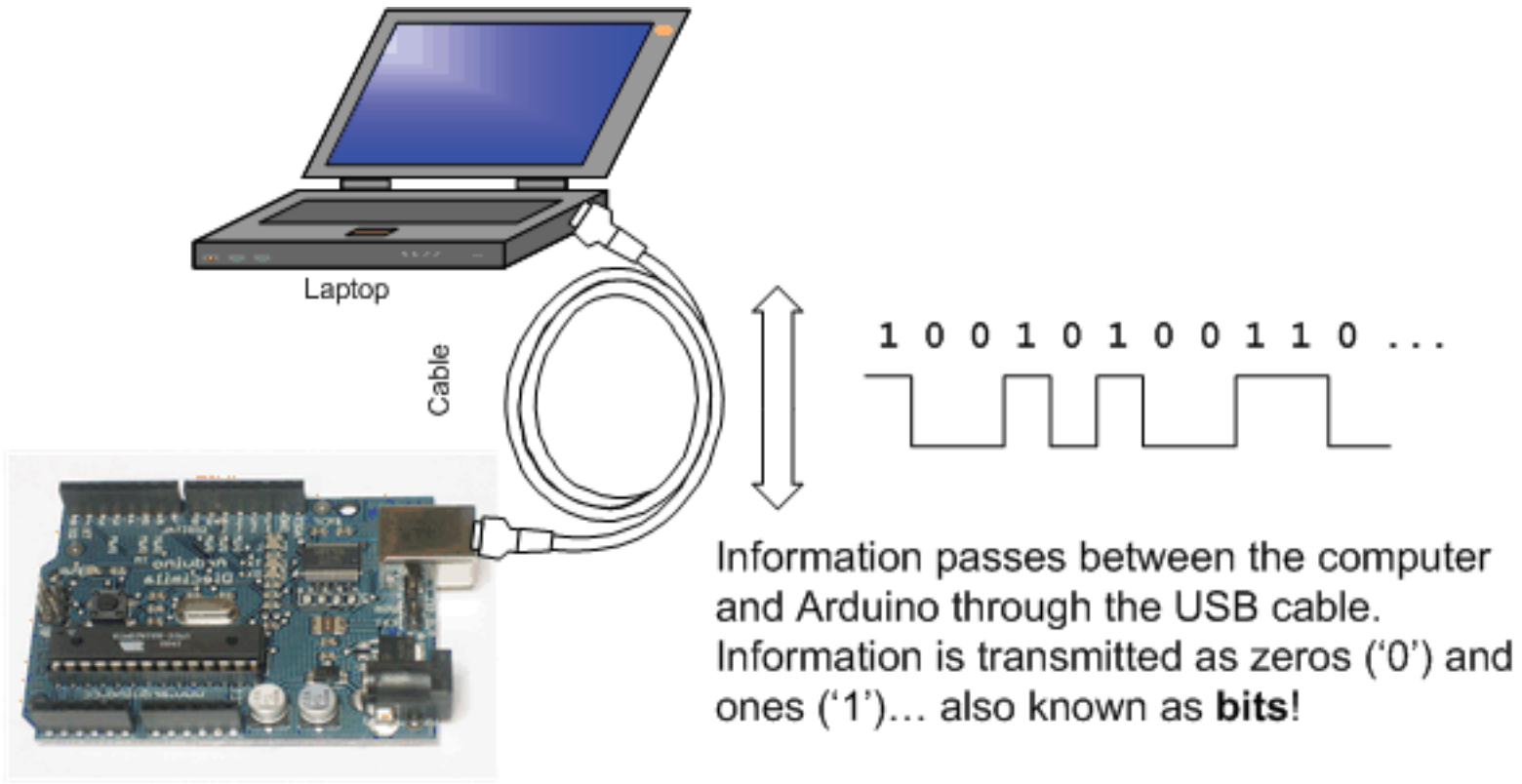
SERIAL COMMUNICATION

Bits, Bytes, Data Rates and Protocols
ASCI interpretation
Using terminal to view serial Data
Serial Out from Arduino
Serial In to Processing, PD, Max/MSP, Flash

SERIAL COMMUNICATION

Communication done one after the other.

