1300 Henley Court
Pullman, WA 99163
509.334.6306
www.store.digilentinc.com

Unit 4 Part 1: Communications – Serial Protocol

Revised May 23, 2017
This manual applies to Unit 4 Part 1

1 Introduction

"Telecommunication is the transmission of signs, signals, writings, images and sounds or intelligence of any nature by wire, radio, optical, or other electromagnetic systems, as defined by the International Telecommunication Union (ITU). Telecommunication occurs when the exchange of information between communication participants includes the use of technology. It is transmitted either electrically over physical media, such as cables, or via electromagnetic radiation. Such transmission paths are often divided into communication channels which afford the advantages of multiplexing. The term is often used in its plural form, telecommunications, because it involves many different technologies."[1]

Telecommunications can be divided into two basic types: digital communications and analog communications. Common residential telephone systems as well as AM and FM radio are forms of analog communications. The focus of this unit is digital communication where the information is communicated as encoded discrete level signals. The digital signals can represent computer generated values or be the result of digitization of analog signals.

Serial communications involves sending data one bit at a time. This is opposed to parallel communications where multiple bits of data are sent simultaneously, such as the interface with the character LCD that was the focus of Unit 3. One of the biggest motivations for implementing serial communications is to minimize the number of processor pins and wires needed to pass data between two points. The longer the physical distance between these two points (the endpoints), the more expensive the communications media becomes, whether that is wireless or wired based.

Unit 4 consists two parts and four labs. Unit 4 Part 1 contains the following overview that addresses different high level views of digital communications. Unit 4 Part 2 addresses asynchronous communications protocols, specifically the Universal Asynchronous Receiver Transmitter (UART) protocol, and applies specifically to Lab 4a and Lab 4b. Lab 4a focuses on full-duplex text communications and Lab 4b uses the communications for control of a real-time operating system. Lab 4c and Lab 4d look at two commonly used synchronous data communications protocols. Lab 4c targets the Serial Peripheral Interface (SPI) protocol while Lab 4d focuses on the Inter-Integrated Circuit (I^2C) protocol.

[1] Telecommunication, https://en.wikipedia.org/wiki/Telecommunication

2 Objectives

1. To understand how communications networks are modeled to partition the various operations and responsibilities.
2. To understand two fundamental data flow management schemes.
3. To understand the common methods of implementing basic sender-receiver handshaking.

3 Basic Knowledge

1. How to map PIC32MX370 I/O pins to special function registers.
2. How to initialize special function using the PLIB functions.
3. How to generate ASCII character strings.
4. How to parse text fields from an ASCII string.

4 Equipment List

4.1 Hardware

1. Basys MX3 trainer board
2. 2 Micro USB cables
3. Workstation computer running Windows 10 or higher, MAC OS, or Linux
4. 4-wire stepper motor (Lab 4b only)
5. 5V, 4A DC power supply (Lab 4b only)

In addition, we suggest the following instruments:

6. Digilent Analog Discovery 2

4.2 Software

The following programs must be installed on your development work station:

1. Microchip MPLAB X® v3.35 or higher
2. PLIB Peripheral Library
3. XC32 Cross Compiler
4. WaveForms 2015
5. PuTTY Terminal Emulator

5 Project Takeaways

1. Understanding of the basics of telecommunications.
2. Understanding of requirements and implementations of asynchronous and synchronous communications.
3. How to recognize the handshaking methods used in a communications protocol.
4. How to set up the Analog Discovery 2 to display communication signal waveforms.

6 Fundamental Concepts

6.1 Digital Communications

Serial protocols fall into one of two categories: either asynchronous or synchronous. Asynchronous protocols do not transmit a synchronizing signal with the data, whereas a synchronous protocol transmits a signal that indicates to the receiver when the data signal should be sampled. In some protocols, the transmitter and receiver are synchronized by a completely separate signal while other synchronous protocols (such as CAN) embed the clock signal in the data stream.

A partial list of existing serial protocols is shown in Appendix B. You may ask, "why so many serial protocols?" The simple answer is that the protocols are optimized for data rates, physical media, physical distance, and connectivity. Some protocols are strictly for point to point communications such as the asynchronous serial communications. Other protocols are for multi drop applications where many senders can be connected to many receivers – a network is an example. In contrast, point-to-point communications is limited to be exclusively between two endpoints, or nodes, and is not an example of a network.

