

**UNCC, Department of Electrical and Computer Engineering
ECGR4101/5101/6090, Fall 2004, Homework #4, Due: 9/27/03, at the beginning of class (20 points)**

You will need to refer to the M16C/20/60 Software Manual, the M16C26 Hardware Manual and M16C C Language Programming Manual to complete this assignment. They are available online through the Documentation contained in the SKP16C26 directories link on the course home page.

0. How long did this homework take you?
1. Assume the following C program is executed. Show the contents of the variables after execution completes, assuming their addresses are as given below. Represent data “naturally” – integers should be shown as big-endian values, strings as a sequence of characters, and characters as the actual character (not the ASCII code). **(NOTE: the bolded line below is a correction)**

```
void main() {
  char s1[20];
  char c, d;
  int i, j, k;
  char *pc;
  int *pi1, *pi2;
  i = 1;
  j = 241;
  k = 314;
  strcpy(s1, "one five");
  pc = &s1[5];
  d = *pc;
  pc = strchr(s1, 'e');
  strcpy(pc, "kcm");
  pc++;
  c = *pc;
  i = 581;
  pi1 = &j;
  *pi1 = i;
  pi2 = &i;
  i -= 21;
  k = *pi1 + *pi2;
  pi2--;
}
```

Var	Addr	Intermediate Values						Final Value
s1	600	_____	_____	_____	_____	_____	_____	
c								
d								
i								
j								
k								
pc								
pi1								
pi2								

For the questions 2 to 4, specify the format string needed to print the value as shown.

2. value = 33, output = 0033

3. value = 33, output = 0x0021

4. value = 21.343, output = 21.3

5. Write C code to declare a character variable a and initialize it with the single character constant \$.

6. Show how the following C arrays are laid out in memory. Remember to pay attention to endianness, indicating which byte is located where.

a. int a[5];

Address	Array Element	Which byte?
a		
a+1		
a+2		
a+3		
a+4		
a+5		
a+6		
a+7		
a+8		
a+9		
a+10		
a+11		
a+12		
a+13		
a+14		
a+15		

b. unsigned char b[4][2]

Address	Array Element	Which byte?
b		
b+1		
b+2		
b+3		
b+4		
b+5		
b+6		
b+7		
b+8		
b+9		
b+10		
b+11		
b+12		
b+13		
b+14		
b+15		

7. Write assembly code to make register A0 point to element $b[i][j]$ of the array b in the previous question. Assume that i is in $r0$ and j is in $r1$. Assume array b is on the stack at $-2[FB]$. This is different from the examples in lecture, where the array is a global variable. Assume the array is completely contained within the address range 0 to $0ffffh$, so only 16 bits of address are needed. You are allowed to overwrite registers as needed. The first instruction (but without its operands) has been provided for you; `mova` (move effective address) computes the address of the source operand and copies that address to the destination operand without actually reading the memory at that address.

`mova`

; A0 now points to the element $b[i][j]$