hroac2HW3

by Hannah Roach

Submission date: 07-Oct-2018 12:22AM (UTC-0500)
Submission ID: 1015191305
File name: hroac2HW3.pdf (1.14M)
Word count: 1018
Character count: 4710

Hannah Roach
CSC-376 Computer Organization

Problem 1a: (3points) What is the output of the following PEP 8 program.

-202
99
cat

Problem 1b: (5points) Explain how each of the 5 outputs are produced from the 4 inputs:

```
      Object
Addr  code   Symbol   Mnemon  Operand     Comment
0000  040009          BR      0x0009      ;Branch around data
0003  FF36            .WORD   0xFF36      ;First Input
0005  00              .BYTE   0x00        ;Second Input
0006  63              .BYTE   'c'         ;Third Input
0007  6174            .WORD   24948       ;Fourth Input
            ;
0009  390003          DECO    0x0003,d    ;First Output
000C  50000A          CHARO   '\n',i
000F  390005          DECO    0x0005,d    ;Second Output
0012  50000A          CHARO   '\n',i
0015  510006          CHARO   0x0006,d    ;Third Output
0018  510007          CHARO   0x0007,d    ;Fourth Output
001B  510008          CHARO   0x0008,d    ;Fifth Output
001E  00              STOP
001F                  .END
```

First Output: **-202**
FF36 (hex)
1111111100110110 (binary)
0000000011001001 (binary negated)
0000000011001010 (two's complement)

Second output: **99**
This is the decimal output of the second and third input.
c = 0110 0011 (binary) = 63 (hex) = 99 (decimal)

Third output: **c**
Character output directly addresses memory location 0006.

Fourth Output: **a**
Character output directly addresses memory location 0007.
24948 (decimal) = 6174 (hex)
Mem[0007] = 61 (hex) = 1100001 (binary) = a (text)

Fifth Output: **t**
Mem[0008] = 74 (hex) = 1110100 (binary) = t (text)

Problem 2: Run the following PEP 8 program, **Problem 2a:** (2 points) Cut & paste the Assembler Listing including the symbol table at the bottom.

Assembler Listing

```
-----------------------------------------------------
      Object
Addr  code    Symbol   Mnemon  Operand     Comment
-----------------------------------------------------
0000  040007            BR      three
0003  0010    one:      .WORD   16
0005  0016    two:      .WORD   0x0016      ;
0007  390003  three:    DECO    one,d
000A  50000A            CHARO   '\n',i
000D  390005            DECO    two,d
0010  50000A            CHARO   '\n',i
0013  390007            DECO    three,d
0016  00                STOP
0017                    .END                  ; Comment
-----------------------------------------------------

Symbol table
--------------------------------------
Symbol    Value        Symbol    Value
--------------------------------------
one       0003         three     0007
two       0005
--------------------------------------
```

Problem 2b: (3 points) Explain the values of the symbols one, two, and three in the symbol table

The symbols eliminate the problem of having to manually determine addresses. The values of the symbols are the addresses of the first byte of the object in memory.

Symbol	Value
one	0003
two	0005
three	0007

Problem 2c: (3 points) Explain the values of the output of DECO one, DECO two & DECO three

DECO one: **16**
DECO 0x0003,d
0003 references .WORD 16.
There are 16 bits/word. So, 16 bits are allocated to the object code 0010.
10 (hex) = 10000 (binary) = 16 (decimal)

DECO two: **22**
DECO 0x0005,d
0x0016 (hex) = 1011 (binary) = 22 (decimal)

DECO three: **14592**
DECO one,d
DECO 0x0003,d
3900 (hex) = 11100100000000 (binary) = 14592 (decimal)

Problem 3: (7 points) Write an assembly language program that prints your first name on the screen. Use immediate addressing with a hexadecimal constant to designate the operand of CHARO for each letter of your name. Comment each line except STOP & .END. Cut & paste the Assembler Listing into your document and paste a screen shot of the Output area of the PEP/8.

