Lab 3

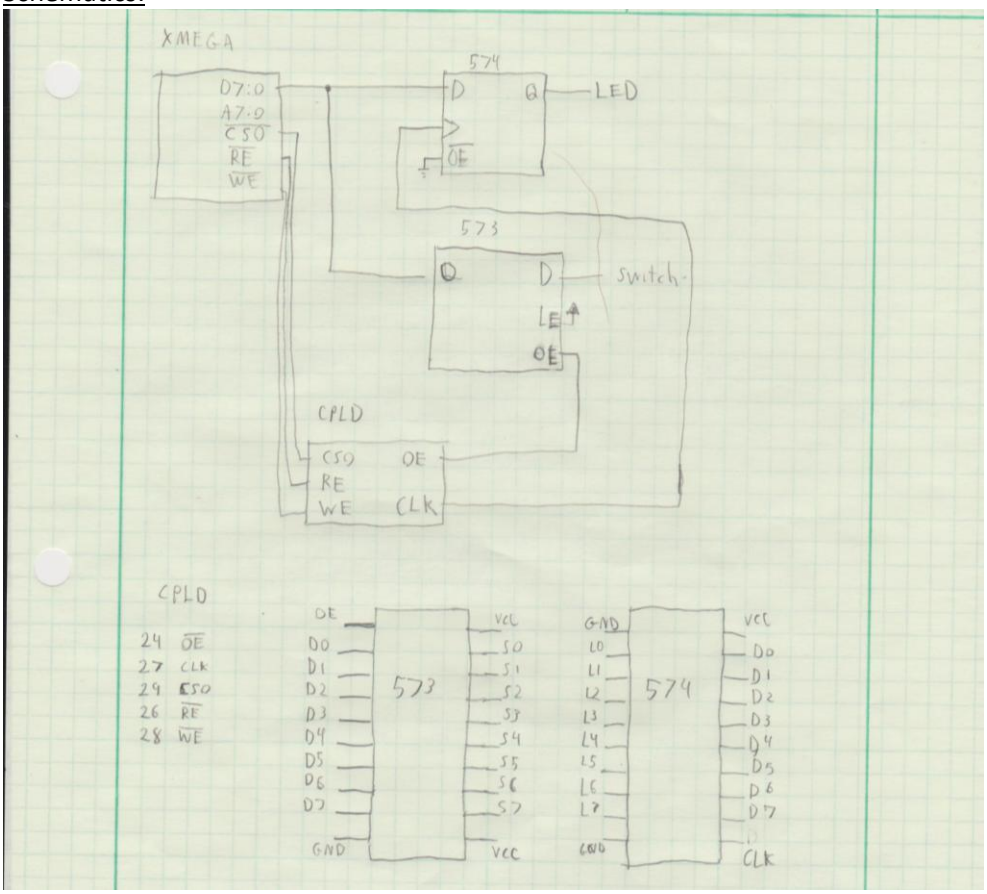
Reyna, Ricardo

Section 75C9

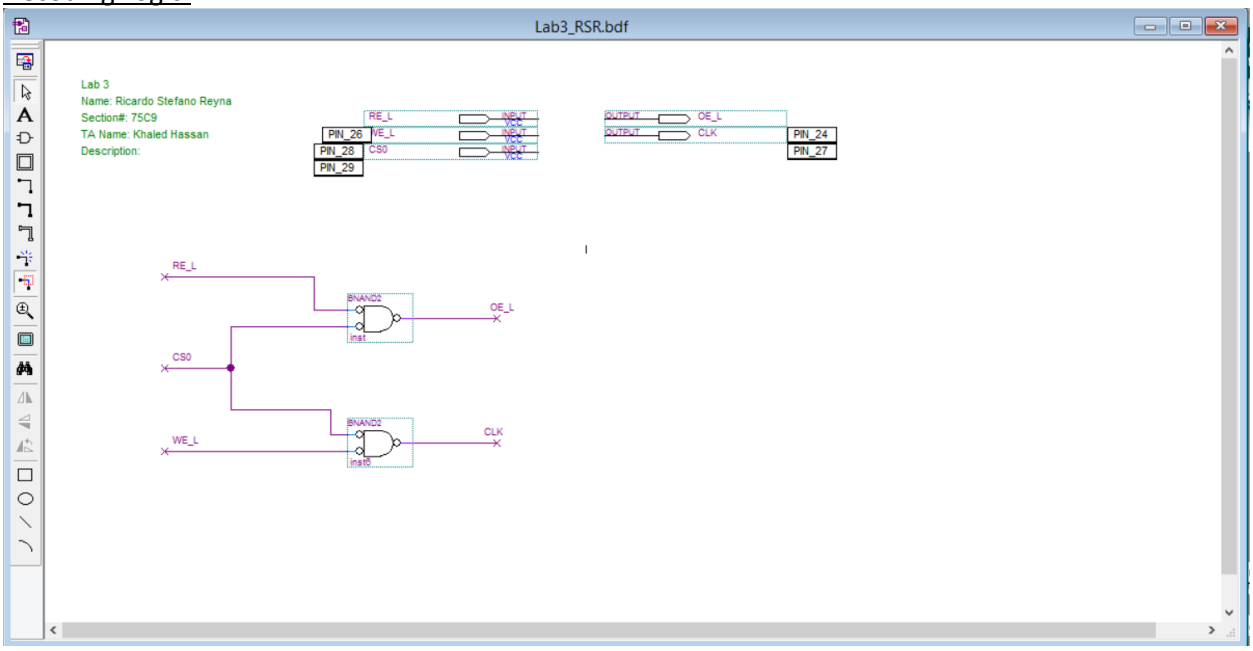
06/11/2015

- Prelab Questions:
  1. What is the advantage of mirroring the input and output ports to $4000 - $4FFF? What would be necessary if you wanted to only place the ports at $4000 and no place else?
  Because it will be using the external memory. Set the IOPORT at address $4000.
  2. Write the address decode equation to put the input and output ports at addresses at $1000 - $7FFF.
  CSO = /A15 + A14 + A13 + A12
  3. What is the complete range of external memory on the XMEGA?
  The XMEGA is 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 these higher address bits? Describe the change(s) necessary to use one of these modes. What additional hardware is needed, if any?
  SRAM ALE12 or 4PORT SRAM using ALE2. We would need a 4PORT SRAM.
- Problems Encountered:
  Some of the LEDs wouldn't turn on because of poor soldering.
- Future Work/Applications:
  Now I can use memory mapped IO in future labs.

- Schematics:



- Decoding Logic:



- Pseudo code:
  Configure ports
  have a counter
  load switch
  echo
  shift check counter

- Program Code:

```
/*
 * Lab3_RSR.asm
 *
 *  Created: 6/9/2015 6:52:15 PM
 *   Author: stefano92
 Lab 3
 Name: Ricardo Stefano Reyna
 Section#: 75C9
 TA: Khaled Hassan
 Description: This program will interact with the EBI to read from the switch and shift
right every 2 seconds 8 times and then read again.
 */

.NOLIST
.include "ATxmega128A1Udef.inc"

.set IOPORT = 0x4000
.set IPORTEND = 0x4FFF

.LIST
.org 0x0000          ;code starts at this address
       rjmp MAIN     ;go to MAIN

.org 0x200
MAIN:
       ldi R16, 0x17
       sts PORTH_DIR, R16                ;configure /WE /RE and ALE


       ldi R16, 0x2                      ;Since /RE is an active low signal, we must set
the
       sts PORTH_OUT, R16                ;default output to 1.


       ldi R16, 0xFF                     ;set all PORTJ pins (D0-D7) to be outputs. As
requried
       sts PORTJ_DIR, R16                ;in the data sheet.


       ldi R16, 0xFF                     ;set all PORTK pins (A0-A15) to be outputs. As
requried
       sts PORTK_DIR, R16                ;in the data sheet.
