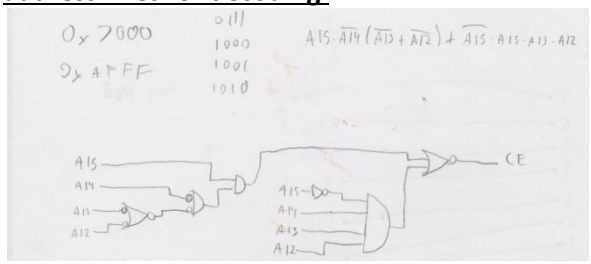
Lab 4

Reyna, Ricardo

Section 75C9

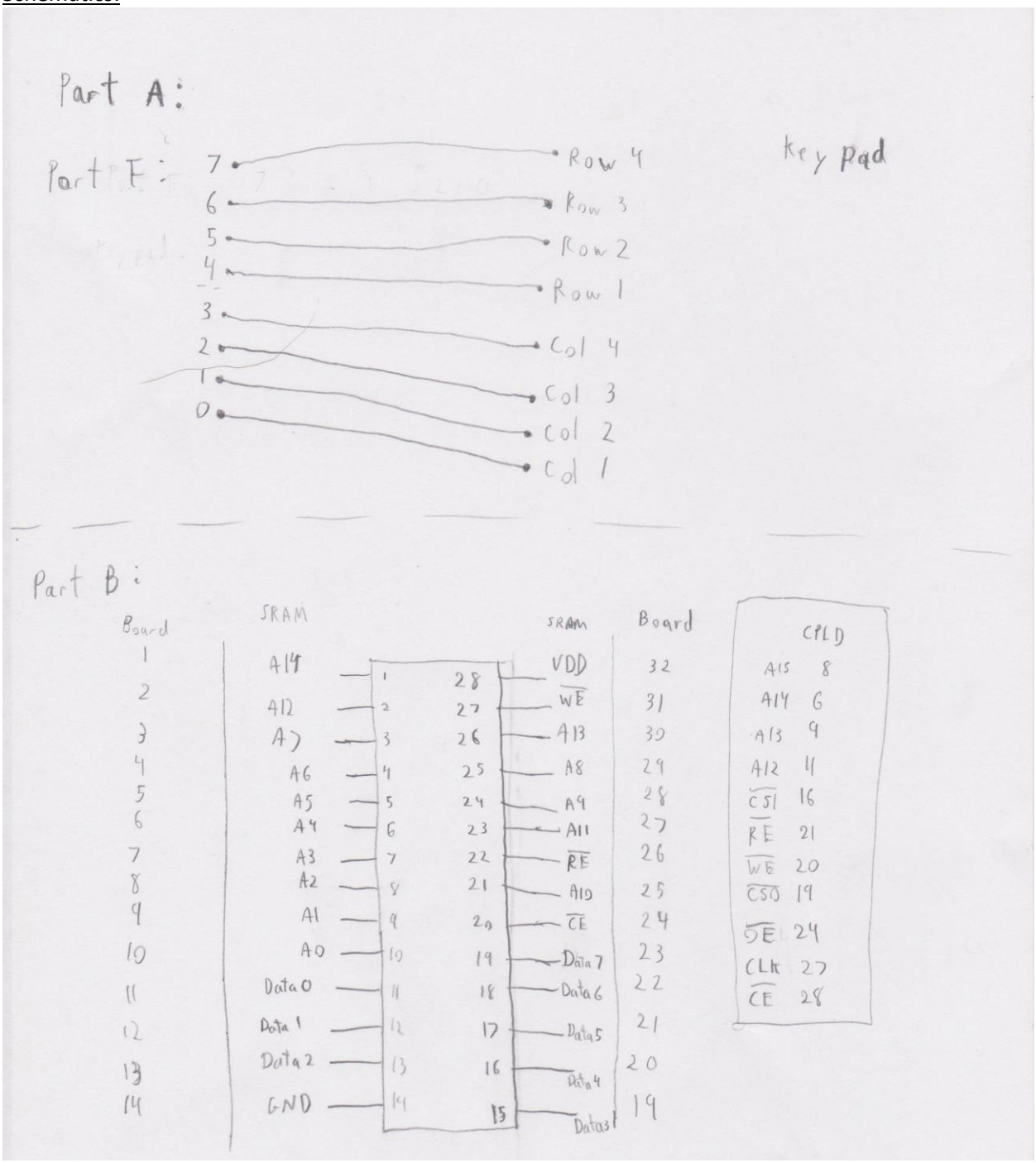
06/18/2015

- Prelab Questions:
  1. What is the size CS1 used in this lab?
  The size of CS1 is 64k.
  2. Which address lines MUST be present in the SRAM chip-enable equation for **full address decoding**?
  A23:0 would need to be present for full address decoding.
  3. Draw a schematic of a 16k x 8 RAM expansion starting at address 0x7000. **You may only use address lines for decoding.**

