Experiment 10: Serial Communication

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME
141	Nov. 23, 2014		H

Contents

1 Objectives

- Introduction to serial communication protocols
- Using the *Serial Peripheral Interface* (SPI) protocol to perform serial communication

2 Parts List

- LPC1769 LPCXpresso board
- USB A-Type to Mini-B cable
- LPCXpresso Base Board (User's Manual)

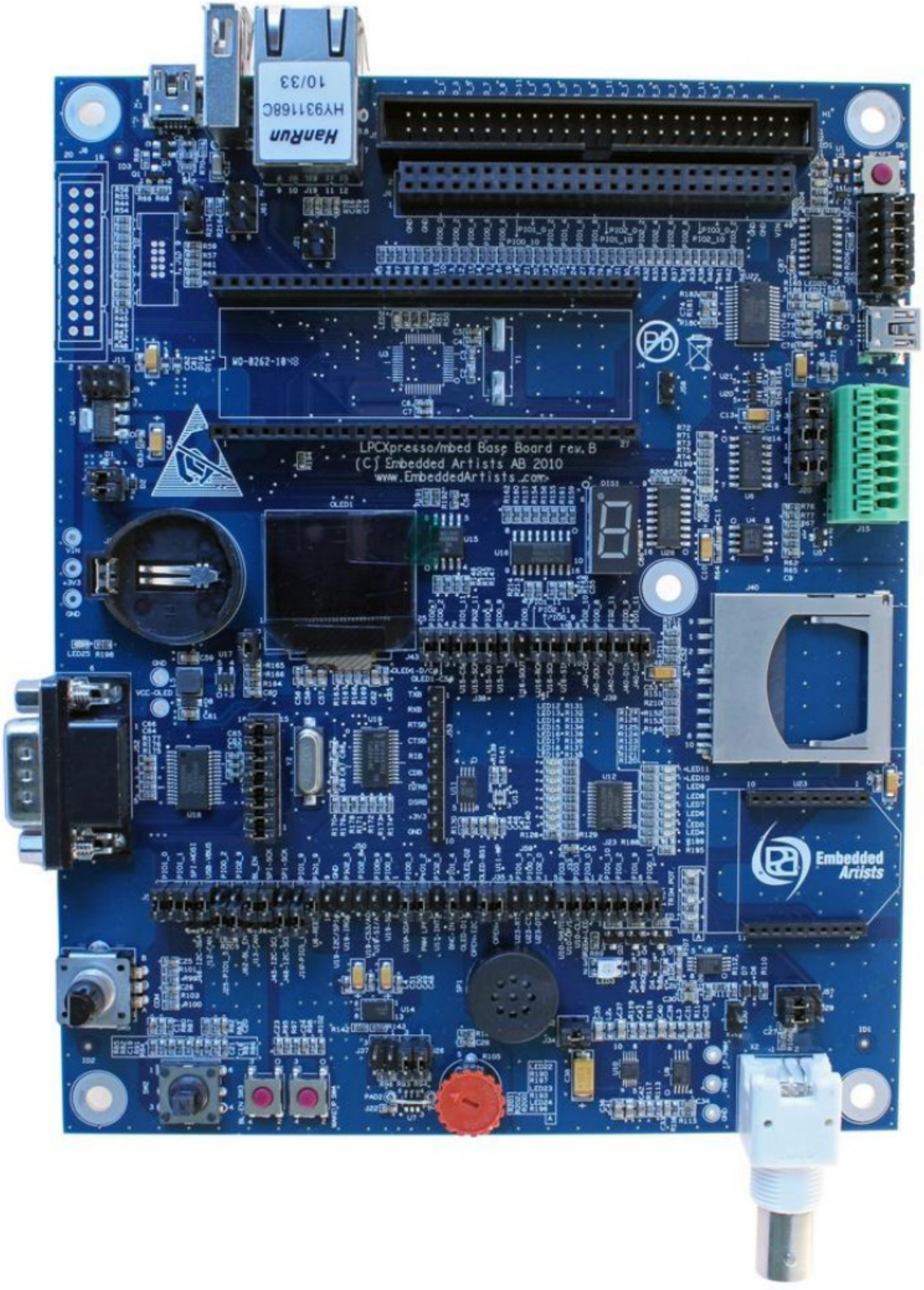

Figure 1: LPCXpresso Base Board

Note
You can conduct this experiment without the LPCXpresso Base Board, but you would then need the following parts instead:

