CPE301 – SPRING 2019

Design Assignment DA1B

Student Name: Jett Guerrero

Student #: 5002377713

Student Email: guerrj1@unlv.nevada.edu

Primary Github address: <https://github.com/guerrj1/Submission_DA.git>

Directory: DA1B - <https://github.com/guerrj1/Submission_DA/tree/master/DA1B>

Submit the following for all Labs:

1. In the document, for each task submit the modified or included code (only) with highlights and justifications of the modifications. Also, include the comments.
2. Use the previously create a Github repository with a random name (no CPE/301, Lastname, Firstname). Place all labs under the root folder ESD301/DA, sub-folder named LABXX, with one document and one video link file for each lab, place modified asm/c files named as LabXX-TYY.asm/c.
3. If multiple asm/c files or other libraries are used, create a folder LabXX-TYY and place these files inside the folder.
4. The folder should have a) Word document (see template), b) source code file(s) and other include files, c) text file with youtube video links (see template).

1. **COMPONENTS LIST AND CONNECTION BLOCK DIAGRAM w/ PINS**

N/A

1. **DEVELOPED CODE OF TASK 1, 2 and 3**

;Jett Guerrero

;2/23/2019

;DA1B

.equ STARTADDS = 0x0200 ;starting location address for X pointer

.equ count = 98 ;counter for 99 numbers - 1-98

.org 0x00 ;starting program at 0x00

clr r0 ;clearing r0

ldi XL, low(STARTADDS) ;loading the lower bits of STARTADDS into XL

ldi XH, high(STARTADDS) ;loading the higher bits of STARTADDS into XH

ldi YL, low(0x400) ;loading the lower bits of 0x400 into YL

ldi YH, high(0x400) ;loading the higher bits of 0x400 into YH

ldi ZL, low(0x600) ;loading the lower bits of 0x600 into ZL

ldi ZH, high(0x600) ;loading the higher bits of 0x600 into ZH

ldi r20, 3 ;loading decimal value 3 into r20 for divisible by 3

ldi r21, count ;loading value of count which is 98 into r21

ldi r23, 11 ;loading decimal value 11 into r23 for comparing

;clear registers

clr r16 ;clearing r16 - Y SUM

clr r17 ;clearing r17 - Y SUM

clr r18 ;clearing r18 - Z SUM

clr r19 ;clearing r19 - Z SUM

;adding the lower and upper bits and storing it to a register

start:

mov r1, XL ;moving the value from XL into r1

add r1, XH ;adding the value in XH into r1

inc2eleven: ;loop for increasing value to greater than 10 => 11

cp r1, r23 ;compares the value in r1 with 11

brsh isgreaterthanten ;if the value in r1 is same or higher than 11 then branch to "isgreaterthanten" label

inc r1 ;if r1 is lower than 11, increment value in r1

jmp inc2eleven ;jump back to inc2eleven label and repeat until r1 is 11

;value is greater than 10

isgreaterthanten:

mov r3, r1 ;moves the value from r1 into r3

st X+, r3 ;stores the value in r3 into X and increments the X pointer

jmp dividebythree ;jumps to dividebythree label to check for divisibility

loop\_greaterthanten: ;when value increments and is greater than 10, jump here

st X+, r3 ;stores the value in r3 into X and increments the X pointer

mov r1, r3 ;moves value from r3 into r1

dividebythree: ;checks the value if it's divisible by three

cp r1, r20 ;compares value in r1 to 3

brlo notdivbythree ;if r1 is lower than 3 then branch to "notdivbythree"

sub r1, r20 ;subtract the value in r1 by 3

cp r1, r20 ;compare the value in r1 with 3

breq isdivbythree ;if the value in r1 equals 3 then branch to "isdivbythree"

rjmp dividebythree ;relative jump to "dividebythree" if r1 is not equal to 3 yet

isdivbythree: ;value is divisible by 3

st Y+, r3 ;store the value in r3 into Y and increment the Y pointer [Y] = XL + XH

