CPE301 – SPRING 2019

Design Assignment 1B

Student Name: Biruk Gebremeskel

Student #: 5002212730

Student Email: gebreb1@unlv.nevada.edu

Primary Github address: https://github.com/gebreb1

Directory: DA1B

Design Assignment 1B:

Q: Write, simulate, and demonstrate using AVRStudio6/7 an assembly code for the AVR ATMEGA328p microcontroller that performs the following functions:

1. Store 99 numbers starting from the STARTADDS=0x0200 location. Populate the value of the memory location by adding high(STARTADDS) and low(STARTADDS). Use the X/Y/Z registers as pointers to fill up 99 numbers that are greater than 10 and less than 255. The numbers can be consecutive or random numbers.

2. Use X/Y/Z register addressing to parse through the 99 numbers, if the number is divisible by 3 store the number starting from memory location 0x0400, else store at location starting at 0x0600.

3. Use X/Y/Z register addressing to simultaneously add numbers from memory location 0x0400 and 0x0600 and store the sums at R16:R17 and R18:R19 respectively. Pay attention to the carry overflow.

4. Verify your algorithm and answers using C or any high-level program.

5. Determine the execution time @ 16MHz/#cycles of your algorithm using the simulation

**Answer**

1. **Code**

; Assignment1B.asm

; Created: 2/23/2019 1:33:36 PM

; Author : Biruk Gebremeskel

; Instructor : Dr. Venki Muthukumar

RESET:

.equ STARTADDS = 0x0200 ; starting adress

.equ count = 109 ; the counter to 109

.org 0

clr r0 ; clearing r0 resister

ldi XL, low(STARTADDS) ; XL = STARTADDS[7:0]

ldi XH, high(STARTADDS) ; XH = STARTADDS[15:8]

ldi YL, low(0x400) ; YL = 0x00

ldi YH, high(0x400) ; YH = 0x04

ldi ZL, low(0x600) ; ZL = 0x00

ldi ZH, high(0x600) ; ZH = 0x06

; initializing count registers

ldi r21, count ; r21 = 109

ldi r22, 10 ; r22 = 10

ldi r20, 3 ; r20 = 3

; Clearing Registers for sums

clr r0 ; r0 = 0

clr r16 ; r16 = 0

clr r17 ; r17 = 0

clr r18 ; r18 = 0

clr r19 ; r19 = 0

clr r5 ; r5 = 0

inc r5 ; r5 = 1

START:

; Adding lower and upper address bits to be stored

mov r1, XL ; R1 = XL

add r1, XH ; R1 = XL + XH

mov r3, r1 ; R3 = R1

st x+, r3 ; [x] = XL + XH

; Checking if divisible by 3

DIVBYTHREE:

cp r1, r20 ; if(R1 < 3)

brlo DIVBAD ; go to DIVBAD

sub r1, r20 ; R1 = R1 - R20

cp r1, r0 ; if (R1 == 0)

breq DIVGOOD ; go to DIVGOOD

rjmp DIVBYTHREE ; go to DIVBYTHREE

DIVGOOD:

st Z+, r3 ;[Z] = XL + XH

add r16, r3 ; R16 = R16 + R3 (lower)

adc r17, r0 ; R17 = R17 + R0 + C (upper)

rjmp DONE ; go to DONE

DIVBAD:

st Y+, r1 ; [Y] = XL + XH

add r18, r3 ; R18 = R18 + R3 (lower)

adc r19, r0 ; R19 = R19 + R0 + C (upper)

rjmp DONE ; go to DONE

DONE:

cp r21, r0 ; if(count == 0)

breq NOTYET ; go to NOTYET

sub r21, r5 ; R21 = R21 - R5

brne START ; if(R21 != 0) go to START

NOTYET:

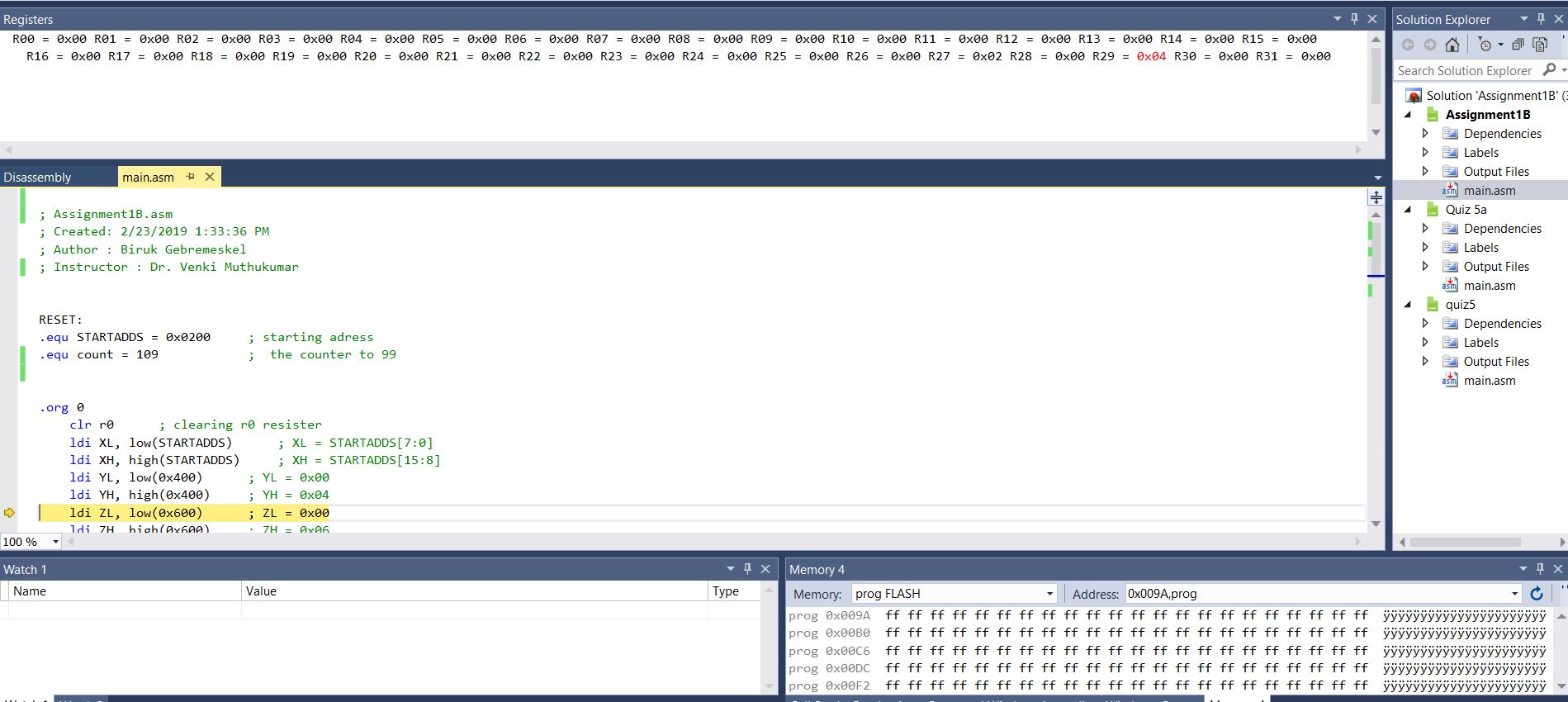
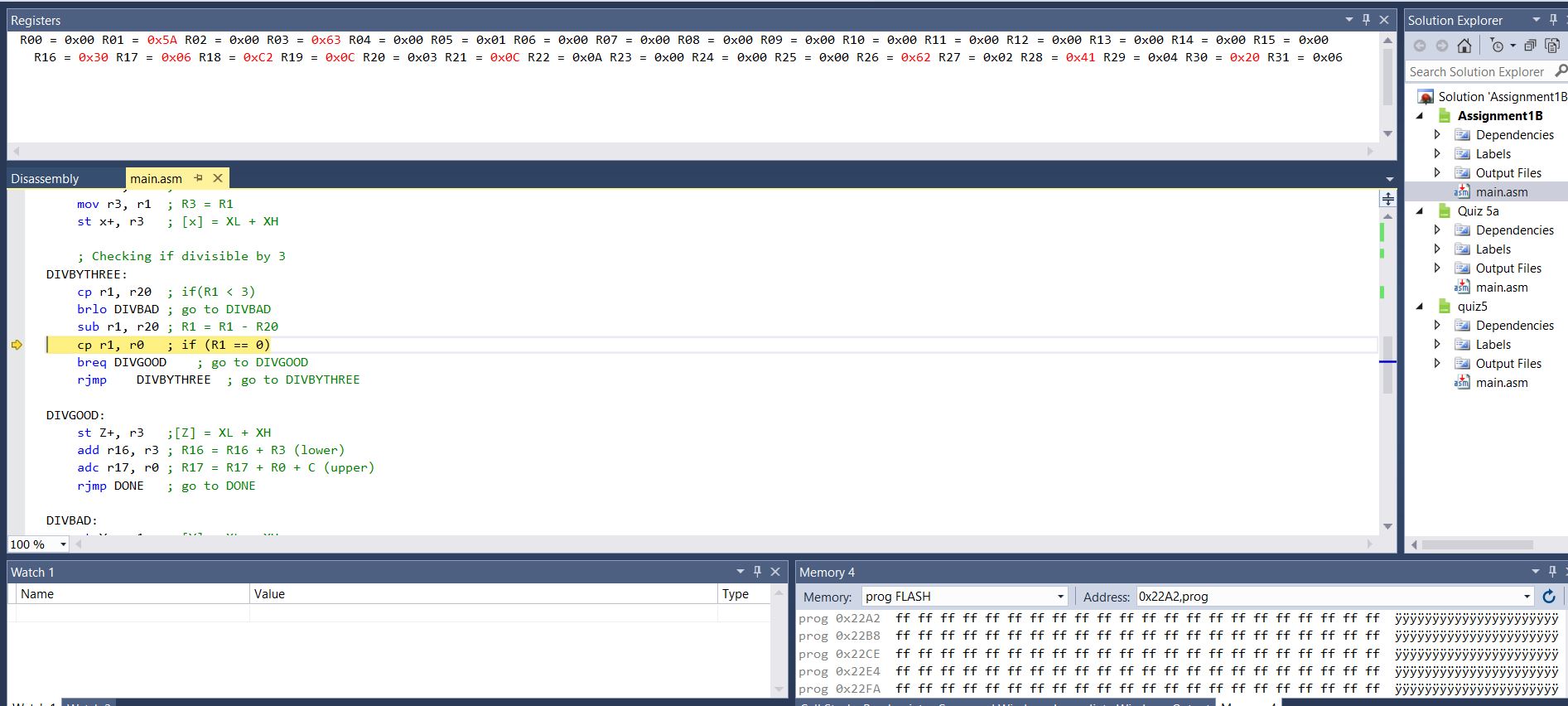
sub r22, r5 ; R22 = R22 - R5

brne START ; if( R22 != 0) go to START

FIN:

rjmp RESET ; Reset

1. **Screenshots**



1. **GITHUB Link**

<https://github.com/gebreb1/DA1B>

1. **Execution time calculation**

E = C\*T

Where T = 1/F = 1/(16MHz) = 0.625us

41 instruction repeated 99 times, the approximate execution time is:

E = 99 \* 41 \* 0.625 = 2.54ms

“This assignment submission is my own, original work”.

Biruk Gebremeskel