Candace Caimol

**CPE301 – SPRING 2016**

Design Assignment 1

**DO NOT REMOVE THIS PAGE DURING SUBMISSION:**

The student understands that all required components should be submitted in complete for grading of this assignment.

|  |  |  |  |
| --- | --- | --- | --- |
| **NO** | **SUBMISSION ITEM** | **COMPLETED (Y/N)** | **MARKS**  **(/MAX)** |
| 1. | INITIAL CODE OF TASK 1/A |  |  |
| 2. | CODE OF TASK 2/B and 3/C |  |  |
| 3. | CODE OF TASK 4/D |  |  |
| 4. | SCREENSHOTS OF EACH TASK OUTPUT |  |  |
| 5. | EXECUTION TIME |  |  |
| 6. | GOOGLECODE LINK OF THE DA |  |  |
|  |  |  |  |
|  |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| 1. | INITIAL CODE OF TASK 1/A |  |  |

The code will be divided according to their respective tasks:

;

; DA1.asm

;

; Created: 2/23/2016 10:36:43 PM

; Author : user

;

; Replace with your application code

.def COUNT=R17

.def SetR7= R22 ;used to set R7.3 when sums are more than 8 bits

.org 0

LDI COUNT, 25 ;number of loops numbers stored

LDI SetR7, 8 ;value to set R7.3

LDI XL, LOW(RAMEND) ;loads lower 8 bits of RAMEND address

LDI XH, HIGH(RAMEND) ;loads upper 8 bits of RAMEND address

LDI YL, 0x7F ;starting midpoint in RAM (lower byte)

LDI YH, 0x04

SUB XL, YL ;XL = XL + YL (to get MID\_RAM)

SUB XH, YH ;XH = XH + YH

StoreNum:

MOV R16, XL ;moves lower 8 bytes of RAMEND/2 to R16 (value stored)

ST X+, R16 ;RAM[X] = R16

DEC COUNT ;COUNT--

BRNE StoreNum ;continue loop until all 25 numbers are stored

|  |  |  |  |
| --- | --- | --- | --- |
| 2. | CODE OF TASK 2/B and 3/C |  |  |

LDI R20, 0 ;initialize sum7 to 0 (lower byte)

LDI R21, 0 ;(higher byte)

LDI R23, 0 ;intialize sum3 to 0 (lower byte)

LDI R24, 0 ;(higher byte)

SUB R9, R9 ;initialize to 0; used to add to upper 8 bits of sum7 (ADC)

SUB R8, R8 ;initialize to 0; used to add to upper 8 bits of sum3 (ADC)

AddNums:

CPI COUNT, 25 ;compare to 25 (number of elements stored)

BREQ CmpSums ;if equal, jump to compare sums

LD R16, -X ;goes backwards toward RAM\_MID

MOV R19, R16 ;R19 = RAM[X] (used to find mod7)

MOV R18, R16 ;R18 = RAM[X] (used to find mod3)

FindModulus7:

SUBI R19, 7 ;R19 = R19 - 7

CPI R19, 7 ;compares R19 to 7

BRSH FindModulus7 ;if >= 7, continue finding modulus7

FindModulus3:

SUBI R18, 3 ;R18 = R18 - 3

CPI R18, 3 ;compares R18 to 3

BRSH FindModulus3 ;if >= 3, continue finding modulus3

INC COUNT ;COUNT++ (number of values parsed)

CPI R19, 0 ;compare mod7 to 0

BRNE CheckModulus3 ;if mod7 != 0, jump without adding

ADD R20, R16 ;lower byte: R20 = R20 + R16

ADC R21, R9 ;upper byte (with carry): R21 = R21 + 0

CheckModulus3:

CPI R18, 0 ;compare mod3 to 0

BRNE AddNums ;jump if mod3 != 0

ADD R23, R16 ;else add onto sum3

ADC R24, R8

JMP AddNums

|  |  |  |  |
| --- | --- | --- | --- |
| 3. | CODE OF TASK 4/D |  |  |

CmpSums:

CPI R21, 0 ;see if sum7 is greater than 8 bits

BREQ End ;if upper byte is empty, end program

CPI R24, 0 ;see if sum 3 is greater than 8 bits

BREQ End ;if upper byte is empty, end program

ADD R7, SetR7 ;else, set R7.3 to 1

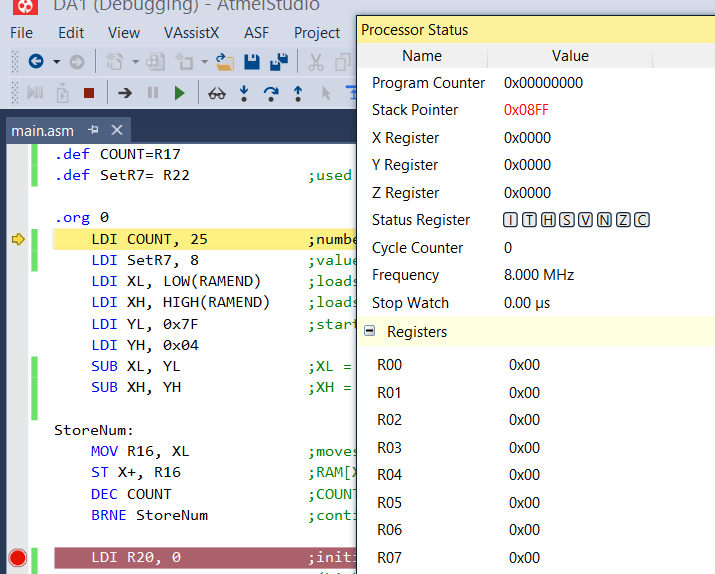
End:

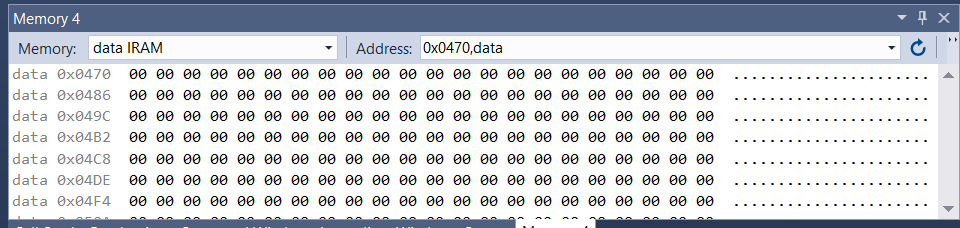
JMP End ;points to end of program

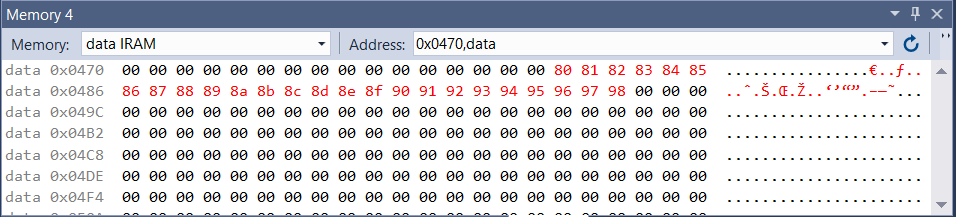
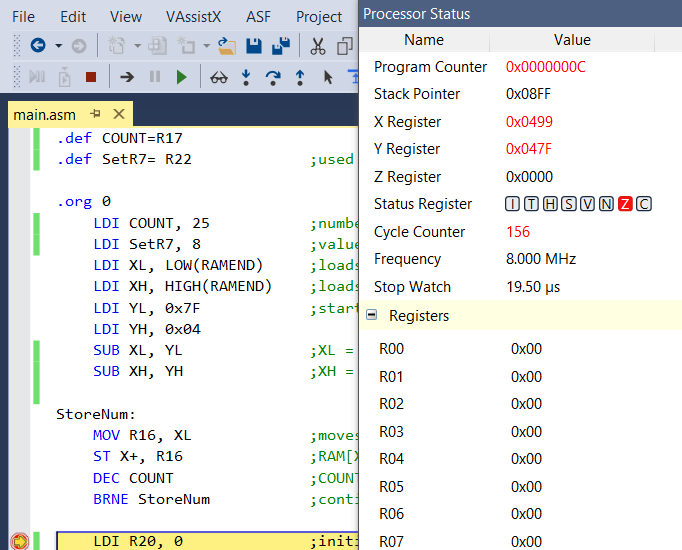
|  |  |  |  |
| --- | --- | --- | --- |
| 4. | SCREENSHOTS OF EACH TASK OUTPUT |  |  |