In the four labs associated with Unit 4, we will look at the most common serial protocols used in the world of computer communications. These include the Universal Asynchronous Receiver Transmitter (UART or USART), the Inter-Integrated Circuit (I^2C), and the Serial Peripheral Interface (SPI) protocol.

As we saw in Unit 3, the communications between the PIC32 processor and the character LCD device required an application of a set of rules that resulted in handshaking. These rules are referred to as the communications protocol, whether the data being exchanged is 8 bits at a time as with the LCD or a single bit at a time as with the serial protocols that will be discussed throughout Unit 4. The basic rules of handshaking apply in either case.

6.2 Network Communications

Fundamentally, a communications network is the connection of multiple devices to exchange information. A network system can consist of one device talking to many listeners (one-to-many) or many devices talking to one listener (many-to-one). A many-to-one system is characteristic of master-slave operations. A one-to-many is characteristic of a broadcast based network. Peer-to-peer networks have many devices talking and listening, making a many-to-many network system.

Networks are characterized by multiple devices connected to a shared data communication resource. Network communications can use the radio frequency (RF) spectrum, fiber optics (FO), electrical wires, and acoustic media such as air, water, or solid materials. Most often, network systems share a common resource – the media over which they communicate. If more than one sender exists on a network, there must be a mechanism to detect and arbitrate conflicts when two devices attempt to transmit at the same time. Master-slave networks with a single master do not require any form of arbitration.

It is common practice to model the processing of information as layers of software, each layer adding information or modifying the presentation of the information. Various network models have evolved over the years. The partitioning and activities represented by the seven layer Open Systems Interconnection (OSI) model are shown in Table 6.1. Network communications does not always require all seven layers of processing. Typically, embedded systems applications use layer 7 that represents a specific application, layer 3 that manages the device addressing, layer 2 that is used for error correction, and layer 1 that is responsible for controlling the electronics that modulate the communications media.

Table 6.1. Seven layer OSI model

OSI Model			
	Data unit	**Layer**	**Function**
Host layers	Data	7.Application	Network process to application
		6.Presentation	Data representation, encryption and decryption, convert machine dependent data to machine independent data
		5. Session	Inter-host communication, managing sessions between applications
	Segments	4. Transport	Reliable delivery of packets between points on a network.
Media layers	Packet/Datagram	3. Network	Addressing, routing and (not necessarily reliable) delivery of datagrams between points on a network.
	Bit/Frame	2. Data link	A reliable direct point-to-point data connection.
	Bit	1. Physical	A (not necessarily reliable) direct point-to-point data connection.

Looking more closely at layer 2 of the OSI model, there are two kinds of data link layer operating modes for multi-drop, or network, configurations; master – slave and peer-to-peer. The master-slave mode of operation is characterized by one processor, acting as the master, which controls when slave devices are permitted to transmit information. Although multiple processors can be connected to a network, only one microprocessor is designated as the master at any given time. The master processor dictates the data direction and the timing of the data exchange between the master and slave nodes.

In contrast to master-slave mode, Peer-to-peer communications allows data exchange at any time and between any two communications devices, or nodes. Further, communications can be initiated by any node at any time. An example of peer-to-peer is the PIC processor communicating with a PC running a terminal emulation application using full duplex UART communications.

The Master-slave operation is similar to the microprocessor-LCD communication considered in Unit 3. The microprocessor is the master and it dictates the data direction and the timing of the data exchange between the master and slave units. SPI and I2C are examples of master-slave networks supported directly by hardware in many microprocessors. I2C is a half-duplex scheme where the slave devices are enabled, or selected, by encoding data in a message sent by the master. SPI uses master controlled dedicated device select signals along with separate data send and receive signals to enable simultaneous communications (full-duplex) between the master and a specific slave device.

Layer 1 has the responsibility in the OSI model of controlling the timing of the communications in both master-slave and peer-to-peer systems. Depending on the communications electronics hardware, both master-slave and peer-to-peer configurations can be implemented using half or full duplex connections. The distinction between the two duplex operations is that for half duplex, nodes can both send and receive but not simultaneously, whereas full duplex allows simultaneous sending and receiving.

