Design Assignment 6

CpE 301 Due: Thursday, March 16th, 2017 by 11:59pm Spring 2017 (Dr. Harris)

Description

The goal of the assignment is to write a C program that displays the value read on an analog pin on a serial terminal (such as PuTTY) on your computer. The program should monitor the value of the analog pin every second and then send it to the PuTTY terminal via a COM port. Use the FTDI chip provided for UART to USB conversion. You can produce the analog value by connecting a voltage divider using a variable resistor or if you have a sensor, you can hook that up to the analog pin.

Some pointers, examples, and hints about setting up the FTDI chip are available at the end of this document.

What to Turn In

The following must be submitted via WebCampus by the due date/time to receive credit for the assignment. Messy, difficult to understand, or disorganized work will receive no credit.

Total points available: 100

0. **Time:** Indicate the amount of time this assignment took in hours. This will not affect your grade (unless omitted) but will help gauge the workload for this and future semesters. **[-5 points if omitted]**

1. **A pdf or word document** that contains details about your design. Be sure to submit them **in this order**:
   a. A 1-paragraph description of your design. **[15 pts]**
   b. Your program, which must have been built (i.e., compiled/assembled) and working. The code should be clear, well-formatted, and well-documented. **[30 pts]**
   c. A flow chart of your program (hand-drawn is fine as long as it's legible). **[20 pts]**
   d. A schematic of your hardware (again, hand-drawn is fine as long as it's legible). **[20 pts]**
   e. A link to a ~1-2 minute Youtube video showing your program working in hardware. **[15 pts]**

Examples and Hints

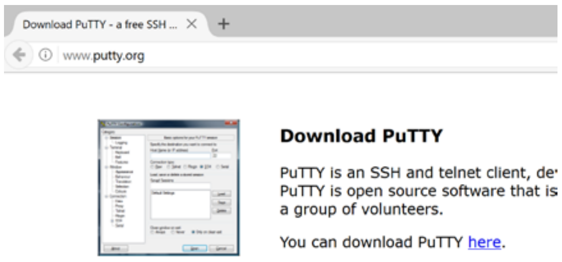
To connect to a serial monitor on your PC to the ATmega328p, you will:

Step 1. If needed, download the PuTTY program that will accept serial data
Step 2. Connect the FTDI chip to the ATmega328p to send serial data via the USART interface (configured as a UART)
Step 3. Compile and run a program on the ATmega328p to send and receive data via the UART.
Step 4. Run and configure the PuTTY program and send serial data to it from the ATmega328p

Details about each step are below.

Step 1. If needed, download the PuTTY program that will accept serial data

You will need a serial monitor that accepts data from one of the COM (USB) ports. If you do not already have one, go to www.putty.org and download the PuTTY installer (see Figure 1). Execute the installer and follow the appropriate steps for your system.



Download PuTTY

PuTTY is an SSH and telnet client, de
PuTTY is open source software that is
a group of volunteers.

You can download PuTTY here.

Figure 1. PuTTY installer download

Step 2. Connect the FTDI chip to the ATmega328p to send serial data via the USART interface

Figure 2 shows the UART to USB converter breakout board. You will connect the pins (shown on the left in the figure) to the ATmega328p and the USB port (shown on the right) – via a mini USB cable – to your computer. The **breakout board provides power** to the ATmega328p, so you will not connect an external power supply as you have done in previous labs. Notice that the jumper (highlighted in red in Figure 2) is placed on the 5V setting, but you could run the system at either setting (3.3V or 5V).