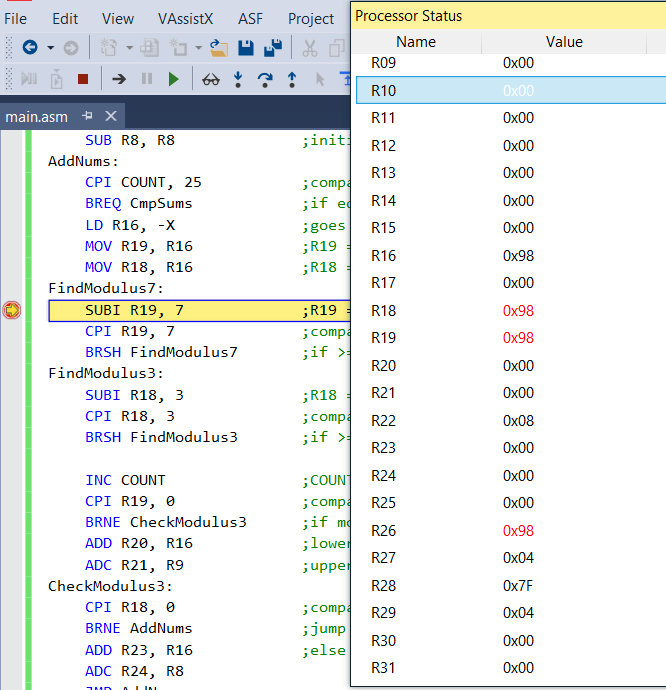
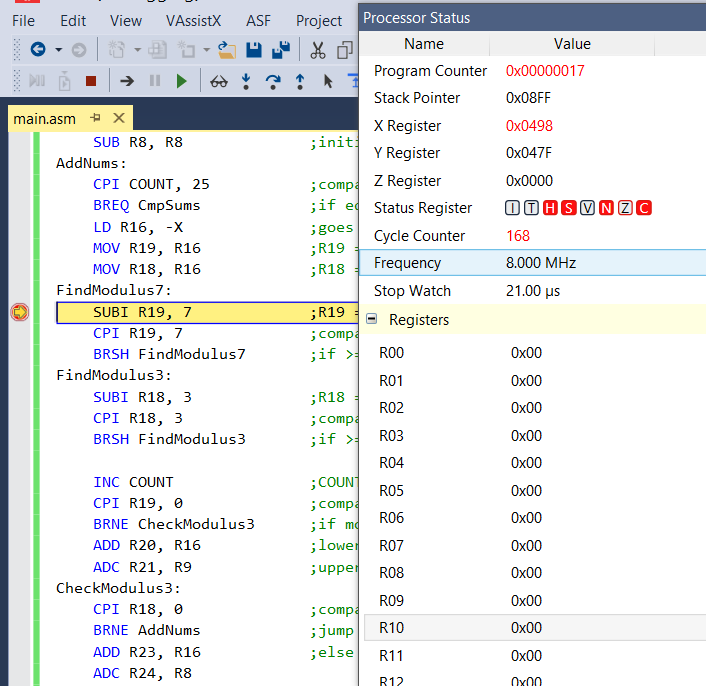
TASK 1/A:

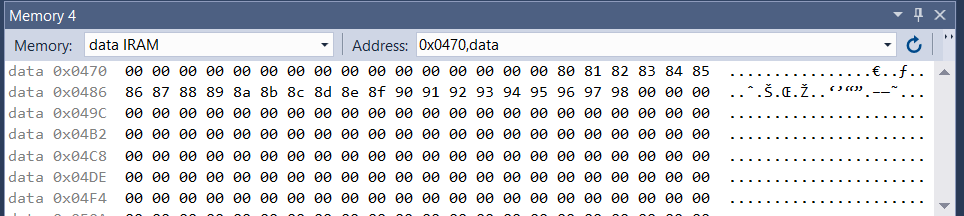
**Beginning of Task**  
X and Y registers are pointing at 0x00 by default. No values have been stored into RAM (all initialized at 0x00). The stopwatch is at 0.00us and the frequency is at 8MHz.