Assembler Listing

Assembler Listing - untitled.pepl

```
                                                                    
      Object                                                        
Addr  code   Symbol   Mnemon  Operand     Comment                  
----------------------------------------------------------------------
0000  500048         CHARO   0x0048,i    ; Character output H
0003  500061         CHARO   0x0061,i    ; Character output a
0006  50006E         CHARO   0x006E,i    ; Character output n
0009  50006E         CHARO   0x006E,i    ; Character output n
000C  500061         CHARO   0x0061,i    ; Character output a
000F  500068         CHARO   0x0068,i    ; Character output h
0012  00             STOP
0013                 .END
----------------------------------------------------------------------
```

Output Area

Output

```
Hannah
```

Problem 4: (7 points) Write an assembly language program that prints your full name on the screen. Use .ASCII pseudo-op to store the characters at the top of your program. Use BR to branch around the characters and use STRO to output your name. Comment each line except STOP & .END. Cut & paste the Assembler Listing into your document and paste a screen shot of the Output area of the PEP/8.

Assembler Listing

Assembler Listing - untitled.pepl

```
                                                                              
      Object                                                                  
Addr  code   Symbol   Mnemon  Operand     Comment                             
                                                                              

0000  040010          BR      main                                            
0003  48616E name:    .ASCII  "Hannah Roach\x00";Store Hannah Roach in a variable
      6E6168                                                                  
      20526F                                                                  
      616368                                                                  
      00                                                                      
0010  410003 main:    STRO    name,d      ; reference name directly           
0013  00              STOP                                                    
0014                  .END                                                    
                                                                              

Symbol table
------------------------------------
Symbol    Value        Symbol    Value
------------------------------------
main      0010         name      0003
------------------------------------
```

Output Area

Output

```
Hannah Roach
```

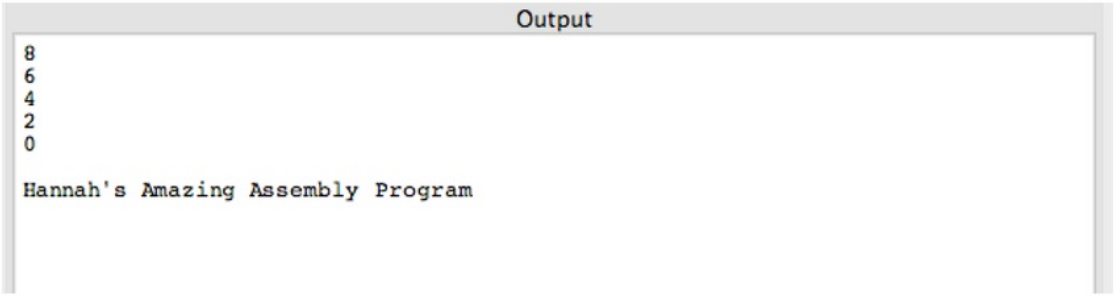
Problem 5a: (5 points) Write an assembly language program (no loops!) that starts at 8 and counts down by 2 to 0. The C++ program is shown below. Comment each line except STOP & .END. Add something to the output that makes this program uniquely yours. Cut & paste the Assembler Listing into your document and paste a screen shot of the Output area of the PEP/8.