- Breadboard
- Shift register IC (74595)
- Seven-segment display
- Jumper wires

3 Background

3.1 Serial vs. Parallel Communication

Serial communication is the process of sending data one bit at a time, sequentially. In contrast, parallel communication involves multiple bits at the same time, as illustrated in the Parallel vs. Serial Communication figure below.

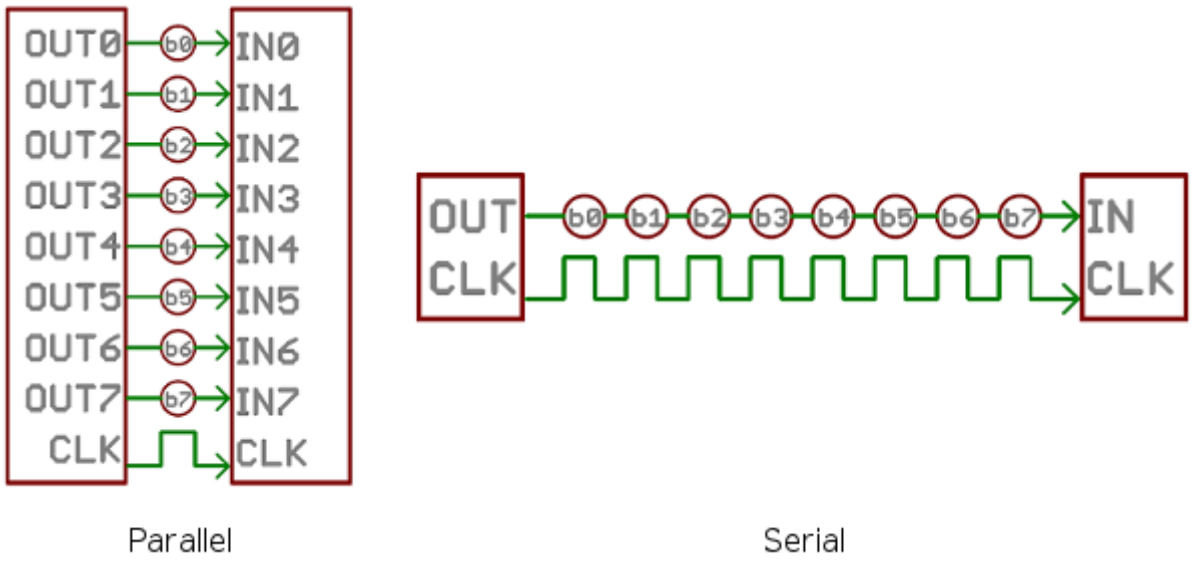

Figure 2: Parallel vs. Serial Communication

Some of the main differences between serial and parallel communication are:

- A parallel link requires more wires, occupying more space and resulting in higher cost.
- To keep all wires in a parallel link synchronized, the link rate is limited. In contrast, serial links can sustain much higher clock rates.
- Parallel links are more susceptible to crosstalk interference.
- Parallel communication between ICs require more pins, increasing the IC cost.
- Parallel communication is easier to implement because it does not require data serialization and deserialization.

Serial communication is becoming more common for transmitting data between a computer and a peripheral device or even another computer, as improved signal integrity and transmission speeds in newer serial technologies have begun to outweigh the parallel bus's advantages.