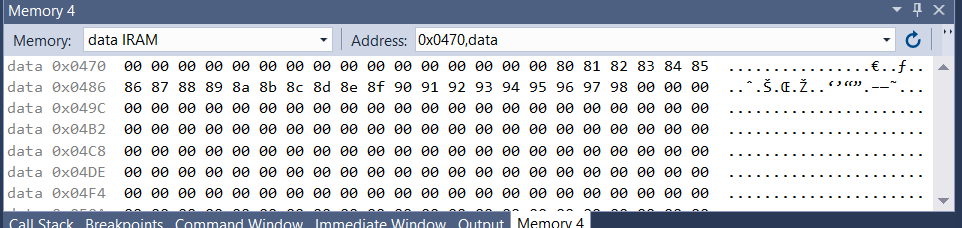
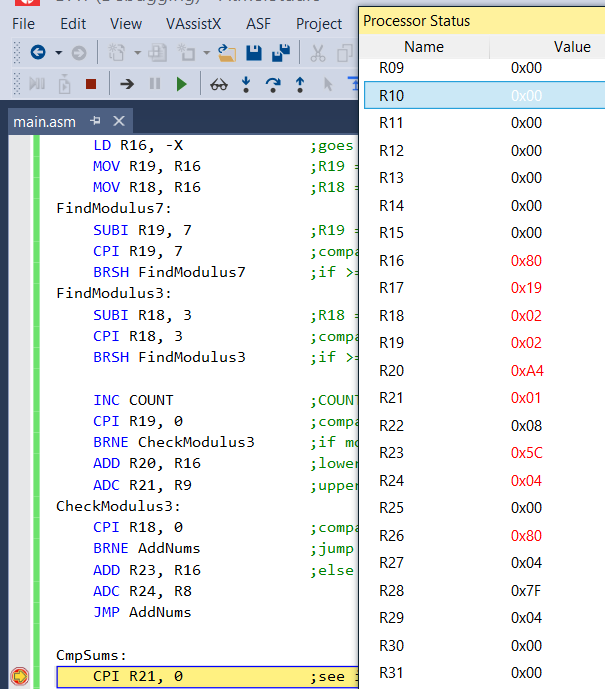
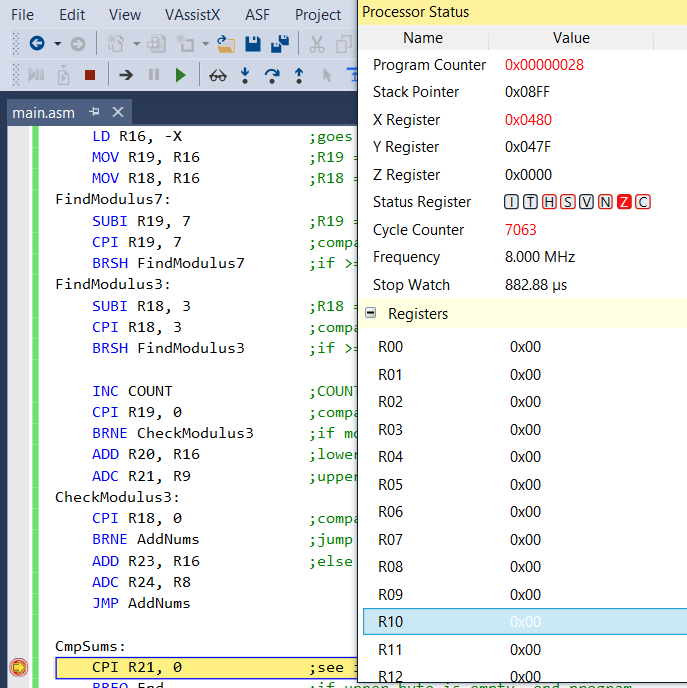


