**REQUIREMENTS NOT MET**

N/A

**PROBLEMS ENCOUNTERED**

N/A

**FUTURE WORK/APPLICATIONS**

<insert a brief paragraph describing how the topics covered in this lab could potentially be used for other applications>

THE ABOVE SHOULD BE LIMITED TO THE FIRST PAGE, AND NOTHING ELSE SHOULD BE INCLUDED, WHICH ALSO IMPLIES THAT THIS SENTENCE OF TEXT SHOULD BE REMOVED.

**PRE-LAB EXERCISES**

**i. For each SRAM configuration within the EBI system of the ATxmega128A1U microcontroller, to which address lines do you have external, physical access? Additionally, for the SRAM 3-PORT ALE1 configuration, is it possible to have external, physical access to any address lines above A15? Why or why not?**

**Normal SRAM Configuration:  
  
 No multiplexing:**

**Address lines 0 through 21**

**Multiplexing byte 0 and 1:**

**If ALE 1 is false, address lines 0 through 7 along with address lines 16 through 19.**

**If ALE 1 is true, address lines 0 through 15 along with 16 through 19 are available.**

**Multiplexing byte 0 and 2:**

**If ALE2 is false, address lines 0 through 15.**

**If ALE2 is true, address lines 0 through 23 are available.**

**Multiplexing byte 0,1 and 2:**

**If both ALE1 and ALE2 are false, address lines 0 through 7 are available.**

**If ALE1 is true and ALE2 is false, address lines 0 through 15 are available.**

**If ALE1 is true and ALE2 is true, address lines 0 through 23 are available.**

**LPC SRAM Configuration:**

**Multiplexing byte 0:**

**If ALE1 is false, address lines 8 through 19 are available.**

**Multiplexing byte 0 and 1:**

**If ALE1 and ALE2 are false, address lines 16 through 19 are available.**

**If ALE1 is true and ALE2 is false, address lines 0 through 7 along with 16 through 19**

**are available.**

**If ALE1 and ALE2 are true, address lines 0 through 19 are available to you.**

**It is possible to access address lines above A15, but only if at least 1 chip select line isn’t used.**

**ii. Describe what performing full address decoding and partial address decoding signifies. Provide examples of both types for [1] an SRAM chip and [2] either an input port or output port.**

**An address consists of two parts:  
 1.Actual memory address  
 2.Chip select lines  
 Full address decoding means you use all of the chip select lines.  
 Partial address decoding means you use a portion of the chip select lines  
   
 SRAM example:  
 You have a 10 bit address that you use to load/store data in a 1k byte SRAM. The SRAM is split into four smaller SRAM’s that are 250 bytes each.  
 With full address decoding, you would allocate the first 2 bits to select the SRAM chip, then you would use the rest of the bits to select the specific byte in the SRAM chip.  
 With partial address decoding, where you aren’t allowed to access the lower two chips, you would ignore the 9th bit, and use the 8th bit to address the SRAM chip. You would then use the rest of the bits to address the specific byte.  
   
 GPIO example:**

**iii. In theory, how many 8-bit I/O ports could be mapped to the external data memory space of an ATxmega128A1U microcontroller, assuming that at most one port utilizes any particular address range? Explain your answer, utilizing your response from Exercise ii.**

**16 million I/O ports could be mapped.**

**The EBI can support up to 16MB of SRAM.  
The address size for the EBI can go up to 24 bits.  
However, the first 16k addresses are reserved for internal SRAM  
that’s 0xFFFFFF – 0x004000 = 0xFFBFFF(16,760,831) possible ports**

**iv. Assume that an SRAM component with the same size as the one on the OOTB Memory Base has to be added to a different computing system containing an ATxmega128A1U but no OOTB Memory Base (i.e., this is not the OOTB µPAD computing system), with the first address of the SRAM starting at address 0x42 4000, instead of the address specified in the “Lab 4 – Hardware Expansion” quiz. Design, on paper, a hardware expansion for this new system, utilizing the same structure of steps laid out in this quiz. Use the same “overall” memory-mapping constraints presented within the quiz.** A picture containing text, sketch, handwriting, drawing

Description automatically generated **Figure 1: EBI expansion with address starting at 0x424000.  
The SRAM needs to be split because the 32k SRAM is bigger than the boundary.**