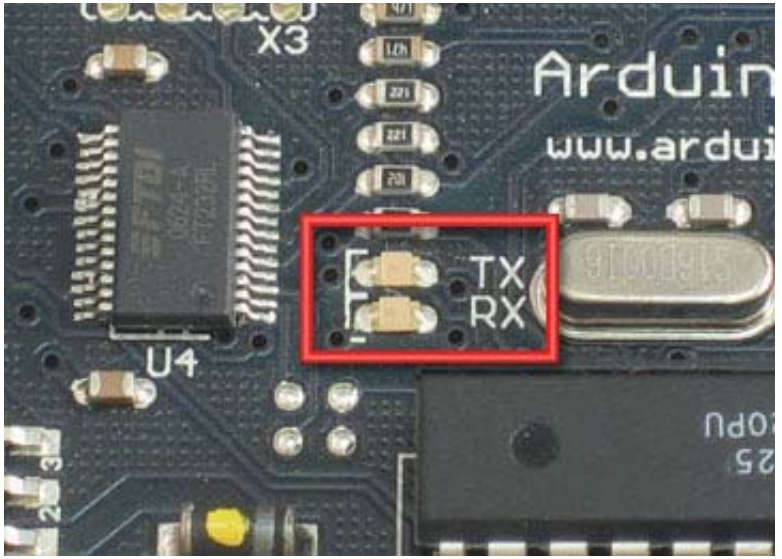


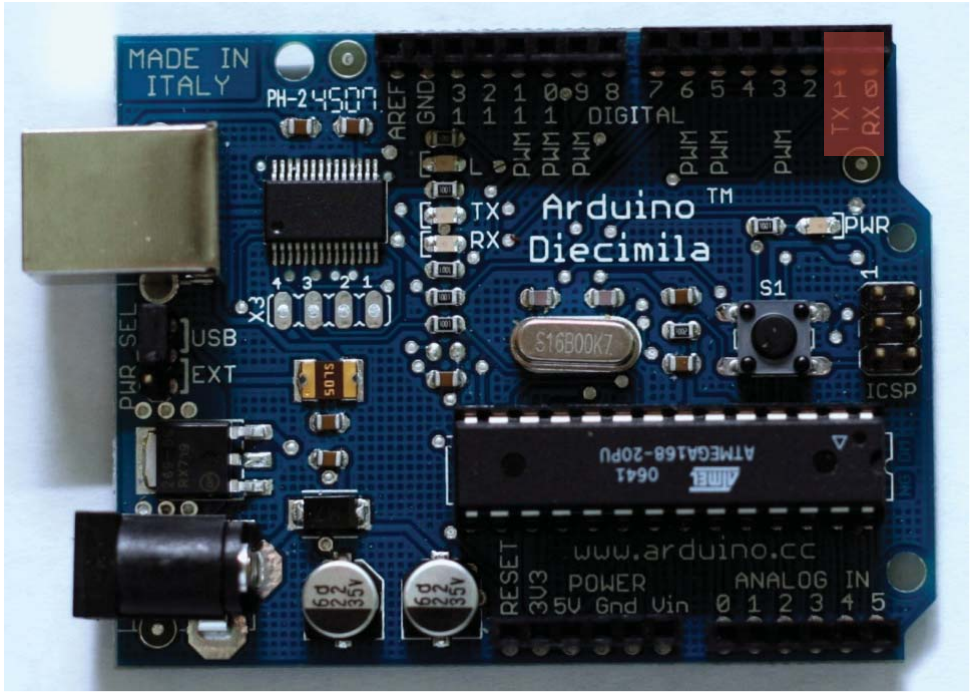
01100010011010010111010001110011001011000010
00000110001001111001011101000110010101110011