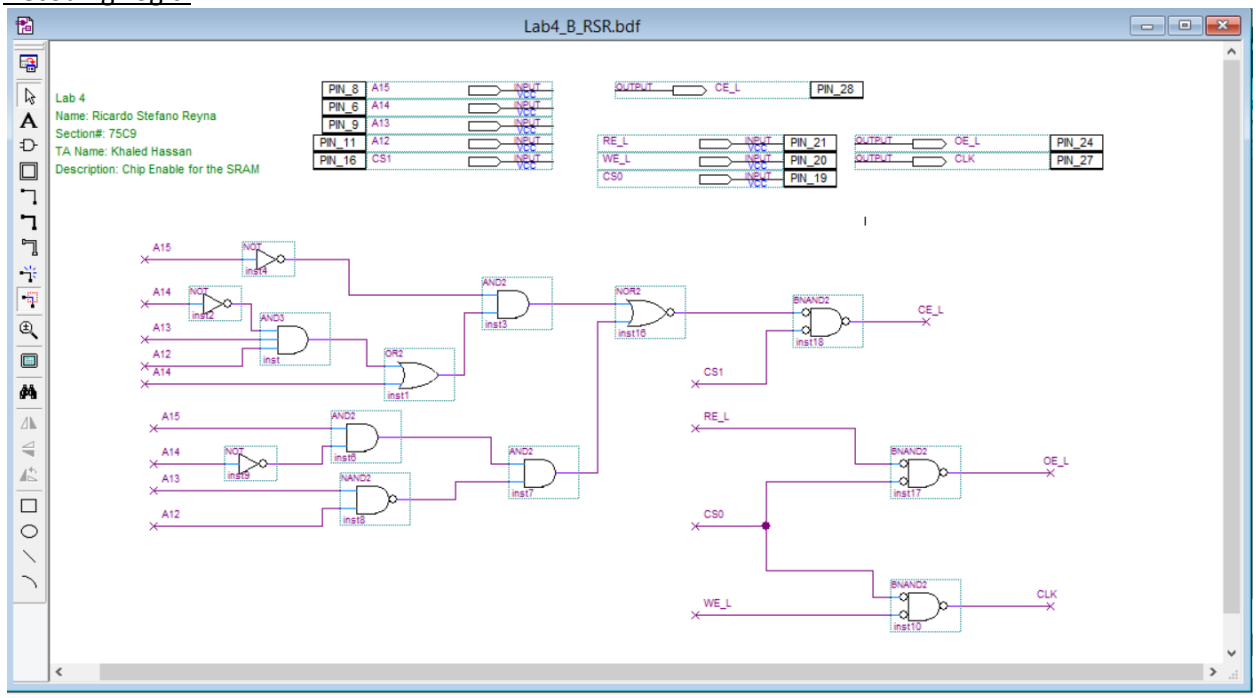


  4. On the uPAD Proto Base board, is the CPLD required for interfacing with the SRAM? If not, explain.
  You don't need the CPLD to decode the SRAM, we can use the CS pins.
  5. The simple memory test program described above is not very good. It checks for neighboring data pins that are shorted or left unconnected, but we have no idea if the address bus is working. Describe a procedure for testing the address lines. Are there any limitations to your procedure (for example, does your procedure test **all** of the address lines)? Explain.
  We can input the values of the addresses inside each address line. This way we can tell which addresses are connected poorly.
  6. In this lab, we configure CS1 to be bigger than the SRAM so that we can divide it for multiple devices. If we want to connect many devices to the memory bus, this is a good way to conserve XMEGA chip select outputs. Suppose we wanted to configure CS1 to span from 0x1A0000 to 0x1BFFFF instead. Is this a valid range for an XMEGA chip select? With the hardware on your uPAD Proto Base as it is now, can we divide this new CS1 the same way do in this lab? Why or why not?
  No, because the CS has a range of 64k and this span exceeds the 64k.
- Problems Encountered:
  I had trouble setting the PINnCTRL for the Keypad
- Future Work/Applications:
  I learned how to use the keypad and to setup the SRAM.

- Schematics:


Part A:
Part E:
7
6
5
4
3
2
1
0
Row 4
Row 3
Row 2
Row 1
Col 4
Col 3
Col 2
Col 1
key pad
Part B:
Board
SRAM
SRAM
Board
CPLD
A14
A12
A7
A6
A5
A4
A3
A2
A1
A0
Data 0
Data 1
Data 2
GND
VDD
WE
A13
A8
A9
A11
OE
A10
CE
Data 7
