CPE301 – SPRING 2019

Design Assignment 1B

Student Name: Ron Joshua Recrio
Student #: 5003825419
Student Email: recrio@unlv.nevada.edu
Primary Github address: https://github.com/recrio/submissions
Directory: submissions/DesignAssignments/DA1B

1. COMPONENTS LIST AND CONNECTION BLOCK DIAGRAM w/ PINS

N/A

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

```
.ORG 0
.EQU STARTADDS = 0x0200
.EQU DIVISIBLES = 0x0400
.EQU NDIVISIBLE = 0x0600

       LDI R23, 0                      ; Holds Zero
       LDI R20, 99                     ; Counter = 99
       LDI R25, 11                     ; First Value = 11
       LDI XL, low(STARTADDS)          ; X points to
       LDI XH, high(STARTADDS)         ; 0x0200
       LDI YL, low(DIVISIBLES)         ; Y points to
       LDI YH, high(DIVISIBLES)        ; 0x0400
       LDI ZL, low(NDIVISIBLE)         ; Z points to
       LDI ZH, high(NDIVISIBLE)        ; 0x0600

POPULATE:
       ST X+, R25                      ; Store R25 into X, increment pointer
       INC R25                         ; Increment R25
       DEC R20                         ; Decrement Counter
       BRNE POPULATE                   ; Keep populating if counter != 0
```

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

```
       LDI XL, low(STARTADDS)          ; X points to
       LDI XH, high(STARTADDS)         ; 0x0200 again
       LDI R20, 99+1                   ; Counter = 99+1 because BREQ
                                       ; checks before the operations

DIV3:
       DEC R20                         ; Decrement Counter
       BREQ DONEDIV                    ; If counter = 0, parsed through all numbers
       LD R25, X                       ; R25 = X data
       LD R24, X+                      ; R24 = X data, increment X pointer
CHECK:
```

```
        CPI R25, 0                      ; Check if R25 = 0
        BREQ DIVIDES                    ; If 0 then divisible
        SUBI R25, 3                     ; Subtract R25 by 3
        BRPL CHECK                      ; If not negative, go back to CHECK
NDIVIDES:
        ST Z+, R24                      ; store R24 in Z
        RJMP DIV3                       ; Jump to DIV3
DIVIDES:
        ST Y+, R24                      ; store R25 in Y
        RJMP DIV3                       ; Jump to DIV3
```

4. DEVELOPED MODIFIED CODE OF TASK 3/A from TASK 2/A

```
DONEDIV:
        LDI R20, 99                     ; Counter set to 99 again
        LDI YL, low(DIVISIBLES)         ; Y points to
        LDI YH, high(DIVISIBLES)        ; 0x0400 again
        LDI ZL, low(NDIVISIBLE)         ; Z points to
        LDI ZH, high(NDIVISIBLE)        ; 0x0600 again

SUM:
        LD R21, Y+                      ; R21 = Y data, increment Y pointer
        LD R22, Z+                      ; R22 = Z data, increment Z pointer
        ADD R17, R21                    ; R17 = R17 + R21
        ADC R16, R23                    ; R16 = R16 + 0 + Carry
        ADD R19, R22                    ; R19 = R19 + R22
        ADC R18, R23                    ; R18 = R18 + 0 + Carry
        DEC R20                         ; Decrement counter
        BRNE SUM                        ; if counter != 0 keep summing

DONE: RJMP DONE
```

5. Full Code

```
.ORG 0
.EQU STARTADDS = 0x0200
.EQU DIVISIBLES = 0x0400
.EQU NDIVISIBLE = 0x0600

        LDI R23, 0                      ; Holds Zero
        LDI R20, 99                     ; Counter = 99
        LDI R25, 11                     ; First Value = 11
        LDI XL, low(STARTADDS)          ; X points to
        LDI XH, high(STARTADDS)         ; 0x0200
        LDI YL, low(DIVISIBLES)         ; Y points to
        LDI YH, high(DIVISIBLES)        ; 0x0400
        LDI ZL, low(NDIVISIBLE)         ; Z points to
        LDI ZH, high(NDIVISIBLE)        ; 0x0600

POPULATE:
        ST X+, R25                      ; Store R25 into X, increment pointer
        INC R25                         ; Increment R25
        DEC R20                         ; Decrement Counter
        BRNE POPULATE                   ; Keep populating if counter != 0

        LDI XL, low(STARTADDS)          ; X points to
```

