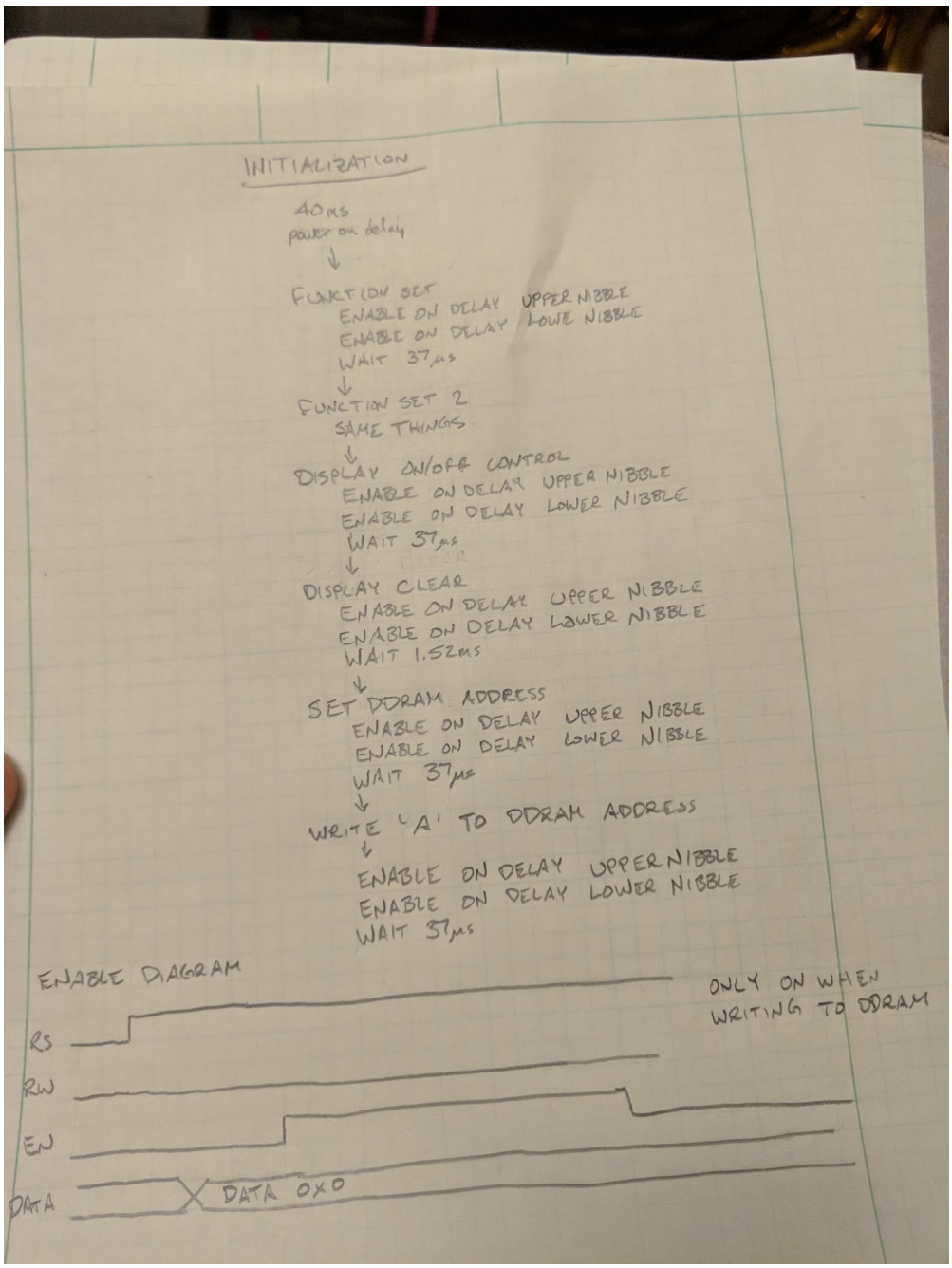
Assignment 3

Questions

1. Without time for instructions: Power on time + Function Set time * 2 + Display ON/OFF Control + Display Clear + Enable bit on time * 8
   40ms + 37micros + 37MICROs + 37MICROs + 1.52ms + 460ns = **41.6347ms**

2. 