Data 6
Data 5
Data 4
Data 3
A15 8
A14 6
A13 9
A12 4
CS1 16
RE 21
WE 20
CS0 19
OE 24
CLK 27
CE 28

- Decoding Logic:



- Pseudo code:

Part A:
EBI Stuff
Set PINnCTRL
set r18 to 0xFF
check column
check each row
if it's true set r18 to that value

PartB:
EBI Stuff
Initiate Y with 0x1B3000
r16 = 0xAA or 0x55
write:
store r16 into Y
increase pointer
check if high Y is 0xB0
if no jump to write
set Y with 0x1B2FFF
set X with table
set r17 with 0
check:
increase Y
check if high Y is 0xB0
if true DONE

check r16 is 0xAA
if true jump to check
store address of Y into X
increase pointer X
check if r17 is 100 yet

- Program Code:

```
/*
 * Lab4_A_RSR.asm
 *
 *  Created: 6/15/2015 4:33:15 PM
 *   Author: stefano92
 Lab 4 part A
 Name: Ricardo Stefano Reyna
 Section#: 75C9
 TA: Khaled Hassan
 Description: This program will read from the keypad and echo in the LEDs
 */

.NOLIST
.include "ATxmega128A1Udef.inc"

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

.LIST
.org 0x0000
	rjmp MAIN	;go to MAIN

.org 0x200
MAIN:
	ldi r16, 0xFF
	out CPU_SPL, r16
	ldi r16, 0x3F
	out CPU_SPH, r16 //init stack pointer

	ldi r16, 0x17
	sts PORTH_DIR, r16 //set port pins as outputs for RE and ALE and WE

	ldi r16, 0x13
	sts PORTH_OUT, r16 //WE and RE is active low so it must be set

	ldi r16, 0xFF
	sts PORTJ_DIR, r16 //set datalines as outputs
	sts PORTK_DIR, r16 //set address lines as outputs

	ldi r16, 0x01
	sts EBI_CTRL, r16 //turn on 3 port SRAM ALE1 EBI

	ldi ZH, HIGH(EBI_CS0_BASEADDR) //all the set up for CS0, since EBI won't work
without it
	ldi ZL, LOW(EBI_CS0_BASEADDR)
	ldi r16, ((IOPORT>>8) & 0xF0)
	st Z+, r16
	ldi r16, ((IOPORT>>16) & 0xFF)
```

