Lab: 03
Nicholas Imamshah
Section: 6957

b) Prelab Questions

1. What is the advantage of mirroring the input and output ports to $28 8000 - $28 9FFF (also known as partial address decoding)? What would be necessary if you wanted to only place the ports at 0x28 8000 and no place else?
    a. Partial address decoding allows us to have simpler address decode equations, which is particularly ideal when we do not anticipate needing every possible address available; it allows us to trade efficiency of space, with simplicity of implementation. In order to place the ports at 0x288000 and nowhere else, we would need to include use the address bits for decoding in addition to the chip select (CS0).
2. Write the address decode equation to put the input and output ports at addresses 0x4000 – 0xCFFF.
    a. $A_{15} \oplus A_{14} + A_{14}/A_{13}/A_{12}$
3. What is the complete range of external memory on the XMEGA?
    a. 0x3000 – 0xFF FFFF (16MB)
4. The uPAD Proto Base is configured to use the EBI in SRAM ALE1 mode, which does not give us address bits A23:16. Which mode(s) can be used to access all of these higher address bits? Describe the change(s) necessary to use one of these modes. What additional hardware is needed, if any?
    a. SRAM 3PORT ALE12, SRAM 4PORT ALE2, and in general most of the configurations that require ALE2, excluding those for SRAM LPC. In the 4PORT cases, we would need an additional port to use such a configuration, as our processor only uses Ports H-K for the EBI configuration.

c) Problems Encountered

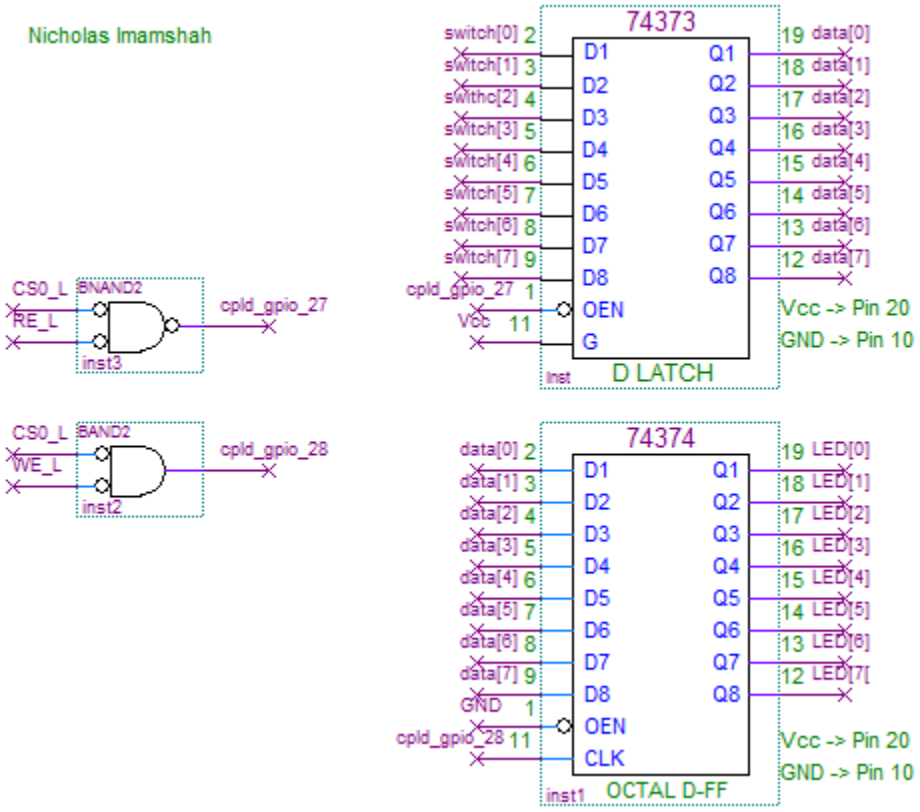
The biggest problem I encountered was in understanding the layout of the architecture; this difficulty led to an initial wrong wire-wrapping of the pins for the input port IC. Following this initial issue, I had a slight confusion understanding how the EBI actually worked. I thought it required more manual control, where we would choose which values to output to Port H as a means of choosing read and write enables, where of course we only need to configure it initially and it manages the signals internally according to loads and stores.

d) Future Work/Applications

The techniques learned in this lab could be broadly applied to expanding any compatible device's I/O ports, which then allows for connection of various other components to the device. In this lab we connected switches and input and used the LED bank as output, but these are general I/O ports as any other on our microprocessor.

