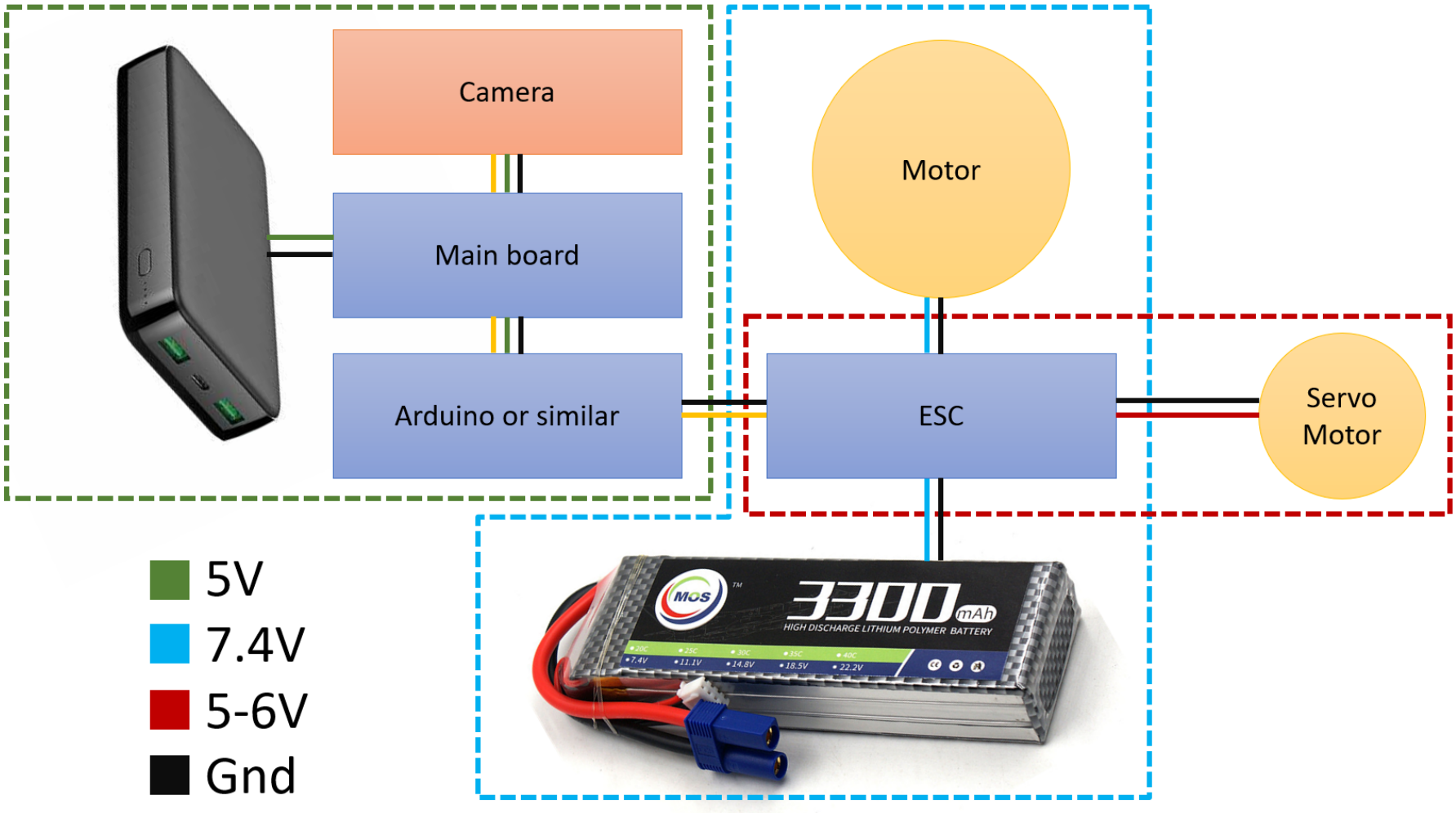
06 - Hardware, communication and protocols

