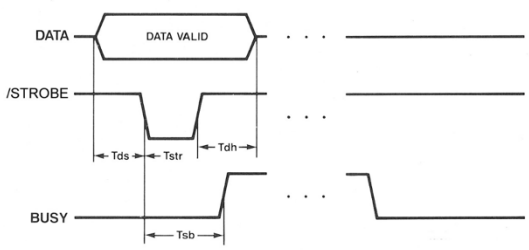
Anthony Bugatto
CPE 406
Assignment 2
11 February 2018

Assignment Description:



This assignment was much easier than the last one in that it was very straightforward to finish given project 1 was completed correctly. Given that we had the 8255A linked and working correctly with the arduino's ATMEGA 2560 the only task in this project was to write a program that imitated an 8-bit printer with a timing diagram of the one to the right. The function worked such that data input (ASCII characters) triggered a STROBE signal to blink and a BUSY signal to go high until the printer is ready to accept another input. Given that we didn't have a printer I wrote my program such that the STROBE and BUSY were simulated and such that there were serial print outputs that triggered each time one of them changed.
In the second trial I instead used a manual BUSY input as I and simulated the STROBE and data to work given the busy signal.

Problems Encountered:

I had a pretty easy time making this project work after I wrote my code. The only issue I had was that the whole process was going too quickly for me to realize what was going on using LED's. My solution was to use serial print out statements that would show me each time data was input or the BUSY/STROBE values were changed.
On the second trial I had a huge problem figuring out what I needed to do to make the system work via oscilloscope timing diagram. I eventually settled on using the oscilloscope to show SCOPE and data and having the BUSY being a manual input from port B. Additionally my oscilloscope wasn't looking right until I switched from the delay() command to the delayMicrosecond() command.

Lessons Learned:

Even though I learned most of what I know about external I/O from project 1, this project taught me a lot about the operations of printers and gave me some real world context for what devices like the 8255A is used for. It is very interesting that a timing diagram such as the DATA/BUSY/STROBE system was used as it seems to say a lot about the operation of printers. In the second trial I learned that in the future I will need an oscilloscope to test out my circuits because the serial output was completely wrong and didn't produce useful oscilloscope data.

Description of Completed Lab:

Draft 1

This lab took an hour to complete since the code was so simple to implement and the debugging was easy, consuming little time. The code is printed in the back of the report and the output is represented by the serial print out statements for one loop cycle of the ASCII data input. I chose to use the LED at the left end of the picture to represent the PA0 in the first picture and the PA5 in the second picture. I chose these because they should light up as 1's in the case that an ASCII 'A' is input (0x41). You can see that they are both lit up as this is after one loop. The other PAx values are 0. The third picture is the serial print out statements.

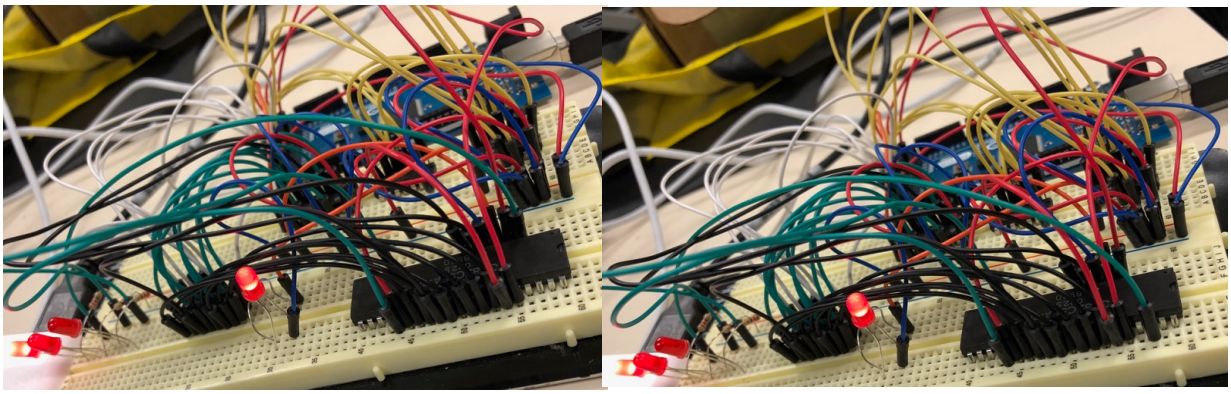


Figure 1: My original version of the project with a serial output interface

Draft 2

The final version of the lab that I made for the second week you gave us was much different than the original one I made. My original code turned out to be way overcomplicated in that I used the serial interface to show that the system worked rather than the oscilloscope. Once I tried using the