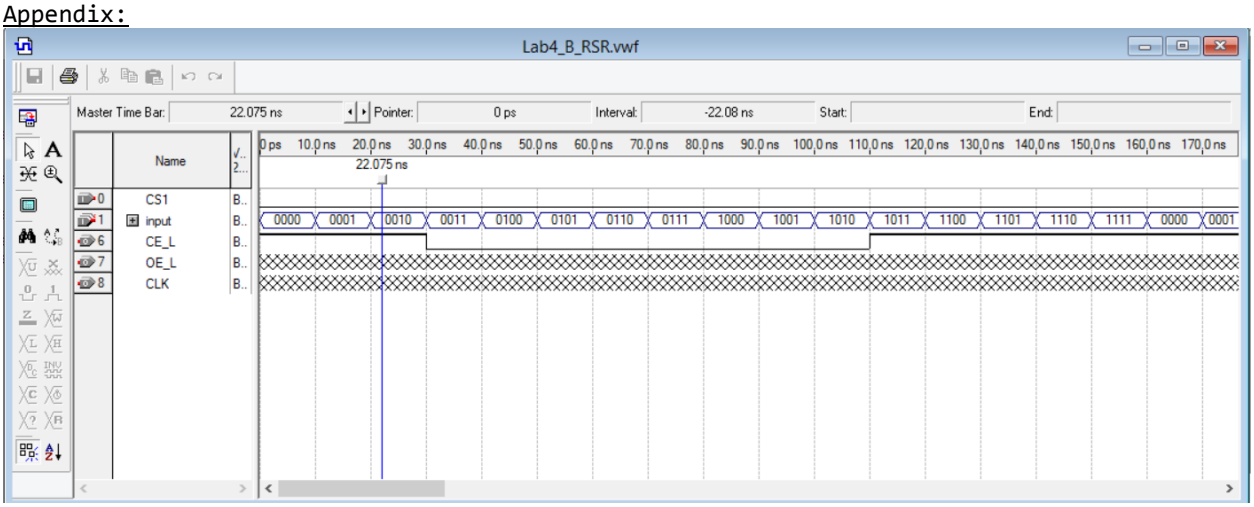
```
        st Z, r16
        ldi r16, 0x11
        sts EBI_CS0_CTRLA, r16
        ldi XH, HIGH(IOPORT)
        ldi XL, LOW(IOPORT)

/*|~~~~~~~~~~~~~~~~~~~~~~~~~
  |PROGRAM OF THE KEYBOARD
  |~~~~~~~~~~~~~~~~~~~~~~~~~
*/
        ldi r16, 0b011000
        sts PORTF_PIN4CTRL, r16
        sts PORTF_PIN5CTRL, r16
        sts PORTF_PIN6CTRL, r16
        sts PORTF_PIN7CTRL, r16
PARTA:
        rcall KEYPAD //this will overwrite r18, will not exit until an input is recieved
        st X, r18 //Hold r18
        rjmp PARTA

KEYPAD:
        push r16
        push r17
        ldi r18, 0xFF //fill 18 with a value that can't get returned.  Notice it does not
get pushed onto the stack!!!
        ldi r16, 0x0F
        sts PORTF_DIR, r16 //make sure the lower four bits are outputs and the upper are
inputs

        ldi r16, 0x0E //column 1
        sts PORTF_OUT, r16 //make the first column low and others high
        NOP
        NOP
        lds r17, PORTF_IN
        SBRS r17, 4   //row 1
        ldi r18, 0x1
        SBRS r17, 5 //row 2
        ldi r18, 0x4
        SBRS r17, 6 //row 3
        ldi r18,  0x7
        SBRS r17, 7 //row 4
        ldi r18, 0xE

        ldi r16, 0x0D //column 2
        sts PORTF_OUT, r16
        NOP
        NOP
        lds r17, PORTF_IN
        SBRS r17, 4   //row 1
        ldi r18, 0x2
        SBRS r17, 5   //row 2
        ldi r18, 0x5
        SBRS r17, 6   //row 3
        ldi r18,  0x8
        SBRS r17, 7   //row 4
        ldi r18, 0x0

        ldi r16, 0x0B //column 3
```

```
        sts PORTF_OUT, r16
        NOP
        NOP
        lds r17, PORTF_IN
        SBRS r17, 4   //row 1
        ldi r18, 0x3
        SBRS r17, 5   //row 2
        ldi r18, 0x6
        SBRS r17, 6   //row 3
        ldi r18,  0x9
        SBRS r17, 7   //row 4
        ldi r18, 0xF

        ldi r16, 0x07 // column 4
        sts PORTF_OUT, r16
        NOP
        NOP
        lds r17, PORTF_IN
        SBRS r17, 4   //row 1
        ldi r18, 0xA
        SBRS r17, 5   //row 2
        ldi r18, 0xB
        SBRS r17, 6   //row 3
        ldi r18,  0xC
        SBRS r17, 7   //row 4
        ldi r18, 0xD

DONE:
        pop r17
        pop r16
        RET
/*
 * Lab4_B_RSR.asm
 *
 *  Created: 6/16/2015 3:27:02 PM
 *   Author: stefano92
 Lab 4 part B
 Name: Ricardo Stefano Reyna
 Section#: 75C9
 TA: Khaled Hassan
 Description: This program will read from the keypad and echo in the LEDs
