

CPE301 – SPRING 2019

Design Assignment DA1B

Student Name: Gabriela Cuicas

Student #: 5002960022

Student Email: cuicas@unlv.nevada.edu

Primary Github address: <https://github.com/cuicattack/cat1>

Directory: <https://github.com/cuicattack/cat1/Cat1Assn2>

1. DEVELOPED CODE OF TASK 1

Task 1 stated to:

Store 99 numbers starting from the STARTADDS=0x0200 location. Populate the value of the memory location by adding high(STARTADDS) and low(STARTADDS). Use the X/Y/Z registers as pointers to fill up 99 numbers that are greater than 10 and less than 255. The numbers can be consecutive or random numbers.

To perform the operation, I started by defining the memory location of my x register to be 0x0200. I initialized a register to 10 and made a loop that would increment that register and continue to store the value into the next space in register x.

```
LDI x1, low(0x0200) ; low byte in x memory
LDI xh, high(0x0200) ; high byte in x memory

LDI R20, 10 ; initialize with 10, but after incremented start register x with value 11
LDI R22, 0 ; always stays zero

num99:
INC R20 ;increment value in register everytime loop begins
ST x+,R20 ; increments place in x after storing.
MOV R21, R20 ; copy the value of R20 without making changes to R20 so that we can manipulate value
JMP div ; now we want to check if the number is divisable by 3
```

2. DEVELOPED CODE OF TASK 2

The second task asked to:

Use X/Y/Z register addressing to parse through the 99 numbers, if the number is divisible by 3 store the number starting from memory location 0x0400, else store at location starting at 0x0600.

To perform this operation, I made a loop that continuously subtracted a number by 3 until the number was less than three. Every time a new number was generated and stored in x it would check to see if the number was divisible by 3. From there it would check to see if the remainder was 0 or a number to branch to the next areas. If the remainder was 0 it would store the original unmodified number of x into an array of y. Otherwise, it would store that value into the z registers.

```
div:
SUBI R21, 3 ; subtract the value by 3
CPI R21, 3 ; Compare the subtracted value by 3
BRSH div ; keep subtracting if = 3 or greater

CPI R21, 3 ; compare if value is equal to 3
BRLO remainders ; if it is lower go to remainders label
```

```

remainders:
    CPI R21, 0 ; compare if value if equal to zero
    BREQ remzero ; if so branch to remzero, because the number is visible by 3
    JMP remanum ; other wise we do have a remainder and go to remanum

remzero:
    ADD R17, R20 ; add all numbers that are divisible by 3
    ADC R16, R22 ; add all numbers that are divisible by 3
    ST y+, R20 ; store all the numers that are divisible by 3 in the y array
    JMP compare ; finally jump to compare to see if we are in the last number of the 99 numbers
remanum:
    ADD R19, R20 ; add all numbers that are not divisible by 3
    ADC R18, R22 ; add all numbers that are not divisible by 3
    ST z+, R20 ; store all the numers that are not divisible by 3 in the z array
    JMP compare ; finally jump to compare to see if we are in the last number of the 99 numbers
compare:
    CPI R20, 109 ; compares with 110 to see if it is the final value of the last 99th number
    BRNE num99 ; if not start process all over again
    JMP loop ; otherwise exit the process and jump into the loop and do nothing

```

3. DEVELOPED CODE OF TASK 3

The third task asked to:

Use X/Y/Z register addressing to simultaneously add numbers from memory location 0x0400 and 0x0600 and store the sums at R16:R17 and R18:R19 respectively. Pay attention to the carry overflow.

To perform this operation, the addition would happen after the check if the number was or wasn't divisible by 3. I knew the generated number in R20 would never be more than 255 so all I had to do was keep adding R20 with the low byte of my sum, R17. Then if I had a carry I would add that to R16, my high byte. The same logic follows for the addition of R19 and R18.

```

remzero:
    ADD R17, R20 ; add all numbers that are divisible by 3
    ADC R16, R22 ; add all numbers that are divisible by 3
    ST y+, R20 ; store all the numers that are divisible by 3 in the y array
    JMP compare ; finally jump to compare to see if we are in the last number of the 99 numbers
remanum:
    ADD R19, R20 ; add all numbers that are not divisible by 3
    ADC R18, R22 ; add all numbers that are not divisible by 3
    ST z+, R20 ; store all the numers that are not divisible by 3 in the z array
    JMP compare ; finally jump to compare to see if we are in the last number of the 99 numbers

