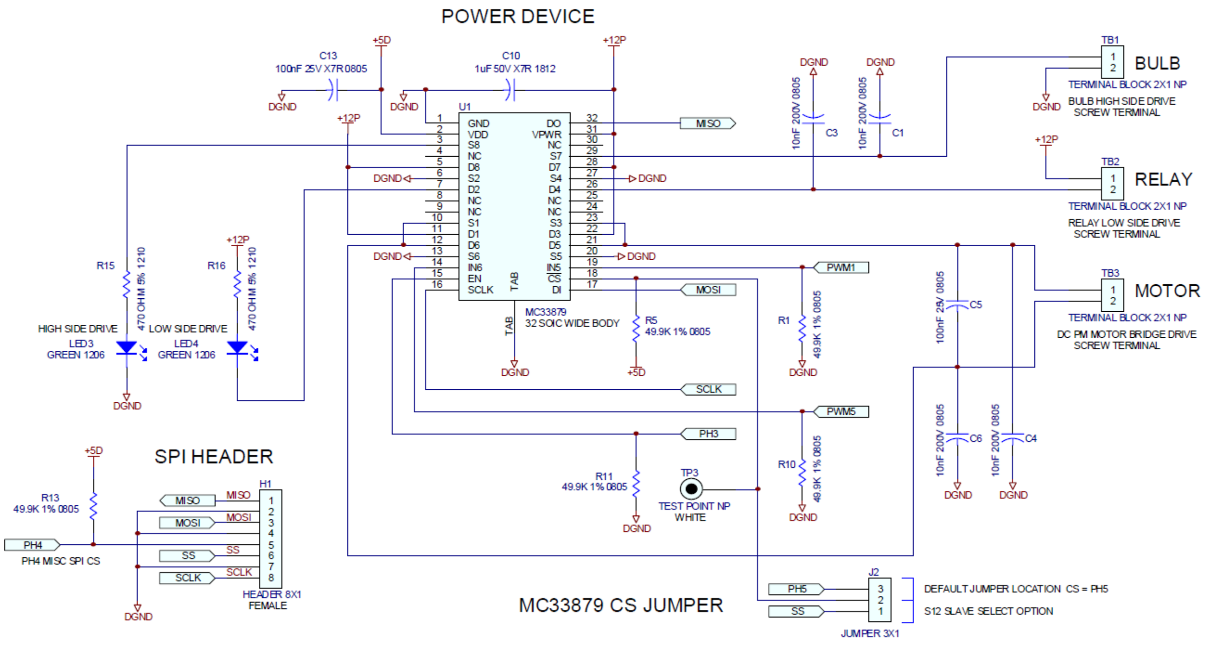

Figure 1 - SPI Power Driver IC on Freescale S12 University Board

K65 SPI Driver Code Example:

- This example demonstrates an initialization and data transfer routine with the K65 board as the master and the S12 board’s power driver IC as the one slave.
- SPI Configuration:

```
/*****************************************************************************
* SPIInit() – Initializes SPI1
* Baud rate: ~3.6MHz, Transfer size: 16 bits
* MCU: Freescale K65
*****************************************************************************/
void SPIInit(void){
    SIM_SCGC6 |= SIM_SCGC6_SPI1_MASK;         //Turn on SPI1 clock
    SIM_SCGC5 |= SIM_SCGC5_PORTE_MASK;        //Turn on PORTE clock

    PORTE_PCR2 = PORT_PCR_MUX(2);             //SCK:  B7  - PTE2
    PORTE_PCR4 = PORT_PCR_MUX(2);             //SS:   B9  - PTE4
    PORTE_PCR1 = PORT_PCR_MUX(2);             //MOSI: B11 - PTE1
    PORTE_PCR3 = PORT_PCR_MUX(2);             //MISO: B10 - PTE3

    //Set prescalar to 2 and scalar to 8, achieves baud rate of ~3.6MHz for a
    //protocol clock of ~60MHz
    SPI1_CTAR0 |= SPI_CTAR_PBR(0);
    SPI1_CTAR0 |= SPI_CTAR_BR(3);
    SPI1_CTAR0 |= SPI_CTAR_FMSZ(15);          //Set transfer size to 16 bits

    SPI1_MCR &= SPI_MCR_HALT(0);              //Disable halt mode
    SPI1_MCR |= SPI_MCR_MSTR(1);              //Enable Master Mode
    SPI1_MCR |= SPI_MCR_PCSIS(1);             //Set SS inactive state to 1

    //Dummy transmission with SS 1 to set Transfer Complete Flag
    SPI1_PUSHR = SPI_PUSHR_TXDATA(0x0000) | SPI_PUSHR_PCS(1);
}
```