Assembler Listing

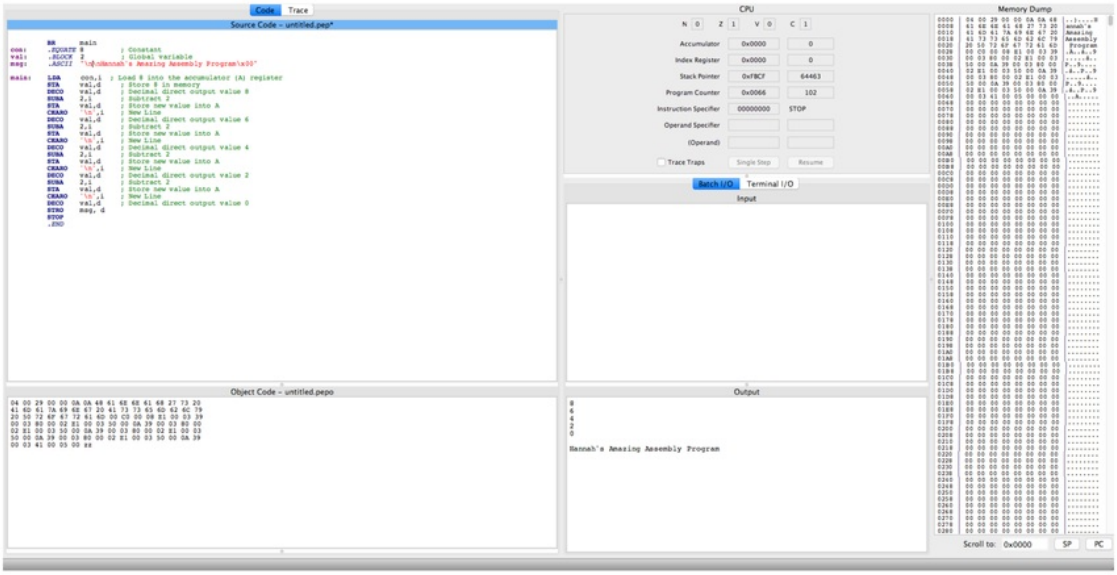
Assembler Listing - untitled.pepl

```
------------------------------------------------------------------------------
     Object
Addr  code   Symbol   Mnemon  Operand     Comment
------------------------------------------------------------------------------

0000  040029           BR      main
            con:       .EQUATE 8           ; Constant
0003  0000    val:     .BLOCK  2           ; Global variable
0005  0A0A48  msg:     .ASCII  "\n\nHannah's Amazing Assembly Program\x00"
      616E6E
      616827
      732041
      6D617A
      696E67
      204173
      73656D
      626C79
      205072
      6F6772
      616D00

0029  C00008  main:    LDA     con,i       ; Load 8 into the accumulator (A) register
002C  E10003           STA     val,d       ; Store 8 in memory
002F  390003           DECO    val,d       ; Decimal direct output value 8
0032  800002           SUBA    2,i         ; Subtract 2
0035  E10003           STA     val,d       ; Store new value into A
0038  50000A           CHARO   '\n',i      ; New Line
003B  390003           DECO    val,d       ; Decimal direct output value 6
003E  800002           SUBA    2,i         ; Subtract 2
0041  E10003           STA     val,d       ; Store new value into A
0044  50000A           CHARO   '\n',i      ; New Line
0047  390003           DECO    val,d       ; Decimal direct output value 4
004A  800002           SUBA    2,i         ; Subtract 2
004D  E10003           STA     val,d       ; Store new value into A
0050  50000A           CHARO   '\n',i      ; New Line
0053  390003           DECO    val,d       ; Decimal direct output value 2
0056  800002           SUBA    2,i         ; Subtract 2
0059  E10003           STA     val,d       ; Store new value into A
005C  50000A           CHARO   '\n',i      ; New Line
005F  390003           DECO    val,d       ; Decimal direct output value 0
0062  410005           STRO    msg,d
0065  00               STOP
0066                   .END
------------------------------------------------------------------------------


Symbol table
-------------------------------------
Symbol    Value        Symbol    Value
-------------------------------------
con       0008         main      0029
msg       0005         val       0003
-------------------------------------
```

Output Area



Problem 5b: (3 points) Cut and paste a screen shot of the Output of the PEP/8



Problem 5c: (2 points) Explain the status bit(s) NZVC at the point that STOP is loaded.

N = 0

This is the sign flag. This would be set to one if the final value was a negative value. Since zero is neither positive or negative, this is not set to one.

Z = 1

This indicates if the result of an operation was zero. Since the last value calculated was zero, this value is set to one.

V = 0

This indicates if the result of an operation was too large to fit the register's width. Since that was not the case, this remains zero.

C = 1

This is the carry flag. It shows if something has been added or subtracted at the bit level.

Problem 6a: (4 points) Write an assembly language program that corresponds to the following C++ program. Comment each line except STOP & .END. Add something to the output that makes this program uniquely yours. Cut & paste the Source Code into your document. (Hint: PEP/8 does not have a divide instruction; however we have discussed an instruction that divides by 2. Please use that instruction.)

