CPE301 – SPRING 2019

Design Assignment 1B

Student Name: Robert Sander

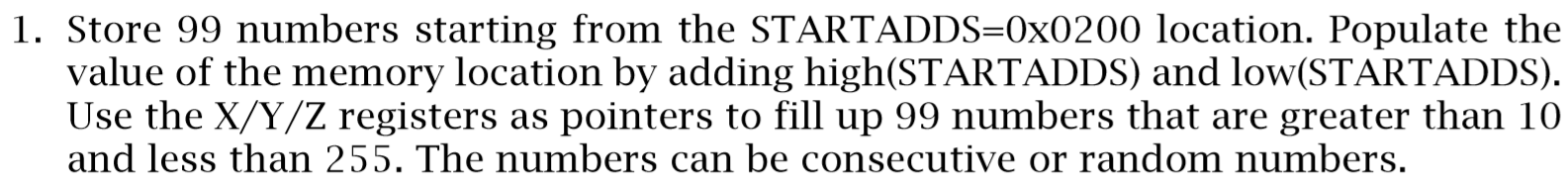
Student #: 5002102412

Student Email: sander1@unlv.nevada.edu

Primary Github address: https://github.com/sanderUNLV/submission\_DA.git

Youtube link: <https://youtu.be/cYE0_ewTX2Q>

1. **DEVELOPED CODE OF TASK 1/B**



; Author : Robert Sander

.include<m328pdef.inc>

.CSEG

.ORG 0x00

LDI R19, 99 ;R19 = 99 (R19 for counter)

LDI R16,0x0B ;load R16 with value 0xB0 (value to be copied)

LDI XL,LOW($200) ;load the low byte of X with value 0x00

LDI XH,HIGH($200) ;load the high byte of X with value 0x2

L1:

ST X+,R16 ;copy R16 to memory location X and increments the location by 1

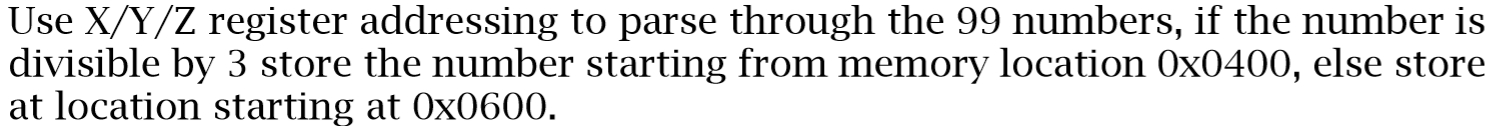
INC R16 ;Increments R16 by 1 every loop, 99 times

DEC R19 ;decrement the counter by 1

BRNE L1 ;loop until R19 (counter) = zero

END: RJMP END

1. **DEVELOPED CODE OF TASK 2/B from TASK 1/B**



; Author : Robert Sander

.include<m328pdef.inc>

.ORG 0x00

LDI R16, 99 ;R16 = 99 (R16 FOR COUNTER)

LDI R17,0x0B ;LOAD R17 WITH VALUE 0x0B

LDI XL,LOW($200) ;LOAD THE LOW BYTE OF X WITH VALUE 0x00 - STARTING POSITION OF THE POPULATED NUMBERS

LDI XH,HIGH($200) ;LOAD THE HIGH BYTE OF X WITH VALUE 0x2 - STARTING POSITION OF THE POPULATED NUMBERS

LDI YL,LOW($400) ;LOAD THE LOW BYTE OF Y WITH VALUE 0x00 - FOR THE NUMBERS THAT ARE DIVISBLE BY THREE

LDI YH,HIGH($400) ;LOAD THE HIGH BYTE OF Y WITH VALUE 0x4 - FOR THE NUMBERS THAT ARE DIVISBLE BY THREE

LDI ZL, LOW($600) ;LOAD THE LOW BYTE OF Z WITH VALUE 0x00 - FOR THE NUMBERS THAT ARE NOT DIVISBLE BY THREE

LDI ZH,HIGH($600) ;LOAD THE HIGH BYTE OF Z WITH VALUE 0x6 - FOR THE NUMBERS THAT ARE NOT DIVISBLE BY THREE

L1:

ST X+, R17 ;COPY R17 TO MEMORY LOCATION X AND INCREMENTS THE LOCATION BY 1

LDI R18, 0 ;LOAD THE VALUE 0 INTO R18

ADD R18, R17 ;ADD R17 TO R18 AND STORE IN R18 - LINE 19

GREATERTHANTHREE:

SUBI R18, 3 ;SUBTRACT 3 FROM R18 AND STORE THE RESULT IN R18

CPI R18, 0 ;COMPARE R18 WITH THE VALUE 0

BRNE DONTSAVE ;IF R18 IS NOT EQUAL TO 0 GO TO 'DONTSAVE:'

ST Y+, R17 ;COPY R17 TO MEMORY LOCATION Y AND INCREMENTS THE LOCATION BY 1 - FOR THE NUMBERS THAT ARE DIVISBLE BY THREE

DONTSAVE:

CPI R18, 3 ;COMPARE R18 WITH THE VALUE 3

BRGE GREATERTHANTHREE ;IF R18 IS GREATER THAN OR EQUAL TO 3 GO TO 'GREATERTHANTHREE:'

CPI R18, 0 ;COMPARE R18 WITH THE VALUE 0

BREQ SKIP ;IF R18 IS NOT EQUAL TO 0 GO TO 'SKIP:'

ST Z+, R17 ;COPY R17 TO MEMORY LOCATION Z AND INCREMENTS THE LOCATION BY 1 - FOR THE NUMBERS THAT ARE NOT DIVISBLE BY THREE

SKIP:

INC R17 ;INCREMENTS R17 BY 1 EVERY LOOP, 99 TIMES

DEC R16 ;DECREMENT THE COUNTER BY 1

BRNE L1 ;LOOP UNTIL R16 (COUNTER) = ZERO

END: RJMP END

;TO CHECK IF DIVISIBLE BY THREE

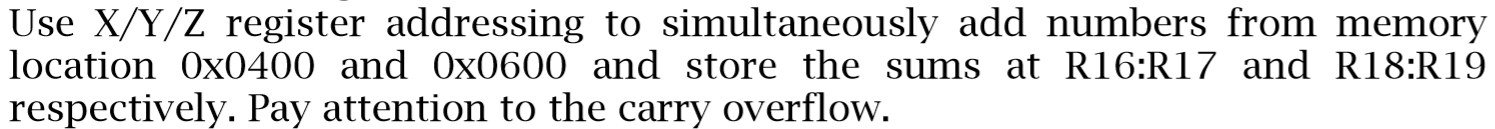
;SUBTRACT 3 UNTIL NUMBER EQUALS ZERO, IF THERE IS A REMAINDER AFTER IT GOES TO ZERO THEN NUMBER IS NOT DIVISIBLE BY THREE

;STORE IN 0x0600

;IF NO REMAINDER, NUMBER IS DIVISBLE BY THREE

;STORE IN 0x0400

1. **DEVELOPED CODE OF TASK 3/B from TASK 2/B**



; Author : Robert Sander

.include<m328pdef.inc>

.ORG 0x00

LDI R16, 99 ;R16 = 99 (R16 FOR COUNTER)

LDI R17,0x0B ;LOAD R17 WITH VALUE 0x0B

LDI XL,LOW($200) ;LOAD THE LOW BYTE OF X WITH VALUE 0x00 - STARTING POSITION OF THE POPULATED NUMBERS

LDI XH,HIGH($200) ;LOAD THE HIGH BYTE OF X WITH VALUE 0x2 - STARTING POSITION OF THE POPULATED NUMBERS

LDI YL,LOW($400) ;LOAD THE LOW BYTE OF Y WITH VALUE 0x00 - FOR THE NUMBERS THAT ARE DIVISBLE BY THREE

LDI YH,HIGH($400) ;LOAD THE HIGH BYTE OF Y WITH VALUE 0x4 - FOR THE NUMBERS THAT ARE DIVISBLE BY THREE

LDI ZL, LOW($600) ;LOAD THE LOW BYTE OF Z WITH VALUE 0x00 - FOR THE NUMBERS THAT ARE NOT DIVISBLE BY THREE

LDI ZH,HIGH($600) ;LOAD THE HIGH BYTE OF Z WITH VALUE 0x6 - FOR THE NUMBERS THAT ARE NOT DIVISBLE BY THREE

L1:

ST X+, R17 ;COPY R17 TO MEMORY LOCATION X AND INCREMENTS THE LOCATION BY 1

LDI R18, 0 ;LOAD THE VALUE 0 INTO R18

ADD R18, R17 ;ADD R17 TO R18 AND STORE IN R18 - LINE 19

GREATERTHANTHREE:

SUBI R18, 3 ;SUBTRACT 3 FROM R18 AND STORE THE RESULT IN R18

CPI R18, 0 ;COMPARE R18 WITH THE VALUE 0

BRNE DONTSAVE ;IF R18 IS NOT EQUAL TO 0 GO TO 'DONTSAVE:'

