CPRE 288

Exam 1 Review Session Solution

Data Types

Write the amount of memory allocated to each of the following declarations

```
char a;                                          1 bytes
short x;                                         2 bytes
long y;                                          4 bytes
int num = 4;                                     4 bytes
double nums[40];                                 320 (8*40) bytes
int **num;                                       4 bytes
int *num;                                        4 bytes
char str[10];                                    10 bytes
typedef struct compound {
        char *c_ptr;                             4 bytes
        int coord[2];                            8 bytes
        union {
                double *num;                     4 bytes
                char str[4];                     4 bytes
                char *my char;                   4 bytes
        };
} compound_t;

compound_t my_var;                               16 bytes
```

What are valid ranges of values for the following data types?

a. unsigned char 0 to 255
b. signed char -128 to 127
c. int -2147483648 to 2147483647

Base conversions

a. What is binary 0b0110 0011 0000 1000 in hex?
   0x6308
b. What is hex 0x8EC4 in binary?
   0b1000111011000100
c. Given the decimal number 7, write its representation in binary and hex for both 4-bit and 8-bit fields.
   4-bit:
   0x7
   0b0111
   8-bit:
   0x07
   0b00000111

Pointers

What is the value of ptr after the following code runs?
ptr=0x00000010 (2*8=16)

```
double x = 47;
double *ptr = &x; //assume address of x is 0x00000000
ptr += 2;
```

What are the values of each memory address after this code runs?

```
typedef struct coord{
  char x;
  char y;
} coord_t;

coord_t  *coord_ptr;
int *num_ptr;
int **p_ptr = &num_ptr;
char a = 0x07;
coord_t my_coord[2];
int num_array[2]={1,4};

int main(){
  coord_ptr = my_coord;

  num_ptr = num_array;

  my_coord[1].x = 0x33;

  coord_ptr++;

  coord_ptr->y = 0x44;

  num_ptr = num_ptr + 2;

  *num_ptr = 0x5040;

  p_ptr++;

  *p_ptr = 0xFEC0;
```

Address	Variable Name	Value
0xFFFF_FFFF	coord_ptr	~~0xFF~~ 00
0xFFFF_FFFE		~~0xFF~~ 00
0xFFFF_FFFD		~~0xFF~~ FE
0xFFFF_FFFC		~~0xEF~~ ~~F1~~ C0
0xFFFF_FFFB	num_ptr	0xFF
0xFFFF_FFFA		0xFF
0xFFFF_FFF9		0xFF
0xFFFF_FFF8		~~0xE7~~ EF
0xFFFF_FFF7	p_ptr	0xFF
0xFFFF_FFF6		0xFF
0xFFFF_FFF5		0xFF
0xFFFF_FFF4		~~0xF8~~ FC
0xFFFF_FFF3	a	0x07
0xFFFF_FFF2	my_coord[1].y	~~0x44~~ 00
0xFFFF_FFF1	my_coord[1].x	~~0x33~~ 00
0xFFFF_FFF0	my_coord[0].y	0x50
0xFFFF_FFEF	my_coord[0].x	0x40
0xFFFF_FFEE	num_array[1]	0x00
0xFFFF_FFED		0x00
0xFFFF_FFEC		0x00
0xFFFF_FFEB		0x04
0xFFFF_FFEA	num_array[0]	0x00
0xFFFF_FFE9		0x00
0xFFFF_FFE8		0x00
0xFFFF_FFE7		0x01

Bitwise Operations

Write 1 line to accomplish each of the following

uint8_t x;
uint8_t y;

a. Check if any of bits 7, 5, 4, or 0 of x are set to 1
   if(x & 0b10110001) //0xB1
b. Check if all bits 6, 3, 2, or 1 of x are set to 1
   if((x & 0b01001110) == 0b01001110) //0x4E
c. Check if all bits 7, 6, and 1 are cleared to 0
   if((x & 0b11000010) == 0) //0xC2
d. Set bits 5, 4, 3, and 1 of y
   y = y | 0x3A //0b00111010
e. Clear bits 5, 3, 2, and 1 of y
   y = y & 0b11010001 //0xD1

What is the value of x after the following code executes?

signed char x = 0x80;
signed char r = 2;

f. x = (x >> 3) | (0x06 >> r)

   x = 0b11110001

x = 0b10000000
r = 0b00000010
0x06 = 0b00000010

x>>3 = 11110000
0x06>>2 = 00000001

11110000 | 00000001 = 11110001 //0xF1

Writing functions

Write functions to achieve the following

a. Write a function that returns a char representing the number of 1s in the input parameter *int input* using bitwise operators

```
char countSetBits(unsigned int n){

    char count = 0;
    while(n)
    {
        char result = n & 0b1;
        total = total + result;
        n = n >> 1;
    }
    return result;

}
```

b. Write complete code to implement the C function *strcat* with pointers.

```
void strConCat(char* arrayA, char* arrayB){

    while((*arrayA) !='\0'){
        arrayA++;
    }
    while((*arrayB) != '\0'){
        *arrayA = *arrayB;
        arrayB++;
        arrayA++;
    }
    arrayA* = '\0'

}
```

Memory Mapped GPIO

Enable port B, pins 0, 2,3,4, and 7 configured for output and pins 1, 5, and 6 for input. Then check if pin 5 is high. (Preserve bits as needed)

```
//There may be other correct answers
//Enable PORTB clock
SYSCTRL_RCGCGPIO_R  |= 0b00000010;

//Pins 0,2,3,4,7 are outputs, so set them to 1.
GPIO_PORTB_DIR |= 0b10011101;

//Pins 1,5,6 are inputs, so and them to a 0
GPIO_PORTB_DIR &= 0b10011101;

//Enable digital functionality for all 8 pins
GPIO_PORTB_DEN |= 0b11111111;

//Check if pin 5 is high (assume active high switch)
if(GPIO_PORTB_DATA_R & 0b00100000) == 0b00100000){ };
```