Brandon Ramos

Homework 5

Problem 1.)

The homework took about a good 2 ish hours just going through it and I was able to ask for TA help. I used the datasheet and schematics. The hardest were some of the problems on problem 3 in general. But it was helpful looking back at the lectures.

Problem 2.)

a.)

TCCR0A |= (1<<WGM01)

ISR(TIMER1\_COMPA\_vect){

}

b.)

the program finishes whatever line it was on, pushes that location onto the stack, goes to the ISR and executes that (while disabling interrupts) then enable interrupts and at least execute one line from main before another interrupt.

c.)

To enable/disable interrupts, use sei() and sli()

d.)

If it is disabled for the second half of the .5ms then it will miss the compare match and not be accurate for how long the clock has been running since power on. But if we do not reach the TOP while disabled for .5ms then the clock will be fine and millis() will work as expected. Even if it was off by that disable interrupt for .5ms you could still use it after interrupts are enabled to keep track of time by temp = millis(); and time = temp – millis();

e.) 16MHz/(1+48)clock cycles = 326.53kHz

f.)

One benefit is that we don’t need to keep checking if data transfer is done. We can send a flag with an interrupt when this is complete. Another is that when in time sensitive operations, we want to focus on the main code. If something happens such as we are getting close to a wall, we want to be able to stop, check the surroundings and act accordingly with interrupts.

g.)

One is that if you have a pin that is constantly changing set with an interrupt, it can bog down your main code and cause noticeable time delays. The seconds is that too many interrupts can be an issue where you must queue the interrupts and very little of your main code is running. Therefore, we write as little as lines as we can in the ISR function.

h.) We are setting an interrupt for when the RX vector is triggered as well, we are checking in uartReceiveByte() if there is information being sent. Instead we can delete that reveivebyte() function and put c\* = DEQUEUE(uart0RxQueue) into the interrupt ISR.

Problem 3.)

a.)

Parallel communication can send information over a shorter period. And parallel communication allows us send information at lower clock rates. Parallel communications allow to have separate select lines available at any time for data transmission.

b.)

00101101 for rising edge

01001000 for falling edge

c.)

Slave Select allows the master to enable and disable specific devices So that the SPI can support multiple devices

d.)

BAUD = clock/16(0+1) =750kHz

BAUD = clock/8(0+1) = 1.5MHz

UBRRn = 0

e.)

baud/data/stop bits = 9600

f.)

UBRR0=15MHz/(2\*28800) -1=259.42

~0% error

g.)

Open drain means to add a pullup resistor to the output. And it is needed in I2C because this implements a wired-AND function which is needed to operate normally.

All TWI devices are open drain, so they are both open drain (SCL and SDA)

h.)

the master writes to the slave’s address where the last bit is 0 if it wants to write, otherwise 1 for read

i.)

arbitration works by watching SDA and SCL for its stop and start conditions, and if the bus is in use. This is how multiple masters can be used, they check if the bus is currently in use.