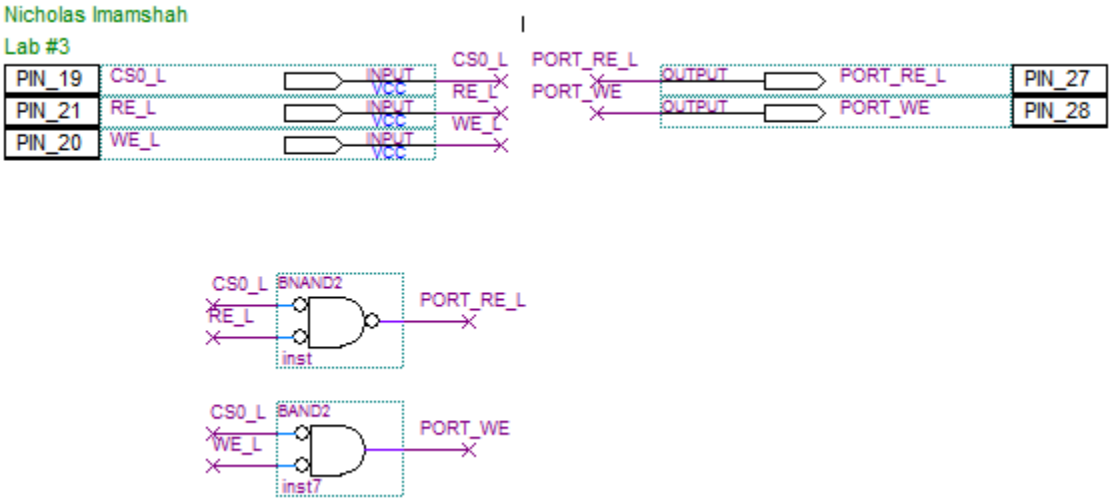
e) Schematics

Connections for 74573 and 74574 (pin numbers corresponding to those chips)

