CPE301 – SPRING 2019

Design Assignment 1A

Student Name: Jasper Estorco
Student #: 2000836259
Student Email: estorco@unlv.nevada.edu
Primary Github address: https://github.com/toobsock/online_submission
Directory: online_submission/DA/1A

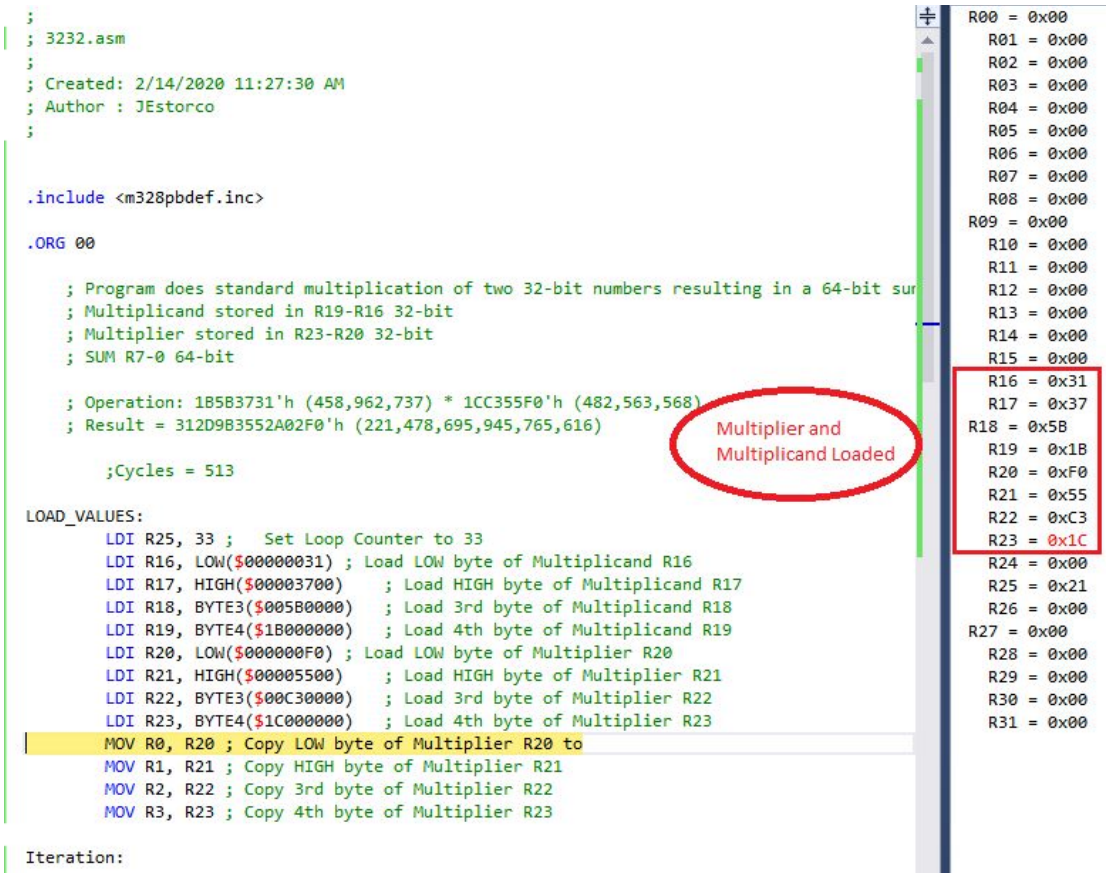
1. DEVELOPED CODE OF TASK 1-2

```
;
; Assign1A.asm
;
; Created: 2/14/2020 11:27:30 AM
; Author : JEstorco
;


.include <m328pbdef.inc>

.ORG 00

        ; Program does standard multiplication of two 32-bit numbers resulting in a 64-bit sum
        ; Multiplicand stored in R19-R16 32-bit
        ; Multiplier stored in R23-R20 32-bit
        ; SUM R7-0 64-bit

   ; Operation: 1B5B3731'h (458,962,737) * 1CC355F0'h (482,563,568)
        ; Result = 312D9B3552A02F0'h (221,478,695,945,765,616)

                ;Cycles = 513

LOAD_VALUES:
                LDI R25, 33     ;       Set Loop Counter to 33
      LDI R20, LOW($000000F0)     ; Load LOW byte of Multiplier R20
      LDI R21, HIGH($00005500)    ; Load HIGH byte of Multiplier R21
      LDI R22, BYTE3($00C30000)   ; Load 3rd byte of Multiplier R22
      LDI R23, BYTE4($1C000000)   ; Load 4th byte of Multiplier R23
        LDI R16, LOW($00000031)   ; Load LOW byte of Multiplicand R16
      LDI R17, HIGH($00003700)    ; Load HIGH byte of Multiplicand R17
```

```
    LDI R18, BYTE3($005B0000)   ; Load 3rd byte of Multiplicand R18
    LDI R19, BYTE4($1B000000)   ; Load 4th byte of Multiplicand R19
    MOV R3, R23     ; Copy 4th byte of Multiplier R23 (Can't use LDI instruction)
      MOV R2, R22 ; Copy 3rd byte of Multiplier R22 (Can't use LDI instruction)
      MOV R1, R21 ; Copy HIGH byte of Multiplier R21 (Can't use LDI instruction)
      MOV R0, R20 ; Copy LOW byte of Multiplier R20 (Can't use LDI instruction)

Iteration:
    ROR R3  ; Shift 4th byte (of Multiplier) R23 to right in R3
    ROR R2  ; Shift 3rd byte (of Multiplier) R22 to right in R2
    ROR R1  ; Shift HIGH byte (of Multiplier) R21 to right in R1
    ROR R0  ; Shift LOW byte (of Multiplier) R20 to right in R0
    DEC R25; Decrement Loop Counter
    BREQ END        ; Branch to end of Program if All 32 Bits Completed
    BRCC IGNORE  ; Branch to skip additon if carry = zero
    ADD R4, R16     ; Add LOW byte (of Multipicand) R16 to R4 and store result to R4
    ADC R5, R17     ; Add HIGH bye (of Multipicand) R17 to R5 and store result to R5
    ADC R6, R18     ; Add 3rd byte (of Multipicand) R18 to R6 and store result to R6
    ADC R7, R19     ; Add 4th byte (of Multipicand) R19 to R7 and store result to R7

IGNORE:
    ROR R7  ; Shift 4th byte (of Multiplicand R19) to right in R7
    ROR R6  ; Shift 3rd byte (of Multiplicand R18) to right in R6
    ROR R5  ; Shift HIGH byte (of Multiplicand R17) to right in R5
    ROR R4  ; Shift LOW byte (of Multiplicand R16) to right in R4
    RJMP Iteration   ; Proceed with remaining Iteration

END:
            RJMP END ; Finish program - Infinite Loop
```

