Serial Peripheral Interface (SPI)

- Synchronous serial data transfers
  - Multipoint serial communication between a "master" and a "slave" device
  - Clock permits faster data rates than async communications *(framing unnecessary)*
  - Signals = clock, data in/out, "slave select"
  - **Master** controls data transfers:
    - transmit a synchronization clock
    - activate slave select signal
  - All device data registers effectively linked into a single “shift register”
Single master, single-slave SPI connections

SCLK – serial clock, generated by the master
MOSI – master output/slave input
MISO – master input/slave output
SS – slave select/enable signal
Single master, multiple slave
SPI implementation

Slave-selects
Single master, multiple slave SPI implementation – daisy chained

- Dout of one device connected to Din of next (creates a single shift register)
- All devices selected concurrently by the Master
SPI serial data timing

- Programmable clock rate and timing for flexibility
  - CPOL = clock polarity (0=active-high, 1=active-low)
  - CPHA = clock phase (sample on leading/trailing pulse edge)

CPHA = 0: data output immediately when PCSn active data sampled on leading edge

CPOL is the “idle” state
SPI serial data timing

- CPHA=1
  - data output on 1st clock edge after PCSn active
  - data sampled on trailing edge
Maxim MAX5154
12-Bit Serial DAC

Data Output
DOUT, CL, PDL
DGND, AGND, VDD, REFA

Data Input
CS, DIN, SCLK, UPO

Chip-Select

Serial Clock

OSA
OUTA, OSB
OUTB

REFB
MAX5154 serial data format

Command and address bits select channel and conversion properties.
VTI Technologies

SCP1000 Pressure Sensor

Supports SPI or I2C (factory programmed)
19 bit pressure, 14 bit temperature
Analog Devices
ADIS16003 Dual-axis accelerometer

12-bit acceleration/10-bit temperature
ADIS16003 Dual-axis accelerometer
SPI interface timing

Figure 3. Accelerometer Serial Interface Timing Diagram

Figure 4. Temperature Serial Interface Timing Diagram
SST: SST25VF016B
16 Mbit SPI serial flash memory

- 50 MHz clock
- 8-lead SOIC package
- 7 us byte program
- 18 ms block erase

Use in DVDs, hard drives, PCs, WLANs, LCD monitors, MP3 players, FPGAs, etc.

- 4-wire SPI interface
- Write protection
- Suspend (hold) serial transfer
**TABLE 5: DEVICE OPERATION INSTRUCTIONS**

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Description</th>
<th>Op Code Cycle</th>
<th>Address Cycle(s)</th>
<th>Dummy Cycle(s)</th>
<th>Data Cycle(s)</th>
<th>Maximum Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read</td>
<td>Read Memory at 25 MHz</td>
<td>0000 0011b (03H)</td>
<td>3</td>
<td>0</td>
<td>1 to ∞</td>
<td>25 MHz</td>
</tr>
<tr>
<td>High-Speed Read</td>
<td>Read Memory at 50 MHz</td>
<td>0000 1011b (0BH)</td>
<td>3</td>
<td>1</td>
<td>1 to ∞</td>
<td>50 MHz</td>
</tr>
<tr>
<td>4 KByte Sector-Erase³</td>
<td>Erase 4 KByte of memory array</td>
<td>0010 0000b (20H)</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>50 MHz</td>
</tr>
<tr>
<td>32 KByte Block-Erase⁴</td>
<td>Erase 32 KByte block of memory array</td>
<td>0101 0010b (52H)</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>50 MHz</td>
</tr>
<tr>
<td>64 KByte Block-Erase⁵</td>
<td>Erase 64 KByte block of memory array</td>
<td>1101 1000b (D8H)</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>50 MHz</td>
</tr>
<tr>
<td>Chip-Erase</td>
<td>Erase Full Memory Array</td>
<td>0110 0000b (60H) or 1100 0111b (C7H)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>50 MHz</td>
</tr>
<tr>
<td>Byte-Program</td>
<td>To Program One Data Byte</td>
<td>0000 0010b (02H)</td>
<td>3</td>
<td>0</td>
<td>1</td>
<td>50 MHz</td>
</tr>
<tr>
<td>AAI-Word-Program⁶</td>
<td>Auto Address Increment Programming</td>
<td>1010 1101b (ADH)</td>
<td>3</td>
<td>0</td>
<td>2 to ∞</td>
<td>50 MHz</td>
</tr>
<tr>
<td>RDSR⁷</td>
<td>Read-Status-Register</td>
<td>0000 0110b (05H)</td>
<td>0</td>
<td>0</td>
<td>1 to ∞</td>
<td>50 MHz</td>
</tr>
<tr>
<td>EWSR</td>
<td>Enable-Write-Status-Register</td>
<td>0101b 0000b (50H)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>50 MHz</td>
</tr>
<tr>
<td>WRSR</td>
<td>Write-Status-Register</td>
<td>0000 0011b (01H)</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>50 MHz</td>
</tr>
<tr>
<td>WREN</td>
<td>Write-Enable</td>
<td>0000 0110b (06H)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>50 MHz</td>
</tr>
<tr>
<td>WRDI</td>
<td>Write-Disable</td>
<td>0000 0100b (04H)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>50 MHz</td>
</tr>
<tr>
<td>RDID⁸</td>
<td>Read-ID</td>
<td>1001 0000b (50H) or 1010 1011b (ABH)</td>
<td>3</td>
<td>0</td>
<td>1 to ∞</td>
<td>50 MHz</td>
</tr>
<tr>
<td>JEDEC-ID</td>
<td>JEDEC ID read</td>
<td>1001 1111b (5FH)</td>
<td>0</td>
<td>0</td>
<td>3 to ∞</td>
<td>50 MHz</td>
</tr>
<tr>
<td>EBSY</td>
<td>Enable SO to output RY/BY# status during AAI programming</td>
<td>0111 0000b (70H)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>50 MHz</td>
</tr>
<tr>
<td>DBSY</td>
<td>Disable SO to output RY/BY# status during AAI programming</td>
<td>1000 0000b (80H)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>50 MHz</td>
</tr>
</tbody>
</table>
Motorola MC14499 7-segment LED display decoder/driver with SPI

