Student Name: Ricky Perez

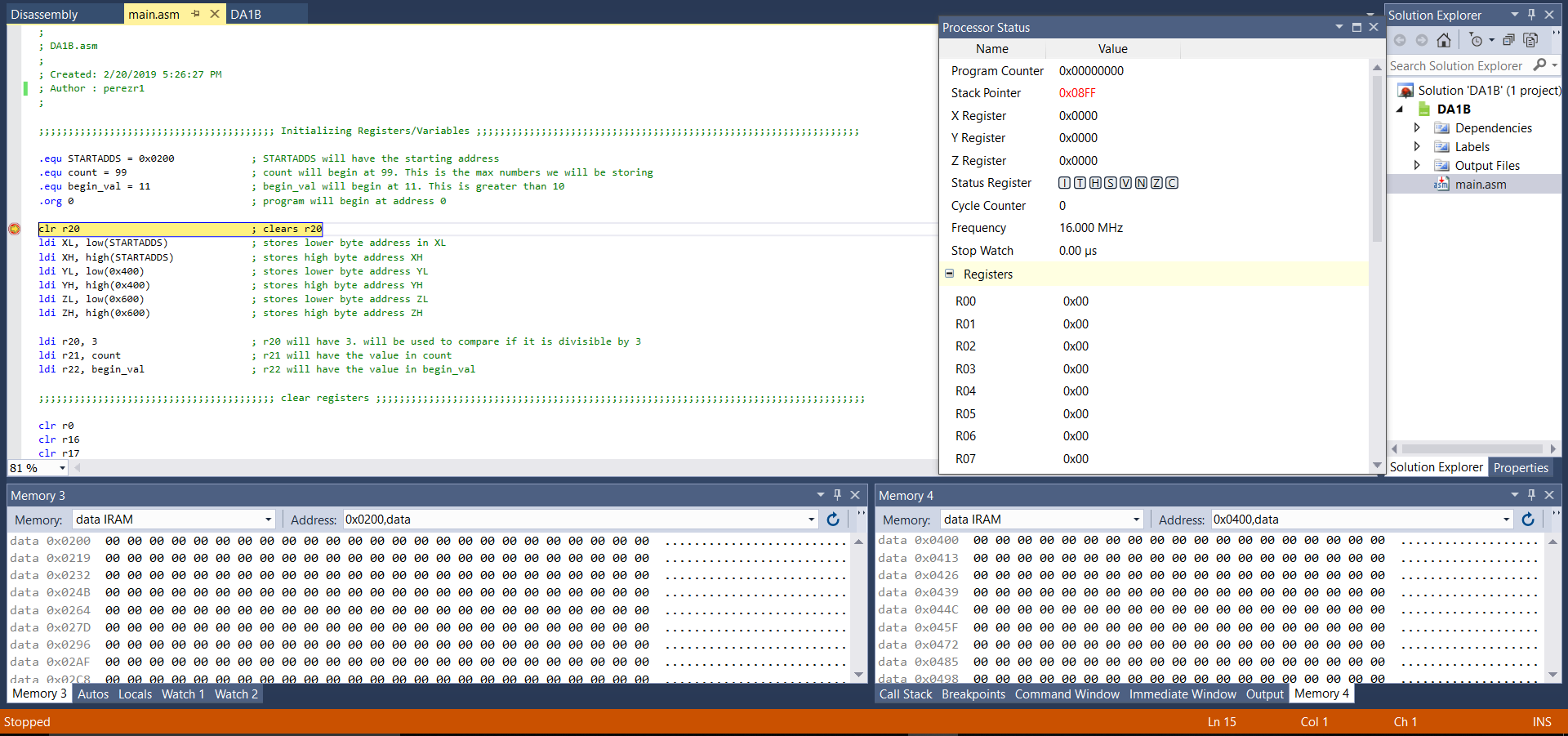
Student #: 5002297620

Student Email: perezr1@unlv.nevada.edu

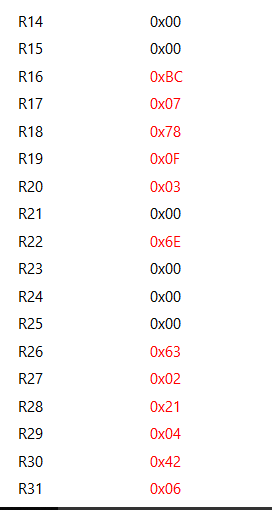
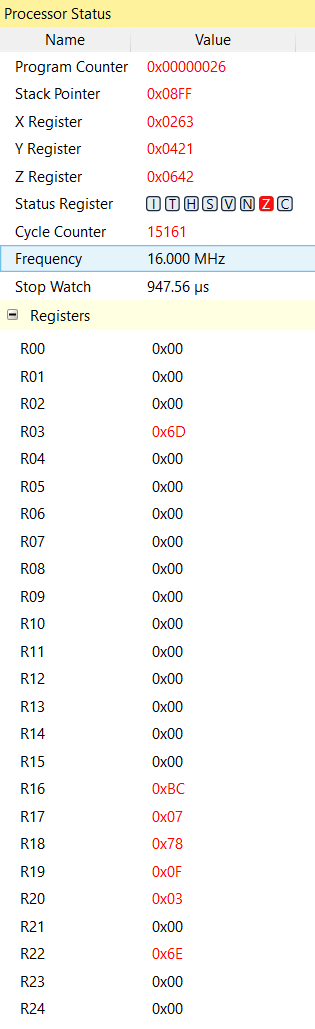
Primary Github address: <https://github.com/RickyPerez79/submission_da.git>

In this program we are to populate either 99 continuous or random numbers starting at address 0x0200(STARTADDS). Using the X, Y, and Z registers we were to store the values that were divisible by 3 starting at 0x0400 address location and the values that were not divisible by 3 starting at 0x0600 address location. Also we are to store the sum of the values that are divisible by 3 in r16:r17 and the values that are not divisible by 3 in r18:19.

Beginning of program @16MHz frequency:



End of Program results:



At 16MHz frequency my clock cycle was 15,161 and my stop watch was 947.56µs.

DA1B ASM Code:

;

; DA1B.asm

;

; Created: 2/20/2019 5:26:27 PM

; Author : perezr1

;

;;;;;;;;;;;;;;;;;; Initializing Registers/Variables ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

.equ STARTADDS = 0x0200 ; STARTADDS will have the starting address

.equ count = 99 ; count will begin at 99. This is the max numbers we will be storing

.equ begin\_val = 11 ; begin\_val will begin at 11. This is greater than 10

.org 0 ; program will begin at address 0

clr r20 ; clears r20

ldi XL, low(STARTADDS) ; stores lower byte address in XL

ldi XH, high(STARTADDS) ; stores high byte address XH

ldi YL, low(0x400) ; stores lower byte address YL

ldi YH, high(0x400) ; stores high byte address YH

ldi ZL, low(0x600) ; stores lower byte address ZL

ldi ZH, high(0x600) ; stores high byte address ZH

ldi r20, 3 ; r20 will have 3. will be used to compare if it is divisible by 3

ldi r21, count ; r21 will have the value in count