- The world isn’t run by weapons anymore, or energy, or money. It’s run by little 1’s and 0’s…
- BIT = 1 / 0
- BYTE = 8 BITs (10010010)
- KB = 1024 B
- MB = 1024 KB
- GB = 1024 MB

SERIAL COMMUNICATION_ARDUINO

- RX – Receiving Data
- TX – Transmitting Data





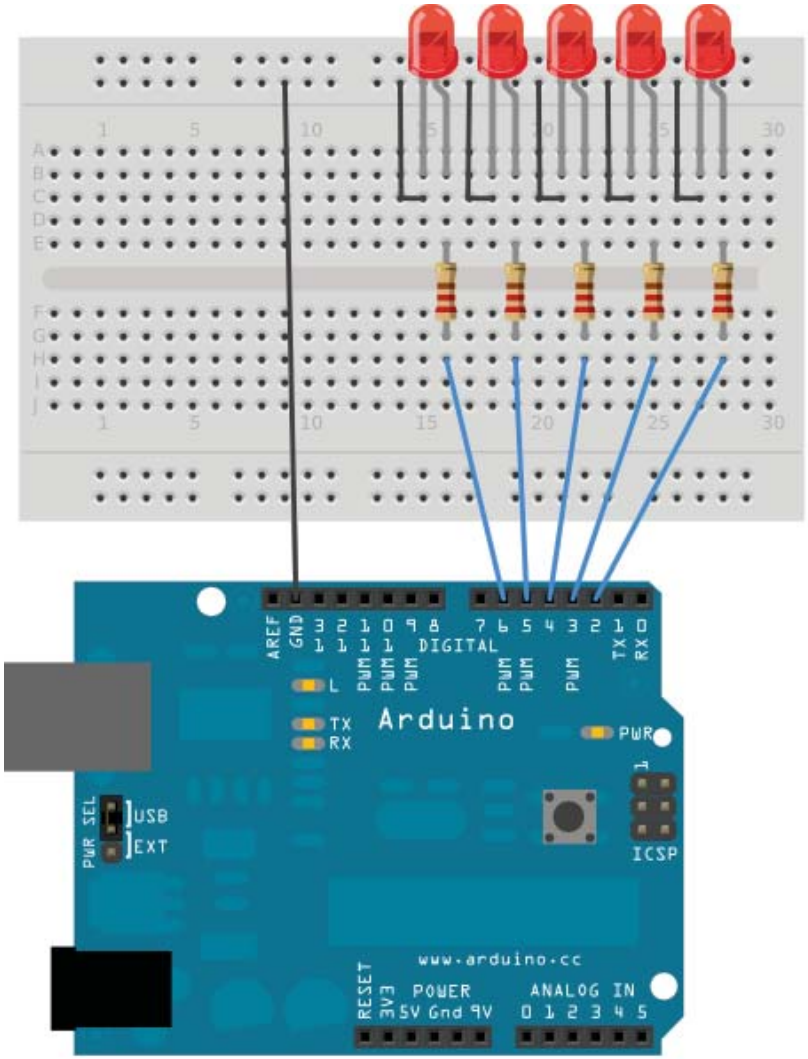
Time to Talk with Arduino board!

- Program to let Arduino say “Hello World!”
- Bit per second (baud rate)
- Note that this communication baud rate is independent of the upload process, which is fixed at 19200 bps.

