CPE301 – SPRING 2019

Design Assignment 1B

Student Name: Victor Martinez

Student #:5003131674

Student Email: martiv6@unlv.nevada.edu

Primary Github address: https://github.com/martiv6/submissions\_da

Directory: 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**

ATMEL studio

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

;

; DA1B.asm

;

; Created: 2/20/2019 11:46:41 AM

; Author : victor

;

.equ STARTADDS = 0x0200 ;where the memory address will start

.equ numvalues = 99 ;the number of values we need

.equ minvalue = 21 ;the min value we start with

.equ dividend =3 ;the value we are dividing by

.org 0

CLR R0 ;clears value in R0

CLR R16 ;clears value in R16

CLR R17 ;clears value in R17

CLR R18 ;clears value in R18

CLR R19 ;clears value in R19

LDI R22, numvalues ;loads the number of values in to R22

LDI R23, minvalue ;loads min value will be in R23 greater then 10

LDI XL, low(STARTADDS) ;sets the lower bits of register X

LDI XH, high(STARTADDS) ;sets the higher bits of register X

LDI YL, low(0x400) ;sets the lower bits of register Y as well is where we will store the values that are divisible by 3

LDI YH, high(0x400) ;sets the higher bits of register Y as well is where we will store the values that are divisible by 3

LDI ZL, low(0x600) ;sets the lower bits of register Z as well is where we will store the values that are NOT divisible by 3

LDI ZH, high(0x600) ;sets the higher bits of register Z as well is where we will store the values that are NOT divisible by 3

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

;

; DA1B.asm

;

; Created: 2/20/2019 11:46:41 AM

; Author : victor

;

.equ STARTADDS = 0x0200 ;where the memory address will start

.equ numvalues = 99 ;the number of values we need

.equ minvalue = 21 ;the min value we start with

.equ dividend =3 ;the value we are dividing by

.equ maxvalue = 255 ;the max value that can be ran

.org 0

CLR R0 ;clears value in R0

CLR R16 ;clears value in R16

CLR R17 ;clears value in R17

CLR R18 ;clears value in R18

CLR R19 ;clears value in R19

**CLR R24 ;added code**

LDI R22, numvalues ;loads the number of values in to R22

LDI R23, minvalue ;loads min value will be in R23 greater than 10

LDI R25, maxvalue ;loads max value will be in R25 less than 255

LDI XL, low(STARTADDS) ;sets the lower bits of register X

LDI XH, high(STARTADDS) ;sets the higher bits of register X

LDI YL, low(0x400) ;sets the lower bits of register Y as well is where we will store the values that are divisible by 3

LDI YH, high(0x400) ;sets the higher bits of register Y as well is where we will store the values that are divisible by 3

LDI ZL, low(0x600) ;sets the lower bits of register Z as well is where we will store the values that are NOT divisible by 3

LDI ZH, high(0x600) ;sets the higher bits of register Z as well is where we will store the values that are NOT divisible by 3

BEGIN:

MOV R24, R23 ;places min value into R24 to BEGIN testing values

ST X+, R23 ;also places min value in X register

RJMP BOUNCER ;sent over to check if able to be divided by 3

BOUNCER:

SUBI R24, dividend ;will subtract by 3

BRLT KICKOUT ;if less then zero its KICKOUT

CP R25, R24 ;makes sure the value isnt over 255

BRGE ITSOVER ;if value over 255 it is not placed in either register

BREQ YOUGOOD ;if its equal to zero its good to be divided

RJMP BOUNCER ;checks the next value

YOUGOOD: ;adds the value to R16 and R17 that are divisible by 3

ST Y+, R23 ;puts it to the high bits of Y

ADD R16, R23 ;adds value of R23 to R16

ADC R17, R0 ;puts any overflow from R0

RJMP ITSOVER ;goes to ITOVER

KICKOUT: ;adds the value to R18 and R19 that are NOT divisible by 3

ST Z+, R23 ;it puts the value that cant be divided into the Z register

ADD R18, R23 ;adds value of R23 to R18

ADC R19, R0 ;puts any overflow from R0

RJMP ITSOVER ;jumps to ITSOVER

ITSOVER:

INC R23 ;adds one to value in R23 to goto the next value

DEC R22 ;subtracts one to R22 bc we did the first value of 99

BRNE BEGIN ;goes to the BEGINning

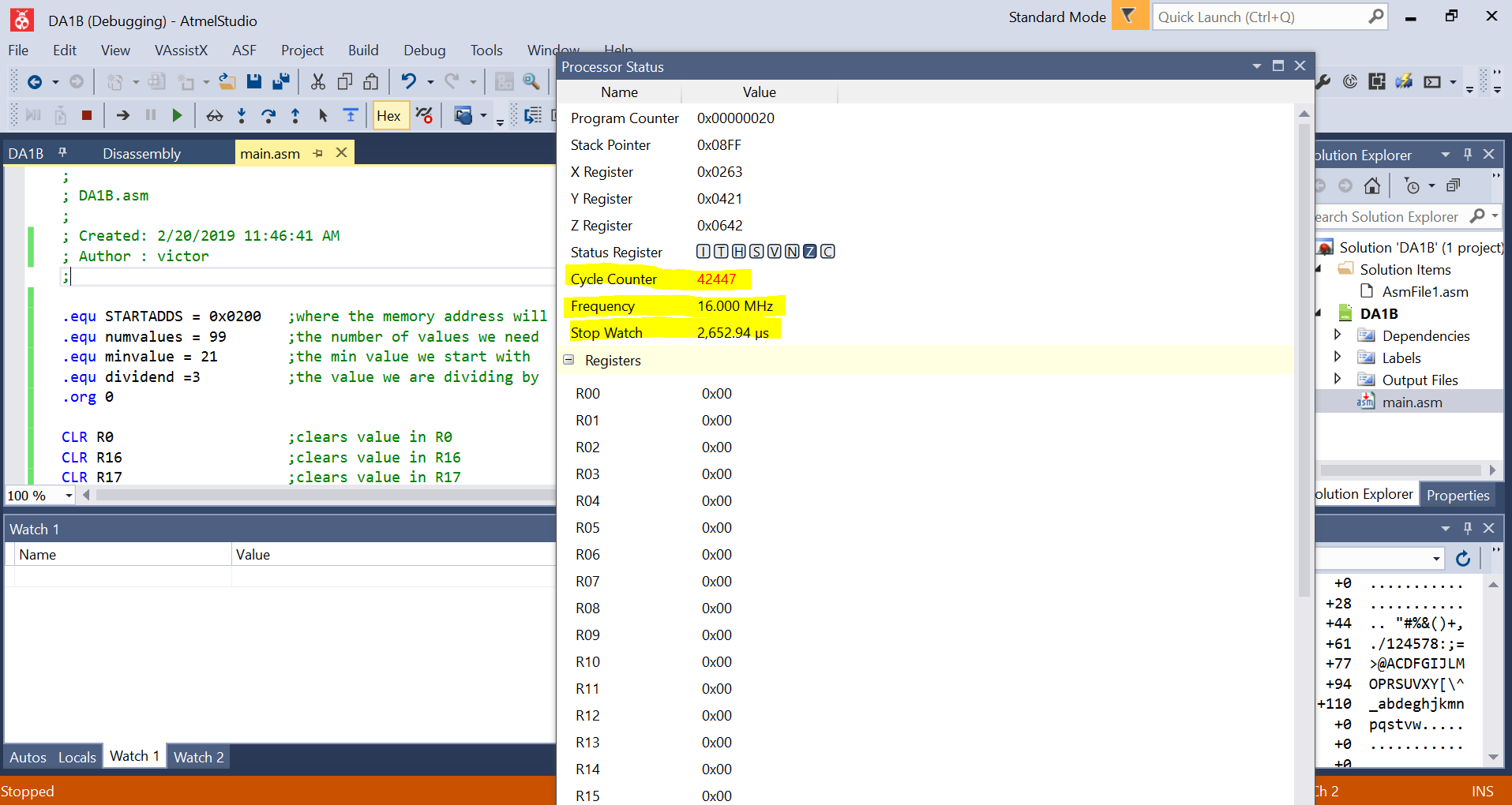
BREAK

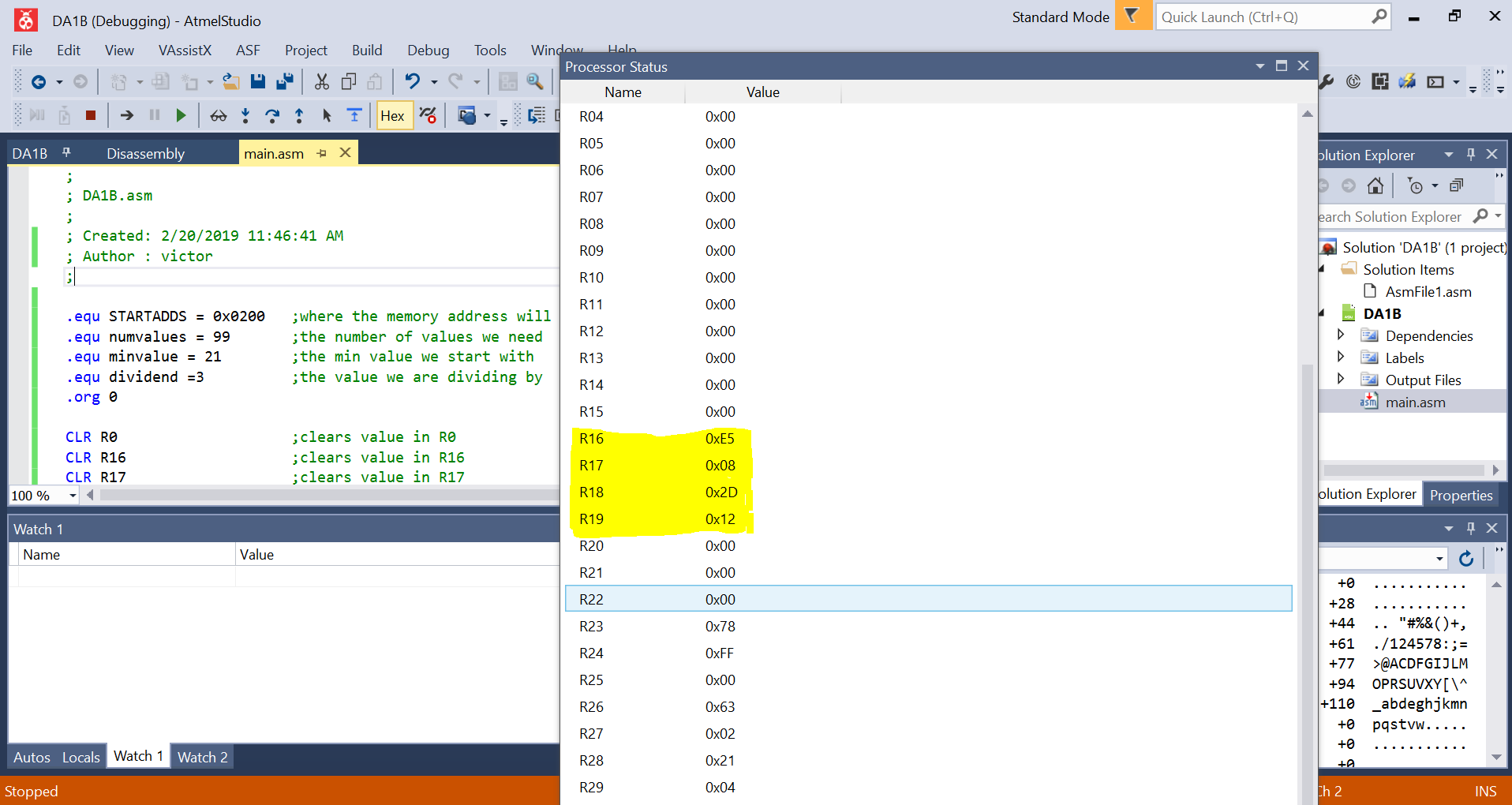
1. **SCHEMATICS**

N/A