- SPI Transfer:

```
/*****************************************************************************
* SPITransfer(INT16U data) – Transfers 16-bit data to SPI IC
*
* MCU: Freescale K65
*****************************************************************************/
void SPITransfer(INT16U data){
     while((SPI1_SR & SPI_SR_TCF_MASK) == 0){} //Wait for previous transmission
     SPI1_SR |= SPI_SR_TCF(1);                 //Reset Transfer Complete Flag

    //Push data to transmit with SS 1
     SPI1_PUSHR = SPI_PUSHR_TXDATA(data) | SPI_PUSHR_PCS(1);
}
```

- Driver Code Notes:
  - Previous transmission's Transfer Complete Flag must be pended on before making another transmission
  - TX FIFO register SPIx_PUSHR usage:
    - Master to slave data (MOSI) is specified
    - Slave Select (SS) is specified
    - Must be written to entirely on one line or multiple transfers execute
  - S12 board's power driver IC SS requirements:
    - SS goes low for duration of transfer
    - Data is transferred to IC only when SS is low
    - Data is latched to IC outputs on SS rising edge
  - Slave Select is often referred to as Chip Select (CS) in documents
  - Data from slave to master (MISO) is transferred every rising clock edge
- Figure 2 displays waveform capture of 16-bit transmission (0x0064) at ~3.6MHz
  - CLK: Signal 1
  - MOSI: Signal 2
  - SS: Signal 3

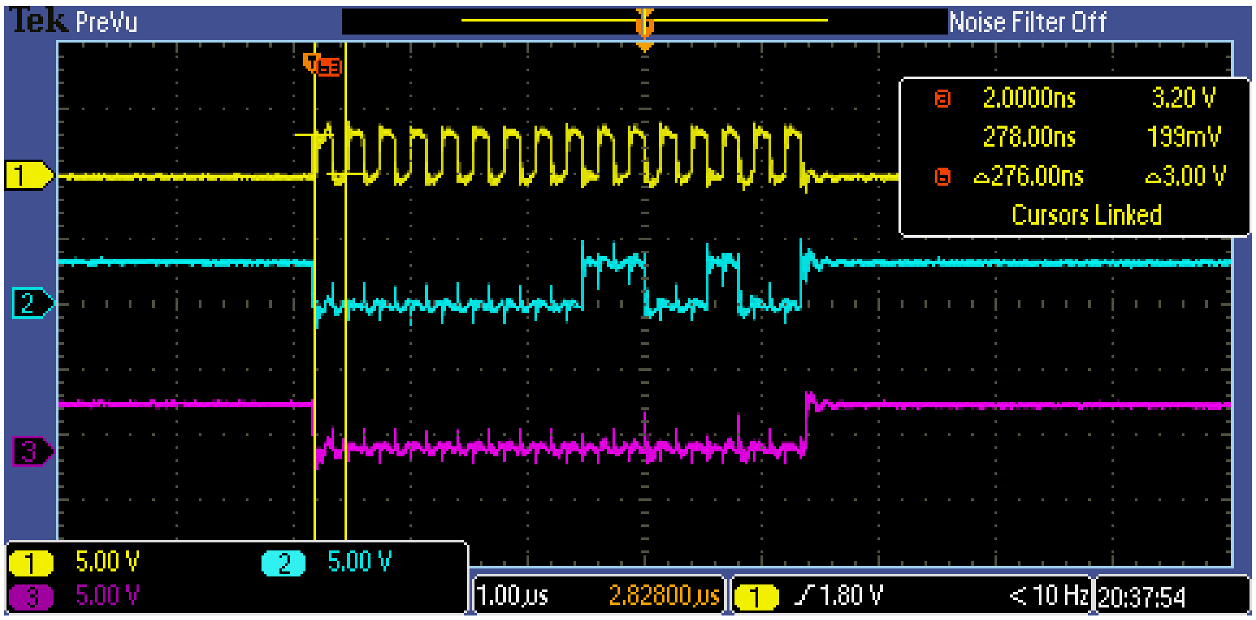

Figure 2 - Waveform Capture of 16-bit SPI Transmission