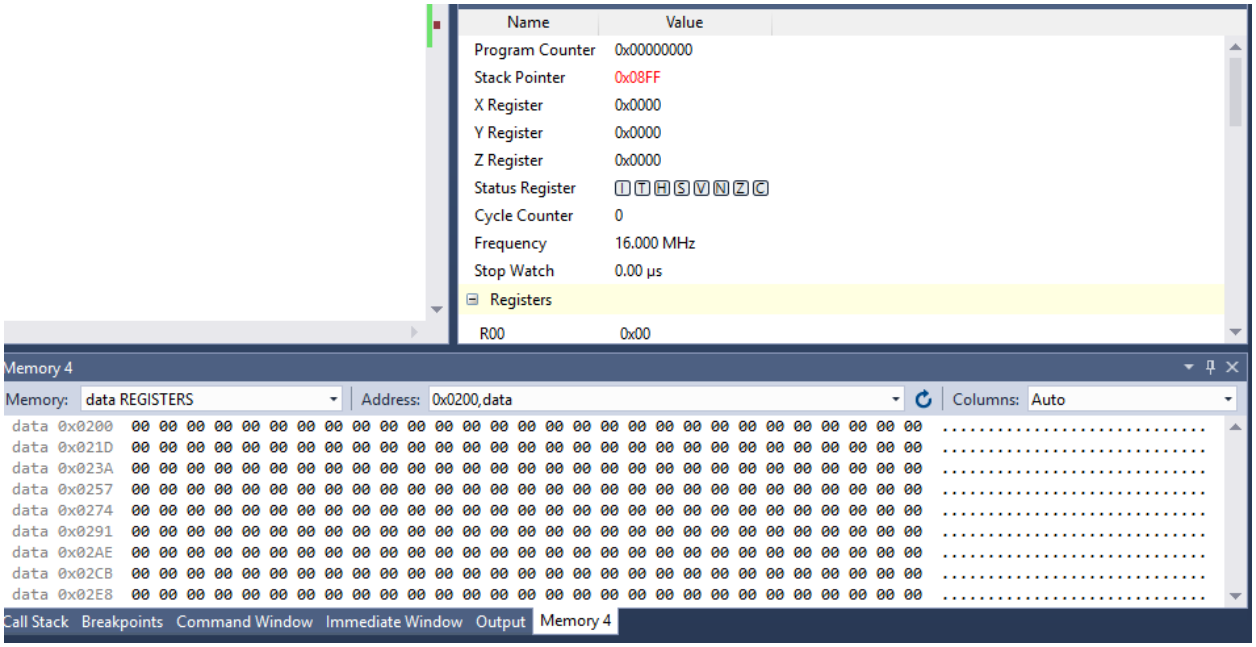
```
        LDI XH, high(STARTADDS)          ; 0x0200 again
        LDI R20, 99+1                    ; Counter = 99+1 because BREQ
                                         ; checks before the operations

DIV3:
        DEC R20                          ; Decrement Counter
        BREQ DONEDIV                     ; If counter = 0, parsed through all numbers
        LD R25, X                        ; R25 = X data
        LD R24, X+                       ; R24 = X data, increment X pointer
CHECK:
        CPI R25, 0                       ; Check if R25 = 0
        BREQ DIVIDES                     ; If 0 then divisible
        SUBI R25, 3                      ; Subtract R25 by 3
        BRPL CHECK                       ; If not negative, go back to CHECK
NDIVIDES:
        ST Z+, R24                       ; store R24 in Z
        RJMP DIV3                        ; Jump to DIV3
DIVIDES:
        ST Y+, R24                       ; store R25 in Y
        RJMP DIV3                        ; Jump to DIV3
```