Video Link: https://photos.app.goo.gl/TVRBHJJmv3J7AJ269

Code:

```
#include "msp.h"
#include "LCD.h"
#include <string.h>
/**
 * main.c
 */
void main(void)
{
	WDT_A->CTL = WDT_A_CTL_PW | WDT_A_CTL_HOLD;

	init_LCD(0xFF); //MSB(1,D,C,B,N,F,x,x)LSB (8bits)

	char* asnmt3 = " Assignment 3";
	uint8_t startAddress = 0x00;
	Write_String_LCD(asnmt3, startAddress);

	char* helloWorld = "Hello World";
	startAddress = 0x40;
	Write_String_LCD(helloWorld, startAddress);

	Home_LCD();

	while(1);
}

#include "msp.h"
#include "delay_us.h"

/*
 * LCD.h
 *
 *  Created on: Apr 12, 2019
 *      Author: Daniel Gutmann
 */

#ifndef LCD_H_
#define LCD_H_

#define ENABLE  ((uint8_t)0x02) //enable pin
#define DISABLE ((uint8_t)0xFD)
```

```
#define RS      ((uint8_t)0x04) //RS pin
#define RW      ((uint8_t)0x01) //RW pin
#define DELAY   100

void init_LCD(uint8_t modeDispCTL); //mode_dispCTL:
MSB(1,D,C,B,N,F,x,x)LSB (8bits)
void Clear_LCD();
void Home_LCD();
void Write_String_LCD(char* string, uint8_t startAddress);

#endif /* LCD_H_ */

#include "LCD.h"
#include <string.h>

/*
 * LCD.c
 *
 *  Created on: Apr 10, 2019
 *      Author: Daniel Gutmann
 */

//Functions only accessible in LCD.c
void Write_Byte(uint8_t data, uint8_t rsrw); //data: (8bits), rsrw:
MSB(x,x,x,x,x,x,rs,rw)LSB (8bits)
void Set_Entry_Mode(uint8_t entryMode);
void Set_CGRAM(uint8_t address);
void Set_DDRAM(uint8_t address);
void wait_Busy_Flag();

void init_LCD(uint8_t modeDispCTL)  //MSB(1,D,C,B,N,F,x,x)LSB (8bits)
{
    P4-> SEL0 = 0;
    P4-> SEL1 = 0;
    P4-> DIR = 0xFF; //set ports to output
    P4-> OUT = 0x00; //Clear ports
    //delay_us(50000); //Power On delay

    //First initialization
    float shortDelay = 1;
    P4-> OUT |= 0x30;
    P4-> OUT |= ENABLE;
    shortDelay ++;
```

```
    P4-> OUT &= DISABLE;
    delay_us(100);

    //Set number of lines and font
    uint8_t dispMode = 0x20 | (modeDispCTL & 0x0F);
    Write_Byte(dispMode, 0); //mode must be 8 bit number the first 4
and last two bits don't matter
    delay_us(100);
    //wait_Busy_Flag();

    //Set number of lines and font
    Write_Byte(dispMode, 0); //mode must be 8 bit number the first 4
and last two bits don't matter
    delay_us(100);
    //wait_Busy_Flag();

    //Set display ON/OFF, cursor, cursor blink
    uint8_t dispOnOff = modeDispCTL>>4; //shift on/off control bits
into lower nibble
    dispOnOff = 0x08 | (modeDispCTL & 0x0F);
    Write_Byte(dispOnOff,0);
    //wait_Busy_Flag();
    delay_us(100);

    Clear_LCD();
    P4-> OUT = 0x00; //clear ports
}

void Clear_LCD()
{
    Write_Byte(0x01,0);
    delay_us(2000);
    //wait_Busy_Flag();
}

void Home_LCD()
{
    Write_Byte(0x02,0);
    delay_us(2000);
}

void Write_String_LCD(char* string, uint8_t startAddress)
{
```

```
    Set_DDRAM(startAddress);
    int len = strlen(string);
    int i;
    for(i=0; i<len; i++)
    {
        Write_Byte(string[i], RS);
        delay_us(1000);
        Set_Entry_Mode(0x06);
    }
    delay_us(100);
}

void Write_Byte(uint8_t data, uint8_t rsrw)
{
    float shortDelay = 1;
    P4-> OUT = rsrw & (RS | RW);

    //UPPER NIBBLE
    P4-> OUT |= data & 0xF0;
    P4-> OUT |= ENABLE;
    shortDelay ++;
    P4-> OUT &= DISABLE;

    data = data<<4;
    P4-> OUT &= 0x0F;//Clear data

    //LOWER NIBBLE
    P4-> OUT |= data & 0xF0;
    P4-> OUT |= ENABLE;
    shortDelay ++;
    P4-> OUT &= DISABLE;
}

void Set_Entry_Mode(uint8_t entryMode)
{
    entryMode &= 0x07;
    entryMode |= 0x04;
    Write_Byte(entryMode,0);
    delay_us(100);
    //wait_Busy_Flag();
}

void Set_CGRAM(uint8_t address)
```

```
{
    address &= 0x7F;
    address |= 0x40;
    Write_Byte(address,0);
    delay_us(100);
    //wait_Busy_Flag();
}

void Set_DDRAM(uint8_t address)
{
    address |= 0x80;
    Write_Byte(address,0);
    delay_us(100);
    //wait_Busy_Flag();
}

void wait_Busy_Flag()
{
    float shortDelay = .1;
    P4-> DIR = ((0x00 | ENABLE) | RW); //Set Data bits to input
    P4-> OUT = RW;
    uint8_t busyFlag;

    do
    {
        P4 -> OUT |= ENABLE;
        busyFlag = P4-> IN;
        shortDelay *= .1;
        P4-> OUT &= DISABLE;
    }while((busyFlag & 0x80) == 0x00);
}

#include "delay_us.h"

/*
 * delay_us.c
 *
 *  Created on: Apr 13, 2019
 *      Author: Daniel Gutmann
 */

void delay_us(int usec)
{
```

```
    SysTick -> LOAD = (((CS->CTL0 &
CS_CTL0_DCORSEL_MASK)|CS_CTL0_DCORSEL_1)>>11)*(usec)/21;
    SysTick->CTRL = 1;
    while((SysTick->CTRL & 0x10000) != 0x10000);
}
```