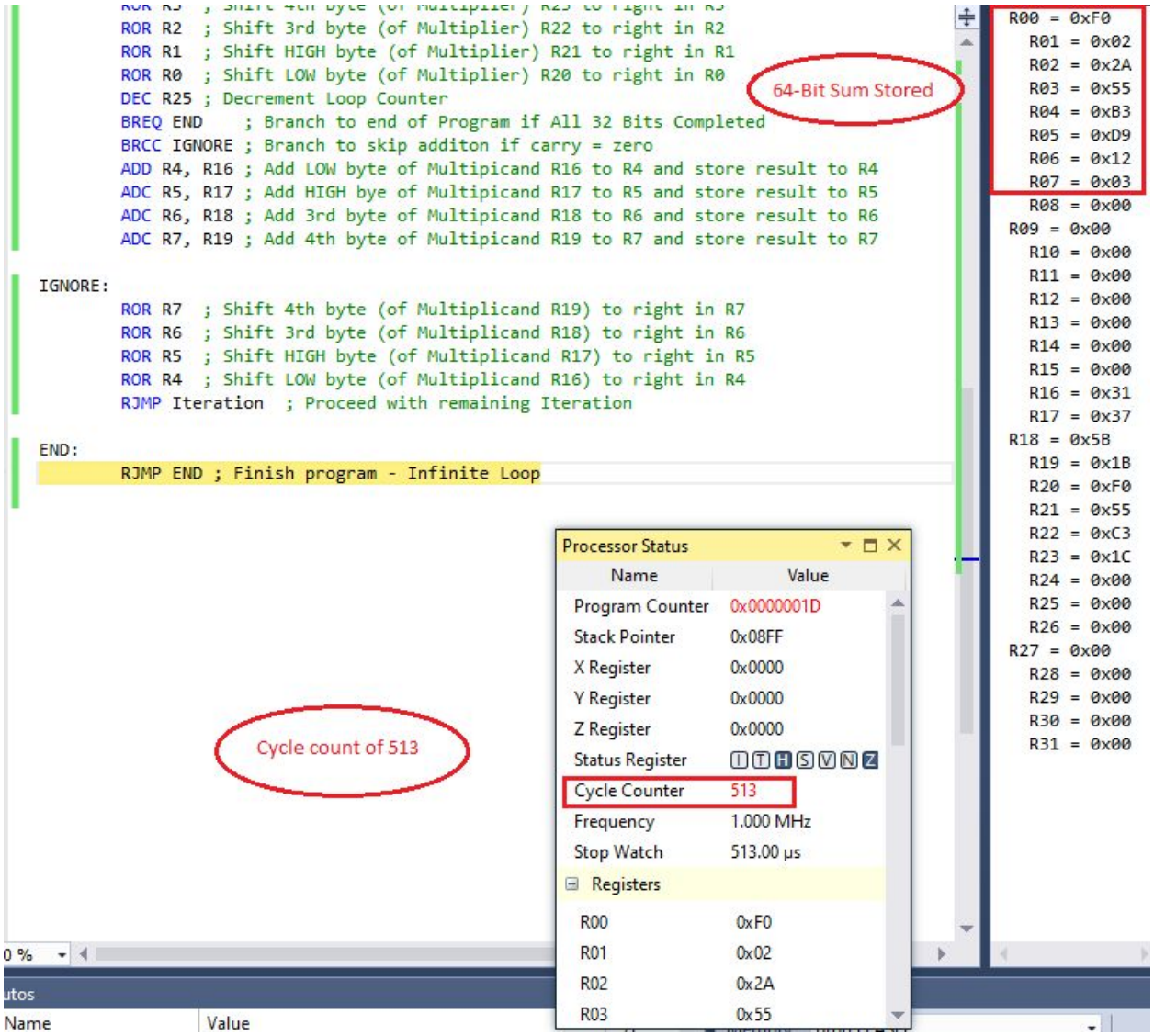
2. SCREENSHOTS OF EACH TASK OUTPUT (ATMEL STUDIO OUTPUT)

