Lab 3
Jineidy Mak
Secton 7F31

Pre-Lab Questions:

1. What is the advantage of mirroring the input and output ports to $8000 - $BFFF (also known as partial address decoding)? What would be necessary if you wanted to only place the ports at 0x8000 and no place else?

The CS0 would need the equation for that particular address space, It would require 16 inputs for one chip select. The advantage is that by mirroring the address we only need A15 and A14 as opposed to A15:0.

2. What is the complete range of external memory on the XMEGA?

16 mb of external memory according to chapter 4.

3. Write the address decode equation to put the input and output ports at addresses 0x4000 – 0xCFFF.

CE = /A15A14 or A15A14

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?

We could use SRAM 3Port ALE12 we would not need any additional hardware. We could also use SRAM 4Port ALE2 we would need additional latches.

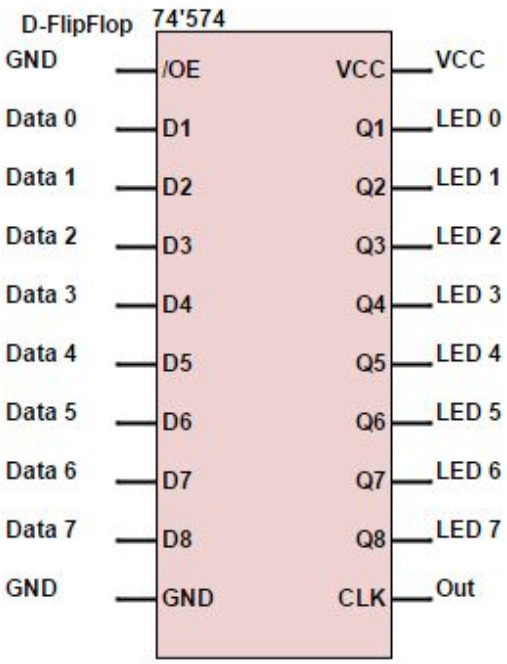
Problems Encountered:

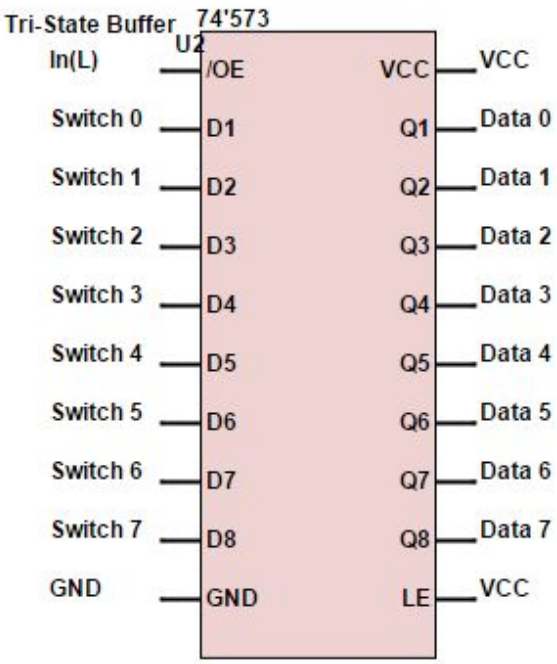
The ROR and ROL instructions would not work as expected. They would not set or use the carry register.

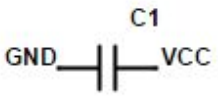
Future Work/Applications:

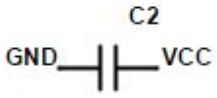
Could use the EBI to expand memory to numerous devices, to write all the inputs to memory. Can be used to output to LCDs.

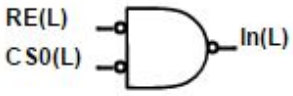
Schematics:

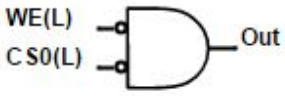

D-FlipFlop
74'574
GND
/OE
VCC
VCC
Data 0
D1
Q1
LED 0
Data 1
D2
Q2
LED 1
Data 2
D3
Q3
LED 2
Data 3
D4
Q4
LED 3
Data 4
D5
Q5
LED 4
Data 5
D6
Q6
LED 5
Data 6
D7
Q7
LED 6
Data 7
D8
Q8
LED 7
GND
GND
CLK
Out


Tri-State Buffer
74'573
U2
In(L)
/OE
VCC
VCC
Switch 0
D1
Q1
Data 0
Switch 1
D2
Q2
Data 1
Switch 2
D3
Q3
Data 2
Switch 3
D4
Q4
Data 3
Switch 4
D5
Q5
Data 4
Switch 5
D6
Q6
Data 5
Switch 6
D7
Q7
Data 6
Switch 7
D8
Q8
Data 7
GND
GND
LE
VCC


C1
GND
VCC


C2
GND
VCC


RE(L)
CS0(L)
In(L)


WE(L)
CS0(L)
Out

Decoding Logic:

```
library ieee;
use ieee.std_logic_1164.all;

entity lab3 is
  port (

    CS0_L    : in  std_logic;
    RE_L     : in  std_logic;
    WE_L     : in  std_logic;
    Out_H    : out  std_logic;
    In_L     : out std_logic

    );

end lab3;


architecture BHV of lab3 is
begin

  In_L <= RE_L or CS0_L;
  Out_H <= WE_L nor CS0_L;

end BHV;
```

Pseudocode/Flowcharts:

Configure Port H
Configure Port J
Configure Port K
Set EBI to 3 Port ALE1 mode.
Set CS0 data space to 16K SRAM mode.
Initialize CS0 Base Address
Initialize pointer to Base Address.
Read Input.
Write to LEDs.
SHIFT LOOP:
Shift Left.
Delay.
Dec Counter.
Back to Shift Loop.
If counter is zero.
Shift Right.
Delay.
Dec Counter.
Back to Shift Loop.