3.2 Serial Communication Protocols

Example serial communication standards include USB, FireWire, Serial ATA (SATA), PCI Express (PCIe), Ethernet. Serial protocols commonly used in embedded systems include RS-232, I^2C, and SPI.

Serial communication protocols can be synchronous or asynchronous. An asynchronous protocol sends a *start signal* prior to each code word, and a *stop signal* after each code word. RS-232 is an asynchronous serial protocol supported by UART interfaces.

A synchronous serial protocol sends a *clock* signal on a dedicated wire. Additional wire(s) are required for data. I^2C and SPI are synchronous serial protocols.

3.3 LPC1769 Serial Interfaces

The LPC1769 microcontroller provides the following serial interfaces (LPC1769 Manual):

- Two Synchronous Serial Port (SSP) controllers with multi-protocol capabilities. They can operate as SPI, 4-wire TI SSI, or Microwire bus
- SPI controller. SSP0 is intended to be used as an alternative for the SPI interface. SPI is included as a legacy peripheral.
- Three enhanced I^2C-bus interfaces, one supporting the full I^2C specification, and two with standard port pins.
- Four UARTs.
- Two-channel CAN controller.
- Ethernet MAC with RMII interface and dedicated DMA controller.
- USB 2.0 full-speed controller that can be configured for either device, host, or OTG operation with an on-chip PHY for device and Host functions and a dedicated DMA controller.

In this experiment, we will use the SSP interface using the SPI protocol.

3.4 Serial Peripheral Interface (SPI)

SPI is a four-wire, full-duplex, master-slave bus that was created by Motorola. There can be only a single master. Multiple slaves are allowed with individual *slave select* (`SS` or `SSEL`) lines. The four wires are:

1. `SCLK`: Serial Clock (output from master)
2. `MOSI`: Master Output, Slave Input (output from master)
3. `MISO`: Master Input, Slave Output (output from slave)
4. `SSEL`: Slave Select (active low, output from master)

The microcontroller is usually the master. It uses the `MOSI` pin to send data, and the `MISO` pin to read data. The `SCLK` pin dictates the transmission rate; a bit is sent/received every clock pulse. A simple timing diagram for writing data is shown below.

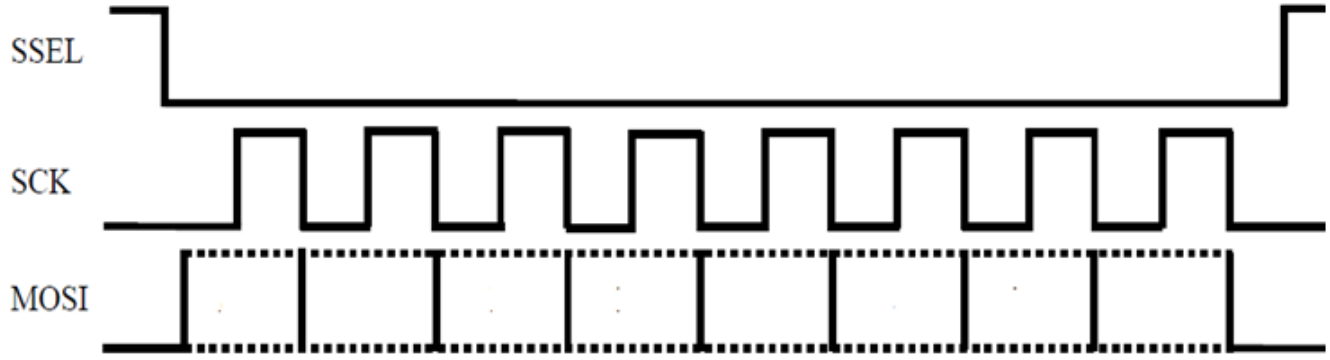

Figure 3: Timing diagram for writing data on a SPI bus

The *slave select* (`SSEL`) signal is used to select the slave in a data transfer. `SSEL` is active low: it must be low before the transaction begins, and must stay low for the duration of the transaction.