 */

.include "atxmega128a1udef.inc"
.set SRAMPORT = 0x1B0000

.org 0x0000
        rjmp MAIN

.dseg
.org 0x2300
Table1: .byte 200

.org 0x2700
Table2: .byte 200
.cseg

.org 0x100
```

```
MAIN:

	ldi r16, 0b00110111
	sts PORTH_DIR, r16 //set port pins as outputs for RE and ALE and WE CS1 and CS0

	ldi r16, 0b00110011
	sts PORTH_OUT, r16 //WE and RE is active low so it must be set

	ldi r16, 0xFF
	sts PORTJ_DIR, r16 //set datalines as outputs
	sts PORTK_DIR, r16 //set address lines as outputs

	ldi r16, 0x01
	sts EBI_CTRL, r16 //turn on 3 port SRAM ALE1 EBI

//CS1 STUFF

	ldi ZH, HIGH(EBI_CS1_BASEADDR) //set up CS1 for the SRAM
	ldi ZL, LOW(EBI_CS1_BASEADDR)
	ldi r16, ((SRAMPORT>>8) & 0xF0)
	st Z+, r16
	ldi r16, ((SRAMPORT>>16) & 0xFF)
	st Z, r16

	ldi r16, 0b00011101
	sts EBI_CS1_CTRLA, r16

PARTB1a:
	ldi r16, 0x1B		//RAMP Y for the SRAM
	out CPU_RAMPY, r16
	ldi YH, 0x30
	ldi YL, 0x00

	ldi r16, 0xAA

WRITEAA:
	st Y+, r16
	cpi YH, 0xB0
	brne WRITEAA

PARTB2a:
	ldi r16, 0x1B		//RAMP Y for the SRAM
	out CPU_RAMPY, r16
	ldi YH, 0x2F
	ldi YL, 0xFF
	ldi r17, 0x00

	ldi XL, low(Table1)		;load low bits of the table in X
	ldi XH, high(Table1) ;load high bits of the table in X

CHECKAA:
	ld r16, Y+ //Pre-increment
	ld r16, Y
	cpi YH, 0xB0
	breq PARTB1b
	cpi r16, 0xAA
	breq CHECKAA
	st X+, YL
```

```
        st X+, YH
        cpi r17, 100
        breq PARTB1b
        inc r17
        rjmp CHECKAA

PARTB1b:
        ldi r16, 0x1B        //RAMP Y for the SRAM
        out CPU_RAMPY, r16
        ldi YH, 0x00
        ldi YL, 0x30

        ldi r16, 0x55

WRITE55:
        st Y+, r16
        cpi YH, 0xB0
        brne WRITE55

PARTB2b:
        ldi r16, 0x1B        //RAMP Y for the SRAM
        out CPU_RAMPY, r16
        ldi YH, 0x2F
        ldi YL, 0xFF

        ldi XL, low(Table2)         ;load low bits of the table in X
        ldi XH, high(Table2) ;load high bits of the table in X

CHECK55:
        ld r16, Y+ //Pre-increment
        ld r16, Y
        cpi YH, 0xB0
        breq DONE
        cpi r16, 0x55
        breq CHECK55
        st X+, YL
        st X+, YH
        cpi r17, 100
        breq DONE
        inc r17
        rjmp CHECK55

DONE: rjmp DONE
```

- Appendix:


Lab4_B_RSR.vwf
Master Time Bar: 22.075 ns
Pointer: 0 ps
Interval: -22.08 ns
Start:
End:
Name
CS1
input
CE_L
OE_L
CLK
22.075 ns
0000
0001
0010
0011
0100
0101
0110
0111
1000
1001
1010
1011
1100
1101
1110
1111
0000
0001