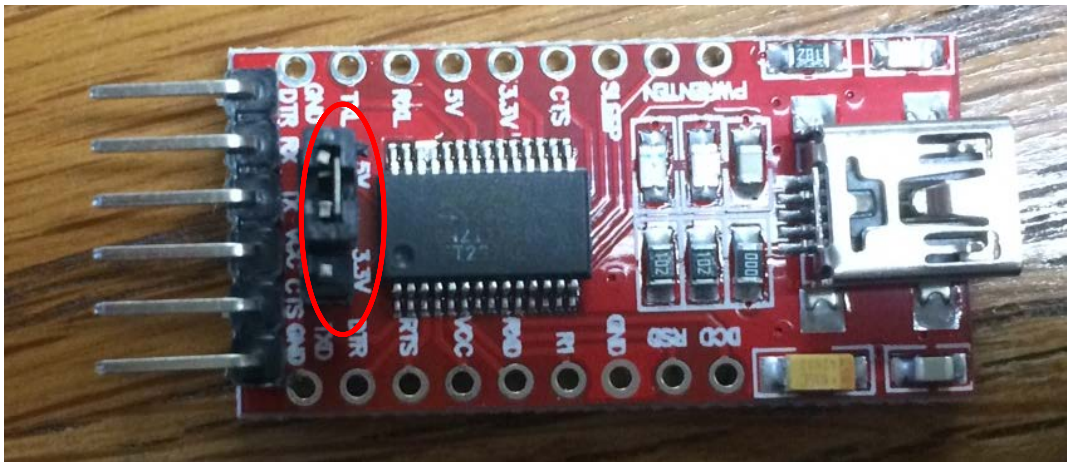


Figure 2. UART to USB converter based on FTDI's FT232RL chip

Connect the breakout board to the ATmega328p by following the connections shown in Table 1. Do not connect the DTR and CTS pins to anything.

Table 1. Breakout board to ATmega328p connections

UART to USB breakout board pin	ATmega328p pin
RX	TXD
TX	RXD
VCC	VCC
GND	GND
DTR	Not connected
CTS	Not connected

You shouldn't need it for this Design Assignment, but here is a link to the FT232RL datasheet, if you're interested:

http://www.ftdichip.com/Support/Documents/DataSheets/ICs/DS_FT232R.pdf

Figure 3 shows the ATmega328p pinout again for your convenience.

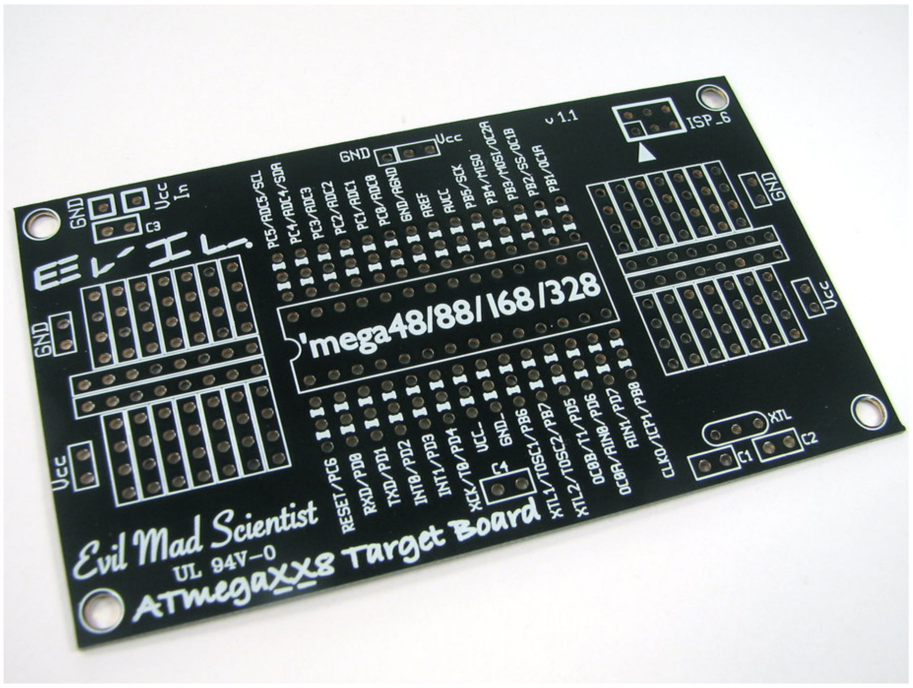


Figure 3. ATmega328p pinout

Step 3. Compile and run a program on the ATmega328p to send and receive data via the UART
Now compile, download, and run two example programs on the ATmega328p that send and/or receive data via the UART. The code is provided on WebCampus and is also shown inline below.

- **DA6_UARTexample.c:** Repeatedly writes the alphabet to the serial terminal
- **DA6_UARTexample_withInterrupts.c:** Writes the users keypresses to the serial terminal

Start by running DA6_UARTexample.c.

```
// DA6_UARTexample.c
#include <avr/io.h>
#define F_CPU 8000000       // Clock Speed
#include <util/delay.h>

#define BAUD  9600

void initUART();
void writeChar(unsigned char c);


int main( void )
{
       unsigned char ch = 'A';

       initUART();
       while (1) {
         writeChar(ch);
         ch++;
```

```c
          if (ch > 'Z')
            ch = 'A';
        }

        return 0;
}

void initUART() {
        unsigned int baudrate;

        // Set baud rate:  UBRR = [F_CPU/(16*BAUD)] -1
        baudrate = ((F_CPU/16)/BAUD) - 1;
        UBRR0H = (unsigned char) (baudrate >> 8);
        UBRR0L = (unsigned char) baudrate;

        UCSR0B |= (1 << RXEN0) | (1 << TXEN0);          // Enable receiver and transmitter
        UCSR0C |= (1 << UCSZ01) | (1 << UCSZ00);        // Set data frame: 8 data bits, 1
                                                        // stop bit, no parity
}

void writeChar(unsigned char c) {
        UDR0 = c;                   // Display character on serial (i.e., PuTTY) terminal
        _delay_ms(200);
}
```