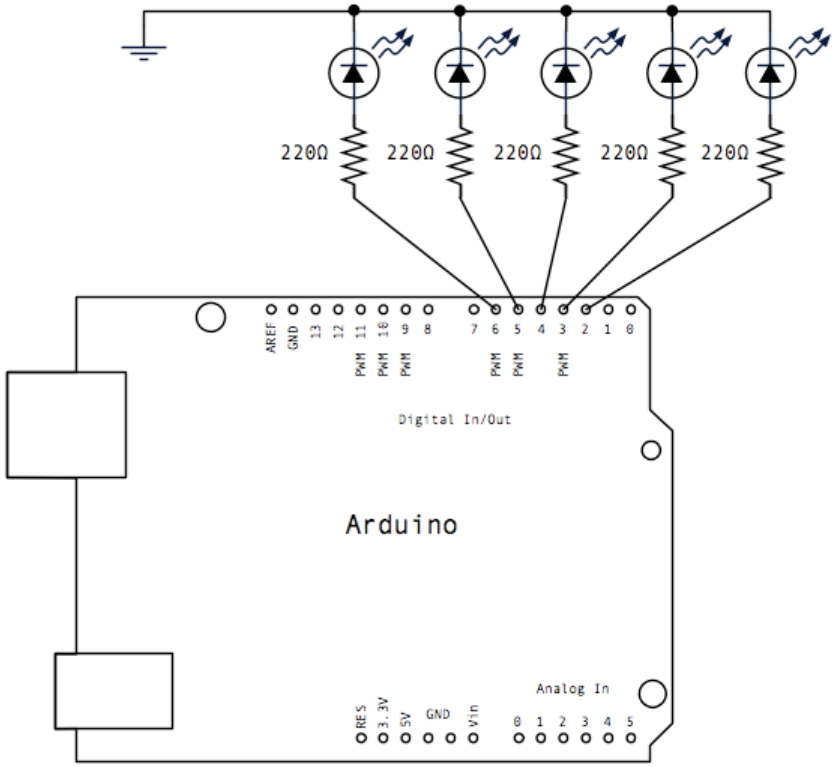
SERIAL AVAILABLE

```
int incomingByte = 0; // for incoming serial data
void setup()
{
Serial.begin(9600); // opens serial port, sets data rate to 9600 bps
}
void loop()
{
        // send data only when you receive data:
        if (Serial.available() > 0)
        {
        // read the incoming byte:
        incomingByte = Serial.read();
        // say what you got:
        Serial.print("I received: ");
        Serial.println(incomingByte);
        }
        else {
        Serial.println("Nothing");
        }
}
```

SERIAL READ_switch case



Code: http://arduino.cc/en/Tutorial/SwitchCase2



SERIAL READ_switch case

EXAMPLES > CONTROL> SWITCHCASE2

```
void setup()
{
        // initialize serial communication:
        Serial.begin(9600);
        // initialize the LED pins:
        for (int thisPin = 2; thisPin < 7; thisPin++)
        {
                pinMode(thisPin, OUTPUT);
        }
}
```

```
void loop()
{
        // read the sensor:
        if (Serial.available() > 0)
        {
                int inByte = Serial.read();
                /*do something different depending on the character received.
                The switch statement expects single number values for each case;
                in this exmaple, though, you're using single quotes to tell
                the controller to get the ASCII value for the character.  For
                example 'a' = 97, 'b' = 98, and so forth:*/
                switch (inByte) {
                case 'a':
                        digitalWrite(2, HIGH);
                        break;
                case 'b':
                        digitalWrite(3, HIGH);
                        break;
                case 'c':
                        digitalWrite(4, HIGH);
                        break;
                case 'd':
                        digitalWrite(5, HIGH);
                        break;
                case 'e':
                        digitalWrite(6, HIGH);
                break;
                default:
                        // turn all the LEDs off:
                        for (int thisPin = 2; thisPin < 7; thisPin++) {
                        digitalWrite(thisPin, LOW);
                        }
                }
        }
}
```

SERIAL FLUSH

- Flushes the buffer of incoming serial data.
- That is, any call to Serial.read() or Serial.available() will return only data received after all the most recent call to Serial.flush().

INTERFACING WITH OTHER SOFTWARE_serial in

- Processing, PureDate(PD), Max/MSP, Flash

http://www.arduino.cc/playground/Main/InterfacingWithSoftware

ARDUINO_SERIAL WRITE

- Writes binary data to the serial port. This data is sent as a byte or series of bytes; to send the characters representing the digits of a number use the **print()** function instead.