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

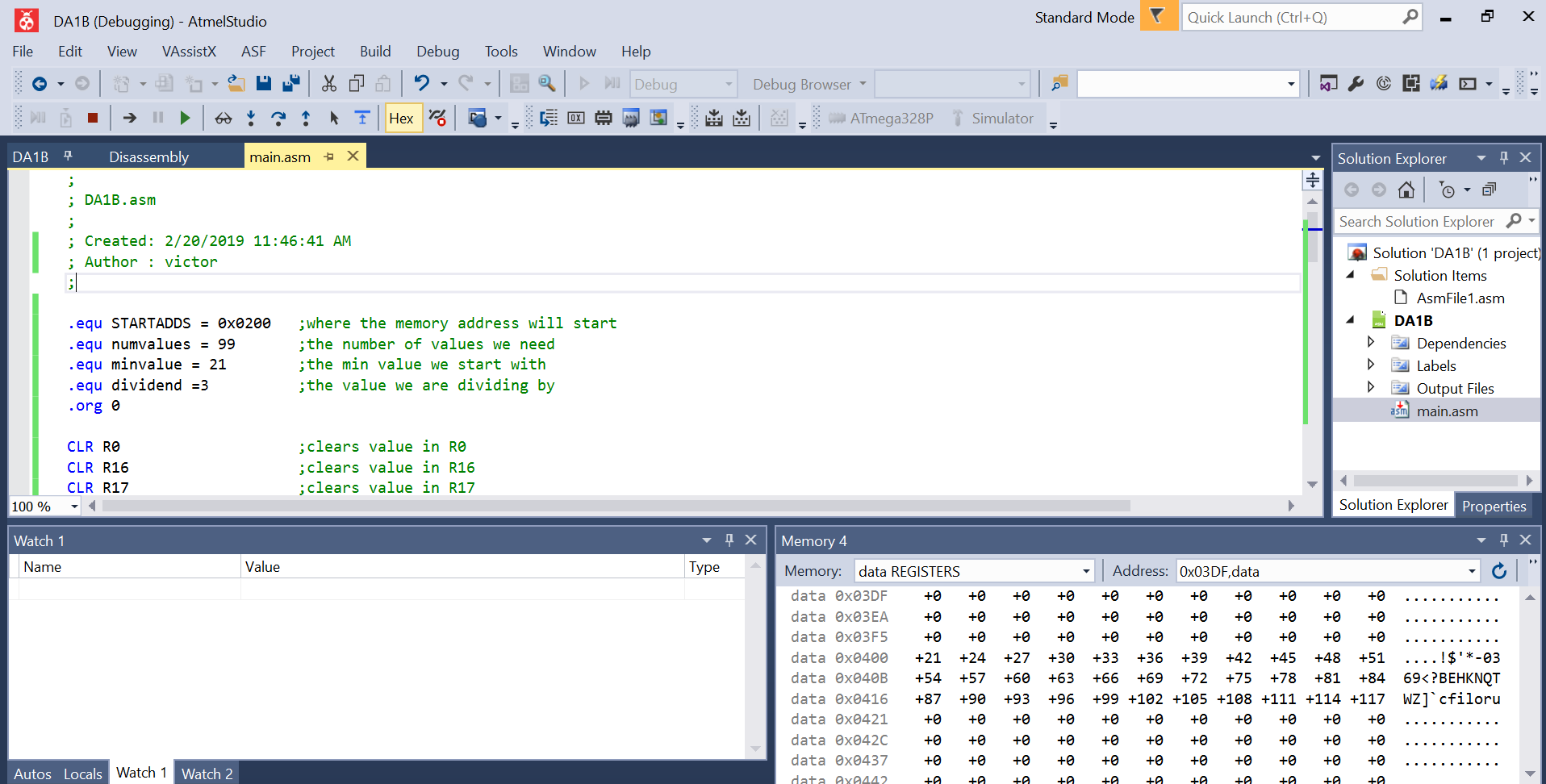
This is without having the register 24 cleared.