Maxime Ellerbach

May 2023

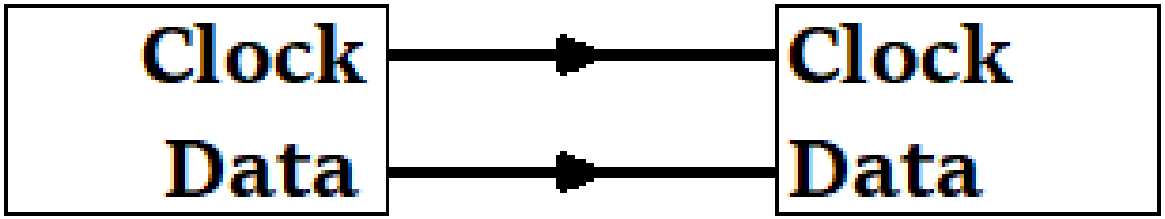
Car Hardware Overview

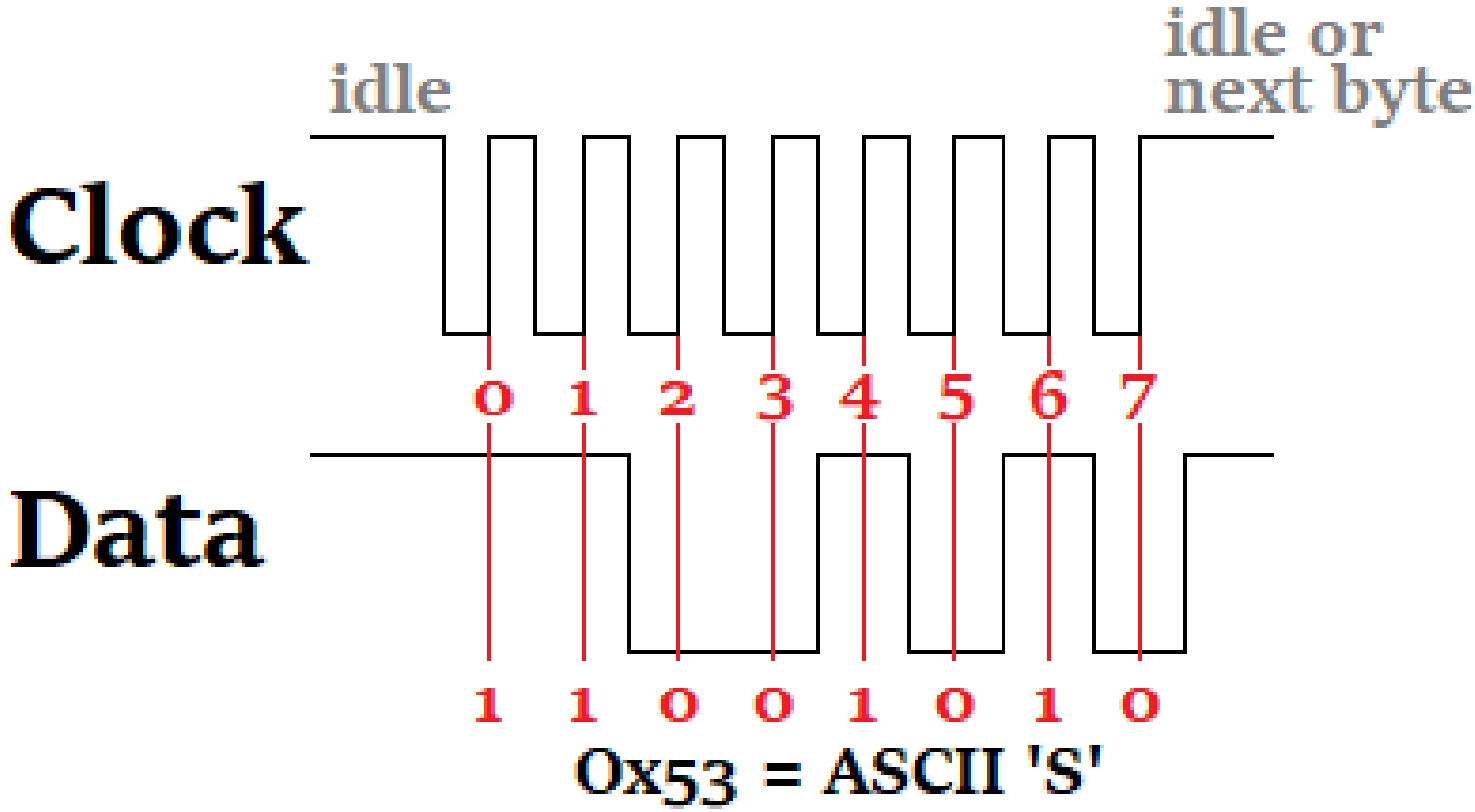


Data Transmission

- Parallel Buses
  - Multiple bits transmitted at the same time through separate lines
  - Requires a lot of wiring e.g. 32 bits at same time = 32 wires
  - ATA, SCSI, PCI, ...
- Serial Buses
  - Transmits data bit by bit sequentially
  - Slower transmission rates
  - Less wiring
  - Telegraph, USB, UART, I2C, CAN, ...