ldi r22, begin\_val ; r22 will have the value in begin\_val

;;;;;;;;;;;;;;;;;;;;;;;;;;;;; clear registers ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

clr r0

clr r16

clr r17

clr r18

clr r19

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;; Will store begin\_val to X register address ;;;;;;;;;;;;;;;;;;;;;;;;;;;

start:

mov r3, r22 ; copies the value in r22 into r3

st X+, r22 ; stores the of r22 into the address of X.

;Afterwords X increments its address by 1.

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;; Will check if the value is divisible by 3 ;;;;;;;;;;;;;;;;;;;;;;;;;;;

divide\_by\_three:

cp r22, r20 ; Will compare r22 and r20.

brlo notDiv ; if the value in r22 is lower than 3 then it will go to the notDiv label

sub r22, r20 ; if r22 is not lower than 3 ==> r22 = r22-3

cp r22, r20 ; will compare again to see if it equals to zero.

breq isDiv ; if it does equal to zero then it is divisible by 3 and will go to the

; isDiv label

rjmp divide\_by\_three ; if it doesnt equal to zero then it will repeat this label to see if its divisible by 3

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;; Is divisible by 3 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

isDiv:

st Y+, r3 ; will store r3 in Y address and then Y will increment to the next address

add r16, r3 ; will add r3 to the lower byte of the sum register(r16) of the numbers

; that are divisable by 3

adc r17, r0 ; if there is a carry it will be added to the high byte of the sum

; register(r17) that are divisable by 3

rjmp done\_adding ; will jump to the done\_adding label

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;; if the value is not divisible by 3 will go to this label;;;;;;;;;;;;;;

notDiv:

st Z+, r3 ; stores the of r3 into the address of Z.