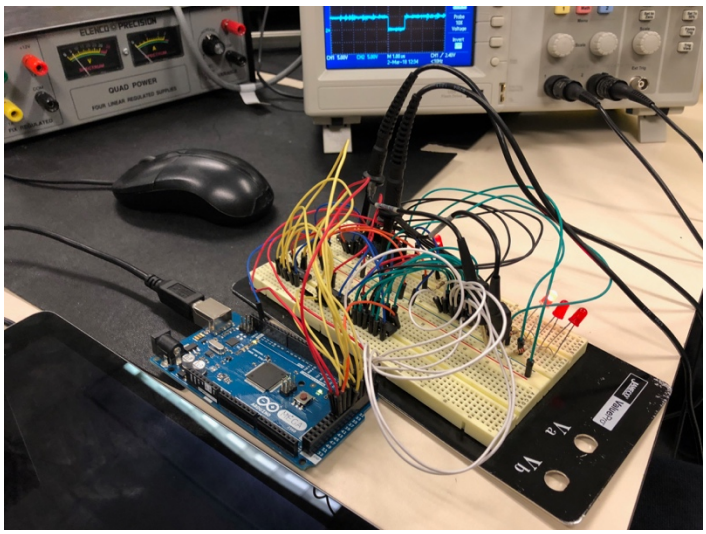


Figure 2: Final Circuit (LED's are not being used)

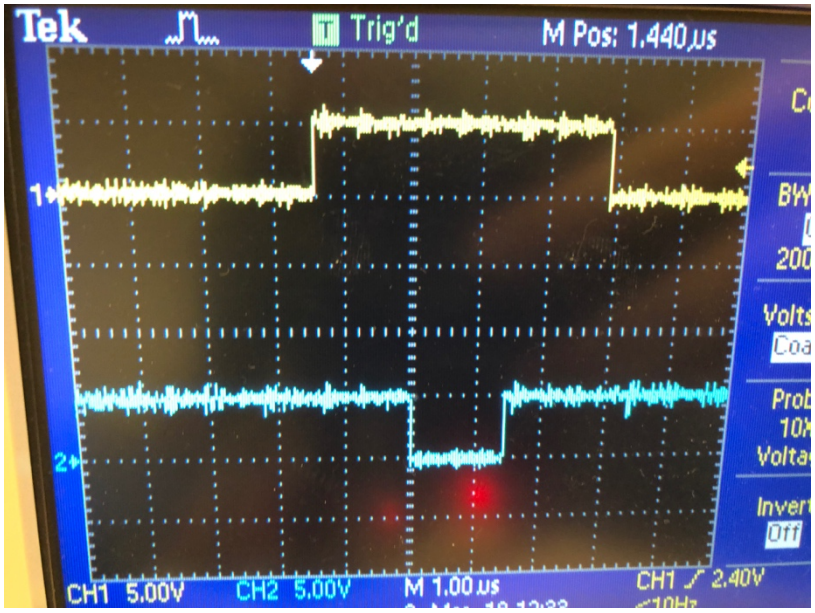
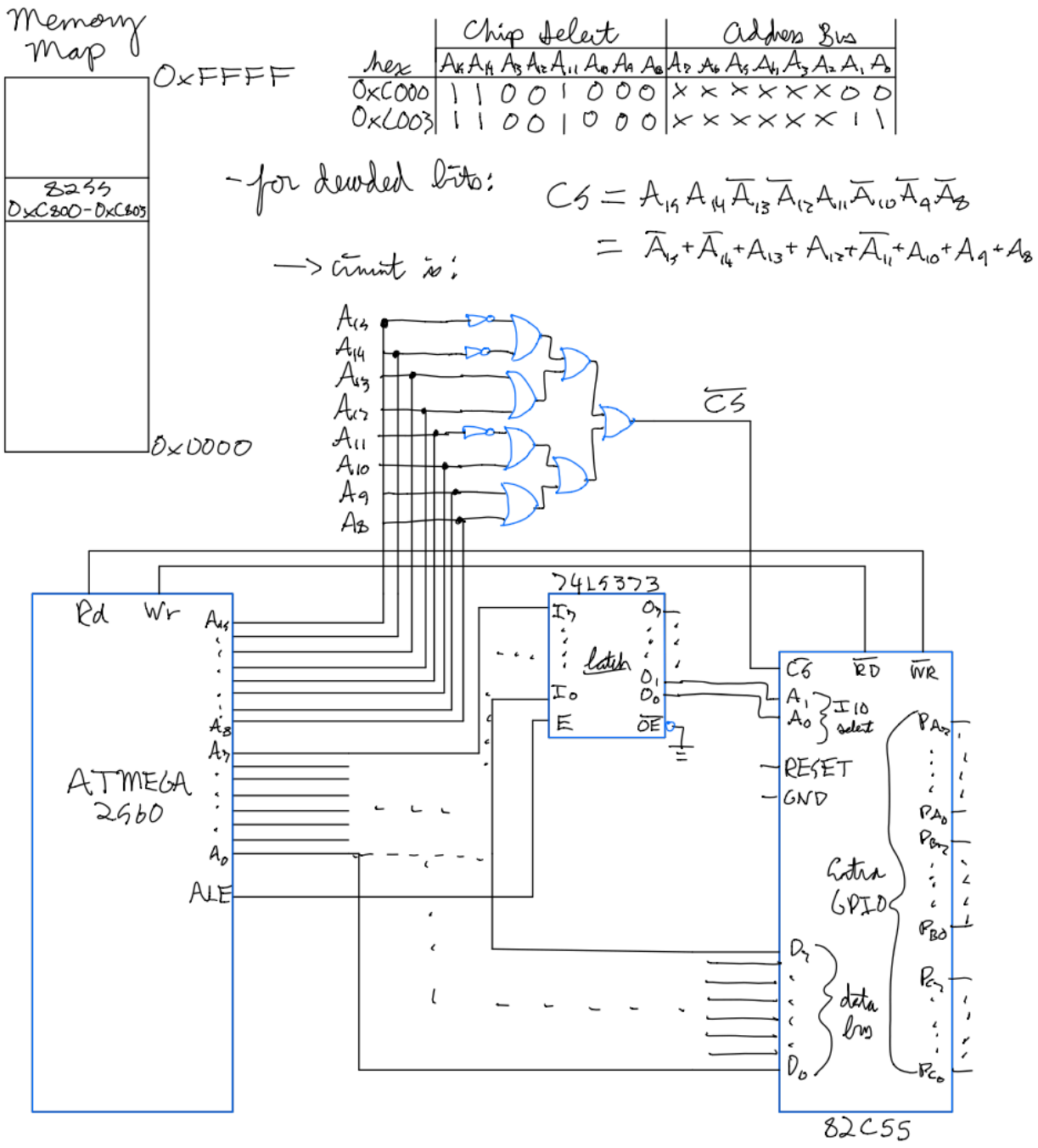