Source Code

```
BR main
aNum: .BLOCK 2 ; global variable
bNum: .BLOCK 2 ; global variable
cNum: .BLOCK 2 ; global variable
dNum: .BLOCK 2 ; global variable
sum: .BLOCK 2 ; global variable
avg: .BLOCK 2 ; global variable
msgSum: .ASCII "sum = \x00" ; message for sum output
msgAvg: .ASCII "avgerage = \x00" ; message for average output
msg: .ASCII "This is Hannah's super cool program"
main: DECI aNum,d ; input first number
DECI bNum,d ; input second number
DECI cNum,d ; input third number
DECI dNum,d ; input fourth number
LDA aNum,d ; load first number
ADDA bNum,d ; add second number
ADDA cNum,d ; add third number
ADDA dNum,d ; add fourth number
STA sum,d ; store value to accessor register
STRO msgSum,d ; output sum message
DECO sum,d ; output sum value
CHARO '\n',i ; new line
ASRA ; divide by 2
ASRA ; divide by 2
STA avg,d ; store average to accessor register
STRO msgAvg,d ; output average message
DECO avg,d ; output average
STOP
.END
```

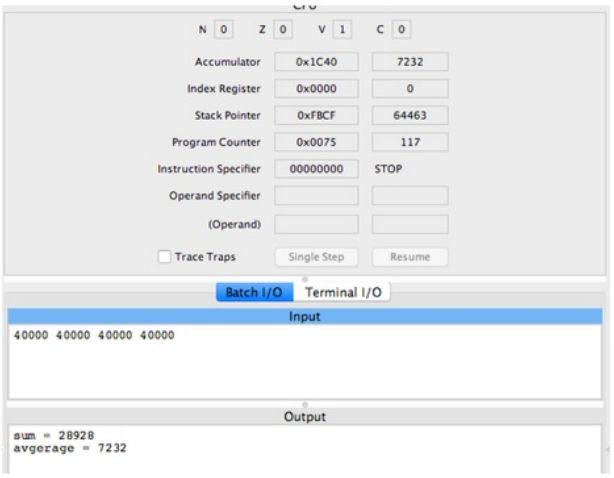
Assembler Listing

```
        BR      main
aNum:   .BLOCK  2           ; global variable
bNum:   .BLOCK  2           ; global variable
cNum:   .BLOCK  2           ; global variable
dNum:   .BLOCK  2           ; global variable
sum:    .BLOCK  2           ; global variable
avg:    .BLOCK  2           ; global variable
msgSum:     .ASCII  "sum = \x00"  ; message for sum output
msgAvg:     .ASCII  "avgerage = \x00" ; message for average output
msg:        .ASCII  "This is Hannah's super cool program"

main:   DECI    aNum,d      ; input first number
        DECI    bNum,d      ; input second number
        DECI    cNum,d      ; input third number
        DECI    dNum,d      ; input fourth number
        LDA     aNum,d      ; load first number
        ADDA    bNum,d      ; add second number
        ADDA    cNum,d      ; add third number
        ADDA    dNum,d      ; add fourth number
        STA     sum,d       ; store value to accessor register
        STRO    msgSum,d    ; output sum message
        DECO    sum,d       ; output sum value
        CHARO   '\n',i      ; new line
        ASRA                ; divide by 2
        ASRA                ; divide by 2
        STA     avg,d       ; store average to accessor register
        STRO    msgAvg,d    ; output average message
        DECO    avg,d       ; output average
        STOP
        .END
```

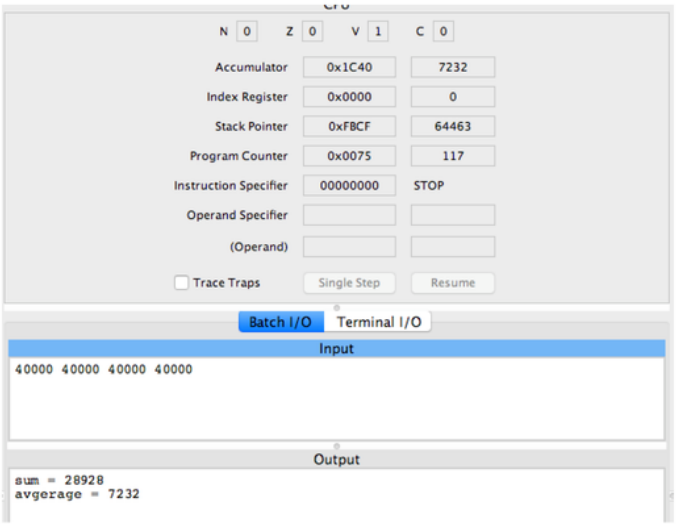
Problem 6b: (4points) Run it twice – once with values that yield a output that is within the range of the PEP/8 and once with values that yield a output that is outside the range of the PEP/8. Explain the limits. Paste screen shots of the Output area of the PEP/8 for both runs.

