# CSE423: Embedded System Summer-2020

## Serial Communication

# Todays Lecture

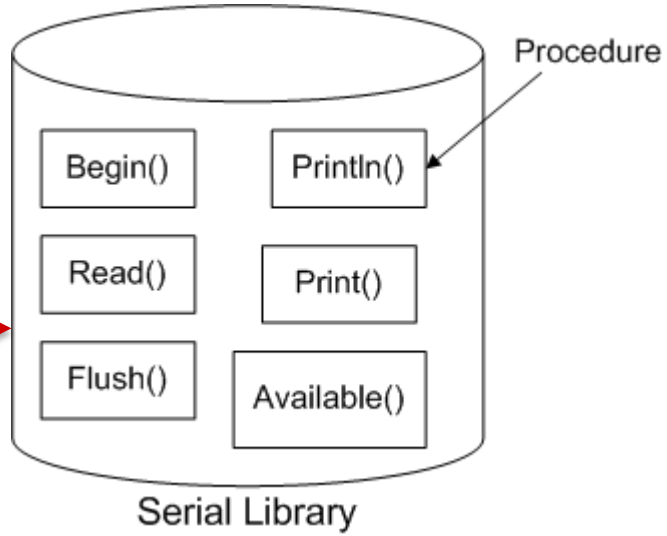

- *Understanding Serial Communication*
- *Understanding Serial Library*
- *Application of Serial Library*
- *Additional functions*

# What is Library

A **procedure** is a list of things to do. A **library** is a big collection of procedures, where all the procedures are related!

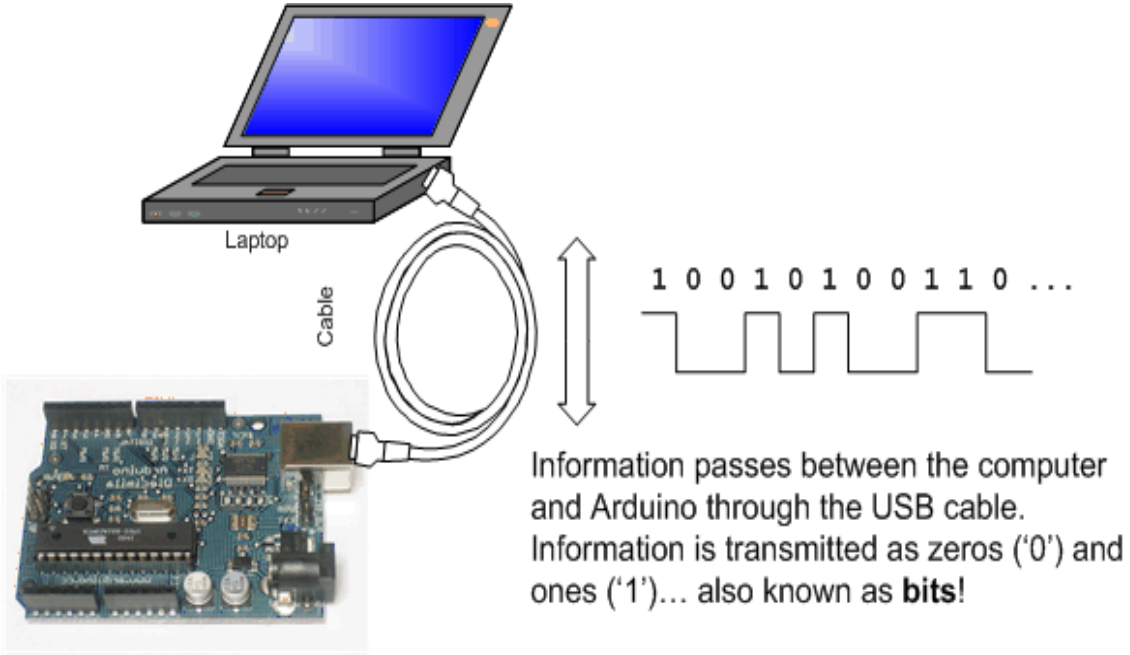
Let's say, we want to control a motor, you may want to find a Motor Control Library: a collection of procedures that have already been written for you and available to use.

The **library** which is widely used in Arduino platform is the **Serial Library**, which allows the Arduino to send data back to the computer



# What is Serial?

The word serial means "one after the other". For example, a serial killer doesn't stop with one murder, but stabs many people one after the other. Serial data transfer is when we transfer data one bit at a time, one right after the other.

