Group 9617 - Juk, Babb, Sall, and Szlosek

Embedded Systems

Week 3 Homework

Due Sept. 15th

1.) What are the main drawbacks of parallel communication that caused serial communication to become so prominent?

During parallel communication, every parallel data bit will not reach its destination at exactly the same time due to many conditions such as path resistance, temperature, and path length. This limits the speed of the parallel data bus to that of the slowest data line since the receiver needs to wait for all the data bits to arrive. This can especially be a problem with faster clock speeds where clock skew becomes an issue. Another draw back is crosstalk between the parallel lines. The parallel lines can interfere with each other causing there to be errors in the received bits. Crosstalk becomes more an issue when the distance between the transmission endpoints lengthens. These combined disadvantages make parallel communication difficult for long distance because errors are more prone to happen when clock skew and crosstalk have combined effects.

2.) How many bits can be sent in one frame of data with UART communication and what is the parity bit?

Usually, only eight bits can be sent in one frame with UART communication,but depending on the device in question, the number of data bits can be set to ten bits instead.

Parity bits are a form of low level error checking. A parity bit is sent after after the data bits. The parity bit keeps track of whether the sent data should have an even or odd number of 1-bits. In even parity, the parity bit would be 0 if the number of 1-bits was even, and in odd parity, the parity bit would be 1 if the number of 1-bits was even.

3.) How many lines are needed for I2C communication and how does the slave device know when to send data or when it is receiving data? How many data lines are needed to connect multiple devices on the I2C bus versus on the SPI bus?

Two lines are needed for I2C communication: a clock line and a data line. The slave device knows when it needs to send data or when to receive data when it sees a start condition on the bus being sent from the master device. The start condition places all slave devices on alert. An 8-bit signal is then sent from the master with an address to one of the slave devices and a read/write bit which will tell the slave whether it should send data or receive it. The SPI bus needs four different lines for communication: MOSI (Master Out Slave In), MISO (Master In Slave Out), SCK (Clock), and SS (Slave Select). For multiple connected slave devices, it will need a slave select line for each slave connected, up to n devices and n SS lines. I2C busses only need two lines total for multiple devices. Different devices can be selected via digital addresses.

5.) Explain when you would use each of the three serial communication protocols.

I2C (inter-integrated circuit) - Great to use when several sensors are needed to be interfaced and a very high throughput is not needed. It also has the capability to have more that one master on one set of I2C lines.

SPI (serial peripheral interface) - Great to use with applications that do not call for several peripherals so we do not need many slave select lines. Also good to use with sensors that have a very specific command structure.

TTL UART (universal asynchronous receiver transmitter) - Commonly used for long distance serial communication. They are commonly included on microcontrollers for a variety of peripheral modules such as Bluetooth and GPS.

6.) How are services and their characteristics distinguished in the BLE protocol?

Each service and its characteristics is distinguished by their unique UUID a 16 bit attribute.

7.) When first setting up a new microcontroller, how do you know if it is compatible with a sensor that you already have?

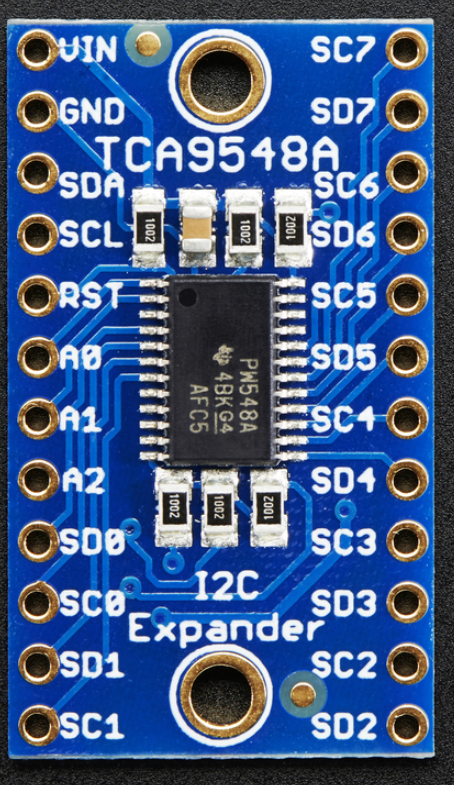
In order to know if a sensor we already have is compatible with the new micro-controller we need to check if the microcontroller has connection ports or inputs compatible with the sensor, check the operating conditions (voltage other physical parameters) and finally check the communication protocol and check if it’s supported by the new microcontroller. If these conditions are met the sensor shouldn’t have any problem to connect to the microcontroller.

8.) When using UART and communication doesn’t seem to be working properly, what things should be checked and in what order?

1. Make sure each UART is wired correctly. Can you physically see data coming in and out of each UART port probed?
2. Did you follow the datasheet of each UART? Are they compatible with the system? If different UART’s are used, are they compatible with each other? Do they use the same signaling scheme (RS-232, TTL, etc.) and voltage level?
3. Configuration settings need to be the same for both UART’s. These settings include:
   * Number of start bits
   * Number of stop bits
   * Number of data bits
   * Parity
   * Baud Rate
4. Were any of the error flags asserted? If so, is the control logic correct and take in account all possible conditions? These include:
   * Did overrun occur?
   * Were the parity bits incorrect?
   * Was a framing error detected?
5. Make sure flow control is configured correctly. Data cannot be transmitted to the receiving UART if a CTS flag is not sent back to the transmitting UART after a RTS flag is sent.
6. Test and debug the control logic or code of the system, making sure it doesn’t contain any bugs and functionally describes the UART protocol.
7. Check to see if there is a hardware issue with one or both of the UART’s.

11.) If you had six leftover i/o pins on your microcontroller and the need to connect three more temperature sensors how would you go about this?

To connect an additional 3 temperature sensors to the microcontroller, you would need an analog I2C multiplexer, such as the TCA9548A. This chip is capable of multiplexing up to eight I2C components with only utilizing five I/O pins.



**Figure 1: TCA9548A I2C Multiplexer**

The SDA, SCL pins of the TCA9548A are directly connected to the microcontroller as part of the I2C protocol. Pins A0-A2 are also connected to the microcontroller and are used as the select lines to the 8-1 mux. The SC0-SC7 and SD0-SD7 are the pins to be connected to the desired I2C component. In this case, you would connect each temperature sensor to one of the SC/SD pairs and would only need 2 of the 3 mux select pins to control them.