- 7-segment alphanumeric LED decoder/driver
- drives 4 characters with decimal points
- NPN output drivers for common-cathode LEDs
STM32 Serial Peripheral Interface (SPI)

- Dual function: **SPI** *(default)* or **I²S**
- Synchronous, serial, full-duplex communication
- Configurable as SPI master or slave
- Programmable clock polarity/phase
- Programmable baud rate
- Supports busy-wait, interrupt, and DMA I/O
STM32 SPI block diagram
SPI data register (SPI\_DR)

Transmit buffer for writing / Receive buffer for reading

SPI 8-bit data frame format (DFF = 0):
   DR[7:0] = data;  DR[15:8] = 00000000

SPI 16-bit data frame format (DFF = 1):
   DR[15:0] = data
### SPI control register 1 (SPI_CR1)

<table>
<thead>
<tr>
<th>Bit 15</th>
<th>Bit 14</th>
<th>Bit 13</th>
<th>Bit 12</th>
<th>Bit 11</th>
<th>Bit 10</th>
<th>Bit 9</th>
<th>Bit 8</th>
<th>Bit 7</th>
<th>Bit 6</th>
<th>Bit 5</th>
<th>Bit 4</th>
<th>Bit 3</th>
<th>Bit 2</th>
<th>Bit 1</th>
<th>Bit 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIDI MODE</td>
<td>BIDIOE</td>
<td>CRC EN</td>
<td>CRC NEXT</td>
<td>DFF</td>
<td>RX ONLY</td>
<td>SSM</td>
<td>SSI</td>
<td>LSB FIRST</td>
<td>SPE</td>
<td>BR [2:0]</td>
<td>MSTR</td>
<td>CPOL</td>
<td>CPHA</td>
<td></td>
<td></td>
</tr>
<tr>
<td>rw</td>
<td>rw</td>
<td>rw</td>
<td>rw</td>
<td>rw</td>
<td>rw</td>
<td>rw</td>
<td>rw</td>
<td>rw</td>
<td>rw</td>
<td>rw</td>
<td>rw</td>
<td>rw</td>
<td>rw</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**BIDI MODE:** 0 = 2-line/unidirectional, 1 = 1-line/bidirectional  
**BIDIOE:** bidirectional mode output enable (0 = receive, 1 = xmit)  
**CRCEN:** hardware CRC calculation enable  
**CRCNEXT:** 1 = next xfer is data (no CRC), 0 = next xfer is CRC  
**DFF:** data frame format (0 = 8-bit, 1 = 16-bit)  
**RXONLY:** receive only (0 = full duplex, 1 = output disabled/receive-only)  
**SSM:** software slave management – NSS pin ignored (1 = enable)  
**SSI:** internal slave select (this bit forced onto NSS pin if output enabled: SSOE)  
**LSBFIRST:** frame format (0 = shift out MSB first, 1 = shift out LSB first)  
**SPE:** SPI enable  
**BR[2:0]** – baud rate control (master)  
\[\text{Fbaud} = \frac{\text{Fpclk}}{2^{(\text{BR}+1)}}\]  
**MSTR:** master selection (0 = slave, 1 = master)  
**CPOL:** clock polarity (idle value)  
**CPHA:** clock phase (0 = 1\textsuperscript{st} clk transition to capture data, 1 = 2\textsuperscript{nd})
### SPI control register 2 (SPI_CR2)