Figure 3: The Oscilloscope output with Source 1 representing the Data I/O from PortA and Source 2 representing the STROBE signal.

oscilloscope for the testing it quickly became clear that it wouldn't work at asll and that the code was wrong. I spent a couple hours figuring this out and it came down to my overcomplicating of the code as well as the use of

the delay() function, which doesn't work well in this case for some reason and instead it's easier to use the delayMicrosecond() function. After I got this set up the code worked flawlessly as long as I made sure to manually change the BUSY to prove that the code worked.

Schematic:



Code:

hww

```
volatile unsigned char* myCLKPR = (unsigned char*) 0x61; //prescaler for system clock

volatile unsigned char* myXMCRA = (unsigned char*) 0x74; //initialize special functiona
volatile unsigned char* myXMCRB = (unsigned char*) 0x75; //control wait states

//addresses of the 8255
volatile unsigned char* myPortA = (unsigned char*) 0xC800;
volatile unsigned char* myPortB = (unsigned char*) 0xC801;
volatile unsigned char* myPortC = (unsigned char*) 0xC802;
volatile unsigned char* myControl = (unsigned char*) 0xC803;

////////////////////////////////

void printchar(unsigned char);

////////////////////////////////

void setup() {
  *(myCLKPR) = 0x81; //prescale system clock by 2

  //set external memory functions to on and set wait to one cycle
  *(myXMCRA) = 0x84;
  *(myXMCRB) = 0x00;

  *(myControl) = 0x82; //initializes all registers and modes

  Serial.begin(9600);
}

void loop() {
  printchar(0xFF);
  printchar(0x00);
}

void printchar(unsigned char in) { //simulates the printer function
  while((*(myPortB) & 0x80) != 0x00) {
    *(myPortA) = in;
    delayMicroseconds(.5);
    *(myPortC) = 0xFB;
    delayMicroseconds(1);
    *(myPortC) = 0xFF;
    delayMicroseconds(.5);
    *(myPortA) = 0x00;
  }
}
```

Done Saving.

```
Sketch uses 2,142 bytes (0%) of program storage space. Maximum is 253,952 bytes.
Global variables use 196 bytes (2%) of dynamic memory, leaving 7,996 bytes for local variables. Maximum is 8,192 bytes.
```