6. SCHEMATICS

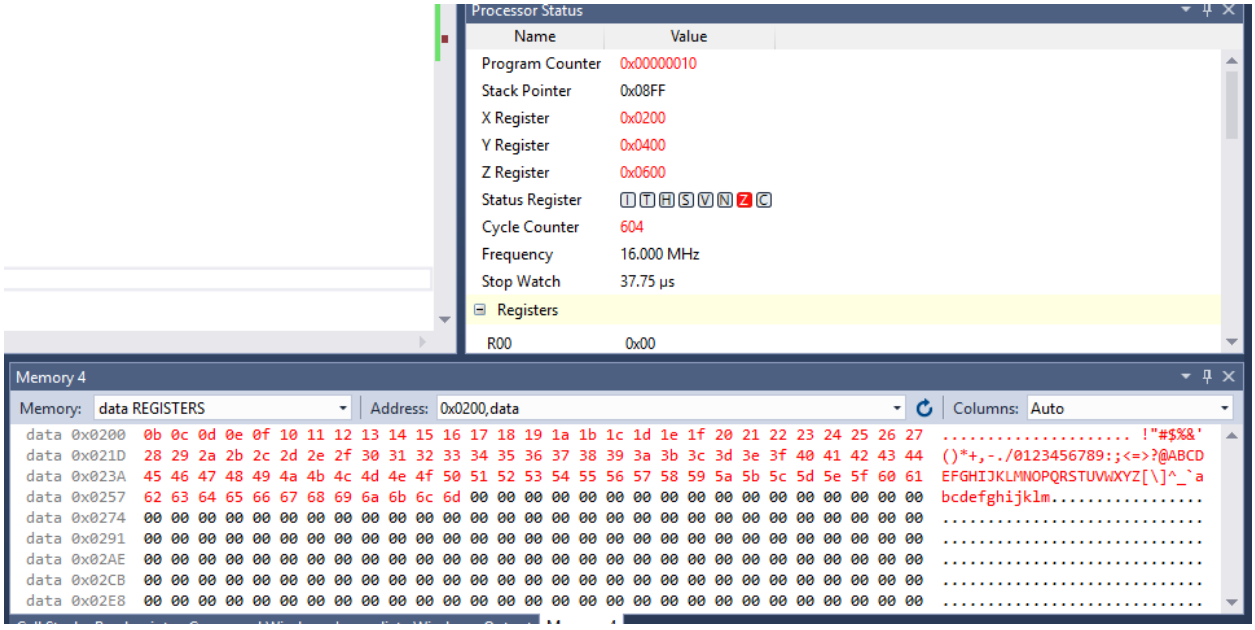
N/A

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

