NCSU - ECE 306- Exam 1 – February 13, 2003

Name: _____ User ID _____

Question	1-20	21-26	27-28	29	Total
Score	/60	/30	/20	/40	/150

You are permitted 75 minutes to take this test, no more. This is a closed book/closed notes test. You are allowed the following item for the test: pencils and erasers. You are not permitted to have any of the following on your desk during the test: calculators, books, notes, homework, labs, or other electronic assistance. Failure to abide by this policy will result in a zero for the test and a visit to the NCSU judicial board. Put your answers on this paper. Use only this paper.

Please read and sign this statement: I have not received from anyone nor assisted others while taking this test. I have also notified the test proctor of any of these violations noted above.

Signature: _____

Multiple Choice - Questions 1-20: Each of these multiple choice questions is worth 3 points for a correct answer, 0 points for an incorrect answer. Circle the correct answer. Multiple circles will be marked as incorrect.

1. Which of the following are typically not embedded systems?
   a) Cell Phones
   b) Computers
   c) PDAs
   d) TV remote controls
   e) Digital clocks

2. What is one reason given by Jack Ganssle that EE are ill equipped in today's market?
   a) Technology is changing so quickly that they can't keep up.
   b) They are not getting enough programming experience in college.
   c) They are too set in their ways.
   d) They are not getting enough hands on training in college.
   e) None of the above.

3. What technology was used for the microprocessor in the cell phone for Lab 1?
   a) DIP
   b) BGA
   c) DFP
   d) QFP
   e) TDMA

4. What is a benefit of an ASIC over a microprocessor?
   a) Production cost
   b) Flexibility of reprogramming
   c) Can be used for a variety of tasks
   d) Acronyms are always better
   e) Faster because direct gate/IO relation

5. What is the order of compilation of C code?
   a) Preprocessor, Compiler, Linker
   b) Preprocessor, Linker, Compiler
   c) Linker, Compiler, Postprocessor
   d) Compiler, Linker, Postprocessor
   e) Compile, Link, Postcompile

6. How much RAM does our MCU unit have for the user to use?
   a) 256 Bytes
   b) 512 Bytes
   c) 256 Kbytes
   d) 22.5 Kbytes
   e) 896 Bytes

7. What file extension does the executable for the Mitsubishi board use?
   a) .bat
   b) .exe
   c) .x30
   d) .obj
   e) .a30

8. What is the name of the circuitry that the processor has to "kick" every so often to prevent the processor from being restarted?
   a) Anti-freeze
   b) Housecat
   c) Watchdog
   d) Canary
   e) Doesn't have a name

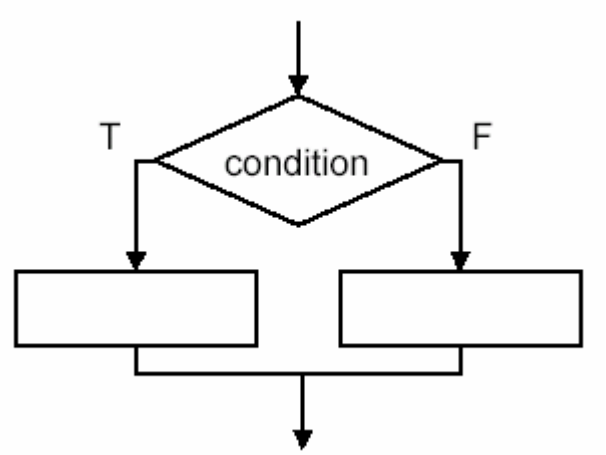
9. Which of the following statements is not true about a code repository?
   a) A code repository stores a complete history log of each file in the source tree is maintained, which is useful for finding who did what, when.
   b) A code repository system makes it easier for many people to review and make changes to existing code, or update it with more efficient code.
   c) A code repository allows certain files to be "checked out," preventing others access to the files until they are "checked in."
   d) A code repository system is not safe because the entire project exists on a central server, making it more susceptible to data loss. Individual project files should be stored on each employee's workstation.
   e) The code repository system decreases the risk of conflicting file versions.