The range appears to be between +/- 100,000. Any bit greater or less than this input value will generate an incorrect answer. This is because there are 17 bits in 100,000 and the input value must be less than or equal to 16 bits.

In Range



Out of Range



Problem 6c: (2 points) Explain the status bit(s) NZVC at the point that STOP is loaded for the invalid run.

The status bits for an input of 4 4 4 4 was N = 0, Z=0, V=0 and C=0. This is because the final value is not negative, the result was not zero, there was no overload, the there was no carry.

hroac2HW3

GRADEMARK REPORT

FINAL GRADE

50 /50

GENERAL COMMENTS

Instructor

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5

PAGE 6

PAGE 7

PAGE 8

PAGE 9

RUBRIC: SPR2015-HW3-POINTS 50 / 50

1A 3 / 3

FULL CREDIT
(3)

MINUS 1
(2)

MINUS 2
(1)

MINUS 3
(0)

MINUS 4
(0)

MINUS 5
(0)

MINUS 6
(0)

NO CREDIT
(0)

1B 5 / 5

FULL CREDIT
(5)

MINUS 1
(4)

MINUS 2
(3)

MINUS 3
(2)

MINUS 4
(1)

MINUS 5
(0)

MINUS 6
(0)

NO CREDIT
(0)

2A 2 / 2

FULL CREDIT
(2)

MINUS 1
(1)

MINUS 2
(0)

MINUS 3
(0)

MINUS 4
(0)

MINUS 5
(0)

MINUS 6
(0)

NO CREDIT
(0)

2B 3 / 3

FULL CREDIT
(3)

MINUS 1
(2)

MINUS 2
(1)

MINUS 3
(0)

MINUS 4
(0)

MINUS 5
(0)

MINUS 6
(0)

NO CREDIT
(0)

2C 3 / 3

FULL CREDIT
(3)

MINUS 1
(2)

MINUS 2
(1)

MINUS 3
(0)

MINUS 4
(0)

MINUS 5
(0)

MINUS 6
(0)

NO CREDIT
(0)

3 7 / 7

FULL CREDIT
(7)

MINUS 1
(6)

MINUS 2
(5)

MINUS 3
(4)

MINUS 4
(3)

MINUS 5
(2)

MINUS 6
(1)

NO CREDIT
(0)

4 7 / 7

FULL CREDIT
(7)

MINUS 1
(6)

MINUS 2
(5)

MINUS 3
(4)

MINUS 4
(3)

MINUS 5
(2)

MINUS 6
(1)

NO CREDIT
(0)

5A 5 / 5

FULL CREDIT
(5)

MINUS 1
(4)

MINUS 2
(3)

MINUS 3
(2)

MINUS 4
(1)

MINUS 5
(0)

MINUS 6
(0)

NO CREDIT
(0)

5B 3 / 3

FULL CREDIT

(3)

MINUS 1
(2)

MINUS 2
(1)

MINUS 3
(0)

MINUS 4
(0)

MINUS 5
(0)

MINUS 6
(0)

NO CREDIT
(0)

5C 2 / 2

FULL CREDIT
(2)

MINUS 1
(1)

MINUS 2
(0)

MINUS 3
(0)

MINUS 4
(0)

MINUS 5
(0)

MINUS 6
(0)

NO CREDIT
(0)

6A 4 / 4

FULL CREDIT
(4)

MINUS 1
(3)

MINUS 2
(2)

MINUS 3
(1)

MINUS 4
(0)

MINUS 5
(0)

MINUS 6
(0)

NO CREDIT
(0)

6B 4 / 4

FULL CREDIT
(4)

MINUS 1
(3)

MINUS 2
(2)

MINUS 3
(1)

MINUS 4
(0)

MINUS 5
(0)

MINUS 6
(0)

NO CREDIT
(0)

6C 2 / 2

FULL CREDIT
(2)

MINUS 1

(1)

MINUS 2
(0)

MINUS 3
(0)

MINUS 4
(0)

MINUS 5
(0)

MINUS 6
(0)

NO CREDIT
(0)