Even though the `SSEL` signal is a part of the SPI protocol, it is not uncommon to leave its control to the software instead of the SPI/SSP controller. The LPC176x manual states that "This signal is not directly driven by the master. It could be driven by a simple general purpose I/O under software control." In the LPCXpresso Base Board, `SSEL` is connected to GPIO P2.2. It should be driven low (by software) prior to placing data in the *Data Register* (`DR`), and then switched back to high.

3.5 Using SSP/SPI in LPC1769

The section describes how to use the SSP interface of the LPC1769 microcontroller as an SPI interface by listing the involved registers and their functions.

3.5.1 Data Register (DR)

The data to be sent serially must be loaded into the SSP *Data Register* ('LPC_SSP1→DR'). The serial transfer rate is controlled by the SSP clock as described below.

3.5.2 SSP Control Registers

There are two control registers for the `SSP1` interface (see `LPC17xx.h`):

1. `SSP1CR0`: can be accessed as `LPC_SSP1->CR0`
2. `SSP1CR1`: can be accessed as `LPC_SSP1->CR1`

`CR0` has 5 fields:

1. Data size (bits 0-3): the number of bits transferred in each frame.
2. Frame Format (bits 4-5): the serial protocol to be used.

00	SPI
01	TI
10	Microwire
11	Not supported

3. Clock Out Polarity (bit 6): should be 0 in our application.
4. Clock Out Phase (bit 7): should be 0 in our application.
5. Serial Clock Rate (`SCR`) (bits 8-15): used with the *Clock Prescale Register* (`CPR`) to control the SSP clock. This is crucial when the SSP peripheral requires a specific value or range of frequencies.

Note
The minimum value for `CPR` is 2. For details, see Table 375 in the LPC176x manual.

The `CR1` register has 4 fields, the most crucial of which is bit 1: *SSP enable*.

Note
For details, see Table 371 and Table 372 in the LPC176x manual.

3.6 Shift Register (74595)

The 74595 IC is a buffered shift register: the parallel outputs do not change during the serial loading process. It contains two registers; a shifting register and an output register. The shifting register is SPI-compatible: it receives data serially, bit by bit, through the input pin (pin 14) by clocking the shifting clock pin (pin 11).

Tip
Consult the 74595 shift register data sheet.

The data in the shift register are not immediately available as output. The `Q` output pins are connected to the second register: the output (or storage) register. The data are moved from the shift register to the output register by setting a select pin high. In some data sheets, this pin is called *storage register clock*. This pin should be low during shifting, and should be set high to indicate that a full byte has been received to pass it to the output register.

In the LPCXpresso Base Board, a 74595 shift register is used to connect the seven-segment display to the microcontroller in order to reduce the number microcontroller pins used by the connection. The serial interface of the shift register is connected to the SPI interface of the microcontroller, whereas the parallel interface of the shift register is connected to the seven-segment display pins.

4 Tasks

Use the LPC1769's SSP/SPI interface to operate a seven-segment display through a shift register.

Tip
Use can use the seven-segment display decoder library you have made in Experiment 6 to generate the seven-segment display bit patterns, which are to be sent to the shift register through the SSP/SPI interface.

If you are using the LPCXpresso Base Board, then the connections are already in place. If you are using a discrete 74595 shift register IC and a seven-segment display, you will need to connect them properly on a bread board.

5 Resources

[base-board-manual]
Embedded Artists AB. *LPCXpresso Base Board Rev B User's Guide*. 2013-01-25.
http://www.embeddedartists.com/sites/default/files/support/xpr/base/LPCXpresso_BaseBoard_rev_B_Users_Guide.pdf

[lpc1769-manual]
NXP Semiconductors. *UM10360 LPC176x/5x User manual*. Rev. 3.1. 2 April 2014.
http://www.nxp.com/documents/user_manual/UM10360.pdf