LAB 4: Writing a small application in C.

This lab is a one-week lab.

This lab consists of the following tasks:

- Write a function which implements the algorithm for the rand48 pseudo-random number generators in C (the methodology is given below.)
- Write setup code to initialize bits RB2, RB3, ..., RB9 as outputs.
- Write code to initialize the pseudo-random number generator (the methodology is given below.)
- Debug and simulate your code.
- Wire up the circuit as shown.
- Demonstrate that you have done this correctly.

The behavior of your circuit should be that it initially starts with all LED's off. Then, for each press of the switch, it should display a pattern on the LED's indicating, in binary (with off representing 1, and on representing 0) the number it generated. You'll have to debounce your switch properly to make this happen.

Before you come to lab you should build the project, and, using the MPLABX simulator, you should have spent some time simulating the operation of the program so that you are comfortable that it should work.

The lab TA has the sequence of pseudo-random numbers your circuit should display and will immediately be able to tell if you have succeeded or failed.

The test circuit for this lab consists of hooking up the power and ground, the two capacitors, plus a resistor to tie the reset pin high (inactive), as well as 8 LED's and current-limiting resistors and the switch. The schematic shows an additional connector (SV1), but that is optional. This circuit, will be used for the fourth lab as well.

1. Turn in a printed copy of your code.
2. Use the simulator and its stopwatch to measure the time required to compute the random number.
3. Also measure the time required for the 48-bit multiply.

Remember, **Always put decoupling capacitors in your circuit, even if not shown!**

rand48 functions work by generating a sequence of 48-bit integers, X_i, according to the linear congruential formula:

$X_{n+1} = (a \times X_n + c)\ mod\ m$, where $n \geq 0$

The parameter $m = 2^{48}$, hence 48-bit integer arithmetic is performed. a and c are given by:

```
a = 0x5DEECE66D;
c = 11;
```

X_0 is initialized using setseed48, given here as a C function:

```
void setSeed(unsigned long int seed)
{
    seedx = seed;
    seedx = seedx << 16;
    seedx = seedx + 0x330E;
}
```

By default, setSeed is called at initialization with seed = 1.