Beginning of Simulation:



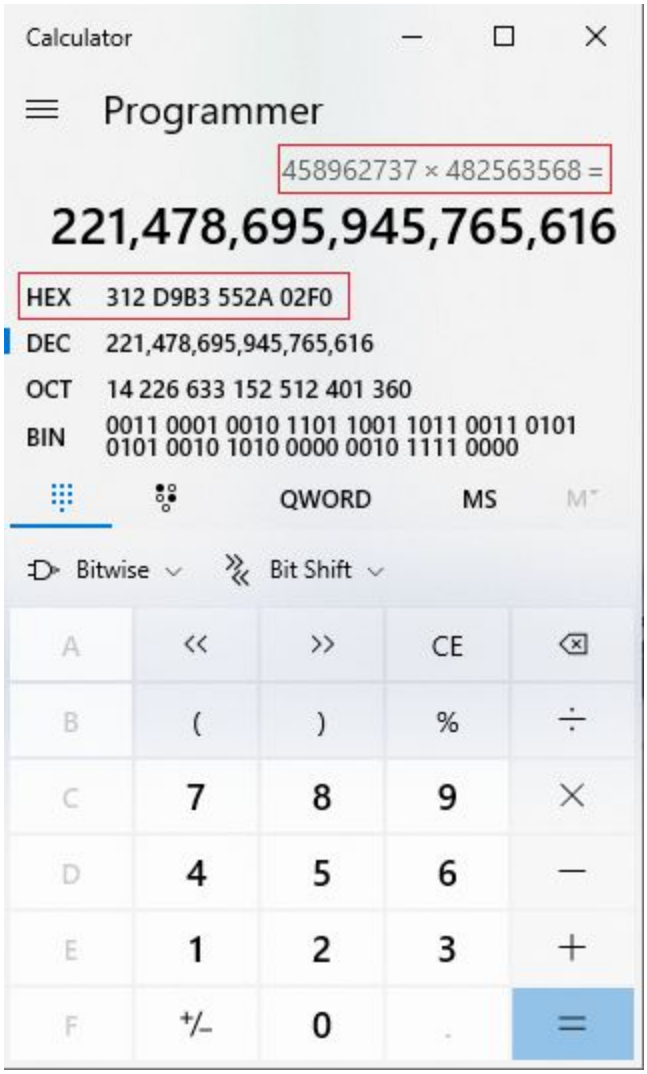
Multiplicand and Multiplier Loaded into the Registers

End of Simulation:



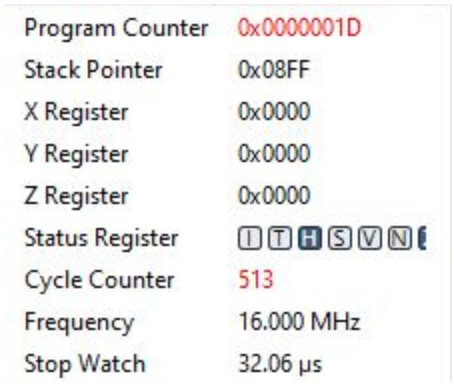
Multiplication with Iterations completed, sum stored in 8 Registers for a 64-bit result. Processor Status window from debugger measures 513 cycles.

Calculation:



Execution Time:

16MHz clock / 513 cycles = 31.2 µs



3. GITHUB LINK OF THIS DA

https://github.com/toobsock/online_submission/tree/master/DA/1A

Student Academic Misconduct Policy
http://studentconduct.unlv.edu/misconduct/policy.html

"This assignment submission is my own, original work".
Jasper Estorco