EE 109 Homework 2

Name: _____

Due: See class website **Score: _____**

Enter your answers on Blackboard..Assignments..HW..Homework 2

Data Representation

1.) Perform the following number system conversions. Note: It may be easier to convert them to the desired base in a different order than shown here. (2 pts. per conversion)

a. $1100101.1011_2 = ?_8 = ?_{16} = ?_{10}$
b. $1A9.D_{16} = ?_8 = ?_2 = ?_{10}$
c. $617_8 = ?_{16} = ?_2 = ?_{10}$

2.) Convert the powers of 2 shown below to its approximate decimal value using K to represent 10^3, M for 10^6, G for 10^9, and T for 10^{12}. (e.g. $2^{12} \approx 4K$) [2 pts. each]

a. $2^{19} = ?$ i. 9K ii. 512K iii. 512M iv. 256K v. 256M	b. $2^{43} = ?$ i. 8G ii. 8T iii. 16M iv. 16G v. 16T
c. $2^{36} = ?$ i. 64M ii. 64G iii. 8M iv. 8G v. 8T	d. $2^{24} = ?$ i. 4K ii. 4M iii. 8M iv. 16M v. 16G

Bit Fiddling and Digital I/O

3.) **True / False**: To force some bits to 0 while leaving other bits in a register unaffected we should use the bitwise 'AND' operation

4.) Match the following operations to their descriptions.

Set bit 4 of PORTD	PORTD &= ~(0x04);
Set bit 2 of PORTD	PORTD |= 0x10;
Clear bit 4 of PORTD	PORTD &= ~(0x10);
Clear bit 2 of PORTD	PORTD |= 0x04;

5.) Taking an 8-bit value such as PORTD and ANDing it with the value 0x20 as in: **`PORTD & 0x20`** can result in which possible outcomes/values (select all that apply)

a. 0x00
b. 0x20
c. 0xff
d. 0xdf

6.) Match the register with the correct description of its use.

PORTx (Port Output Register)	Grabs (samples) the voltage value currently present on a pin being used as input
DDRx (Data Direction Register)	Determines the value of a pin as output OR enables the internal pull-up resistor on a pin used as input
PINx (Port Input Register)	Determines whether a pin is used as input or output

7.) Poor Billy Bruin connected pushbuttons to port B1 and B0 of his Arduino. He wants to turn on an LED on port D7 if someone pushes both buttons at the same time (assume he uses buttons that output 1 when pressed). He writes the code below but when he runs it, it does NOT work. What is a likely problem?

```
DDRD = (1<<PD7);
while( 1  ) {
   if( (PINB & 0x03) == 0x03 ){
      // turn on the LED
      PORTD |= 0x80;
}  }
```

a. He didn't set the PORTB bits to inputs as in:
```
DDRB = 0x00;
```
b. He didn't enable the pull-up resistors for port B1 and B0 by doing the following initialization:
```
DDRB = 0x00;
PORTB = 0x03;
```
c. He could have just checked whether both PORTB bits 1 and 0 were both 1 by changing the if statement to read:
```
if( PINB == 0x03 ){  …  }
```
d. He needed to grab the values on PINB and isolate bits 1 and 0 by saving it to a variable first then examining the variable in an 'if' statement as in:
```
char status = PINB & 0x03;
if (status == 0x03 ){ … }
```

8.) Which statement below initializes PORTC to use bits 7-5 and 1-0 as outputs?

a. `DDRD = 0x1c;`
b. `PORTC = 0xe3;`
c. `DDRC |= 0xe3;`
d. `DDRC = DDRC | 0x1c;`
e. `PORTC = 0b11100011;`

9.) Little Billy Bruin wants to clear bit 3 of PORTD but isn't sure how. Select the correct method for poor Billy.

a. `PORTD &= 0x08;`
b. `PORTD |= 0b00001000;`
c. `PORTD &= 0x03;`
d. `PORTD &= ~(1 << PD3);`

10.) Assume we have a pushbutton connected to port D5 or our Arduino and it requires the pull-up resistor to be enabled. How would we do that and then check whether the input voltage on pin D5 is actually a 1.

a.
```
DDRD =  (1 << PD5);
if( PIND & (1 << PD5 ){
     // executes if the volatge on pin D5 is high = logic 1
}
```

b.
```
DDRD = 0x00;
PORTD |= (1 << PD5);
if( PIND & (1 << PD5 ){
     // executes if the volatge on pin D5 is high = logic 1
}
```

c.
```
DDRD = 0x00;
PORTD |= (1 << PD5);
if( PIND && (1 << PD5 ){
     // executes if the volatge on pin D5 is high = logic 1
}
```

d.
```
DDRD = 0x00;
PORTD |= (1 << PD5);
char pressed = PIND & ~(1 << PD5);
if( pressed ){
     // executes if the volatge on pin D5 is high = logic 1
}
```

11.) Assume we have two pushbuttons connected to port C3 and C2 of our Arduino. They each require a pull-up resistor to be enabled. How would we do that and then check whether the input voltage on pin C3 is a '1' while at the same time the voltage on C2 is a 0

a.
```
DDRC = 0x00;
PORTC |= 0x0c;
if( (PINC & (1 << PC3   |   0 << PC2)  ){
     // executes if the voltage on pin D5 is high = logic 1
}
```

b.
```
DDRC = 0x00;
PORTC |= 0x0c;
char status = PINC & 0x08;
if( status ){
     // executes if the voltage on pin C3 = 1 and C2 = 0
}
```

c.
```
DDRC = 0xff;
PORTC |= 0x0c;
char status = PINC & 0x0c;
if( status == 0x0c ){
     // executes if the voltage on pin C3 = 1 and C2 = 0
}
```

d.
```
DDRC = 0x00;
PORTC |= (1 << PC3) | (1 << PC2);
char status = PINC & 0x0c;
if( status == 0x08 ){
     // executes if the voltage on pin C3 = 1 and C2 = 0
}
```

12.) Suppose we are using all 8 bits of PORTD as outputs (already setup in the DDR register) but don't know what is in PORTD prior. We want to assign the value 0x3a into PORTD. What would be the correct way to achieve this? **(Select all correct answers)**

a.
```
PORTD |= 0x3a;
PORTD &= 0x3a;
```

b.
```
PORTD |= 0x3a;
```

c.
```
PORTD = 0x3a;
```

d.
```
PORTD = 0;
PORTD |= 0x3a;
```

13.) Suppose we are using all 8 bits of PORTD as outputs (already setup in the DDRD register). The upper 4-bits (i.e. bit 7 through bit 4) are being used to control one set of LEDS while bits 3 through 0 are controlling another. If we want only the LED's associated with bit 3 and 2 to turn on and the LEDs associated with bit 1 and 0 to turn off but not affect the LED's on bits 7-4, how should we do it? **(Check all correct answers)**

a.
```
PORTD &= 0xf0;
PORTD |= 0x0c;
```

b.
```
PORTD |= 0x0c;
```

c.
```
PORTD = 0x0c;
```

d.
```
PORTD &= 0xfc;
```

14.) How could we correctly copy the upper 3 bits of PORTD (whatever they are) to the upper 3 bits of PORTB.

a.
```
PORTB = (PORTD & 0xe0);
```

b.
```
PORTB = PORTD;
```

c.
```
PORTB &= ~(0xe0);
PORTB |= (PORTD & 0xe0);
```

d.
```
PORTB &= ~(PORTD & 0xe0);
```