add r16, r3 ;add the value in r3 into r16 for SUM

adc r17, r0 ;add 0 into r17 plus any carry

rjmp counter\_check ;relative jump to "counter\_check"

notdivbythree: ;value is not divisible by 3

st Z+, r3 ;stores value in r3 into Z and increments the Z pointer [Z] = XL + XH

add r18, r3 ;adds the value in r3 into r18 for SUM

adc r19, r0 ;add 0 into r19 plus any carry

rjmp counter\_check ;relative jump to "counter\_check"

counter\_check: ;checks to see if program ran for 99 values

cp r21, r0 ;compare r21 which is 98 to 0

breq end ;if r21 equals 0 then branch to

dec r21 ;else decrement value in r21

inc r3 ;increment value in r3 and continue populating X

jmp loop\_greaterthanten ;relative jump to "loop\_greaterthanten"

end: jmp end ;end label and keep jumping to this label when finished

Functions of Program:

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

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

3. Uses X/Y/Z register addressing to simultaneously add numbers from memory location 0x0400 and 0x0600 and stores the sums at R16:R17 and R18:R19 respectively.

1. **DEVELOPED CODE FOR TASK 4**

C Code Verification

#include <stdio.h>

int main(void)

{

int x = 11;

int y = 0;

int z = 0;

for(int i = 0; i < 99; i++ )

{

if(x % 3 == 0)

{

y = x + y;

}

else

{

z = x + z;

}

x++;

}

printf("SUM of Numbers Divisible by 3\n");

printf("Y = 0x%x\n",y);

printf("SUM of Numbers Not Divisible by 3\n");

printf("Z = 0x%x",z);