10. What indicates the address of the instruction to be executed?
   a) User Stack Pointer
   b) Frame Base Register
   c) Address Register
   d) Flag Register
   e) Program Counter

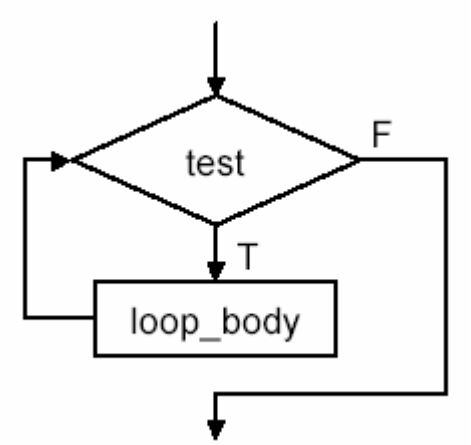
11. In which direction does the stack grow?
   a) Toward larger addresses
   b) Starts in middle and filters outward
   c) Toward smaller addresses
   d) Randomly
   e) none of the above

12. What base 10 value does the code "`mov.w #0011H,R0`" put in register R0?
   a) 17
   b) 11
   c) 3
   d) 1
   e) mov.w is not an instruction

13. Which assembly code command shown below subtracts the value of global variable globalD from the value stored in R1?
   a) `sub.w  R1,_globalD`
   b) `sub.w  R1,globalD`
   c) `sub.w  globalD,R1`
   d) `sub.w  _globalD,R1`
   e) none of the above



14. The diagram above shows a:
   a) Switch statement
   b) While loop
   c) For loop
   d) Subroutine
   e) None of the above



15. The diagram above shows a(n):
- a) While loop
- b) For loop
- c) Switch statement
- d) If-else statement
- e) None of the above

16. When partitioning memory into near and far areas, the far section has what type of pointer?
- a) 16 bit pointer
- b) 16 byte pointer
- c) 20 bit pointer
- d) 20 byte pointer
- e) 8 bit pointer

17. Which assembly language command does the following?
- Push FB onto the stack
- Copy SP to FB
- Subtract n from SP – automatically allocate n bytes of space on the stack

a) Enter #n
b) Exitd
c) FRAME AUTO
d) .align
e) .glb

Consider the following code for Questions 18 and 19:

```
Line 1:    int j = 1;
Line 2:    char c = 0;
Line 3:    int i, k;
Line 4:    const char cc = 'a';
Line 5:    void main(void)
Line 6:    {
Line 7:         int l, m;
Line 8:         i = i +k;
Line 9:    }
```

18. In line 3, int i, k are:
- a) Automatic variables that are stored in the stack section of RAM.
- b) Variables without initial values that are stored in the bss section.
- c) Static variables without initial values that are stored in the bss section.
- d) Static variables without initial values that are stored in the data_l section.
- e) None of the above.

19. In line 4, const char cc = 'a' is stored at:
- a) The rom section of the stack.
- b) The stack section of the stack.
- c) The program section of the stack.
- d) The bss section of the stack.
- e) None of the above.

20. When returning from a function, what register is used to pass back an integer return value?
- a) R0
- b) R1
- c) R2
- d) FB
- e) SP

Multiple Choice - Questions 21-26: Each of these multiple choice questions is worth 5 points for a correct answer, 0 points for an incorrect answer. Circle the correct answer. Multiple circles will be marked as incorrect.

21. Consider the following assembly code:

```
mov.w  _inLocal,r0
shr    r0
mov.w  r0,_inLocal
```

What is the main objective of this code?

a) R0 = _inLocal
b) R0 = _inLocal / 2
c) R0 = _inLocal * 2
d) _inLocal = _inLocal / 2
e) _inLocal = _inLocal *2