**v. Assume that some external 128 KB SRAM must be fully mapped to the data memory space of the ATxmega128A1U, with the first memory location of the SRAM corresponding to data memory address 0x7A 0000. Design, on paper, a hardware expansion for this new system, utilizing the same structure of steps laid out in the relevant pre-lab quiz. Use the same “overall” memory-mapping constraints presented within this quiz. Hint: Consider how many address signals are needed to make the SRAM fully addressable, and then consider how you will gain access to all these signals.**

A picture containing text, handwriting, sketch, diagram

Description automatically generated **Figure 2: EBI expansion with address starting at 0x7A0000.  
This version requires 17 address bits, so an extra latch is needed**

**vi. Assume that some 8-bit input port should be accessible via the 256 consecutive addresses starting at 0x23 A000 within the ATxmega128A1U data memory space. Design, on paper, a hardware expansion for this port, utilizing the same structure of steps laid out in the relevant pre-lab quiz. Use the same “overall” memory-mapping constraints presented within this quiz.**A picture containing text, handwriting, sketch, diagram

Description automatically generated **Figure 3: EBI expansion starting at address 0x23A000.**

**vii. Assume that some external 1 MB SRAM must be fully mapped to the data memory space of the ATxmega128A1U, with the first memory location of the SRAM corresponding to data memory address 0x60 0000. Design, on paper, a hardware expansion for this new system, utilizing the same structure of steps laid out in the relevant pre-lab quiz. Use the same “overall” memory-mapping constraints presented within this quiz, except that which is concerning the usage of the SRAM 3-PORT ALE1 EBI configuration. Hint: you might consider the use of some other EBI configuration.**

A picture containing text, sketch, diagram, plan

Description automatically generated **Figure 4: 1 MB SRAM expansion. This configuration uses 4 port no ALE. Allowing ALE1 and ALE2 to be used as address lines 16 and 17, and CS0 and CS1 to be used as address 18 and address 19.**

**viii. How would one go about utilizing the SRAM 4PORT NOALE mode? (Note: Looking at the Alternate Pin Functions only ports H, J, and K by default can be used with the EBI system. How can the other port be activated)?**

**(Tell him what values get loaded into what registers)**

**PSEUDOCODE/FLOWCHARTS**

**SECTION 2**

**A picture containing text, screenshot, font, number

Description automatically generated  
Figure 5: Flowchart for “lab4\_2.asm”**

**SECTION 3**

**A picture containing text, diagram, plan, technical drawing

Description automatically generated  
Figure 6: Main routine for “lab4\_3a.asm”**

**A picture containing text, diagram, screenshot, parallel

Description automatically generated  
Figure 7: Subroutines for “lab4\_3a.asm”**

**A picture containing text, screenshot, diagram, font

Description automatically generated  
Figure 8: Main loop and subroutines for “lab4\_3b.asm”**

**PROGRAM CODE**

**SECTION 2**

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;Lab 4, Section 2

;Name: Steven Miller

;Class #: 11318

;PI Name: Anthony Stross

;Description: Writes switch values from the external switches to the external LED's

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*INCLUDES\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

.include "ATxmega128a1udef.inc"

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*END OF INCLUDES\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*EQUATES\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

.EQU STACKEND = 0x3fff ;top of stack

.EQU STACKBEGIN = 0x2000 ;bottom of stack

.EQU INPUT = 0B00000000

.EQU OUTPUT = 0B11111111

.EQU IO\_START\_ADDRESS = 0X224000

.EQU SRAM\_START\_ADDRESS = 0X128000

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*END OF EQUATES\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*DEFS\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*END OF DEFS\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;\*\*\*\*\*\*\*\*\*\*\*MEMORY CONFIGURATION\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;\*\*\*\*\*\*\*\*\*\*\*END OF MEMORY CONFIGURATION\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;\*\*\*\*\*\*\*\*\*\*\*MAIN PROGRAM\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

.CSEG

.ORG 0X0000

RJMP MAIN

.ORG 0X0200

MAIN:

;INITIALIZE STACK POINTER

LDI R16, LOW(STACKBEGIN)

out CPU\_SPL, R16

LDI R16, HIGH(STACKEND)

OUT CPU\_SPH, R16

RCALL EBI\_INIT

;LOAD IO ADDRESS

LDI XL, BYTE1(IO\_START\_ADDRESS)

LDI XH, BYTE2(IO\_START\_ADDRESS)

LDI R16, BYTE3(IO\_START\_ADDRESS)

OUT CPU\_RAMPX, R16

LOOP:

ld r16, x

st x,r16

RJMP LOOP

END:

RJMP END

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*SUBROUTINES\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;NAME:EBI\_INIT

;PURPOSE: INITIALIZES EBI SYSTEM

;REGISTERS AFFECTED: EBI\_CNTRL

;INPUTS AFFECTED:PORTJ

;OUTPUTS AFFECTED:PORTK,RE, CS0, CS2

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

EBI\_INIT:

;SAVE RELATIVE REGISTERS

PUSH R16

;INIITIALIZE EBI CONTROL SIGNALS

LDI R16,0B01010011

STS PORTH\_OUTSET, R16

LDI R16,0B00000100

STS PORTH\_OUTCLR, R16

;SET EBI CONTROL SIGNALS TO OUTPUT

LDI R16, 0B01010111

STS PORTH\_DIRSET, R16

;SET ADDRESS SIGNALS TO OUTPUT

LDI R16,0XFF

STS PORTK\_DIRSET, R16

;SET EBI TYPE TO 3 PORT SRAM ALE1

LDI R16, 0B00000001

STS EBI\_CTRL, R16

;CONFIGURE CS0

LDI R16, 0B00011101

STS EBI\_CS0\_CTRLA, R16

LDI R16, BYTE2(SRAM\_START\_ADDRESS )

STS EBI\_CS0\_BASEADDR, R16

LDI R16, BYTE3(SRAM\_START\_ADDRESS )

STS EBI\_CS0\_BASEADDR+1, R16

;CONFIGURE CS2

LDI R16, 0B00000001

STS EBI\_CS2\_CTRLA, R16

LDI R16, BYTE2(IO\_START\_ADDRESS)

STS EBI\_CS2\_BASEADDR,R16

LDI R16, BYTE3(IO\_START\_ADDRESS)

STS EBI\_CS2\_BASEADDR+1,R16

;RESTORE REGISTERS

POP R16

RET

**SECTION 3A**

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;Lab 4, Section 3a

;Name: Steven Miller

;Class #: 11318

;PI Name: Anthony Stross

;Description: Loads data from "sram\_data.asm" from program memory into sram, then loads that data onto the output leds

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*INCLUDES\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

.include "ATxmega128a1udef.inc"

.include "SRAM\_DATA.ASM"

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*END OF INCLUDES\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*EQUATES\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

.EQU STACKEND = 0x3fff ;top of stack

.EQU STACKBEGIN = 0x2000 ;bottom of stack

.EQU INPUT = 0B00000000

.EQU OUTPUT = 0B11111111

.EQU IO\_START\_ADDRESS = 0X224000

.EQU SRAM\_START\_ADDRESS = 0X128000

.EQU SRAM\_END\_ADDRESS = 0X12FFFF

.EQU SYSCLK = 2000000

.EQU PRESCALAR = 64

.EQU DESIREDPERIOD = .30

.EQU RECIPROCAL = 1/.30

.EQU OFFSET = 0

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*END OF EQUATES\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*DEFS\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*END OF DEFS\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;\*\*\*\*\*\*\*\*\*\*\*MEMORY CONFIGURATION\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;\*\*\*\*\*\*\*\*\*\*\*END OF MEMORY CONFIGURATION\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;\*\*\*\*\*\*\*\*\*\*\*MAIN PROGRAM\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

.CSEG

.ORG 0X0000

RJMP MAIN

.ORG 0X0200

MAIN:

;initialize stack pointer

LDI R16, LOW(STACKBEGIN)

out CPU\_SPL, R16

LDI R16, HIGH(STACKEND)

OUT CPU\_SPH, R16

;INITIALIZE EBI AND TIMER

RCALL EBI\_INIT

RCALL TIMER\_INIT

;LOAD IO ADDRESS INTO X

LDI XL, BYTE1(IO\_START\_ADDRESS)

LDI XH, BYTE2(IO\_START\_ADDRESS)

LDI R16, BYTE3(IO\_START\_ADDRESS)

OUT CPU\_RAMPX, R16

;LOAD SRAM ADDRESS INTO Y

LDI YL, BYTE1(SRAM\_START\_ADDRESS)

LDI YH, BYTE2(SRAM\_START\_ADDRESS)

LDI R16, BYTE3(SRAM\_START\_ADDRESS)

OUT CPU\_RAMPY, R16

;LOAD SRAM DATA ADDRESS INTO Z

LDI ZL, BYTE1(SRAM\_DATA<<1)

LDI ZH, BYTE2(SRAM\_DATA<<1)