;Afterwords Z increments its address by 1.

add r18, r3 ; will add r3 to the lower byte of the sum register(r18) of the

; numbers that are not divisable by 3

adc r19, r0 ;if there is a carry it will be added to the high byte of the sum

; register(r17) that are not divisable by 3

rjmp done\_adding ; will jump to the done\_adding label

;;;;;;;;;;;;;;;;;; Will go here once done adding;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

done\_adding:

mov r22, r3 ; reinitialize the value that was in r22

inc r22 ; increments r22 by 1

cp r21, r0 ; will check if the count is zero

breq Done ; if r21 equals zero then it will go to the Done label

dec r21 ; will decrement r21 by 1

brne start ; if r21 doesnt equal to zero then go to the start label

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

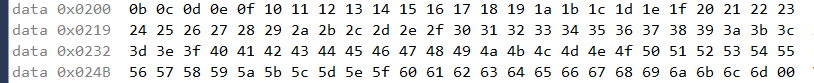
;;;;;;;;;;;;;;;;;;;;;; The Program has finished running ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

Done:

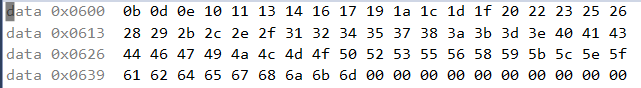
BREAK

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

Values stored in X, Y, and Z registers. X will store the values we are populating, Y will store the values that are divisible by 3, and Z will store the values that are not divisible by 3. X register address starts at 0x0200, Y register address starts at 0x0400, and Z register address starts at 0x0600.The values that are displayed in hex.

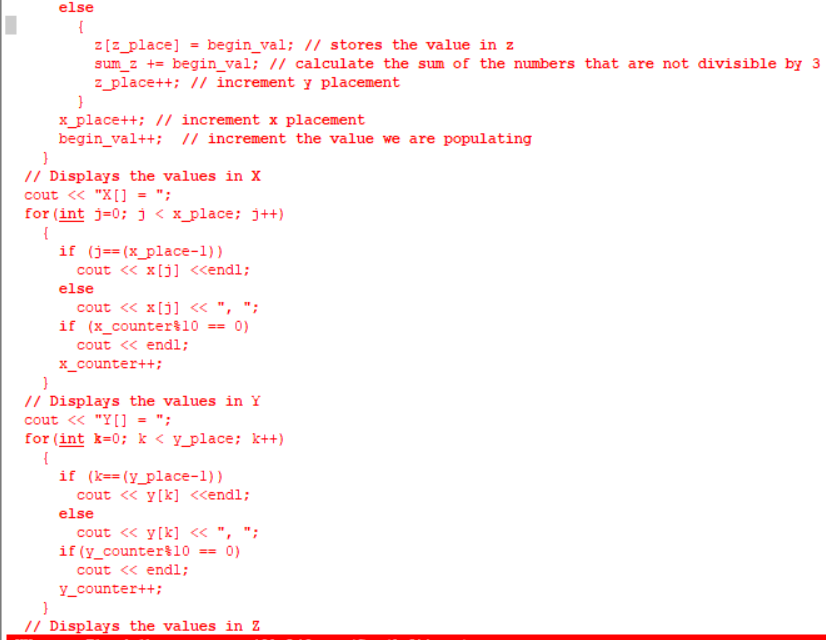


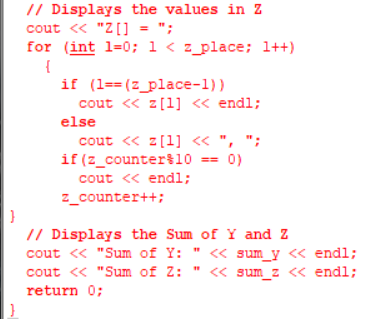




Used C++ to verify my assembly program:







C++ Verification results (displayed as decimal):