Synchronous Serial Bus



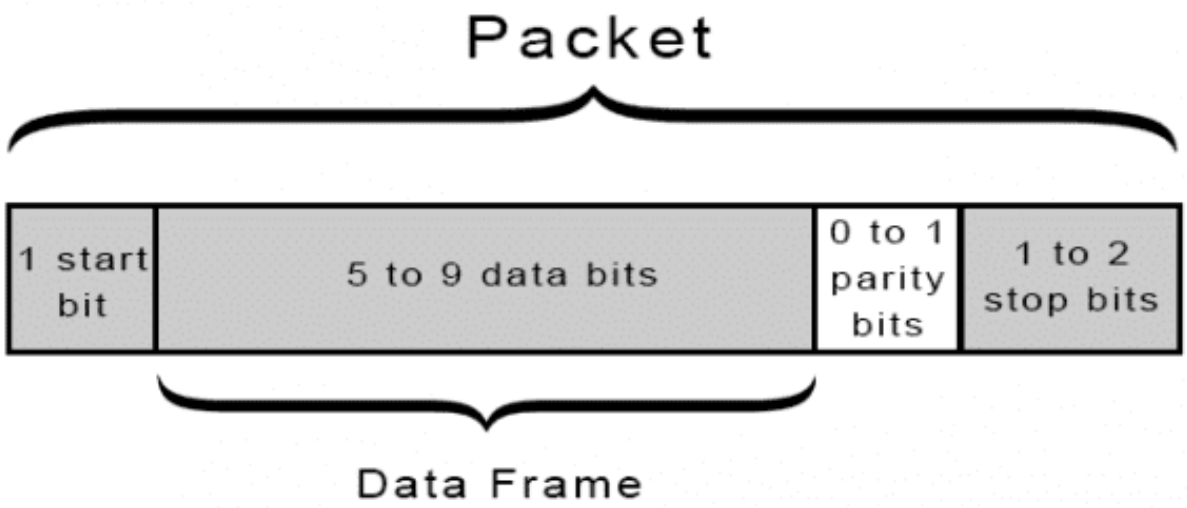


Unicast and Multicast

- Unicast
  - Single sender & receiver
  - point to point
  - Limited scalability
- Multicast
  - Single sender & multiple receiver
  - One-to-many or many-to-many
  - Efficient when having lots of receiver

UART - Universal Asynchronous Receiver/Transmitter

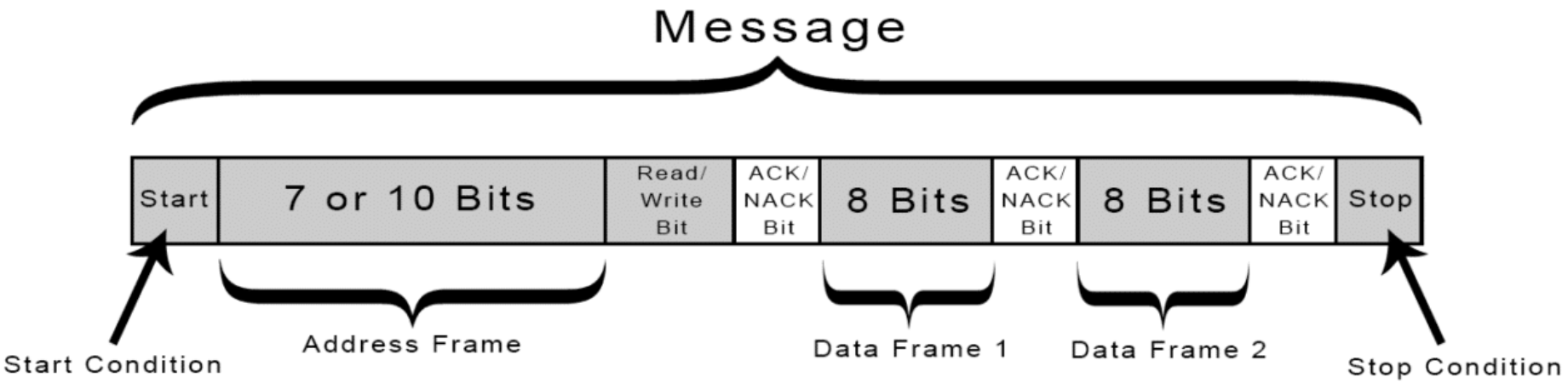
- **Asynchronous**: no shared clock
- **Serial**: single data line (RX / TX)
- **Half/Full Duplex**: can receive and send data simultaneously