// DA6_UARTexample_withInterrupts.c

```c
#include <avr/io.h>
#define F_CPU 8000000       // Clock Speed
#include <avr/interrupt.h>

#define BAUD  9600

void initUART();

int main( void )
{
        initUART();
        while (1)           // Wait for interrupt
              ;

        return 0;
}

void initUART() {
        unsigned int baudrate;

        // Set baud rate:  UBRR = [F_CPU/(16*BAUD)] -1
        baudrate = ((F_CPU/16)/BAUD) - 1;
        UBRR0H = (unsigned char) (baudrate >> 8);
        UBRR0L = (unsigned char) baudrate;

        UCSR0B |= (1 << RXEN0) | (1 << TXEN0);    // Enable receiver and transmitter
        UCSR0C |= (1 << UCSZ01) | (1 << UCSZ00); // Set data frame: 8 data bits, 1 stop
                                                 // bit, no parity
        UCSR0B |= (1 << RXCIE0);                 // Enable receiver interrupt
        sei();                                   // Enable global interrupts
}

// UART receiver interrupt handler
ISR (USART_RX_vect)
{
	unsigned char receivedChar;

	receivedChar = UDR0;				// Read data from the RX buffer
	UDR0 = receivedChar;				// Write the data to the TX buffer
}
```

Step 4. Run and configure the PuTTY program and send serial data to it from the ATmega328p

Now that you have setup the physical connection setup and downloaded the program onto the ATmega328p, open a serial terminal to read and/or send serial data (characters). First, check which COM port the computer is using to connect to the breakout board by opening the Device Manager (Control Panel → Device Manager). It should display the COM port (in this case, COM5, as highlighted in red) to which the FTDI breakout board is connected. If needed, remove other USB connections to find out which one is the connected to the FTDI breakout board. And make sure the FTDI breakout board is connected to the computer.

Driver installation for the FTDI breakout board should be done automatically, but if you need to, you can download drivers for the FTDI chip here:

http://www.ftdichip.com/Drivers/VCP.htm

This is a Virtual COM port driver that makes the USB device look like another COM port to the PC – in this case, COM5.