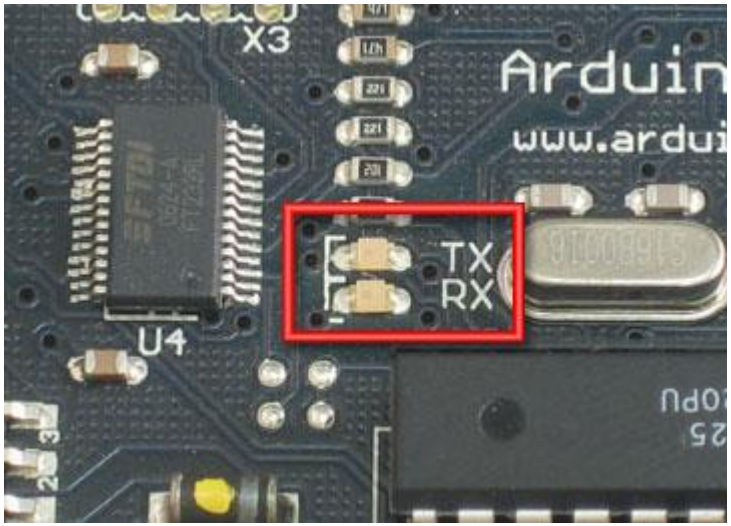


# What is Serial?

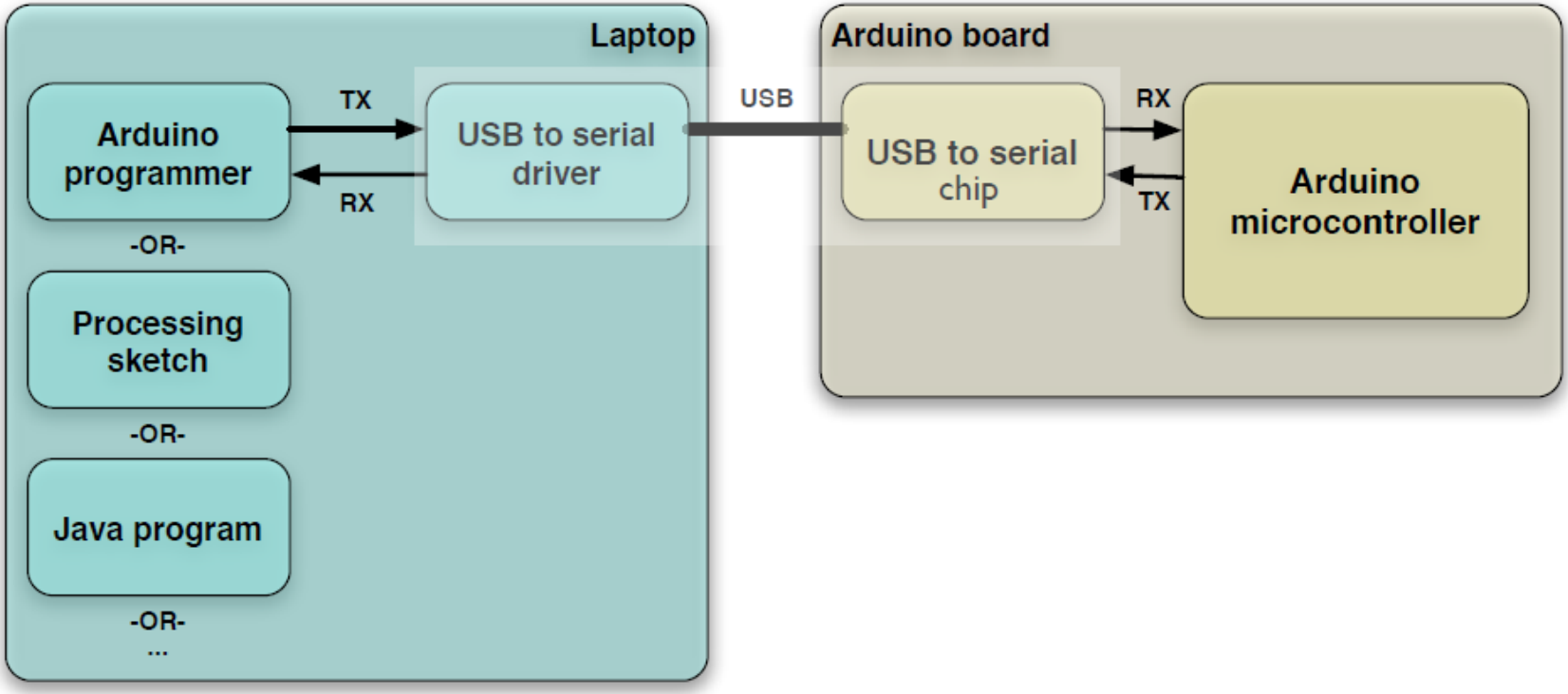
Information is passed back & forth between the computer and Arduino by, essentially, setting a pin high or low