Program Code:

```
;
; lab3.asm
;
; Created: 6/13/2016 6:28:22 PM
; Author : James Mak
;


.include "ATxmega128A1Udef.inc"

.set Port_Start = 0x8000                                    ;Base address of our expansion port.
.set Port_End = 0xBFFF                                      ;Ending address of our expansion port.
.equ stack_init = 0x2FFF                    ;Initialize stack pointer.


.org 0x0000
                                            rjmp MAIN                                       ;Jump to the start of our main
program.

.org 0x0200

MAIN:
                                            ldi YL, low(stack_init)                         ;Initialize low byte of stack pointer.
                                            out CPU_SPL, YL
                                            ldi YL, high(stack_init)            ;Initialize high byte of stack pointer.
                                            out CPU_SPH, YL                             ;We can use the same register YL for both
high and low.

                                            ldi R16, 0b10111                                ;Bit 4 = CS0(L), Bit 2 = ALE1(H), Bit 1 =
RE(L), Bit 0 = WE(L)
                                            sts PORTH_DIRSET, R16                           ;Configure PortH setting bits 4,2,1,0 as
output bits.

                                            ldi R16, 0b10011                                ;The manual says that active low pins
should be set to 1 and that active high pins should be set to 0.
                                            sts PORTH_OUTSET, R16                           ;Output 1 at bits 4,1,0 (active low pins).

                                            ldi R16, 0xFF                                       ;Bit configurations for port K.
(A15-A0)
                                            sts PORTK_DIRSET, R16                           ;Set all Port K pins to output pins.

                                            ldi R16, 0xFF                                       ;Bit configurations for port J.
(D7-D0)
                                            sts PORTJ_DIRSET, R16                           ;Set all Port J pins to output pins.

                                            ldi R16, 0x01                                       ;Configuration bits for EBI
mode.
                                            sts EBI_CTRL, R16                               ;Set IFMODE (EBI mode) to 3-Port
Configuration.

                                            ldi ZH, high(EBI_CS0_BASEADDR)  ;Point High bytes of Z register to point to base
address register for CS0.
                                            ldi ZL, low(EBI_CS0_BASEADDR)       ;Point Low bytes of Z register to point to base
address register for CS0.
```

```
                    ldi R16, byte2(Port_Start)     ;Load the second byte of Starting address 0x8000, Byte 2 =
0x80.
                    st  Z+, R16                                         ;Store Byte 2 into
base address register.

                    ldi R16, byte3(Port_Start)          ;Load the third byte of the starting address.
                    st  Z, R16                                  ;Store Byte 3 into base
address register.

                    ldi R16, 0x19                                       ;16K chip select space
0x8000 - 0xBFFF. SRAM mode.
                    sts EBI_CS0_CTRLA, R16

                    ldi R16, byte3(Port_Start)          ;Load the third byte of Starting address to R16.
                    sts CPU_RAMPX, R16

                    ldi XH, high(Port_Start)            ;Point X register to high bytes of starting address.
                    ldi XL, low(Port_start)                     ;Point X register to low bytes of starting
address.

TEST:
                    ld R16, X                                   ;Test input port to R16.
                    nop
                    st X, R16                                   ;Output to LED.

                    ldi R17, 0x08                               ;Load counter value 8 to R19.
SHIFT_LEFT:
                    sbrc R16, 7                         ;If the most significant bit of r16 is a 1 then
set the carry to one.
                    sec
                    rol R16                                     ;Rotate R16 left.
                    rcall DELAY                                 ;Wait .25 seconds.
                    rcall DELAY                                 ;Wait .25 seconds.
                    st X, R16                           ;Display shifted number to
LEDs.
                    dec R17                                     ;Decrement the
counter.
                    cpi R17, 0x00                               ;Is the counter zero?
                    brne SHIFT_LEFT                             ;Do the shift 8 times.
                    ldi R17, 0x08                               ;Reset counter.
SHIFT_RIGHT:
                    sbrc R16, 0                                 ;If bit 0 is a 0 skip
next instruction.
                    sec
                    ror R16                                     ;Rotate R16 right.
                    rcall DELAY                                 ;Wait .25 seconds.
                    rcall DELAY                                 ;Wait .25 seconds.
                    rcall DELAY                                 ;Wait .25 seconds.
                    rcall DELAY                                 ;Wait .25 seconds.
                    st X, R16                           ;Display shifted number to
LEDs.
                    dec R17                                     ;Decrement the
counter.
                    cpi R17, 0x00                               ;Is the counter zero?
                    brne SHIFT_RIGHT                    ;Do the shift 8 times.
```

```
                rjmp TEST                                   ;Repeat after 8
times.



DELAY:                                                      ;500 us delay
divided by two for 50% duty cycle.
                ldi R18, 125                        ;125 cycles for .25ms of Delay. 125 x
500 for .5s of Delay.
                ldi R19, 250                            ;250x125 for .25s of Delay.
delay_250:
                dec R19             ;Decrement R19.
                cpi R19, 0x00                       ;Compare R19 with zero.
                brne delay_125          ;If not equal to zero,loop.
                ret
delay_125:
                dec R18             ;Decrement R18.
                cpi R18, 0x00                       ;Compare R18 with zero.
                brne delay_125          ;If not equal to zero loop.
                rjmp delay_250
```

Appendix:

Waveform .5s Delay