```

4. CCOMPLETE CODE

```

.org 0

LDI x1, low(0x0200) ; low byte in x memory
LDI xh, high(0x0200) ; high byte in x memory

LDI y1, low(0x0400) ; low byte in y memory
LDI yh, high(0x0400) ; high byte in y memory

LDI z1, low(0x0600) ; low byte in z memory
LDI zh, high(0x0600) ; high byte in z memory

LDI R20, 10 ; initialize with 10, but after incremented start register x with value 11
LDI R22, 0 ; always stays zero

LDI R16, 0 ; high byte of sum of y
LDI R17, 0 ; low byte of sum of y

LDI R18, 0 ; high byte of sum of z
LDI R19, 0 ; low byte of sum of z

num99:
    INC R20 ;increment value in register everytime loop begins
    ST x+,R20 ; increments place in x after storing.
    MOV R21, R20 ; copy the value of R20 without making changes to R20 so that we can manipulate value
    JMP div ; now we want to check if the number is divisible by 3
div:
    SUBI R21, 3 ; subtract the value by 3
    CPI R21, 3 ; Compare the subtracted value by 3
    BRSH div ; keep subtracting if = 3 or greater

    CPI R21, 3 ; compare if value is equal to 3
    BRLO remainders ; if it is lower go to remainders label
remainders:
    CPI R21, 0 ; compare if value if equal to zero
    BREQ remzero ; if so branch to remzero, because the number is visible by 3
    JMP remanum ; other wise we do have a remainder and go to remanum

remzero:
    ADD R17, R20 ; add all numbers that are divisible by 3
    ADC R16, R22 ; add all numbers that are divisible by 3
    ST y+, R20 ; store all the numers that are divisible by 3 in the y array
    JMP compare ; finally jump to compare to see if we are in the last number of the 99 numbers
remanum:
    ADD R19, R20 ; add all numbers that are not divisible by 3
    ADC R18, R22 ; add all numbers that are not divisible by 3
    ST z+, R20 ; store all the numers that are not divisible by 3 in the z array
    JMP compare ; finally jump to compare to see if we are in the last number of the 99 numbers
compare:
    CPI R20, 109 ; compares with 110 to see if it is the final value of the last 99th number
    BRNE num99 ; if not start process all over again
    JMP loop ; otherwise exit the process and jump into the loop and do nothing
loop:
    RJMP loop

```

5. SCREENSHOTS OF EACH TASK OUTPUT

99 numbers stored in x:

Memory:	data IRAM	Address:	0x0200,data
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		
data 0x0219	24 25 26 27 28 29 2a 2b 2c 2d 2e 2f 30 31 32 33 34 35 36 37 38 39 3a 3b 3c		
data 0x0232	3d 3e 3f 40 41 42 43 44 45 46 47 48 49 4a 4b 4c 4d 4e 4f 50 51 52 53 54 55		
data 0x024B	56 57 58 59 5a 5b 5c 5d 5e 5f 60 61 62 63 64 65 66 67 68 69 6a 6b 6c 6d 00		

Numbers divisible by 3 stored in y:

Memory:	data IRAM	Address:	0x0400,data
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		
data 0x0419	57 5a 5d 60 63 66 69 6c 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		

Numbers not divisible by 3 stored in z:

Memory:	data IRAM	Address:	0x0600,data
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		
data 0x0619	31 32 34 35 37 38 3a 3b 3d 3e 40 41 43 44 46 47 49 4a 4c 4d 4f 50 52 53 55		
data 0x0632	56 58 59 5b 5c 5e 5f 61 62 64 65 67 68 6a 6b 6d 00 00 00 00 00 00 00 00		

Values of sum stored in registers:

R16 0x07
R17 0xBC
R18 0x0F
R19 0x78

Sum of numbers divisible by 3 = 07BC = 1980 in decimal

Sum of numbers not divisible by 3 = 0F78 = 3960 in decimal

Time code takes to run

Processor	
Name	Value
Program Counter	0x00000026
Stack Pointer	0x08FF
X Register	0x0263
Y Register	0x0421
Z Register	0x0642
Status Register	I T H S V N Z C
Cycle Counter	10108
Frequency	16.000 MHz
Stop Watch	631.75 µs

6. VERIFICATION IN PYTHON

```

x = []
y = []
z = []
sum_div_3 = 0
sum_not_div_3 = 0

for n in range(11,110):
    x.append(n)
    if n % 3 == 0:
        y.append(n)
        sum_div_3 += n
    else:
        z.append(n)
        sum_not_div_3 += n

print('Numbers stored in X:')
print(x)

print('\nNumbers stored in Y:')
print(y)
print('Sum of numbers divisible by 3: {}'.format(sum_div_3))

print('\nNumbers stored in Z:')
print(z)
print('Sum of numbers divisible by 3: {}'.format(sum_not_div_3))

Numbers stored in X:
[11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30,
 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50,
 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70,
 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90,
 91, 92, 93, 94, 95, 96, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106, 107, 108
, 109]

Numbers stored in Y:
[12, 15, 18, 21, 24, 27, 30, 33, 36, 39, 42, 45, 48, 51, 54, 57, 60, 63, 66, 69,
 72, 75, 78, 81, 84, 87, 90, 93, 96, 99, 102, 105, 108]
Sum of numbers divisible by 3: 1980

Numbers stored in Z:
[11, 13, 14, 16, 17, 19, 20, 22, 23, 25, 26, 28, 29, 31, 32, 34, 35, 37, 38, 40,
 41, 43, 44, 46, 47, 49, 50, 52, 53, 55, 56, 58, 59, 61, 62, 64, 65, 67, 68, 70,
 71, 73, 74, 76, 77, 79, 80, 82, 83, 85, 86, 88, 89, 91, 92, 94, 95, 97, 98, 100
, 101, 103, 104, 106, 107, 109]
Sum of numbers divisible by 3: 3960

```

7. GITHUB LINK OF THIS DA

Student Academic Misconduct Policy

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

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

NAME OF THE STUDENT