**Serial Communication:**

We've actually used the Serial communications capability already by sending sketches to the Arduino! When you **Compile/Verify** what you're really doing is turning the sketch into **binary data** (ones and zeros). When you **Upload** it to the Arduino, the bits are shoved out one at a time through the USB cable to the Arduino where they are stored in the main chip.
Next time you upload a sketch, look carefully at the two LEDs near the USB connector, they'll blink when data is being transmitted. One blinks when the Arduino is receiving data **(RX)** and one blinks when the Arduino is transmitting data (**TX**)



# Serial communication



# What happens if we write this?


```
void setup()                          // run once, when the sketch starts
{
  Serial.begin(9600);                  // set up Serial library at 9600 bps
  Serial.println("Hello world!");     // prints hello with ending line break
}


void loop()                           // run over and over again
{
                                      // do nothing!
}
```

# Understanding Serial Library


We definitely see that there is a **Serial** thing going on, and it looks like there is a procedure call as well. This is a **library procedure call**. The library is called **Serial** and inside the library is a procedure called **begin**.

| library name | . | procedure name | (input values) | ; |
|---|---|---|---|---|
| Serial | . | begin | (9600) | ; |

If there's no library name, it means that the procedure is in the 'default' collection of procedures we use.

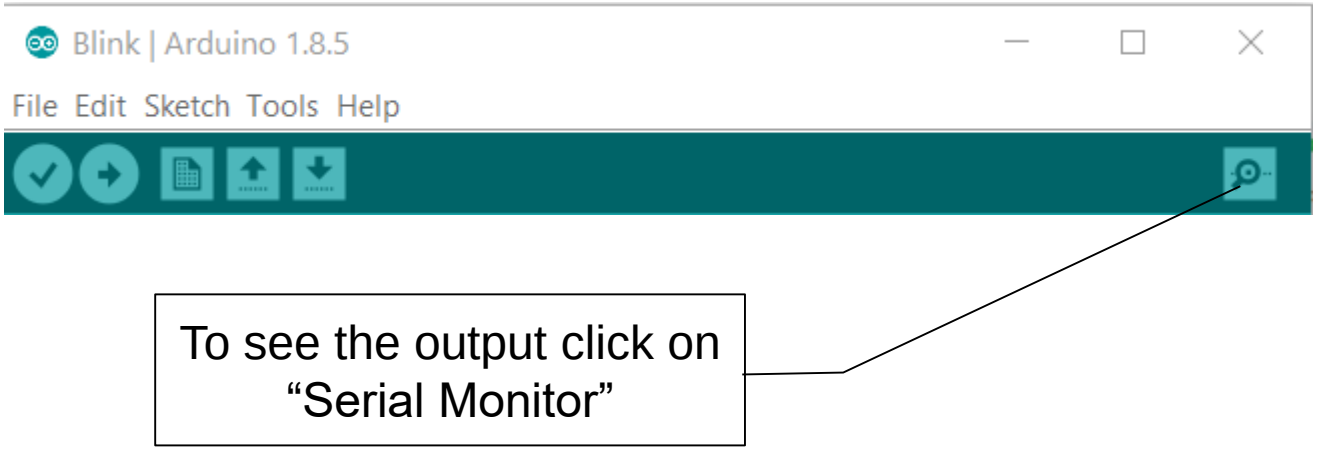
So there's some mystery procedure that's called **begin**, well it's not too tough to figure out what it might do. It's the procedure that gets the Serial stuff ready. But what's the **9600** about?The comment says **9600 bps**, and just so you know bps stands for bits-per-second (we will refer to this as the **baud rate**).

OK so Serial.begin sets up the Arduino with the transfer rate we want, in this case 9600 bits per second

# Understanding Serial Library

Serial.println("Hello world!");

This line also uses the **Serial** library, this time it's calling a procedure called **println** which is just a shorthand for "print line". Note that the 6th letter in **println** is the letter **L** not the number 1. This time the input is a quotation, the line of text we would like it to print. We use two **"**'s (double quotes) to indicate the beginning and end of a line of text.



# Understanding Serial Library

Output is shown in Separate Window


As serial communication passes information back and forth so, Arduino Board must be connected to see this output.

# Understanding Serial Library


Here, we are addressing couple of most popular libraries related with Serial:

| | | |
|---|---|---|
| Serial.available() | Serial.flush() | Serial.find() |
| Serial.write() | Serial.readString() | Serial.read() |

Reference Link:
https://www.arduino.cc/reference/en/language/functions/communication/serial/