/\* HW2- Question 6.23

Name: Pengzhao Zhu

Section#: 112D

TA Name: Chris Crary

Description: This program find the largest of thirty-two 8-bit unsigned numbers in 32 successive RAM memory locations.

It then places the answer in the 33rd location (Result).

I didn't program any data onto the SRAM (can't find a short way to do it and I didn't have time as this is almost 8am), so I can't show any results. but this code works. this code points to where 'filtable' starts (lets assume that is where the 32 data points are at) and run the loop 32 times.After 32 times. Y pointer increments and program the

result there. I tried my best, please take it easy.

\*/

.include "ATxmega128A1Udef.inc"

.list

.equ stack = 0x2FFF

.org 0x0000 ;start at address 0x000 rjmp MAIN ;jump to main code

.dseg

.org 0x3000

store: .byte 1

.cseg

.org 0x200

MAIN:

ldi YL, low(stack)

out CPU\_SPL, YL

ldi YL, high(stack)

out CPU\_SPH, YL ;initialize high byte of stack pointer

rcall routine

DONE:

rjmp DONE

routine:

push YL

push YH

push XL

push XH

push r16

push r17

push r20

ldi YL, low(filtable) ;Y pointer point to where the table will start

ldi YH, high(filtable) ;low and high bytes

ldi XL, low(store)

ldi XH, high(store)

ldi r17, 32 ;counter

Third:

ld r16, Y+

mov r20, r16

ld r16, X

cp r20, r16

brlo store16

st X, r20

rjmp compare

store16:

st X, r16

compare:

cpi r17, 0

breq startpop

dec r17

rjmp Third

startpop:

ld r16, X

st Y, r16

pop r20

pop r17

pop r16

pop XH

pop XL

pop YH

pop YL

ret

/\* HW2- Stack Pointer

Name: Pengzhao Zhu

Section#: 112D

TA Name: Chris Crary

Description: Stack Pointer Example. Trying to get myself familiar how to use stack pointers.

\*/

.include "ATxmega128A1Udef.inc"

.list

.equ stack=0x2FFF

.EQU length= 10 ;test vector length

.ORG 0x100

VECTOR: .DB 1,2,3,4, 5, 6, 7, 8, 9 ;trying to mess around the table with pointers. but couldn't do it

.org 0x00

rjmp MAIN

.org 0x200

MAIN:

ldi YL,low(stack)

out CPU\_SPL, YL

ldi YH, high(stack)

out CPU\_SPH, YH

ldi XL, 0x37 ;push 0x37

push XL

ldi XH, 0xAB ;push 0xAB

push XH

ldi YL, 0x12 ;push EF12

push YL

ldi YH, 0xEF

rcall routine ;call the subroutine

rjmp DONE

DONE:

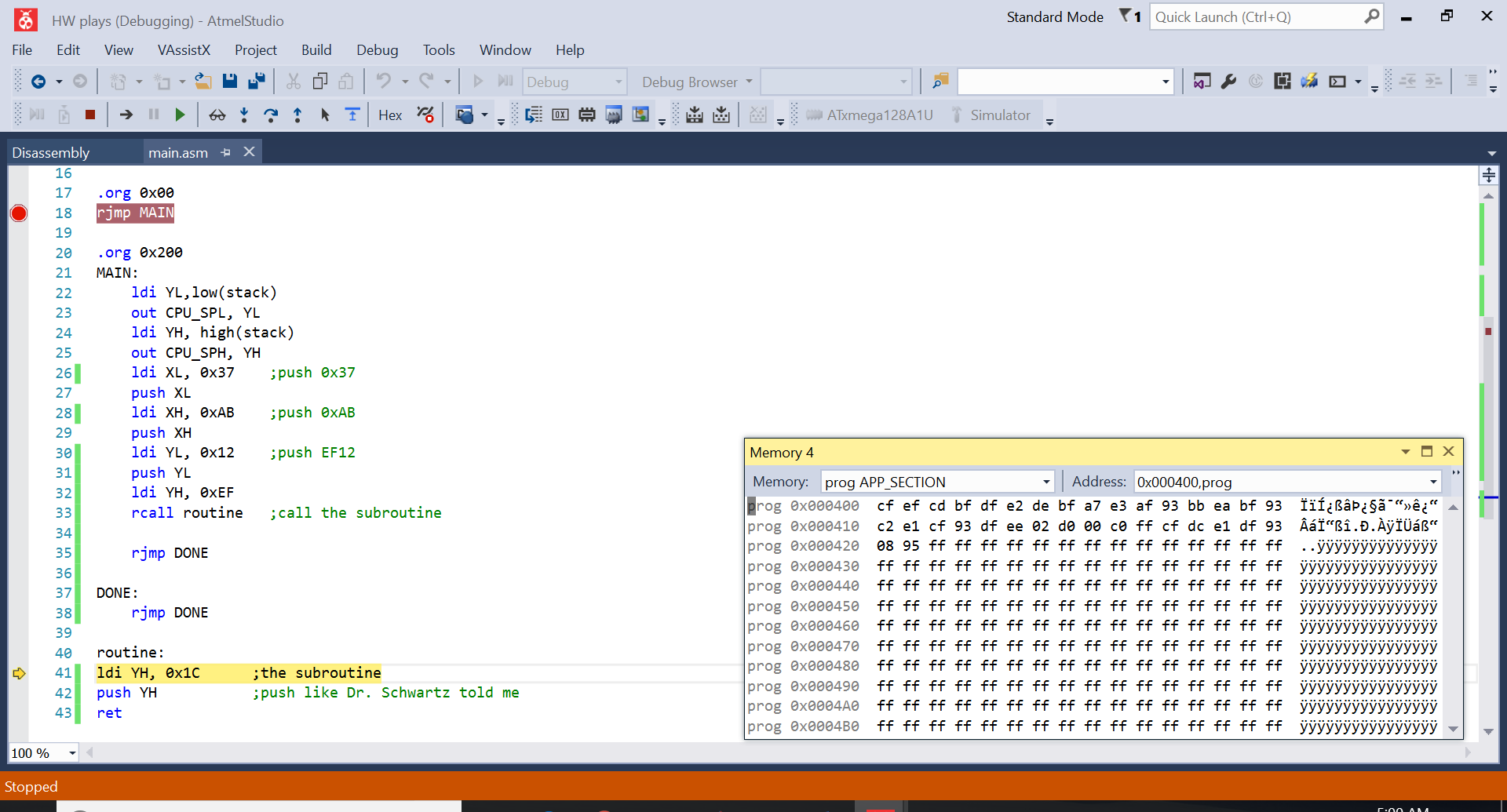
rjmp DONE

routine:

ldi YH, 0x1C ;the subroutine

push YH ;push like Dr. Schwartz told me

ret



Yeah, it didn’t work. I tried my best.