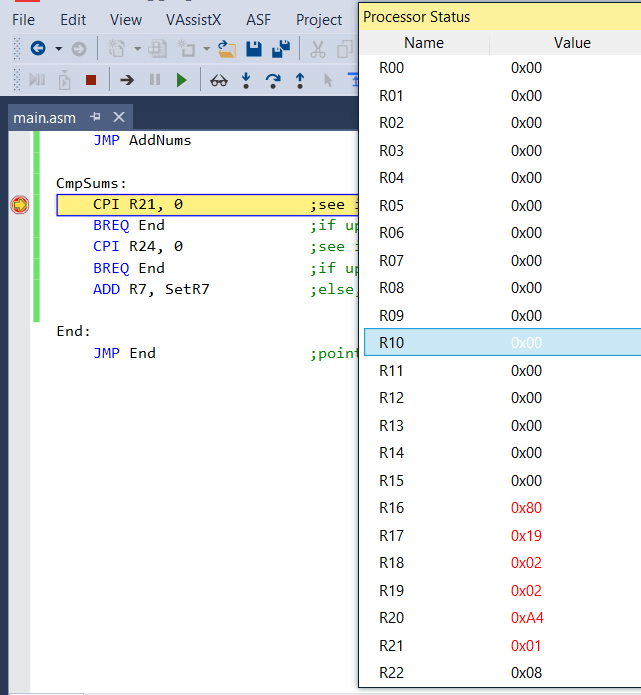


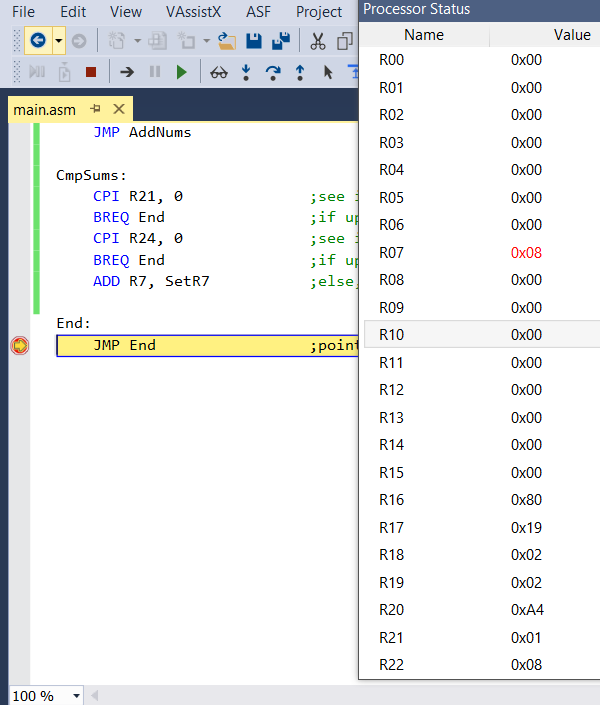
**End of Task**25 slots in RAM have been filled from location 0x0480 to 0x0498 with the last 8 bits of each respective address. Y is shown to be pointing to address 0x047F, which is the exact middle location of RAM (this value was subtracted from the initial RAMEND). X is pointing at RAM address 0x0499, which points to the memory slot right after the last stored element (due to the X incrementation). The stop watch is at 19.50us.   


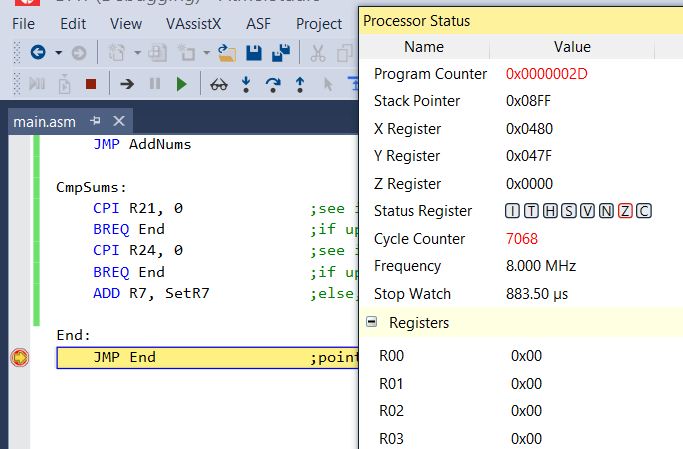
Task 2/B and 3/C  
**Beginning of Tasks**Registers have been initialized, including R21:R20 and R24:R23, which the sums of tasks 2/B and 3/C respectively. Since the pointer X has been set to traverse in the opposite direction (to parse through the stored numbers), X has gone from pointing to address 0x0499 to 0x0498. This address contains the last of the 25 elements. It can be noticed that these 25 elements still remain in their designated places in RAM. Meanwhile, R16, R18 and R19 all contain 0x98, the number stored at address 0x0498. R18 and R19 are used in the repeated subtraction loops in order to find modulus 7 and modulus 3, respectively. Meanwhile, R16 is the preserved value at the address, which comes into play if the value fits the criteria to be added onto either one of the total sums. At this point, the stop watch is at 25us.   