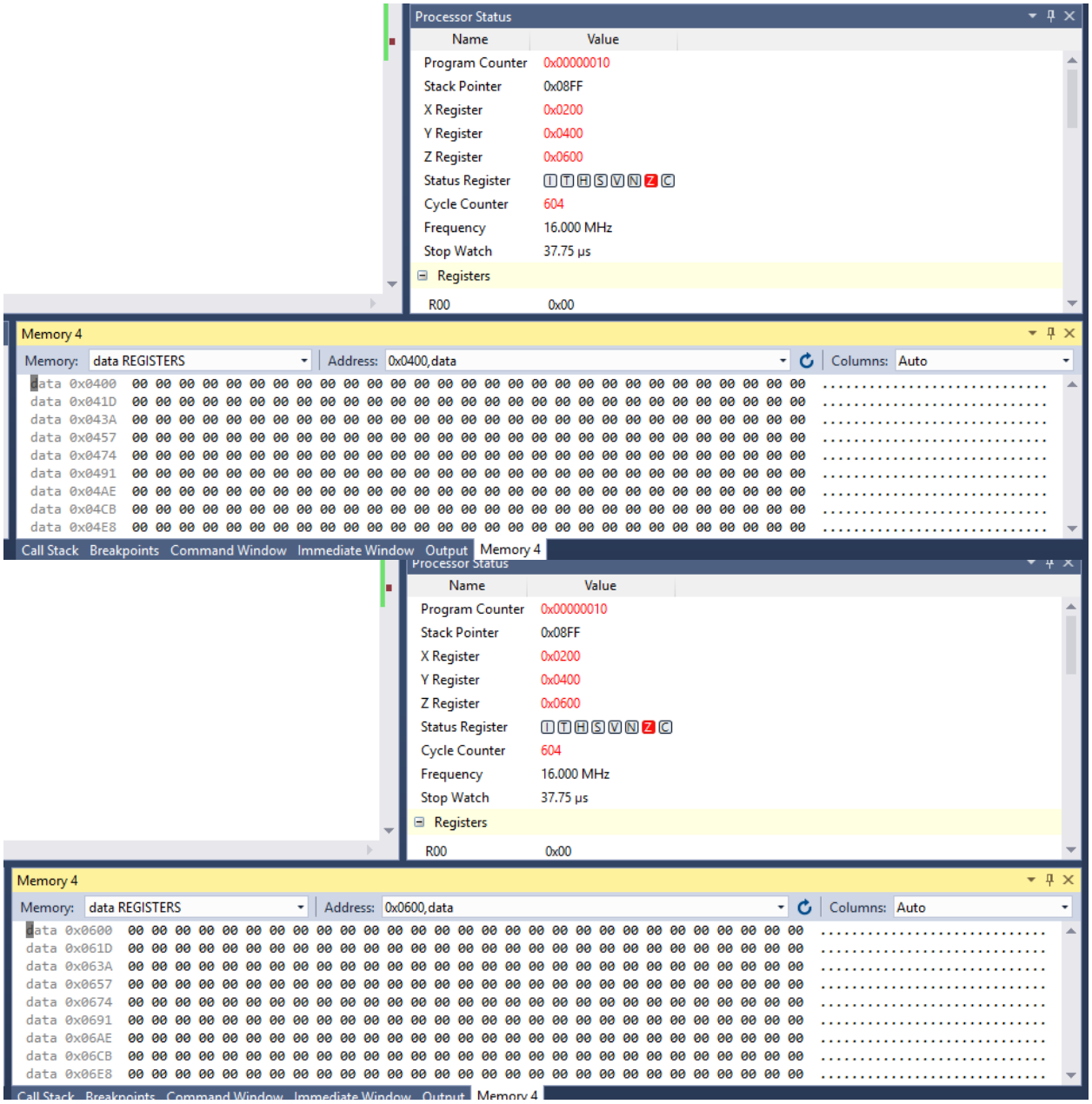
Task 1: Store 99 numbers starting from 0x0200
Before:



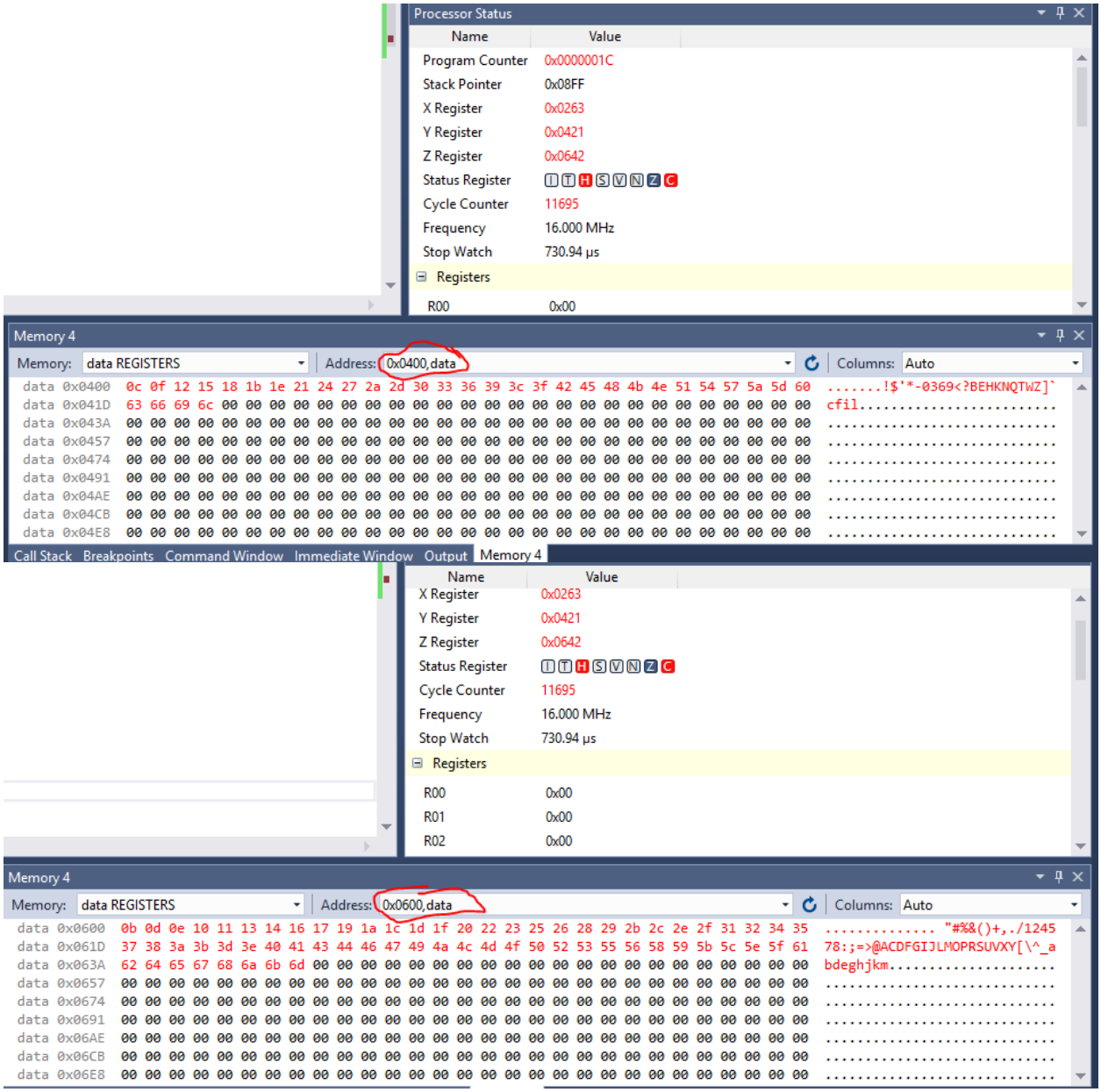
After:


Processor Status
Name
Value
Program Counter
0x00000010
Stack Pointer
0x08FF
X Register
0x0200
Y Register
0x0400
Z Register
0x0600
Status Register
I T H S V N Z C
Cycle Counter
604
Frequency
16.000 MHz
Stop Watch
37.75 µs
Registers
R00
0x00
Memory 4
Memory: data REGISTERS
Address: 0x0200,data
Columns: Auto
data 0x0200 0b 0c 0d 0e 0f 10 11 12 13 14 15 16 17 18 19 1a 1b 1c 1d 1e 1f 20 21 22 23 24 25 26 27 ..... !"#$%&'
data 0x021D 28 29 2a 2b 2c 2d 2e 2f 30 31 32 33 34 35 36 37 38 39 3a 3b 3c 3d 3e 3f 40 41 42 43 44 ()*+,-./0123456789:;<=>?@ABCD
data 0x023A 45 46 47 48 49 4a 4b 4c 4d 4e 4f 50 51 52 53 54 55 56 57 58 59 5a 5b 5c 5d 5e 5f 60 61 EFGHIJKLMNOPQRSTUVWXYZ[\]^_`a
data 0x0257 62 63 64 65 66 67 68 69 6a 6b 6c 6d 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 bcdefghijklm.....
data 0x0274 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
data 0x0291 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
data 0x02AE 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
data 0x02CB 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
data 0x02E8 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....