}

1. **TASK 5**

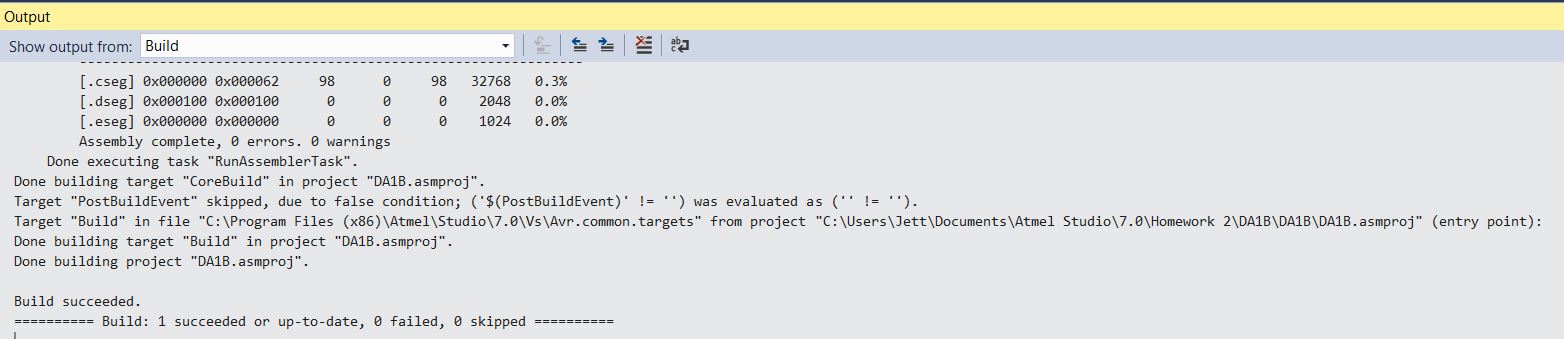
Execution Time @ 16MHz = 951.13us

Number of Cycles = 15,218

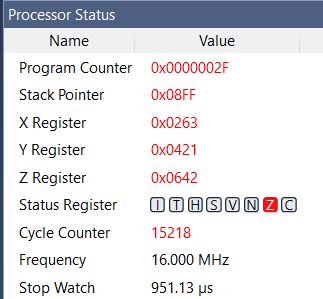
1. **SCHEMATICS**

N/A

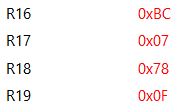
1. **SCREENSHOTS OF EACH TASK OUTPUT (ATMEL STUDIO OUTPUT)**

****

Output from Assembly Build

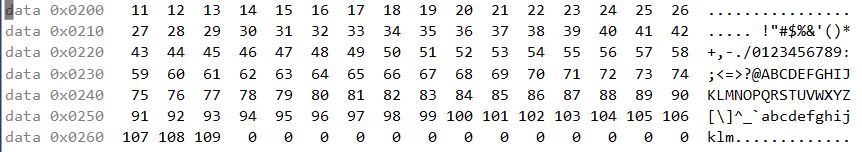
****

Processor Status

****

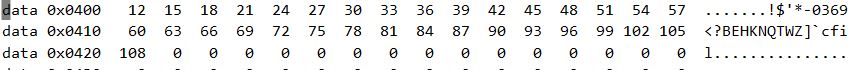
Sum in the Registers

R16:R17 = 0x07BC R18:R19 = 0x0F78

****

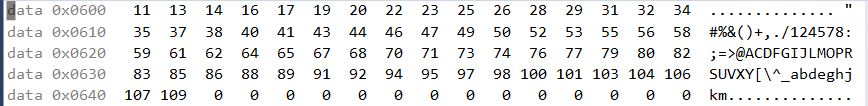
X: 0x200

99 Numbers starting at 0x200 that are greater than 10 and less than 255

****

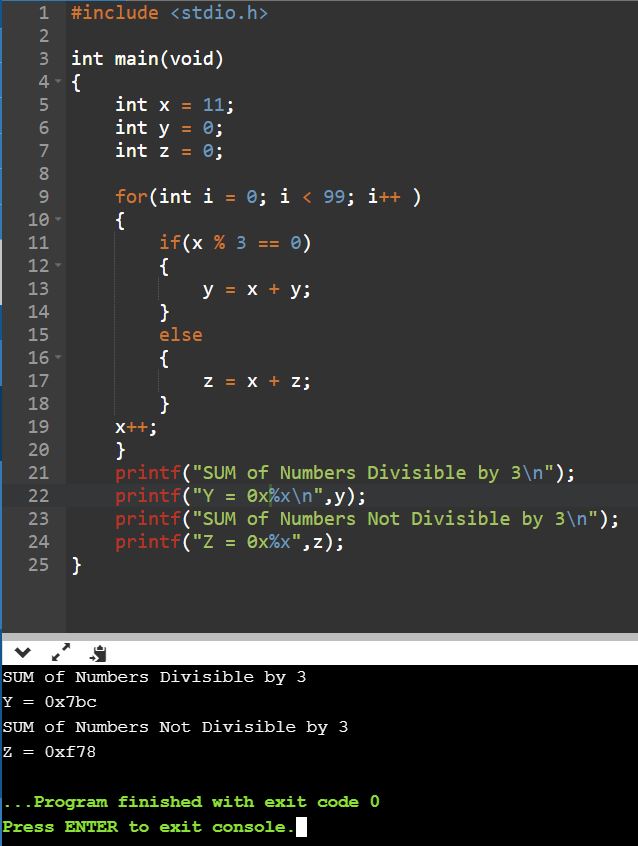
Y: 0x400

Values that are divisible by 3 stored starting at 0x400

****

Z: 0x600

Values that are not divisible by 3 stored starting at 0x600



C Code with Verification Results

SUM of Numbers Divisible by 3

Y = 0x7BC

SUM of Numbers Not Divisible by 3

Z = 0xF78

Execution results verify that the assembly code Sum values are correct

1. **SCREENSHOT OF EACH DEMO (BOARD SETUP)**

N/A

1. **VIDEO LINKS OF EACH DEMO**

N/A

1. **GITHUB LINK OF THIS DA**

<https://github.com/guerrj1/Submission_DA/tree/master/DA1B>

**Student Academic Misconduct Policy**

<http://studentconduct.unlv.edu/misconduct/policy.html>

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

Jett Guerrero