7 Background Information

The theory behind asynchronous and synchronous communication is essentially the same: Point B needs to know when a transmission from Point A begins, when it ends, and if it was processed correctly. However, the difference lies in how the transmission is broken down. Think of the difference in terms of a friendly chat. With asynchronous communication you would need to stop after every word to make sure the listener understood your meaning, and

knew that you were about to speak the next word. With synchronous communication, you would establish with your listener that you were speaking English, that you will be speaking words at measured intervals, and that you would utter a complete sentence, or paragraph, or extended soliloquy, before pausing to confirm understanding. Further, you would establish with your listener beforehand that any extraneous noises you make during the speech or between speeches (coughing, burping, and hiccupping) should be ignored. Clearly the second approach is much faster, even though initializing communication may take slightly longer. In fact, by replacing the start, stop, and parity bits around individual words with start, stop, and control (processing instructions and error checking) sequences around large continuous data blocks, synchronous communication is about 30% faster than asynchronous communication, before any other factors are considered.

Synchronous vs. asynchronous communications pertain to the individual symbol or bit timing and is controlled at layer 1 in the OSI model. Synchronous serial communication describes a serial communication protocol in which "data is sent in a continuous stream at a constant rate."[2]

Synchronous communication requires that the clocks in the transmitting and receiving devices are *synchronized* – running at the same rate – so the receiver can sample the signal at the same time intervals used by the transmitter. No start or stop bits are required. For this reason, "synchronous communication permits more information to be passed over a circuit per unit time" than asynchronous serial communication. Over time the transmitting and receiving clocks will tend to drift apart.

7.1 Asynchronous Digital Communications

In telecommunications, asynchronous communication is transmission of data, generally without the use of an external clock signal, where data can be transmitted intermittently rather than in a steady stream. Any timing required to recover data from the communication symbols is encoded within the symbols. One characteristic of asynchronous communications is that data is formed in units of transmission with well-defined size (number of bits) and symbol rate. Since over time the transmitting and receiving clocks will tend to drift apart resynchronization is required. The resynchronization is accomplished using a "START" symbol while the units of transmission are framed between a START and STOP symbol. The timing between units of transmission is arbitrary depending on the application requirements. Unit 4 Part 2 along with Lab 4a and Lab 4b explore asynchronous communication concepts and applications further.

7.2 Synchronous Digital Communications

Synchronous serial communication describes a serial communication protocol in which "data is sent in a continuous stream at a constant rate." Synchronous communication requires that the clocks in the transmitting and receiving devices are synchronized – running at the same rate – so the receiver can sample the signal at the same time intervals used by the transmitter. No START or STOP bits are required. The message framing is accomplished using a separate handshaking device select signal. The processor manages the data stream in units of transmission based on a pre-assigned number of bits. All PIC32 processors can be configured for 8, 16, or 32-bit transmission units. The net result is that synchronous communication permits more information to be passed over a circuit per unit time than asynchronous serial communication.

The synchronous communications circuit uses a separate clock signal or embeds the clock signal inside the data bit waveform. In either case, the clock implements handshaking and indicates when the receiving device should sample the input to determine the symbol value. As such, each symbol can have differing and arbitrary periods. As

[2] https://en.wikipedia.org/wiki/Synchronous_serial_communication#cite_note-1

you will see in Lab 4d, the I^2C protocol uses a 9-bit partition for each data byte (8 bits of information). The example explored in Lab 4c uses the SPI protocol, which segments the data stream into 8-bit data bytes.

8 References

1. Embedded Computing and Mechatronics with the PIC32 Microcontroller, 1st Edition, by Kevin Lynch (Author), Nicholas Marchuk (Author), Matthew Elwin (Author), https://www.amazon.com/Embedded-Computing-Mechatronics-PIC32-Microcontroller/dp/0124201652
2. "PIC32MX330/350/370/430/4450/470 32 Bit Microcontroller Datasheet (60001185E)", http://ww1.microchip.com/downloads/en/DeviceDoc/60001185E.pdf
3. Serial Protocols Compared, http://www.embedded.com/design/connectivity/4023975/Serial-Protocols-Compared
4. PIC32MX330/350/370/430/450/470 Family Data Sheet