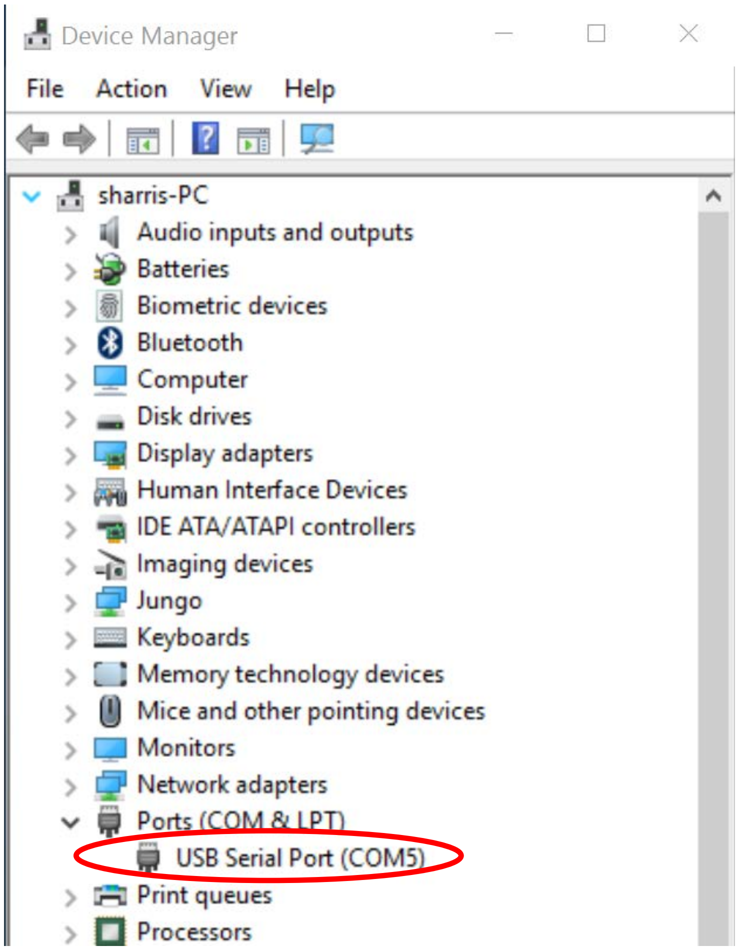
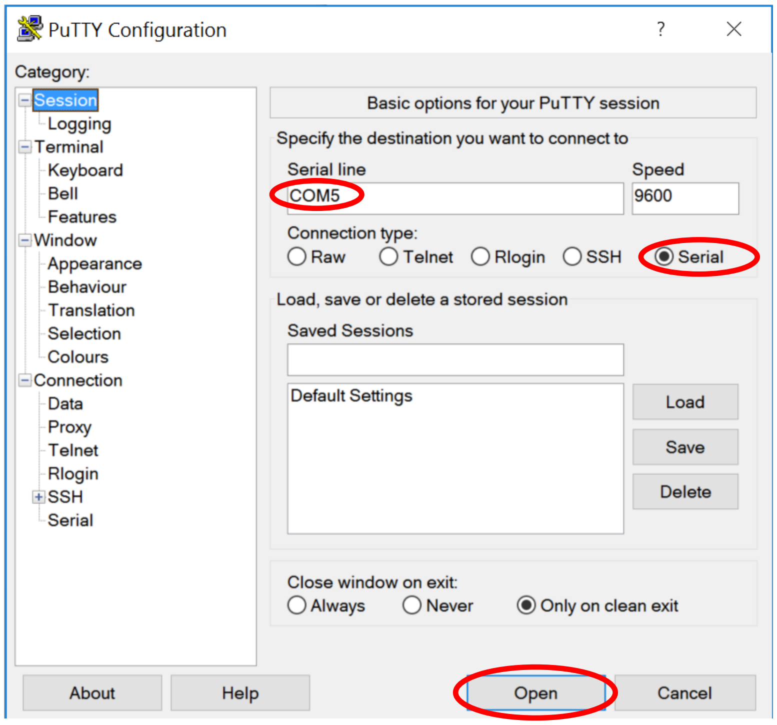


Figure 4. Device Manager

Next, execute the PuTTY application to create a serial terminal. After launching PuTTY, the PuTTY Configuration window will open (see Figure 5). Click on the **Serial** radio button and enter the COM port number (in this example, COM5) in the **Serial line** field. Leave the **Speed** at 9600 baud, as shown. Then click Open.


PuTTY Configuration

Category: Session

Basic options for your PuTTY session

Specify the destination you want to connect to

Serial line: COM5

Speed: 9600

Connection type: Raw Telnet Rlogin SSH [✓ Serial]

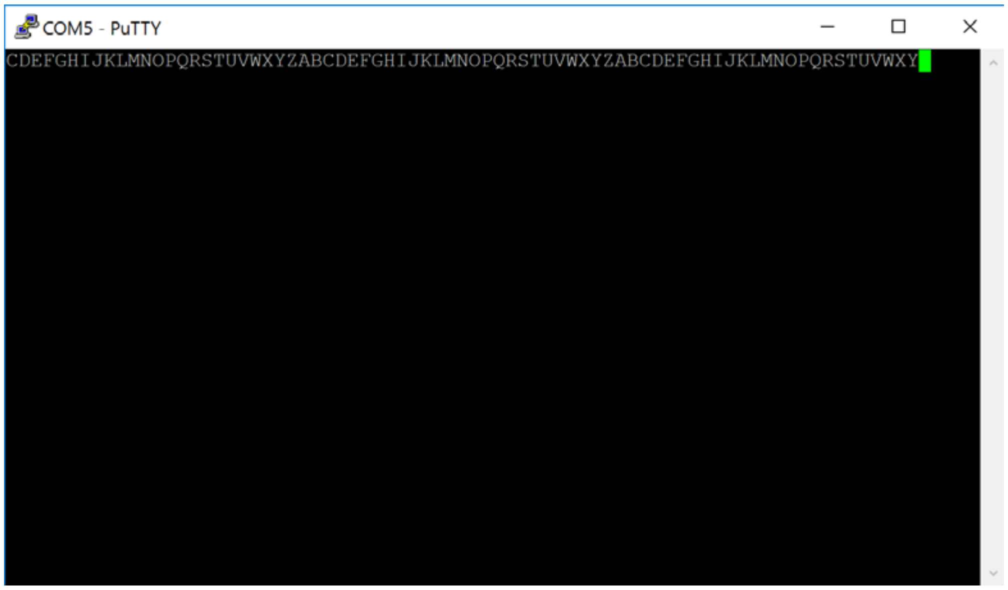
Load, save or delete a stored session

Saved Sessions: Default Settings

Close window on exit: Always Never [✓ Only on clean exit]