ST Y+, R17 ;COPY R17 TO MEMORY LOCATION Y AND INCREMENTS THE LOCATION BY 1 - FOR THE NUMBERS THAT ARE DIVISBLE BY THREE

DONTSAVE:

CPI R18, 3 ;COMPARE R18 WITH THE VALUE 3

BRGE GREATERTHANTHREE ;IF R18 IS GREATER THAN OR EQUAL TO 3 GO TO 'GREATERTHANTHREE:'

CPI R18, 0 ;COMPARE R18 WITH THE VALUE 0

BREQ SKIP ;IF R18 IS NOT EQUAL TO 0 GO TO 'SKIP:'

ST Z+, R17 ;COPY R17 TO MEMORY LOCATION Z AND INCREMENTS THE LOCATION BY 1 - FOR THE NUMBERS THAT ARE NOT DIVISBLE BY THREE

SKIP:

INC R17 ;INCREMENTS R17 BY 1 EVERY LOOP, 99 TIMES

DEC R16 ;DECREMENT THE COUNTER BY 1

BRNE L1 ;LOOP UNTIL R16 (COUNTER) = ZERO

LDI R20, 99 ;R20 = 99 (R16 for counter)

LDI R16, 0 ;LOAD THE VALUE 0 INTO R16

LDI R17, 0 ;LOAD THE VALUE 0 INTO R17

LDI R18, 0 ;LOAD THE VALUE 0 INTO R18

LDI R19, 0 ;LOAD THE VALUE 0 INTO R19

CLC ;CLEAR THE CARRY FLAG

L2:

LD R21, -Y ;LOAD VALUE AT ADDRESS Y INTO R21 AND DECREMENT Y BY 1

ADC R16, R21 ;ADD WITH CARRY R21 AND R16 AND STORE THE RESULT IN R16

BRCC NOCARRY\_Y ;IF CARRY FLAG IS CLEAR GO TO 'NOCARRY\_Y:'

INC R17 ;INCREMENT R17 BY 1 IF CARRY IS NOT CLEAR

NOCARRY\_Y:

CLC ;CLEAR THE CARRY FLAG

LD R22, -Z ;LOAD VALUE AT ADDRESS Z INTO R22 AND DECREMENT Z BY 1

ADC R18, R22 ;ADD WITH CARRY R22 AND R18 AND STORE THE RESULT IN R18

BRCC NOCARRY\_Z ;IF CARRY FLAG IS CLEAR GO TO 'NOCARRY\_Z:'

INC R19 ;INCREMENT R19 BY 1 IF CARRY IS NOT CLEAR

NOCARRY\_Z:

CLC ;CLEAR THE CARRY FLAG

DEC R20 ;DECREMENT THE COUNTER BY 1

BRNE L2

END: RJMP END

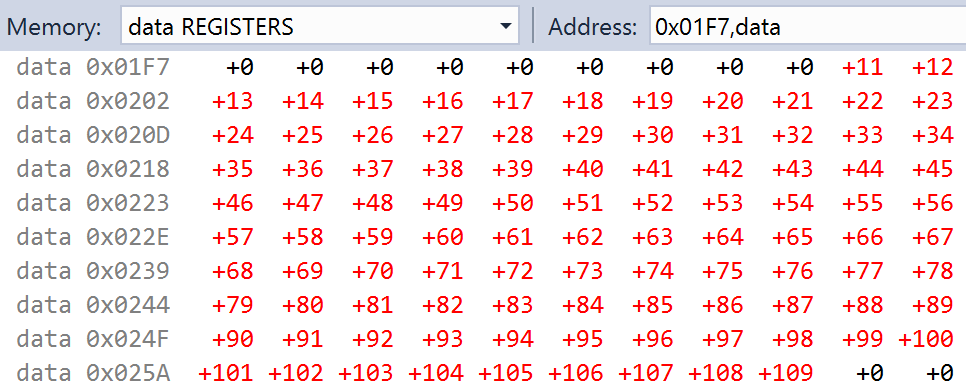
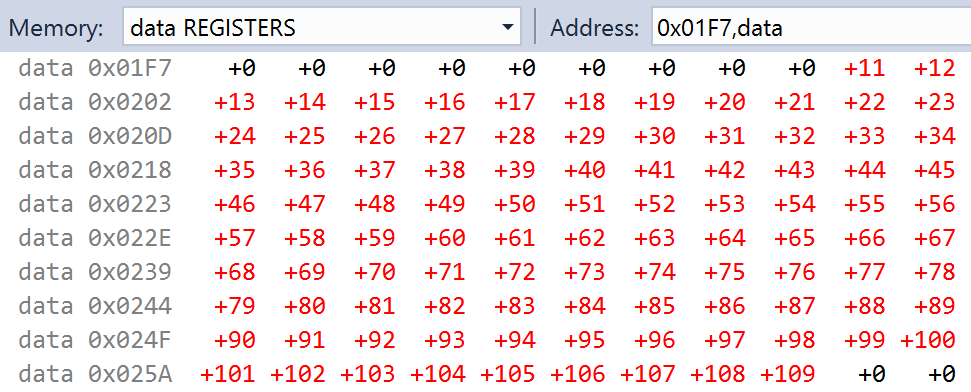
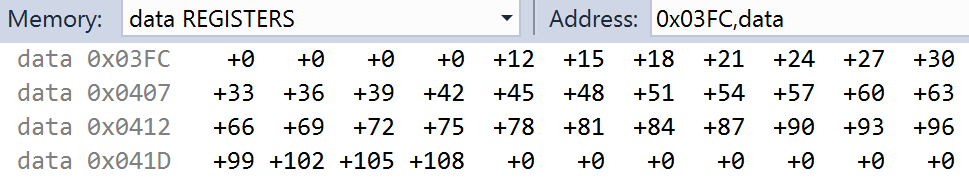
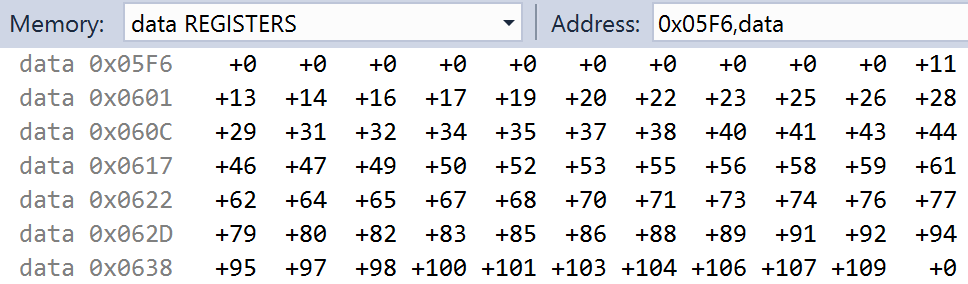
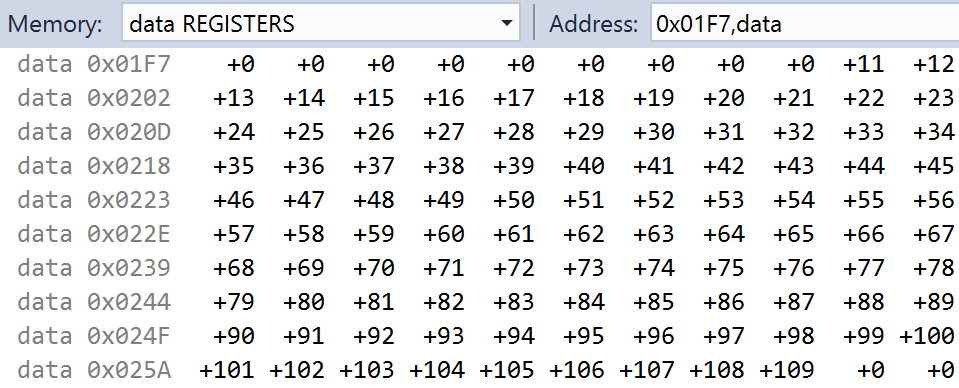
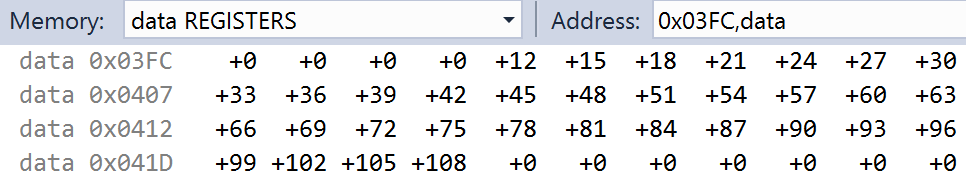
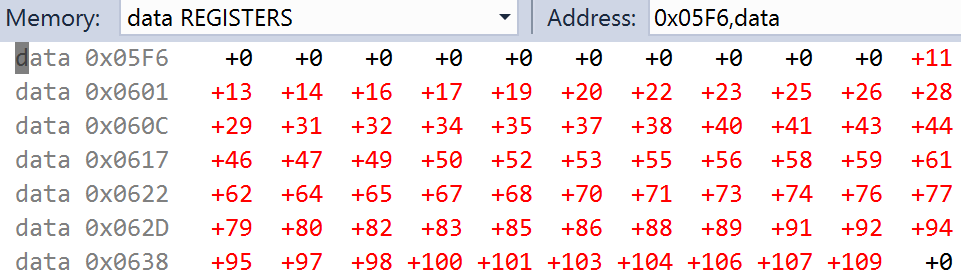
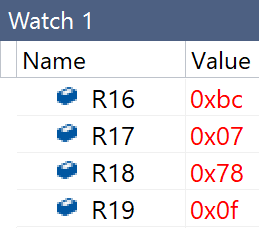
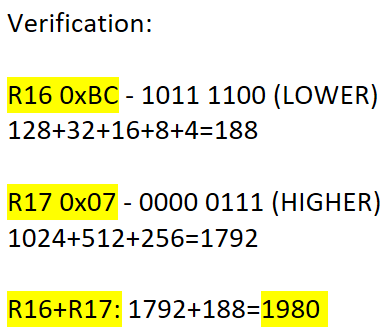
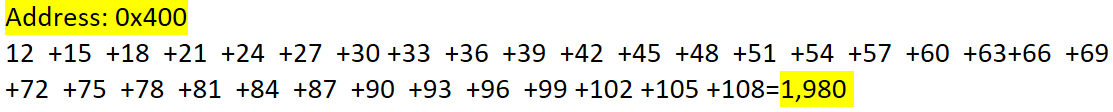
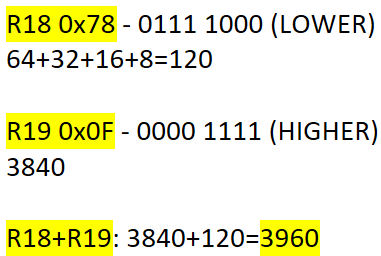
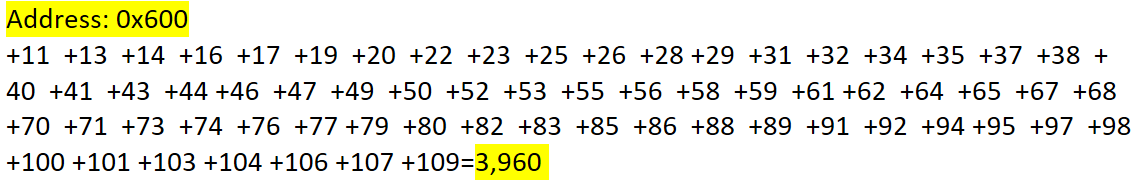
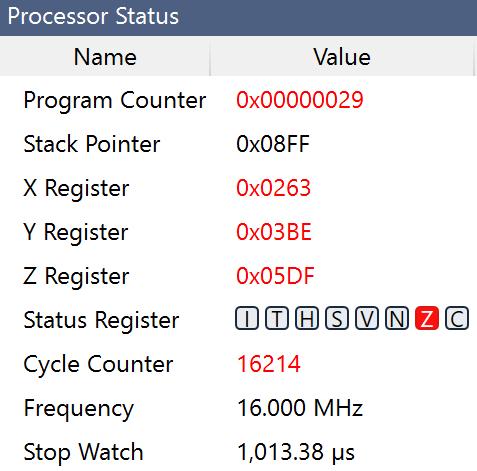
;TO CHECK IF DIVISIBLE BY THREE

;SUBTRACT 3 UNTIL NUMBER EQUALS ZERO, IF THERE IS A REMAINDER AFTER IT GOES TO ZERO THEN NUMBER IS NOT DIVISIBLE BY THREE

;STORE IN 0x0600

;IF NO REMAINDER, NUMBER IS DIVISBLE BY THREE

;STORE IN 0x0400

1. **SCREENSHOTS OF EACH TASK OUTPUT (ATMEL STUDIO OUTPUT)**
   1. 
   2.   
   3.    
   4.    
   5. 

This subroutine takes 1013.38µs to execute

1. **VIDEO LINKS OF EACH DEMO**

<https://youtu.be/cYE0_ewTX2Q>

1. **GITHUB LINK OF THIS DA**

<https://github.com/sanderUNLV/submission_DA.git>

**Student Academic Misconduct Policy**

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

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

-Robert Sander