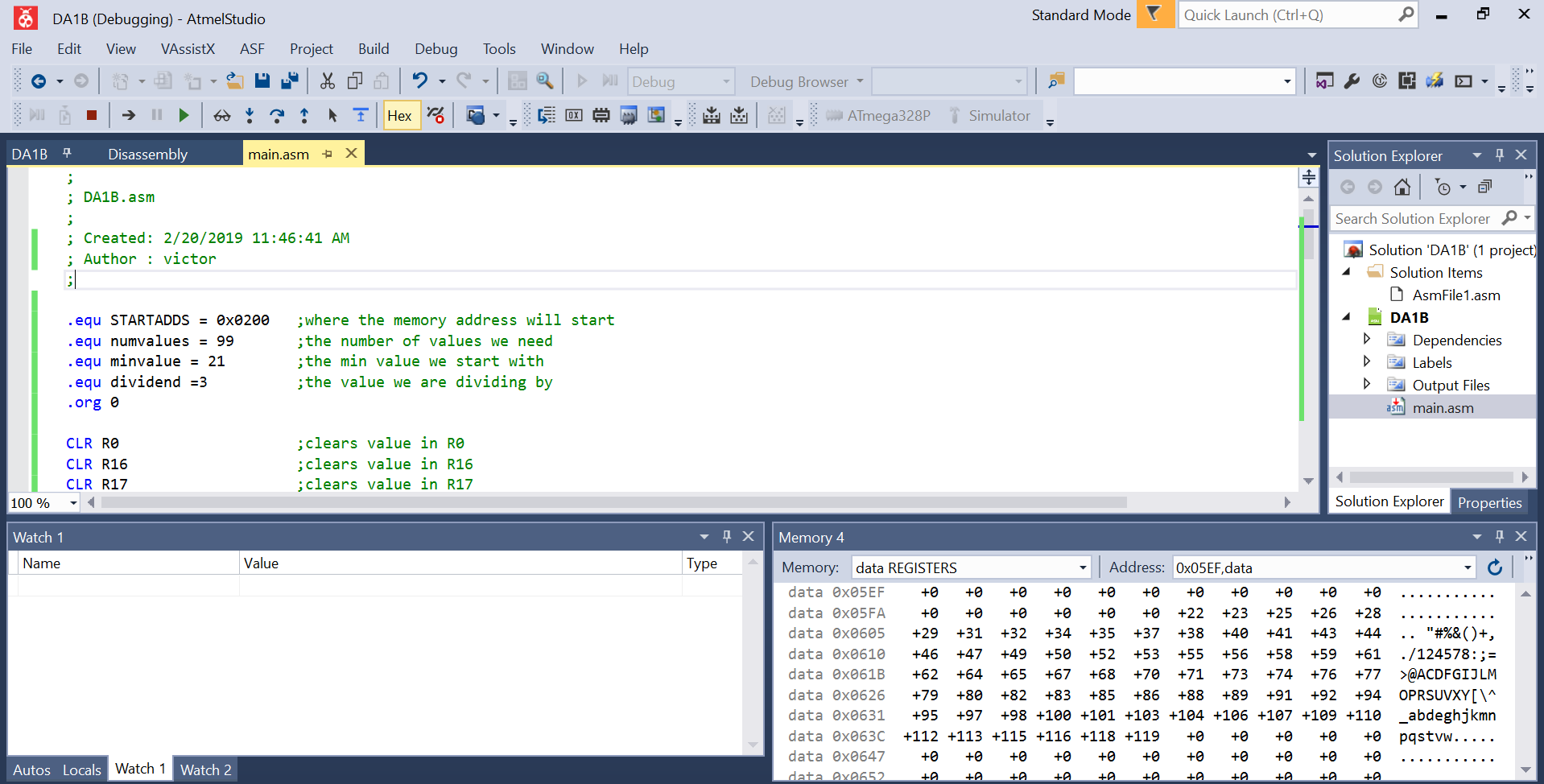




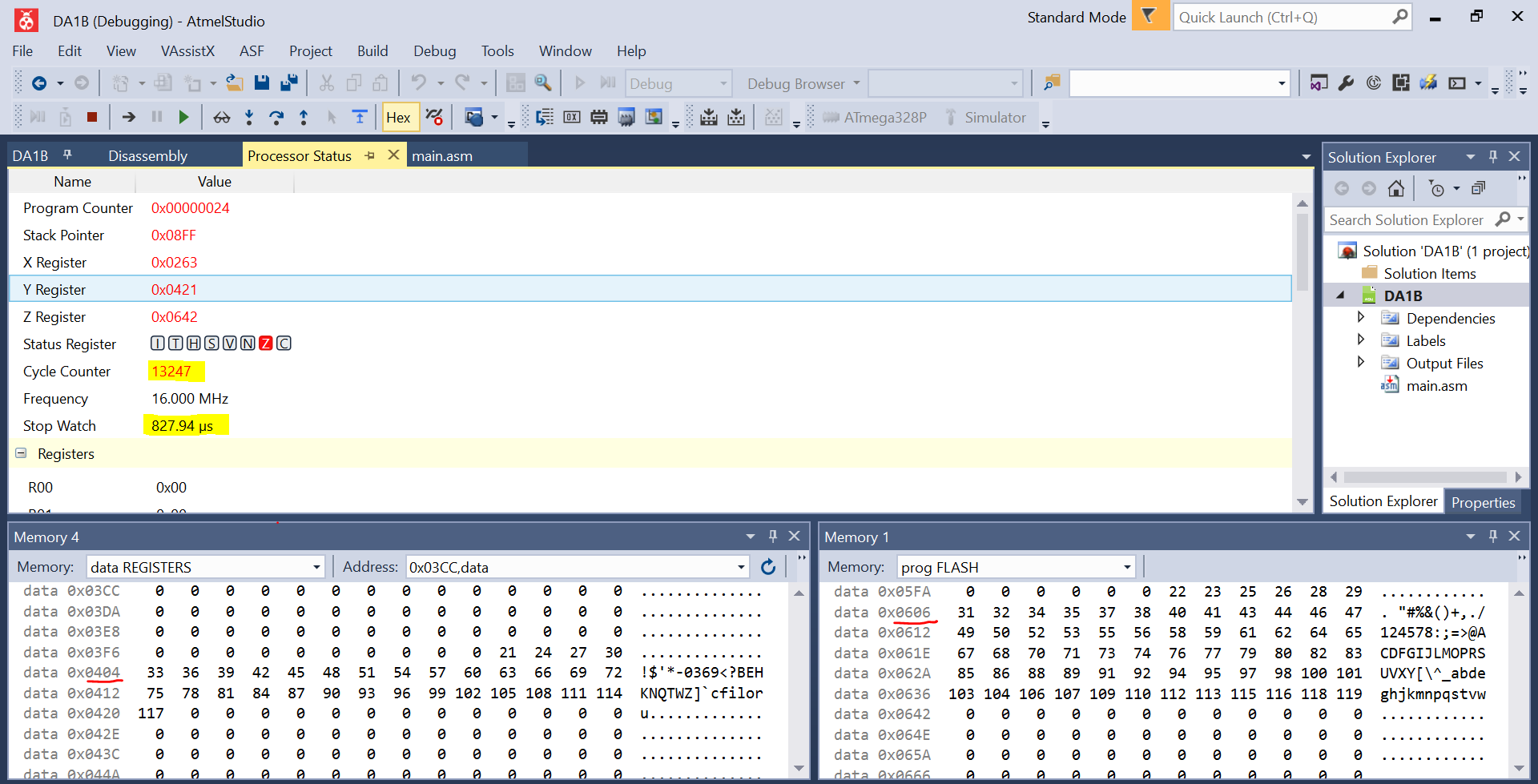
0x08E5= 2277

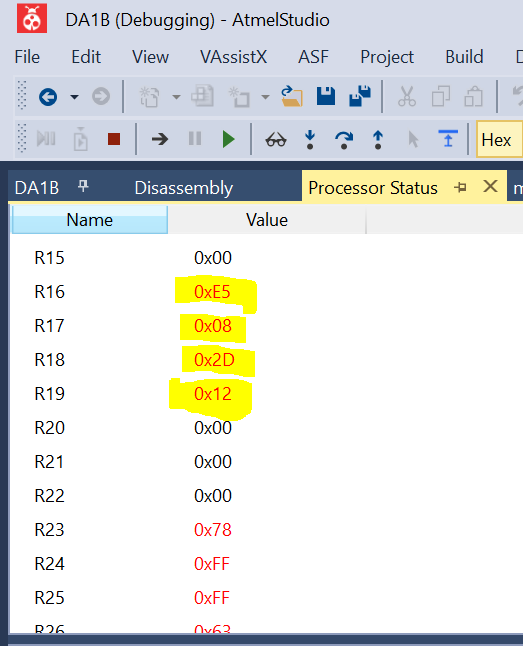
0x122D= 4653





When I added the line to clear R24 the code ran almost 3 times faster. It had no change for anything else the values were the same it was just faster by adding one line of code