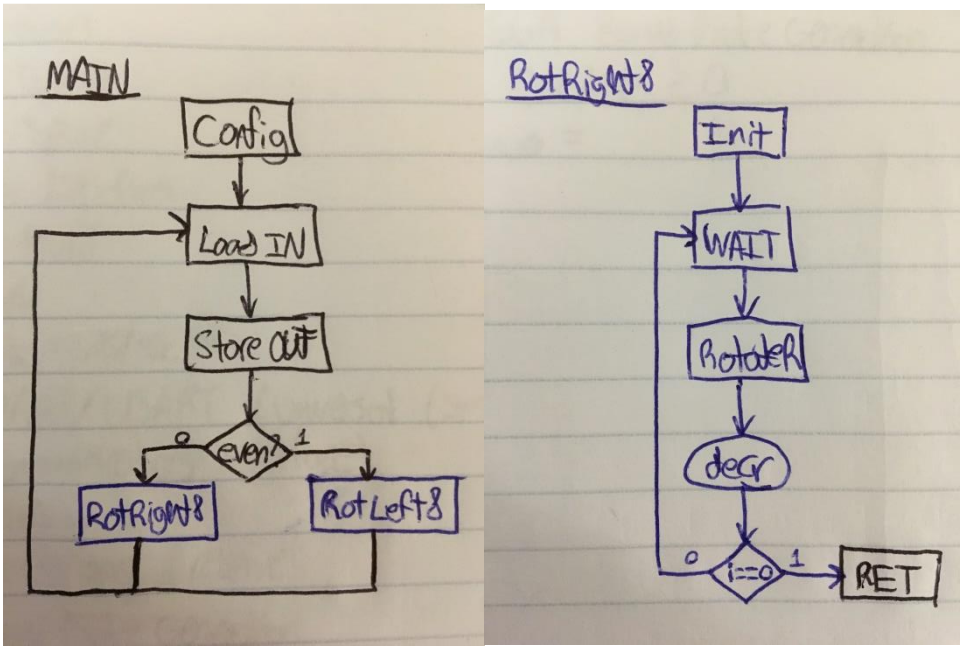


f) Decoding Logic

Control Equations for R/W



g) Pseudocode/Flowcharts



h) Program Code

```
/* lab3.asm
 *
 * Lab 3
 * Name: Nicholas Imamshah
 * Section: 6957
 * TA Name: Daniel Gonzalez
 * Description: The purpose of this program is to use EBI and expansion ports to read input and
conditionally control output.
 */

.nolist
.include "ATxmega128A1Udef.inc"
.list

.set PORT = 0x288000
.set PORT_END = 0x289FFF

.equ jcycles = 20
.equ kcycles = 246

.org 0x0000
        rjmp MAIN

.org 0x200
MAIN:
        ldi r18, 0x00                           ;register holding 0 for later adc

        ldi r16, 0x17                   ;Configure PORTH bits 4, 2, 1, and 0 as outputs.
        sts PORTH_DIRSET, r16    ;       These are the CS0(L), ALE1(H), RE(L), and WE(L) outputs.
                                                ;       (CS0 is bit 4; ALE1 is bit 2; RE is bit
1; WE is bit 0)

        ldi r16, 0x13                   ;Default CS0(L), RE(L), and WE(L) to H = false.
```

```
        sts PORTH_OUTSET, r16    ;       ALE defaults to 0 = L = false.

        ldi r16, 0xFF                    ;Set all PORTK pins (A15-A0) to be outputs.
        sts PORTK_DIRSET, r16

        ldi r16, 0xFF                    ;Set all PORTJ pins (D7-D0) to be outputs.
        sts PORTJ_DIRSET, r16

        ldi r16, 0x41                    ;Store 0x41 in EBI_CTRL reg to select 3 port EBI(H,J,K)
        sts EBI_CTRL, r16                ;       mode and SRAM ALE1 mode

;Initialize the Z pointer to point to the base address for CS0 in memory
        ldi ZH, high(EBI_CS0_BASEADDR)
        ldi ZL, low(EBI_CS0_BASEADDR)

;Load the middle byte (A15:8) of the three byte addr into a reg and store it as the
;       LOW byte of the Base Address, BASEADDRL.
        ldi r16, byte2(PORT)
        st Z+, r16

;Load the highest byte (A23:16) of the three byte addr into a reg and store it as the
;       HIGH byte of the Base Address, BASEADDRH.
        ldi r16, byte3(PORT)
        st Z, r16

        ldi r16, 0x15                    ;Set to 8KB CS space and turn on SRAM mode, 0x288000 - 0x289FFF
        sts EBI_CS0_CTRLA, r16

;Steps for using the port expansion
        ldi r16, byte3(PORT)     ;initialize a pointer to point to the base addr of the PORT
        sts CPU_RAMPX, r16               ;use the CPU_RAMPX reg to set the third byte of the pointer

        ldi XH, high(PORT)               ;set the middle (XH) and low (XL) bytes of the pointer as usual
        ldi XL, low(PORT)

LOAD:
        ld r16, X                                ;read the input port into r16
        nop
        nop

STORE:
        st X, r16                                ;write the input stored in r16 to the outport
        nop

        ldi r17, 8
        sbrc r16, 0
        rjmp ODD

EVEN:
        ;delay
        ldi r20, 50
        rcall DELAY_X_MS

        lsl r16
        adc r16, r18
        st X, r16
        dec r17
        breq LOAD
        rjmp EVEN
```

```
ODD:
        ;delay
        ldi r20, 50
        rcall DELAY_X_MS

        lsr r16
        brcs ODD_CARRY

ODD_STORE:
        st X, r16
        dec r17
        breq LOAD
        rjmp ODD

ODD_CARRY:
        ori r16, 0x80
        rjmp ODD_STORE



DELAY_X_MS:
        dec r20
        rcall JLOOP_INIT
        cpi r20, 0
        brne DELAY_X_MS
        ret

JLOOP_INIT:
        push r20
        ldi r20, jcycles
JLOOP:
        dec r20
        rcall KLOOP_INIT
        cpi r20, 0
        brne JLOOP
        pop r20
        ret

KLOOP_INIT:
        push r20
        ldi r20, kcycles
KLOOP:
        dec r20
        nop
        brne KLOOP
        pop r20
        ret
;.include "delayx10ms.inc"
```

i) Appendix

Screenshot to verify the frequency of rotation in lab3.asm

Lab: 03
Nicholas Imamshah
Section: 6957