Appendix A: Unit 4 Parts Configuration

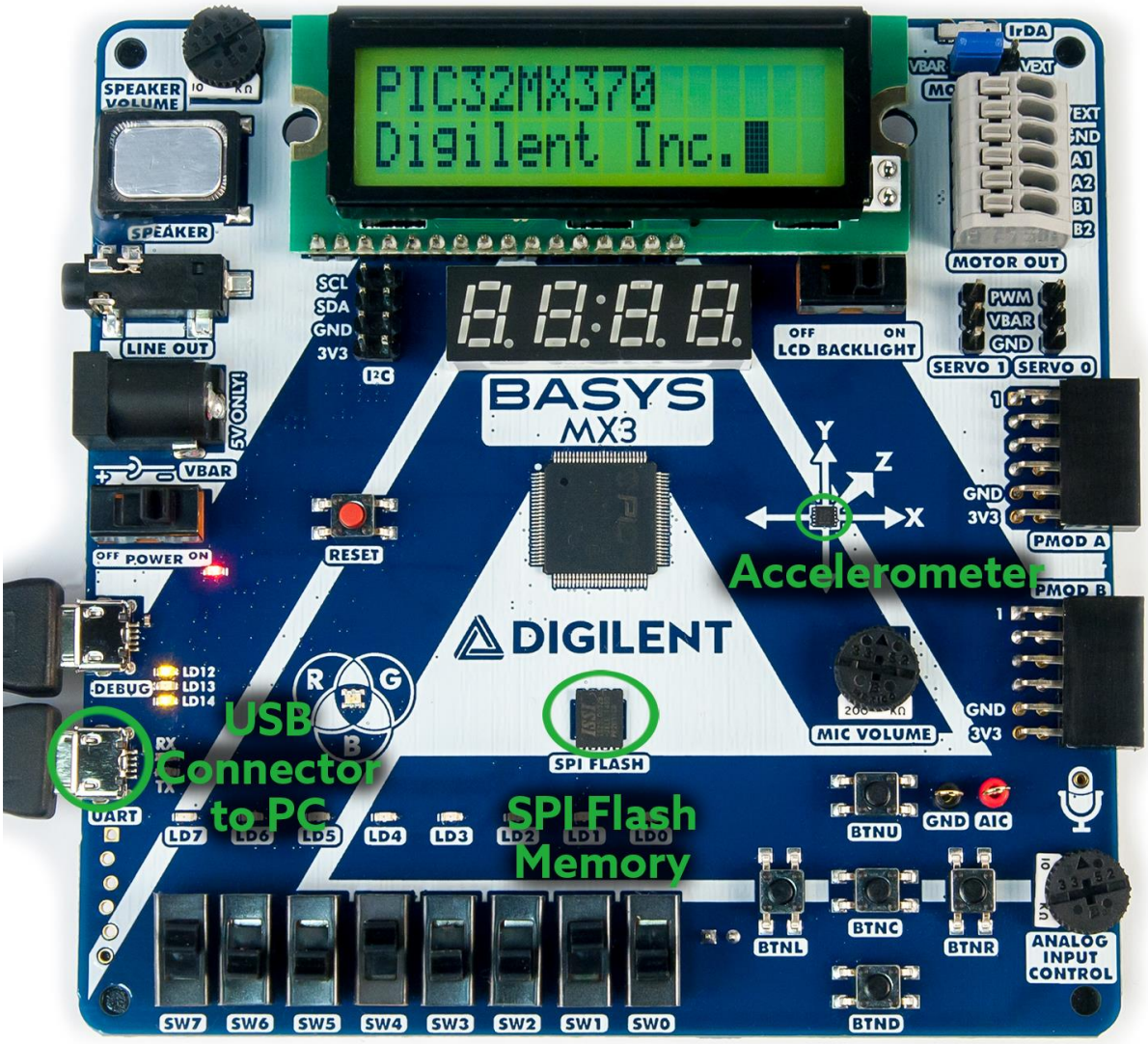

Figure A.1. Unit 4 hardware and instrumentation configuration.

Appendix B: Partial List of Popular Serial Protocols

- Morse code telegraphy
- RS-232, RS-422, RS-423, RS-485 Asynchronous communications (Project 7)
- CAN Controller Area Network
- LIN Local Interconnect Network
- I^2C Inter-Integrated Circuit
- SPI Serial Peripheral Interface
- SDLC/HDLC High Level Data Link Control (Traffic Control)
- 1-wire i-Button
- ARINC 818 Avionics Digital Video Bus
- USB Universal Serial Bus (moderate-speed, for connecting peripherals to computers)
- FireWire IEEE 1394
- Ethernet IEEE 802.3
- Fiber Channel (high-speed, for connecting computers to mass storage devices)
- InfiniBand (very high speed, broadly comparable in scope to PCI)
- MIDI control of electronic musical instruments
- DMX512 control of theatrical lighting
- SDI-12 industrial sensor protocol
- SpaceWire Spacecraft communication network