<table>
<thead>
<tr>
<th>Bit 8</th>
<th>Bit 7</th>
<th>Bit 6</th>
<th>Bit 5</th>
<th>Bit 4</th>
<th>Bit 3</th>
<th>Bit 2</th>
<th>Bit 1</th>
<th>Bit 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>TXEIE</td>
<td>RXNEIE</td>
<td>ERRIE</td>
<td>FRF</td>
<td>Res.</td>
<td>SSOE</td>
<td>TXDMAEN</td>
<td>RXDMAEN</td>
<td></td>
</tr>
<tr>
<td>rw</td>
<td>rw</td>
<td>rw</td>
<td>rw</td>
<td></td>
<td>rw</td>
<td>rw</td>
<td>rw</td>
<td></td>
</tr>
</tbody>
</table>

**TXEIE**: Tx buffer empty interrupt enable (on TXE flag set)

**RXNEIE**: Rx buffer not empty interrupt enable (on RXNE flag set)

**ERRIE**: error interrupt enable (CRCERR, OVR, MODF in SPI mode)

**FRF**: frame format (0 = Motorola mode, 1 = TI mode)

**SSOE**: SS output enable (if in Master mode)

**TXDMAEN**: Tx buffer DMA enable (DMA request when TXE flag set)

**RXDMAEN**: Rx buffer DMA enable (DMA request when RXNE flag set)

DMA automatically xfers data between memory and SPI_DR
SPI status register (SPI_SR)

<table>
<thead>
<tr>
<th></th>
<th>9</th>
<th>8</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>FRE</td>
<td>BSY</td>
<td>OVR</td>
<td>MODF</td>
<td>CRCERR</td>
<td>UDR</td>
<td>CHSIDE</td>
<td>TXE</td>
<td>RXNE</td>
<td></td>
</tr>
<tr>
<td>r</td>
<td>r</td>
<td>r</td>
<td>r</td>
<td>r</td>
<td>rc_w0</td>
<td>r</td>
<td>r</td>
<td>r</td>
<td>r</td>
<td></td>
</tr>
</tbody>
</table>

**FRE**: frame format error (for SPI TI slave mode or I2S slave mode)
**BSY**: SPI/I2S busy communicating (set/cleared by hardware)
**OVR**: overrun error – master sends before RXNE cleared by slave
**MODF**: master mode fault – master NSS pin lulled low (SPI only)
**CRCERR**: CRC error in received value (SPI only)
**UDR**: underrun error (I2S only) 1st clock before data in DR
**CHSIDE**: channel side to xmit/has been received (0 = left/1 = right) (I2S only)
**TXE**: 1 = Tx buffer empty: can load next data to buffer; clears on DR write
**RXNE**: 1 = Rx buffer not empty: valid received data in buffer; clears on DR read

- Use TXE/RXNE rather than BSY for each transmission.
- Trigger SPI interrupts with TXE, RXNE, MODF, OVR, CRCERR, FRE
Master Operation Setup

MOSI pin = data output; MISO pin = data input.

SPI_CR1:
1. Select BR[2:0] bits to define the serial clock baud rate
2. Select CPOL and CPHA bits to define one of the four relationships between the data transfer and the serial clock.
3. Select DFF bit to define 8- or 16-bit data frame format
4. Select LSBFIRST bit to define the frame format (MSB or LSB first).
5. Set MSTR and SPE bits.
6. If the NSS pin is required in input mode, in hardware mode, connect the NSS pin to a high-level signal during the complete byte transmit sequence. In NSS software mode, set the SSM and SSI bits in the SPI_CR1 register. If the NSS pin is required in output mode, the SSOE bit only should be set.
7. Select FRF bit in SPI_CR2 to select the Motorola or TI SPI protocol.