Task 2: Store divisible numbers at 0x0400 and the rest at 0x0600
Before:



After:


Processor Status
Name
Value
Program Counter
0x0000001C
Stack Pointer
0x08FF
X Register
0x0263
Y Register
0x0421
Z Register
0x0642
Status Register
I T H S V N Z C
Cycle Counter
11695
Frequency
16.000 MHz
Stop Watch
730.94 µs
Registers
R00
0x00
Memory 4
Memory: data REGISTERS
Address: 0x0400,data
Columns: Auto
data 0x0400 0c 0f 12 15 18 1b 1e 21 24 27 2a 2d 30 33 36 39 3c 3f 42 45 48 4b 4e 51 54 57 5a 5d 60 .....!$'*-0369<?BEHKNQTWZ]`
data 0x041D 63 66 69 6c 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 cfil
data 0x043A 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
data 0x0457 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
data 0x0474 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
data 0x0491 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
data 0x04AE 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
data 0x04CB 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
data 0x04E8 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
Call Stack Breakpoints Command Window Immediate Window Output Memory 4
Name
Value
X Register
0x0263
Y Register
0x0421
Z Register
0x0642
Status Register
I T H S V N Z C
Cycle Counter
11695
Frequency
16.000 MHz
Stop Watch
730.94 µs
Registers
R00
0x00
R01
0x00
R02
0x00
Memory 4
Memory: data REGISTERS
Address: 0x0600,data
Columns: Auto
data 0x0600 0b 0d 0e 10 11 13 14 16 17 19 1a 1c 1d 1f 20 22 23 25 26 28 29 2b 2c 2e 2f 31 32 34 35 ..... "#%&()+,./1245
data 0x061D 37 38 3a 3b 3d 3e 40 41 43 44 46 47 49 4a 4c 4d 4f 50 52 53 55 56 58 59 5b 5c 5e 5f 61 78:;=>@ACDFGIJLMOPRSUVXY[\^_a
data 0x063A 62 64 65 67 68 6a 6b 6d 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 bdeghjkm
data 0x0657 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
data 0x0674 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
data 0x0691 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
data 0x06AE 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
data 0x06CB 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
data 0x06E8 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

Task 3: Simultaneously add up 0x0400 and 0x0600 numbers to R16:R17 and R18:R19 respectively

Before:

```
DONEDIV:
    LDI R20, 99             ; Counter set to 99 again
    LDI YL, low(DIVISIBLES) ; Y points to
    LDI YH, high(DIVISIBLES); 0x0400 again
    LDI ZL, low(NDIVISIBLE) ; Z points to
    LDI ZH, high(NDIVISIBLE); 0x0600 again

SUM:
    LD R21, Y+              ; R21 = Y data, increment Y pointer
    LD R22, Z+              ; R22 = Z data, increment Z pointer
    ADD R17, R21            ; R17 = R17 + R21
    ADC R16, R23            ; R16 = R16 + 0 + Carry
    ADD R19, R22            ; R19 = R19 + R22
    ADC R18, R23            ; R18 = R18 + 0 + Carry
    DEC R20                 ; Decrement counter
    BRNE SUM                ; if counter != 0 keep summing

DONE: RJMP DONE
```

Processor Status

Name	Value
Program Counter	0x0000001C
Stack Pointer	0x08FF
X Register	0x0263
Y Register	0x0421
Z Register	0x0642
Status Register	I T H S V N Z C
Cycle Counter	11695
Frequency	16.000 MHz
Stop Watch	730.94 µs

Watch 1

Name	Value	Type
R16	0x00	byte{reg
R17	0x00	byte{reg
R18	0x00	byte{reg
R19	0x00	byte{reg

