29. Long Problem (40 points)

Consider the following C code:

```
int compute(int x, int y);
int squared(int r);
constant int globalD=6;

void main() {
   int a, b, c;
   a = 10;
   b = 15;
   c = compute(a,b);
}

int compute(int x, int y) {
   int z;
   z = squared(x);
   z = z + squared(y) + globalD;
   return(z);
}

int squared(int r) {
   return (r*r);
}
```

It will roughly produce the following assembly code. Note: Some extraneous lines have been removed.

```
;## # FUNCTION main
;## # FRAME AUTO (c) size 2, offset -6
;## # FRAME AUTO (b) size 2, offset -4
;## # FRAME AUTO (a) size 2, offset -2

_main:
   enter    #06H
   mov.w    #000aH,-2[FB] ;  a
   mov.w    #000fH,-4[FB] ;  b
   mov.w    -4[FB],R2   ;  b
   mov.w    -2[FB],R1   ;  a
   jsr      $compute
   mov.w    R0,-6[FB]   ;  c
   exitd

;## # FUNCTION compute
;## # FRAME AUTO  (z) size   2, offset -6
;## # FRAME AUTO  (y) size   2, offset -2
;## # FRAME AUTO  (x) size   2, offset -4
;## # REGISTER ARG (x) size 2, REGISTER R1
;## # REGISTER ARG (y) size 2, REGISTER R2
   .glb     $compute
$compute:
   enter    #06H
   mov.w    R1,-4[FB]   ;  x  x
   mov.w    R2,-2[FB]   ;  y  y
   mov.w    -4[FB],R1   ;  x
   jsr      $squared
   mov.w    R0,-6[FB]   ;  z
   mov.w    -2[FB],R1   ;  y
   jsr      $squared
   add.w    -6[FB],R0   ;  z
   add.w    _globalD,R0
   mov.w    R0,-6[FB]   ;  z
   exitd

;## # FUNCTION squared
;## # FRAME AUTO (r) size 2, offset -2
;## # REGISTER ARG (r) size 2, REGISTER R1
;## # ARG Size(0) Auto Size(2) Context Size(5)
   .glb     $squared
$squared:
   enter    #02H
   mov.w    R1,-2[FB]   ;  r  r
   mov.w    -2[FB],R0   ;  r
   mul.w    -2[FB],R0   ;  r ####THIS POINT####
   exitd

   .SECTION data_NE,DATA
   .glb     _globalD
_globalD:
   .blkb    2
   .SECTION data_NEI,ROMDATA
   .word    0006H
   .END
```

Assume you are starting execution of the program at main. Assume the drawing to the right is the stack which is 8-bits wide.

a) Show the contents of the stack after the program has run and just completed executing the line of code at **`####THIS POINT####`** for the first time. (35 points)
- Where you can show values on the stack, put them in the memory space.
- Where specific values are not known, but something has been put on the stack, describe the contents in enough detail to convey your knowledge.
- Where specific information has not been put on the stack, write "XX". This means that the space beyond the top of the stack should have XX, but there are other areas on the stack which are empty. As a hint, two spots of memory have already been answered.
- Assume both FB and SP point to the bottom of the stack before execution starts.

b) Show the location that SP and FB point to on the stack at **`####THIS POINT####`** (5 points)

00763h	XX
00764h	
00765h	
00766h	
00767h	
00768h	
00769h	
0076Ah	
0076Bh	
0076Ch	
0076Dh	
0076Eh	
0076Fh	
00770h	
00771h	
00772h	
00773h	
00774h	
00775h	
00776h	
00777h	
00778h	
00779h	
0077Ah	
0077Bh	
0077Ch	
0077Dh	
0077Eh	
0077Fh	
00780h	XX