**End of Tasks**Since all 25 stored elements have been parsed, the X register is once again pointing to address location 0x0480, the location it originated from. Another good indicator that all 25 elements have been parsed is the value that the value in R17 (which is the counter indicating the current number of elements parsed) is now 0x19. The registers R21:R20 , which hold the sum of the numbers divisible by 7, contain the hex value 0x01A4. Similarly, registers R24:R23, which hold the sum of the numbers divisible by 3, together contain the hex value 0x45C. In spite of all of the elements being parsed, it can be seen that these values still remain stored in RAM. The stopwatch is now at 882.88us.   


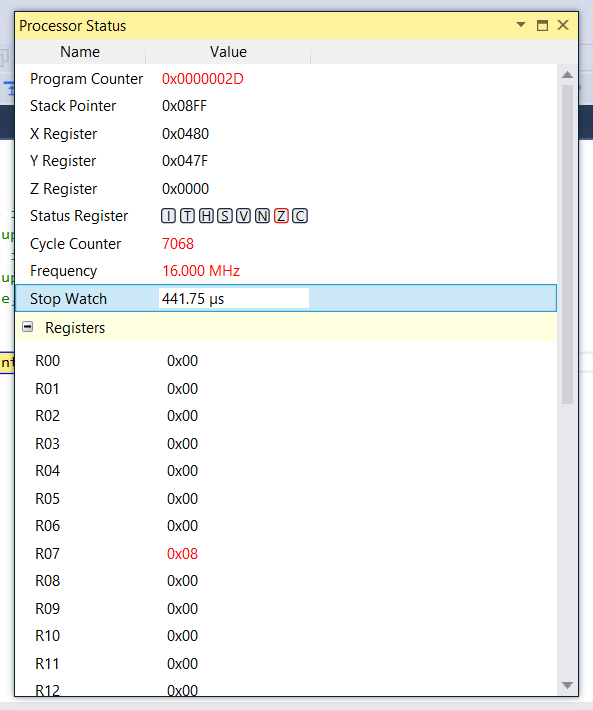
Task 4/D  
**Beginning of Task**This final task is designed to set R7.3 if both sums are greater than 8 bits long. At this point, R7 has been initialized to 0x00. Meanwhile, R22 has been initialized to 0x08, which is the value that will be added onto R7, and will consequently set the 3rd bit if the occasion calls for it.   


**End of Task**The sums were checked to see if they were longer than 8 bits long. This was done by checking to see if R21 and R24 (the upper bytes of the sums) were greater than 0. In this case, both of them are, which triggers R7.3 being set. This is done successfully as the value of R22 (0x08) is added onto the value of R7 (0x00). As a result, R7 now holds the value 0x08, meaning that the 3rd bit is set. The final time on the stopwatch is 883.50us.   




|  |  |  |  |
| --- | --- | --- | --- |
| 5. | EXECUTION TIME |  |  |

The frequency of the original debugging process was at 8MHz. I changed the frequency to 16MHz, and as seen below, the execution time of the whole program was cut roughly in half.



|  |  |  |  |
| --- | --- | --- | --- |
| 6. | GOOGLECODE LINK OF THE DA |  |  |
| https://github.com/candacecaimol/Design-Assignment-1.git | | | |

**Student Academic Misconduct Policy**

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

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

CANDACE CAIMOL