Registers Autos Locals Watch 1 Watch 2

After:

```
    LDI R20, 99             ; Counter set to 99 again
    LDI YL, low(DIVISIBLES) ; Y points to
    LDI YH, high(DIVISIBLES); 0x0400 again
    LDI ZL, low(NDIVISIBLE) ; Z points to
    LDI ZH, high(NDIVISIBLE); 0x0600 again

SUM:
    LD R21, Y+              ; R21 = Y data, increment Y pointer
    LD R22, Z+              ; R22 = Z data, increment Z pointer
    ADD R17, R21            ; R17 = R17 + R21
    ADC R16, R23            ; R16 = R16 + 0 + Carry
    ADD R19, R22            ; R19 = R19 + R22
    ADC R18, R23            ; R18 = R18 + 0 + Carry
    DEC R20                 ; Decrement counter
    BRNE SUM                ; if counter != 0 keep summing

DONE: RJMP DONE
```

Processor Status

Name	Value
Program Counter	0x00000029
Stack Pointer	0x08FF
X Register	0x0263
Y Register	0x0463
Z Register	0x0663
Status Register	I T H S V N Z C
Cycle Counter	12788
Frequency	16.000 MHz
Stop Watch	799.25 µs

Watch 1

Name	Value	Type
R16	0x07	byte{reg
R17	0xbc	byte{reg
R18	0x0f	byte{reg
R19	0x78	byte{reg

Registers Autos Locals Watch 1 Watch 2

0x07BC = 1980
0x0F78 = 3960

Task 4: Verification using C++

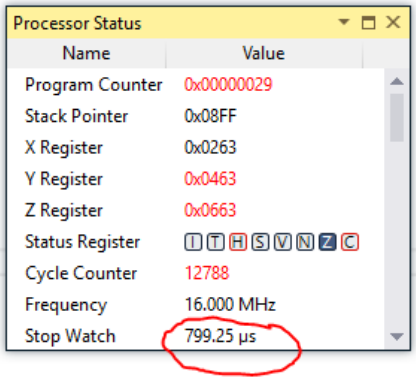
```cpp
// Example program
#include <iostream>
#include <string>
using namespace std;

int main()
{
    int x[99], y[99], z[99];
    int ylen = 0, zlen = 0, sumy = 0, sumz =0;

    for (int i = 0; i < 99; i++) {
        x[i] = i+11;
        if (x[i] % 3 == 0) {
            y[ylen] = x[i];
            ylen++;
        }
        else {
            z[zlen] = x[i];
            zlen++;
        }
    }

    cout << "x = [";

    for (int i = 0; i < 99; i++)
        cout << x[i] << ", ";
    cout << "]\n";

    cout << "y = [";

    for (int i = 0; i < ylen; i++) {
        sumy += y[i];
        cout << y[i] << ", ";
    }
    cout << "]\n";

    cout << "z = [";

    for (int i = 0; i < zlen; i++) {
        sumz += z[i];
        cout << z[i] << ", ";
    }
    cout << "]\n";

    cout << "Sum of Y = " << sumy << endl;
    cout << "Sum of Z = " << sumz << endl;
}
```

Output:

```
x = [11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22
y = [12, 15, 18, 21, 24, 27, 30, 33, 36, 39, 42, 45
z = [11, 13, 14, 16, 17, 19, 20, 22, 23, 25, 26, 28
Sum of Y = 1980
Sum of Z = 3960
```

Task 5: Execution Time

```
SUM:
    LD R21, Y+              ; R21 = Y data, increment Y pointer
    LD R22, Z+              ; R22 = Z data, increment Z pointer
    ADD R17, R21            ; R17 = R17 + R21
    ADC R16, R23            ; R16 = R16 + 0 + Carry
    ADD R19, R22            ; R19 = R19 + R22
    ADC R18, R23            ; R18 = R18 + 0 + Carry
    DEC R20                 ; Decrement counter
    BRNE SUM                ; if counter != 0 keep summing

DONE: RJMP DONE
```



12788/16MHz = 799.25µs

8. SCREENSHOT OF EACH DEMO (BOARD SETUP)

N/A

9. VIDEO LINKS OF EACH DEMO

N/A

10. GITHUB LINK OF THIS DA

https://github.com/recrio/submissions/tree/master/DesignAssignments/DA1B

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

"*This assignment submission is my own, original work*".
Ron Joshua Recrio