22. Consider the following C Code:

```
y = 0;
while (y<9) {
   y = y + 1;
}
```

Pick the correct assembly language fragment of code that best represents the C code above, given that y is stored in `-6[FB]`

a)
```
      mov.w #0000H,-6[FB]
L12:
      cmp.w #0009H,-6[FB]
      jge L12
      add.w #0001H,-6[FB]
      jmp L11
L11:
```

b)
```
      mov.w #0000H,-6[FB]
L11:
      cmp.w #0009H,-6[FB]
      jge L12
      add.w #0001H,-6[FB]
      jmp L11
L13:
```

c)
```
      mov.w #0000H,-6[FB]
L11:
      cmp.w #0009H,-6[FB]
      jge L12
      sub.w #0001H,-7[FB]
      jmp L11
L12:
```

d)
```
      mov.w #0000H,-6[FB]
L11:
      cmp.w #0009H,-6[FB]
      jge L12
      add.w #0001H,-6[FB]
      jmp L11
L12:
```

e)
```
      mov.w #0000H,-6[FB]
L11:
      cmp.w #0009H,-7[FB]
      jge L12
      add.w #0001H,-6[FB]
      jmp L11
L12:
```

23. Which of the following assembly code fragments correctly executes the following C code (assume x is at an offset of -6 from the Frame Base):

```
for (i=0; i <10; i++) x += i;
```

a)
```
      mov.w #0000H, -8[FB]
L1:   cmp.w #000aH, -8[FB]
      jge L1
      add.w -8[FB], -6[FB]
      add.w #0001H, -8[FB]
      jmp  L2
L2:
```

b)
```
      mov.w -8[FB], #0000H
L1:   cmp.w -8[FB], #000aH
      jge L2
      add.w -6[FB], -8[FB]
      add.w -8[FB], #0001H
      jmp  L1
L2:
```

c)
```
      mov.w #0000H, -8[FB]
L1:   cmp.w #000aH, -8[FB]
      jge L2
      add.w -8[FB], -6[FB]
      add.w #0001H, -8[FB]
      jmp  L1
L2:
```

d)
```
      mov.w #000AH, -8[FB]
L1:   cmp.w #0000H, -8[FB]
      jpz L2
      add.w -8[FB], -6[FB]
      sub.w #0001H, -8[FB]
      jmp  L1
L2:
```

e) None of the above

24. Consider the following assembly language code:

```
L1:
   cmp.w #0000H,-6[FB]
   jeq L5
   add.w #0001H, -4[FB]
   sub.w #0001H, -8[FB]
   jmp L6
L5:
   sub.w #0001H, -4[FB]
   add.w #0001H,-8[FB]
L6:
```

Indicate which C-code fragment below most closely corresponds to the assembly code shown above:

```
a) switch(x) {
      case 1: y+=3; break;
      case 2: y-=17; break;
      default: y--;  break;
   }
b) while(x<10) x=x+2;
c) if(x) { y++; z--;}
      else {y--; z++;}
d) while(x<0)  X=x+2;
e) if(x) {y--; z++;}
      else {y++; z--;}
```

25. Assume the following assembly language code and that the variable x is stored in –6[FB]

```
     mov.w #0008H, -6[FB]
   L11:
     cmp.w #000aH, -6[FB]
     jge L12
     add #0001H, -6[FB]
     jmp L11
   L12:
```

The above code is an implementation of what type of structure?

a) switch
b) for
c) while
d) if…else
e) none of the above

26. From the code in problem 25, what is the value stored in –6[FB] after execution?

a) 8
b) 9
c) 10
d) 11
e) 12

Short Answer - Questions 27-28: Write your answer in the space provided. Points as indicated.

27. Given the following assembly language code:

```
mov.w #000Ah,R0
mov.w #0001h,R1
mov.w #0004h,R2
add.w R1,R0
mul.w R1,R2
sub.w R1,R2
sub.w R2,R0
mov.w R0,R1
add.w R0,R1
```

a) What is the value of R0 after the code is run? (5 points)

b) What is the value of R1 after the code is run? (5 points)

c) What is the value of R2 after the code is run? (5 points)

28. A cellular phone has a 720 mAhr battery, and has an average phone current of 4mA. How many days will this phone stay powered-on if it consumes the entire capacity of the battery? (5 points)

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