LDI R16, BYTE3(SRAM\_DATA<<1)

OUT CPU\_RAMPZ, R16

STORE\_LOOP:

;LOAD DATA AT Z INTO R16

ELPM R16, Z+

;STORE DATA AT R16 INTO Y

ST Y+, R16

;CHECK IF Z IS NULL

ELPM R16, Z

CPI R16, 0X00

BRNE STORE\_LOOP

;RESET Y POINTER

LDI YL, BYTE1(SRAM\_START\_ADDRESS)

LDI YH, BYTE2(SRAM\_START\_ADDRESS)

LDI R16, BYTE3(SRAM\_START\_ADDRESS)

OUT CPU\_RAMPY, R16

LOAD\_LOOP:

;CHECK IF END OF SRAM

LDS R16, CPU\_RAMPY

CPI R16, 0X13

BREQ END

;LOAD DATA FROM Y INTO R16

LD R16, Y+

;STORE R16 INTO X

ST X, R16

;START TIMER

RCALL TIMER

RJMP LOAD\_LOOP

end:

rjmp end

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;NAME:EBI\_INIT

;PURPOSE: INITIALIZES EBI SYSTEM

;REGISTERS AFFECTED: EBI\_CNTRL

;INPUTS AFFECTED:N/A

;OUTPUTS AFFECTED:PORTK,RE, CS0, CS2

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

EBI\_INIT:

;SAVE RELATIVE REGISTERS

PUSH R16

;INIITIALIZE EBI CONTROL SIGNALS

LDI R16,0B01010011

STS PORTH\_OUTSET, R16

LDI R16,0B00000100

STS PORTH\_OUTCLR, R16

;SET EBI CONTROL SIGNALS TO OUTPUT

LDI R16, 0B01010111

STS PORTH\_DIRSET, R16

;SET ADDRESS SIGNALS TO OUTPUT

LDI R16,0XFF

STS PORTK\_DIRSET, R16

;SET EBI TYPE TO 3 PORT SRAM ALE1

LDI R16, 0B00000001

STS EBI\_CTRL, R16

;CONFIGURE CS0

LDI R16, 0B00011101

STS EBI\_CS0\_CTRLA, R16

LDI R16, BYTE2(SRAM\_START\_ADDRESS )

STS EBI\_CS0\_BASEADDR, R16

LDI R16, BYTE3(SRAM\_START\_ADDRESS )

STS EBI\_CS0\_BASEADDR+1, R16

;CONFIGURE CS2

LDI R16, 0B00000001

STS EBI\_CS2\_CTRLA, R16

LDI R16, BYTE2(IO\_START\_ADDRESS)

STS EBI\_CS2\_BASEADDR,R16

LDI R16, BYTE3(IO\_START\_ADDRESS)

STS EBI\_CS2\_BASEADDR+1,R16

;RESTORE REGISTERS

POP R16

RET

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;NAME:TIMER\_INIT

;PURPOSE: INITIALIZES TIMER ON PORT C

;REGISTERS AFFECTED: TCC0\_CNT, TCC0\_PER, TCC0\_INTFLAGS

;INPUTS AFFECTED:N/A

;OUTPUTS AFFECTED:N/A

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

TIMER\_INIT:

PUSH R16

;INITIALIZE COUNT REGISTER

LDI R16, 0B00000000

STS TCC0\_CNT, R16

STS TCC0\_CNT+1, R16

;INITIALIZE PERIOD REGISTER

LDI r16,low(((sysclk/prescalar)/reciprocal)+offset)

STS TCC0\_PER, r16

LDI r16,high(((sysclk/prescalar)/reciprocal)+offset)

STS TCC0\_PER+1,r16

;CLEAR INTERRUPT FLAGS

LDI R16, 0B00000001

STS TCC0\_INTFLAGS, R16

POP R16

RET

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;NAME:TIMER

;PURPOSE: THE ACTUAL TIMER

;REGISTERS AFFECTED: TCC0\_CNT, TCC0\_CTRLA

;INPUTS AFFECTED:N/A

;OUTPUTS AFFECTED:N/A

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

TIMER:

PUSH R16

;INITIALIZE CLKSEL

LDI R16, TC\_CLKSEL\_DIV64\_gc

STS TCC0\_CTRLA, R16

TIMER\_LOOP:

;check ov flag

lds r17,TCC0\_INTFLAGS

;branch if we have overflow

sbrs r17,TC0\_OVFIF\_bp

rjmp TIMER\_LOOP

;CLEAR COUNT

LDI R16, 0B00000000

STS TCC0\_CNT, R16

STS TCC0\_CNT+1,R16

;TURN OFF TIMER

LDI R16, 0B00000000

STS TCC0\_CTRLA, R16

;CLEAR OVF

LDI R16, TC0\_BYTEM0\_bm

STS TCC0\_INTFLAGS, R16

POP R16

RET

**SECTION 3B**  
;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;Lab 4, Section 3b

;Name: Steven Miller

;Class #: 11318

;PI Name: Anthony Stross

;Description: Writes switch values from the external switches to the external LED's

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*INCLUDES\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

.include "ATxmega128a1udef.inc"

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*END OF INCLUDES\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*EQUATES\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

.EQU STACKEND = 0x3fff ;top of stack

.EQU STACKBEGIN = 0x2000 ;bottom of stack

.EQU INPUT = 0B00000000

.EQU OUTPUT = 0B11111111

.EQU IO\_START\_ADDRESS = 0X224000

.EQU SRAM\_START\_ADDRESS = 0X128000

.EQU READ\_LOCATION = 0X128000

.EQU WRITE\_LOCATION = 0X128100

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*END OF EQUATES\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*DEFS\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*END OF DEFS\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;\*\*\*\*\*\*\*\*\*\*\*MEMORY CONFIGURATION\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;\*\*\*\*\*\*\*\*\*\*\*END OF MEMORY CONFIGURATION\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;\*\*\*\*\*\*\*\*\*\*\*MAIN PROGRAM\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

.CSEG

.ORG 0X0000

RJMP MAIN

.ORG 0X0200

MAIN:

;INITIALIZE STACK POINTER

LDI R16, LOW(STACKBEGIN)

out CPU\_SPL, R16

LDI R16, HIGH(STACKEND)

OUT CPU\_SPH, R16

RCALL EBI\_INIT

;LOAD SRAM ADDRESS INTO Y

LDI YL, BYTE1(SRAM\_START\_ADDRESS)

LDI YH, BYTE2(SRAM\_START\_ADDRESS)

LDI R16, BYTE3(SRAM\_START\_ADDRESS)

OUT CPU\_RAMPY, R16

;LOAD WRITE ADDRESS INTO Z

LDI ZL, BYTE1(WRITE\_LOCATION )

LDI ZH, BYTE2(WRITE\_LOCATION )

LDI R16, BYTE3(WRITE\_LOCATION )

OUT CPU\_RAMPZ, R16

;INITIALIZE SRAM ADDRESSES

LDI R16, 0XE

ST Y,R16

LDI R16, 0XF

ST Z,R16

LOOP:

;READ FROM SRAM READ ADDRESS

LD R16,Y

;LOAD R16 WITH 0XF

LDI R16, 0XF

;WRITE TO SRAM WRITE ADDRESS

ST Z, r16

RJMP LOOP

END:

RJMP END

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*SUBROUTINES\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;NAME:EBI\_INIT

;PURPOSE: INITIALIZES EBI SYSTEM

;REGISTERS AFFECTED: EBI\_CNTRL

;INPUTS AFFECTED:PORTJ

;OUTPUTS AFFECTED:PORTK,RE, CS0, CS2

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

EBI\_INIT:

;SAVE RELATIVE REGISTERS

PUSH R16

;INIITIALIZE EBI CONTROL SIGNALS

LDI R16,0B01010011

STS PORTH\_OUTSET, R16

LDI R16,0B00000100

STS PORTH\_OUTCLR, R16

;SET EBI CONTROL SIGNALS TO OUTPUT

LDI R16, 0B01010111

STS PORTH\_DIRSET, R16

;SET ADDRESS SIGNALS TO OUTPUT

LDI R16,0XFF

STS PORTK\_DIRSET, R16

;SET EBI TYPE TO 3 PORT SRAM ALE1

LDI R16, 0B00000001

STS EBI\_CTRL, R16

;CONFIGURE CS0

LDI R16, 0B00011101

STS EBI\_CS0\_CTRLA, R16

LDI R16, BYTE2(SRAM\_START\_ADDRESS )

STS EBI\_CS0\_BASEADDR, R16

LDI R16, BYTE3(SRAM\_START\_ADDRESS )

STS EBI\_CS0\_BASEADDR+1, R16

;CONFIGURE CS2

LDI R16, 0B00000001

STS EBI\_CS2\_CTRLA, R16

LDI R16, BYTE2(IO\_START\_ADDRESS)

