CPE301 – SPRING 2019

Design Assignment 1B

Student Name: Chris Barr

Student #: 2000682859

Student Email: barrc1@unlv.nevada.edu

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

Directory: https://github.com/BarrChris/submission\_da.git

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. **INITIAL/MODIFIED/DEVELOPED CODE OF TASK 1/A**

.include <m328pdef.inc> ; Standard library

.EQU STARTADDS = 0x0200 ; Given address to start at

LDI XL, LOW(STARTADDS) ; Stores ALL 99 numbers

LDI XH, HIGH(STARTADDS) ; starting at address 0x0200

LDI YL, LOW(0x400) ; Stores divisible numbers by 3

LDI YH, HIGH(0x400) ; starting at address 0x0400

LDI ZL, LOW(0x600) ; Stores NOT divisible numbers by 3

LDI ZH, HIGH(0x600) ; starting at address 0x0600

; R22 stores the X/Y/Z pointer with values

MOV R25, XL ; R25 = 0x00 + 0x00 = 0x00

ADD R25, XH ; R25 = 0x00 + 0x02 = 0x02

LDI R22, 99 ; Counter from 0 to 99

LDI R23, 0 ; Temp Register for checking divisibility

LDI R20, 0 ; Zero

Wait: ; This label waits before it can populate

CPI R25, 11 ; "Compares the R25 register with 11

BREQ Populate1 ; so it can branch over to start populating X/Y/Z.

INC R25 ; Else, it's going to increment R25

RJMP Wait ; and loop until it reaches 11."

Populate1: ; "This label checks to see if current number

RJMP DIV3 ; is divisible by 3 before going to the next incrementation."

Populate2: ; This label will store the value of the next X register and increment current number

ST X+, R25 ; Stores value in register R25 into next X register

INC R25 ; Goes to the next incrementation value and stores it in register R25

DEC R22 ; Decrements the counter for register R22

BRNE Populate1 ; If the decrementation counter reaches 0, the program finished all 99 numbers

done: rjmp done ; For debugging purposes

;==================================================================

; DIVISIBILITY CHECK

;==================================================================

DIV3: ; This label will start the beginning process of checking the divisibility

MOV R23, R25 ; Uses register R23 as a temp register to find out if register R25 is divisible by 3

check: ; This label will do the arithmetic to determine if the number is divisble by 3

SUBI R23, 3 ; We subtract our number by 3 to check

BREQ zero ; If the number is zero, it will jump to label 'zero'

BRMI negative ; If the number is negative, it will jump to label 'negative'

RJMP check ; Else, go back to subtracting the number until it becomes a negative or a zero

zero: ; This label will store the numbers into address 0x0400 because it's divisible

ST Y+, R25 ; Stores the divisible numbers into 0x0400 and onwards

ADD R16, R25 ; Adds the number to find the sum of the divisible numbers

ADC R17, R20 ; Then adds the carry for overflow

RJMP Populate2 ; Goes back to continue incrementing and populating X/Y/Z registers until we reach all 99 numbers

negative: ; This label will store the numbers into address 0x0600 because it's not divisible

ST Z+, R25 ; Stores the non-divisible numbers into 0x0600 and onwards

ADD R18, R25 ; Adds the number to find the sum of the non-divisble numbers

ADC R19, R20 ; Then adds the carry for overflow

RJMP Populate2 ; Goes back to continue incrementing and populating X/Y/Z registers until we reach all 99 numbers

;==================================================================

1. **DEVELOPED MODIFIED CODE OF TASK 2/A from TASK 1/A**

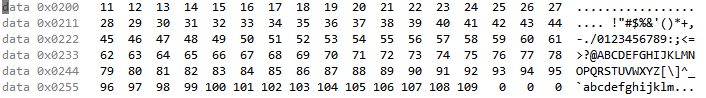
N/A

1. **SCHEMATICS**

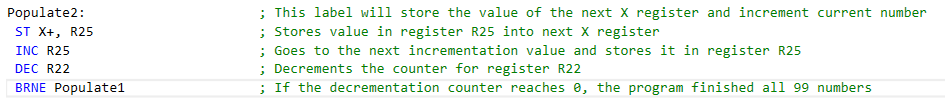
N/A

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

Task 1:

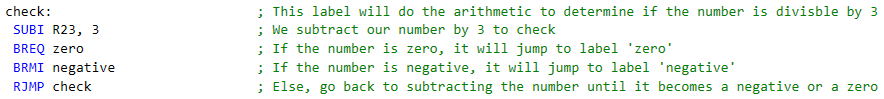


Starting at address STARTADDS = 0x0200, all 99 numbers are stored here in a consecutive order.



This portion of the code will use the X pointer to store values into 0x0200

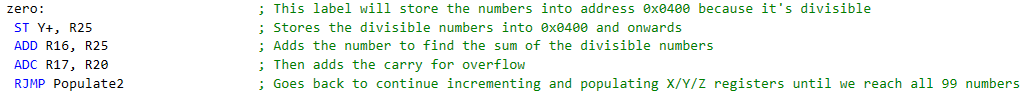
Task 2:



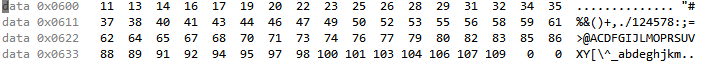
This part of the code performs the arithmetic to determine if the number is divisible by 3 by subtracting the number over and over until the number become a zero or a negative



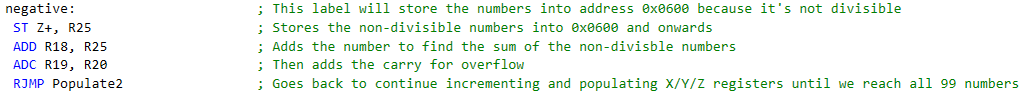
Starting at address 0x0400 address, we store all the divisible by 3 numbers in these registers



This is the portion of the code responsible for ‘storing’ the numbers in address 0x0400 using the Y pointer

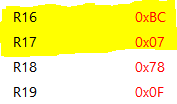


Starting at address 0x0600 address, we tore all the non-divisible by 3 numbers in these registers



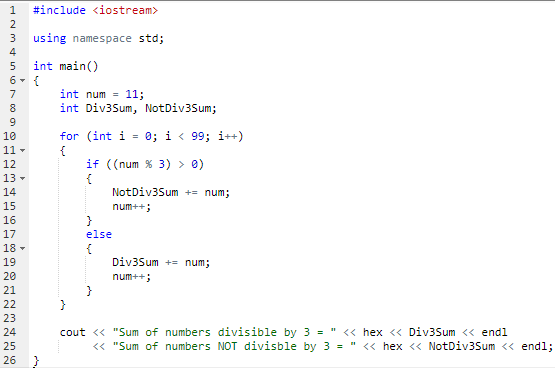
This code uses the Z pointer to ‘store’ the numbers into address 0x0600

Task 3:



The highlighted portion (R17:R16) is the total sum of divisible numbers – 0x07BC. And the non-highlighted portion (R19:R18) is the total sum of the non-divisible numbers – 0x0F78. The code that adds the sum is in the *negative* and *zero* label.

Task 4:



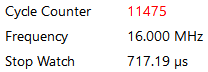
Verification code used to determine if the sum is correct



Output of verification code

The hex value is the exact same obtained from Atmel Studio. This confirms my .asm code is correct.

Task 5:



@ 16 MHz

Execution Time = 717.19 us

# of Cycles = 11475

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

N/A

1. **VIDEO LINKS OF EACH DEMO**

N/A

1. **GITHUB LINK OF THIS DA**

**Student Academic Misconduct Policy**

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

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

NAME OF THE STUDENT