```

```
	ldi R16, 0x01			;Store 0x01 in EBI_CTRL register to select 3
port EBI(H, J, K)
							;mode and SRAM ALE1 mode
	sts EBI_CTRL, R16

	ldi ZH, HIGH(EBI_CS0_BASEADDR);reserve a chip-select zone for our input port.
	ldi ZL, LOW(EBI_CS0_BASEADDR) ;the base address register is made up 12 bits
							;for the address, with the lower
4 bits being reserved
							;the lower 12 bits of the
address are assumed to be
							;zero. This limits our choice of
the base addresses
							;for our zone since we can only
choose A23:A12

	ldi R16, ((IOPORT>>8) & 0xF0)	;Store BASEADDRL. We only choose the upper 12
bits
	st Z+, R16				;of the address. It would make
sense to shift
								;12 bits here, but we shift
right 8 times. This is
								;because we have to store
the lowest nibble
								;of our user specified
address lines (A23:A12)
								;into the upper nibble of
the base address reg.
								;The lower 4 bits are
reserved and not used. Also
								;we increment the Z pointer
so that we can load
								;the upper byte of the base
address register

	ldi R16, ((IOPORT>>16)& 0xFF)	;put the upper byte (A23:16) into the upper byte
								;of the base address
register.
	st Z, R16				;Store BASEADDRH

	ldi R16, 0x11				;Set to 4K chip select space and turn on
SRAM mode
	sts EBI_CS0_CTRLA, R16			;address space of the input port will be
								;0x4000 to 0x4FFF

	ldi XH, HIGH(IOPORT)
	ldi XL, LOW(IOPORT)

;DEBUG:
;	ld r18, X
;	ldi r18, 0x55
;	st X, r18
;	rjmp DEBUG

LOOP:

	ldi R19, 0x08 ;counter
	ld R18, X		;load switches
```

```
NEXT:
	st X, R18		;Echo on leds
	rcall WASTE		;delay
	rcall WASTE		;delay
	lsr R18			;shift right
	dec R19			;decrement counter
	cpi R19, 0		;check if 0
	BREQ LOOP		;if true check switches
	rjmp NEXT		;else keep shifting

;
;DELAY LOOP
;

WASTE:
	ldi r21, 0x00
	ldi r22, 0x00
EXTRA:
	inc r21
	cpi r21, 200
	brne DELAY_10ms
	ret
DELAY_10ms:
	nop			; do nothing
	nop
	nop
	nop
	nop
	nop
	nop
	nop
	nop
	nop
	nop
	nop
	nop
	nop
	nop
	nop
	nop
	nop
	nop
	nop
	nop
	nop
	nop
	nop
	nop
	nop
	nop
	nop
	nop
	nop
	nop
	nop
	nop
	nop
```

```
    nop
    inc r22              ;increase r18
    cpi r22, 0xFF ;check if r18 is 28
    brne DELAY_10ms      ;if not loop
    rjmp EXTRA
```

- Appendix:
  None