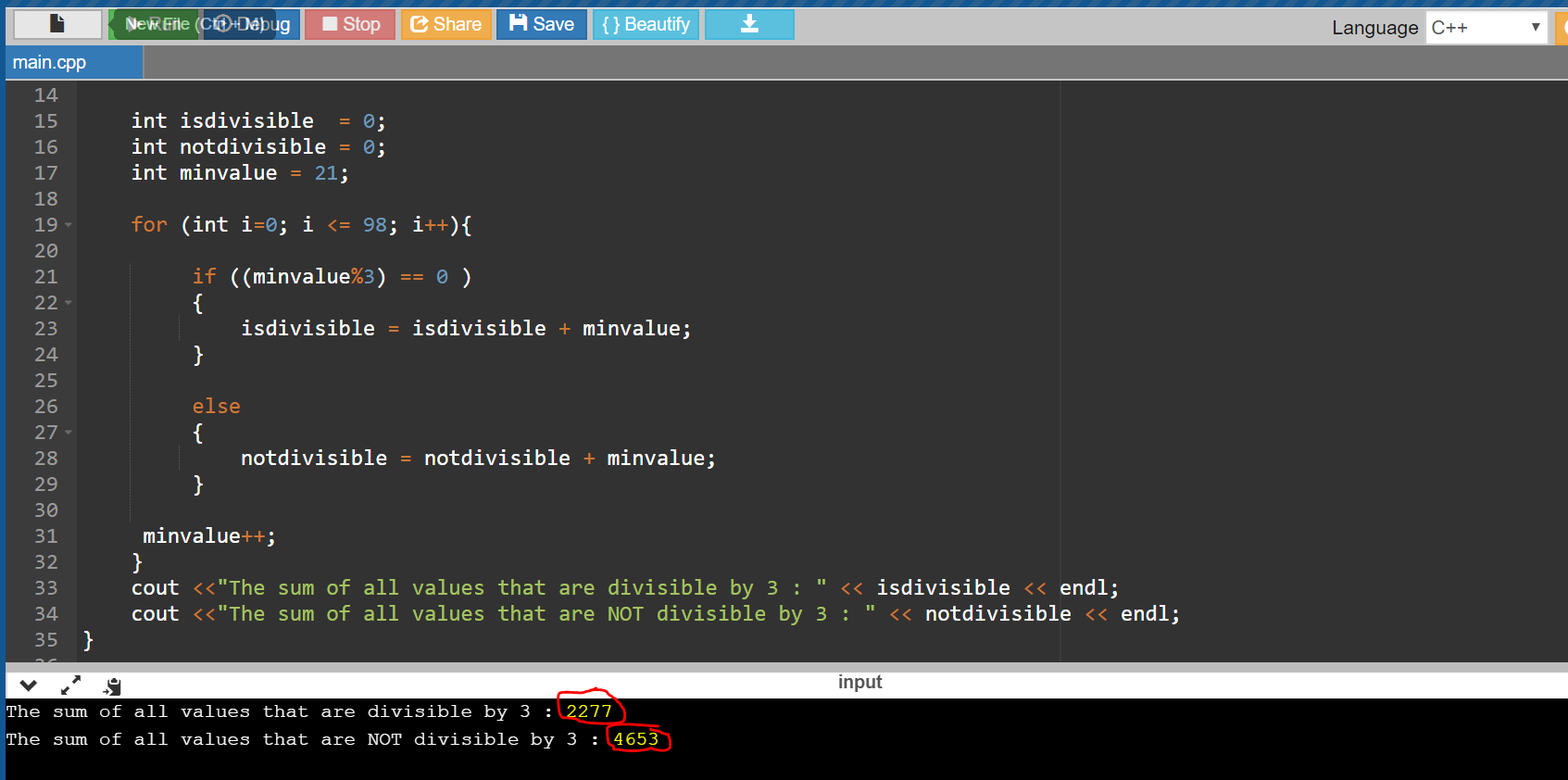


0x08E5= 2277

0x122D= 4653

So clearing the resgister in 24 makes it faster and I still get the same result and also have less clock cycles

Shows that assembly works by verifying with C++



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/martiv6/submissions\_da/tree/master/DesignAssignment/DA1B

**Student Academic Misconduct Policy**

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

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

Victor Martinez