Figure 5. PuTTY Configuration Window

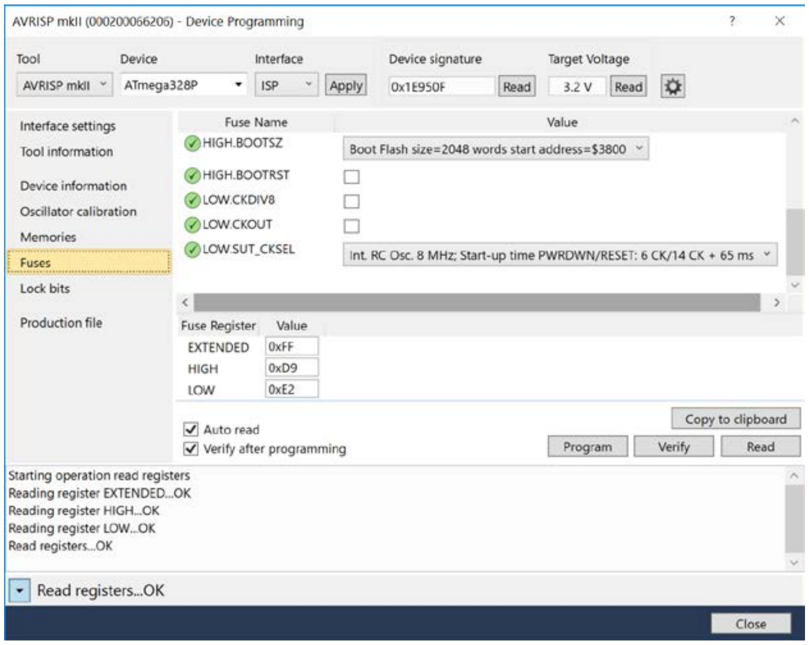
The serial terminal will now appear. The ATmega328p is sending the alphabet to the terminal repeatedly when running DA6_UARTexample.c


COM5 - PuTTY

```
CDEFGHIJKLMNOPQRSTUVWXYZABCDEFGHIJKLMNOPQRSTUVWXYZABCDEFGHIJKLMNOPQRSTUVWXY
```

Figure 6. PuTTY serial terminal when running DA6_UARTexample.c

If you get garbage (i.e., the incorrect characters or garbage characters) on the serial terminal, there is likely an issue with timing. Make sure the clock frequency you're running at (likely 8 MHz, unless you've set up an external oscillator) is consistent in the code as well as the Fuse setup. For example, Figure 8 shows Fuse settings for using the internal 8 MHz oscillator. Make sure that LOW.CKDIV8 is **not** selected, as shown.



AVRISP mkII (000200066206) - Device Programming

Tool	Device	Interface	Device signature	Target Voltage
AVRISP mkII	ATmega328P	ISP	0x1E950F	3.2 V

Fuse Name	Value
HIGH.BOOTSZ	Boot Flash size=2048 words start address=$3800
HIGH.BOOTRST	☐
LOW.CKDIV8	☐
LOW.CKOUT	☐
LOW.SUT_CKSEL	Int. RC Osc. 8 MHz; Start-up time PWRDWN/RESET: 6 CK/14 CK + 65 ms

Fuse Register	Value
EXTENDED	0xFF
HIGH	0xD9
LOW	0xE2

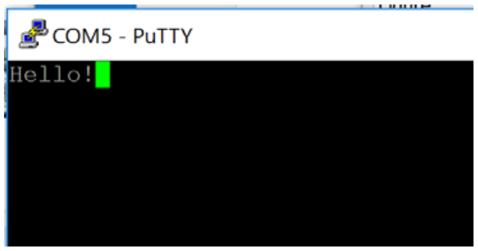
☑ Auto read
☑ Verify after programming

```
Starting operation read registers
Reading register EXTENDED...OK
Reading register HIGH...OK
Reading register LOW...OK
Read registers...OK
```

Read registers...OK

Figure 7. ATmega328p Fuse settings

After you have everything working, download the second program (DA6_UARTexample_withInterrupts.c). This program takes in any user keypresses entered into the serial terminal (i.e., while the PuTTY terminal is selected) and displays them back on the serial terminal. Figure 7 shows the serial terminal after the user has entered the keys "Hello!".



COM5 - PuTTY

```
Hello!
```

Figure 8. PuTTY serial terminal when running DA6_UARTexample_withInterrupts.c