STS EBI\_CS2\_BASEADDR,R16

LDI R16, BYTE3(IO\_START\_ADDRESS)

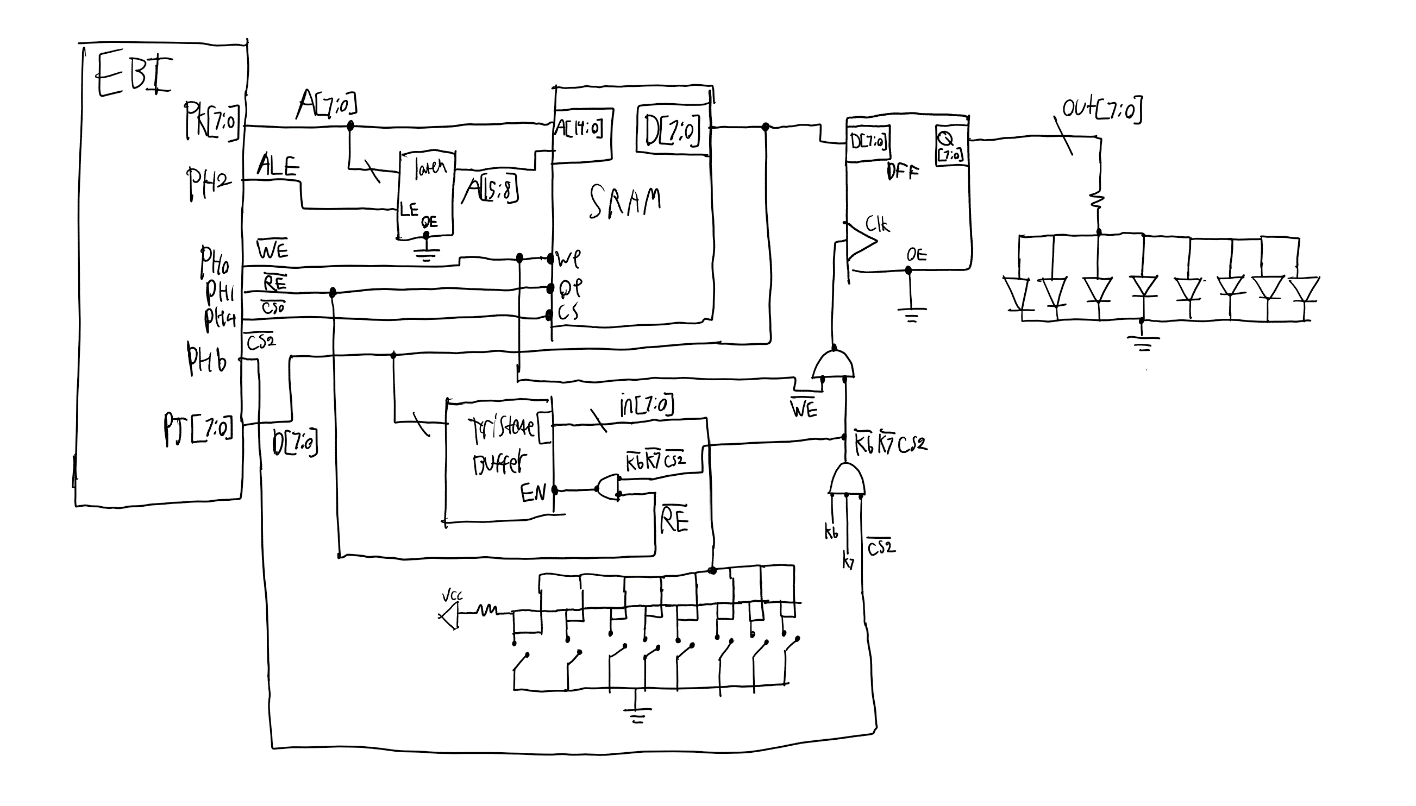
STS EBI\_CS2\_BASEADDR+1,R16

;RESTORE REGISTERS

POP R16

RET

**APPENDIX**

**  
Figure 1: SRAM EBI Extension Schematic. INCORRECT, EDIT LATER.**

**A picture containing line, plot, diagram, text

Description automatically generated  
FIGURE XX: EBI ON LOGIC ANALYZER. (ANNOTATE LATER)  
A picture containing text, diagram, screenshot, plan

Description automatically generated  
FIGURE XX: LOGIC SCHEMATIC**