1. **Port configuring - Know how to use and set registers related to parallel ports, such as PxOUT, PxDIR, PxIN and so on. Understand that ports and their associated registers are 8 bits, and the way you assign values.**
   1. The PxDIR register **directs** pins towards either input or output. It can be considered like a declaration statement, where we just declare a variable but don’t define it. Similarly, we just inform whether a pin will be used for input or output, but we **do not specify the output value (high or low)** at this line itself. We specify the output value using another register PxOUT.
      1. PxDIR, where x is any integer between 1 and 7, is a bank of bits (8 bits in total per every bank, for pin 0 - pin 8).
      2. By default the pins input is set to 0.
      3. If one of those bits is set to 1, that corresponding pin is an output pin. If PxDIR |= 0x01 // or BIT1, which means pin 1 directed towards OUTPUT
      4. PxDIR &= ~BIT 1 // negates, which means the pin is directed towards INPUT
      5. PxIN = it is how you read in values on a specific input pin. It is also addressable by a pin bank, but here you use an AND operation with your BITy (where y is the specific pin in the bank) macro in the MSP430.h header file.
         1. *if ((P2IN & BIT4) == BIT4);* check P2.4 for if it’s pressed, if it is, it will enter the `if` statement.
      6. PxOUT is the same thing; how you set the output high or low, but for this, you will use an inverse AND statement with the BIT statement to turn a pin high.
         1. `P1OUT &= ~BIT4;`
            1. Turn on P1.4.
2. **Software delays** 
   1. **You should understand how to use the clock rate to get the time for clock cycles**
   2. **Any calculations used in the lab, in particular how the for loop delays are working**

* MSP430 operates at 1 MHz. Therefore, time for one second is 1,000,000 clock cycles.
  + A for loop delay takes 10 clock cycles to execute a single loop if there is no body of code execution for it.
  + (loop\_max \* 10 cycles) / 1,000,000 cycles = (125 ms / 1000 ms)
  + \_\_delay\_cycles(500,000) means delay for 500 milliseconds only because we have a 1 MHz operating frequency which operates at 1M cycles per second.

1. **Addressing modes in the MSP430 ISA**
2. **Moving through arrays in assembly**
   1. When moving through an array in assembly, the auto increment function can prove very helpful. The indirect addressing mode will get whatever is at the memory address specified in the register, and is how you get the value of an array. If the auto-increment function is used, it will post-increment the memory value stored in the register, so the next time you call for the value, it’ll get the next one “in line” in memory.
3. **Reading assembly**

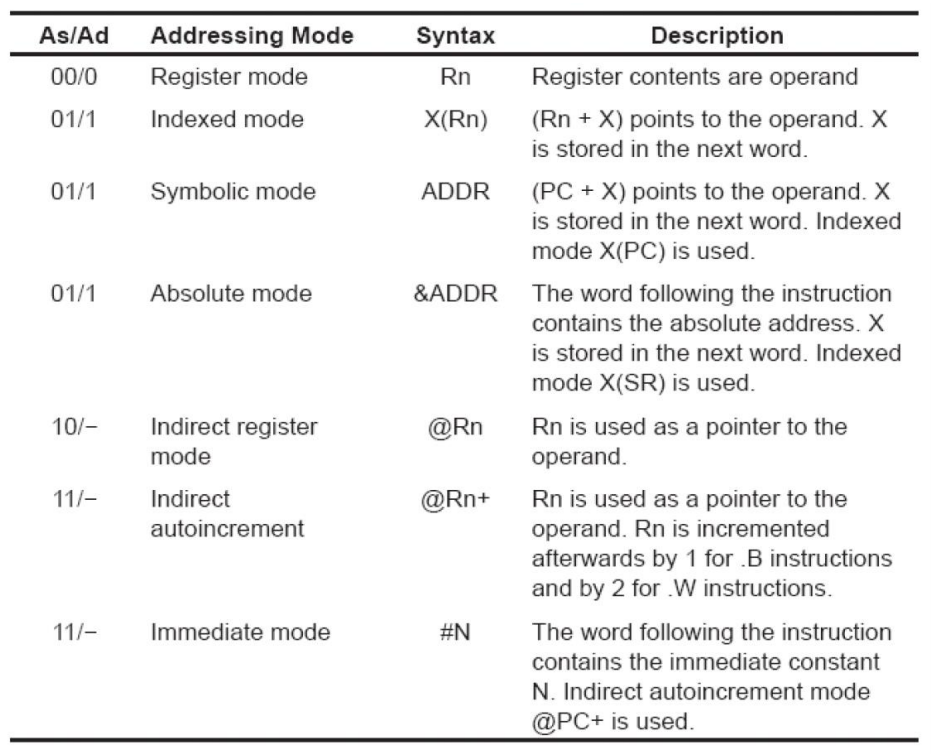
.b vs .w, dadd vs add, stuff like that. Know how jumps work, probably not need to know how subroutines work in the stack or any in-the-stack bullhonkey.

* + 1. macro in the MSP430.h header file.
       1. *if ((P2IN & BIT4) == BIT4);* check P2.4 for if it’s pressed, if it is, it will enter the `if` statement.
    2. PxOUT is the same thing; how you set the output high or low, but for this, you will use an inverse AND statement with the BITy statement to turn a pin high.
       1. `P1OUT &= ~BIT4;`
          1. Turn on P1.4.

1. **Software delays**
   1. **You should understand how to use the clock rate to get the time for clock cycles**
   2. **Any calculations used in the lab, in particular how the for loop delays are working**

* MSP430 operates at 1 MHz. Therefore, time for one second is 1,000,000 clock cycles.
  + A for loop delay takes 10 clock cycles to execute a single loop if there is no body of code execution for it.
  + (loop\_max \* 10 cycles) / 1,000,000 cycles = (125 ms / 1000 ms)
  + \_\_delay\_cycles(500,000) means delay for 500 milliseconds only because we have a 1 MHz operating frequency which operates at 1M cycles per second.

1. **Addressing modes in the MSP430 ISA**

~~~~