https://www.circuitbasics.com/

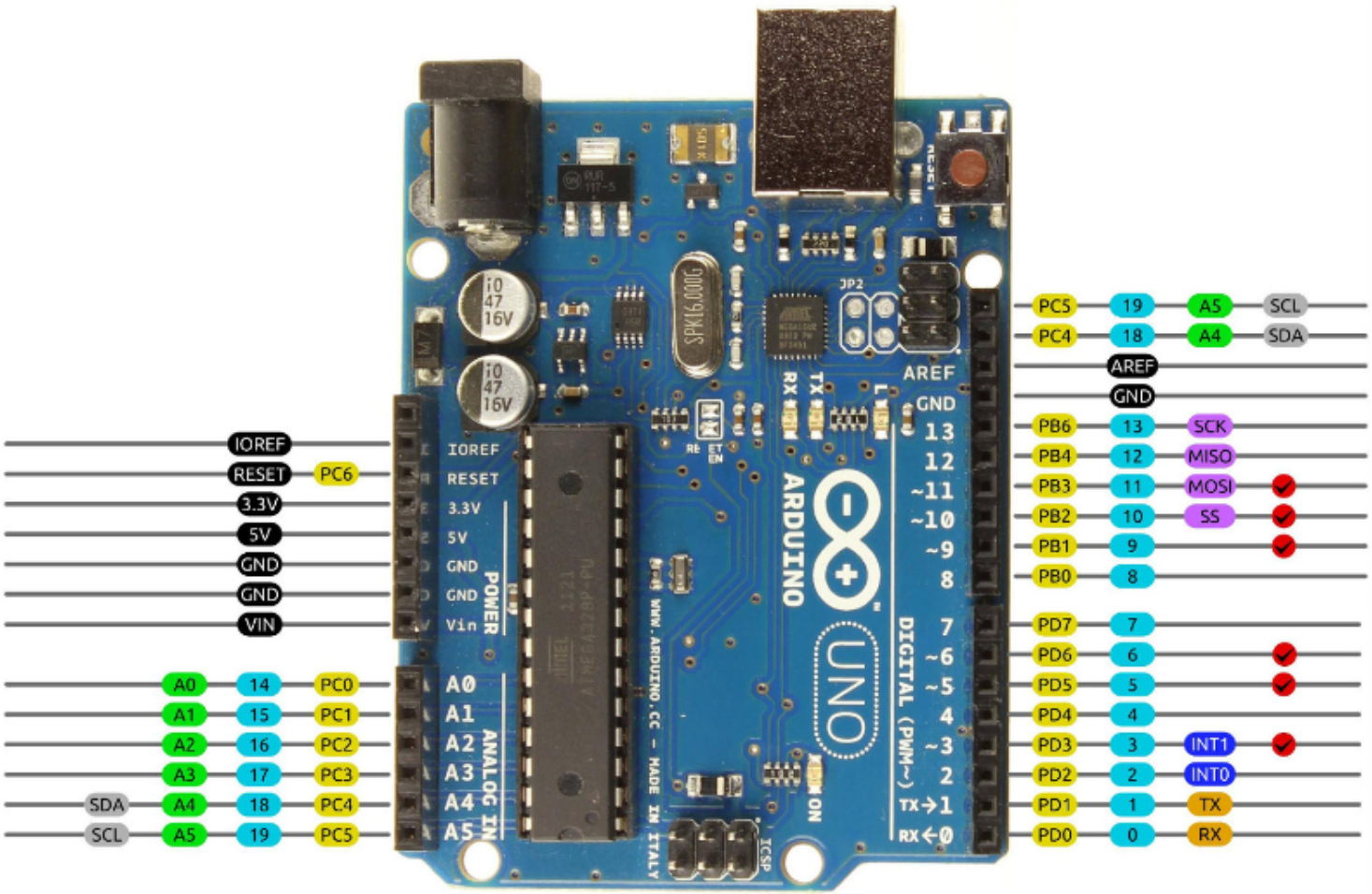
I2C - Inter-Integrated Circuit

- **Two communication lines**
  - Serial Data Line (SDA)
  - Serial Clock Line (SCL)
- **Master-Slave architecture**
  - Supports multiple Master
  - 7 bit addressing



https://www.circuitbasics.com/

Arduino Pinout



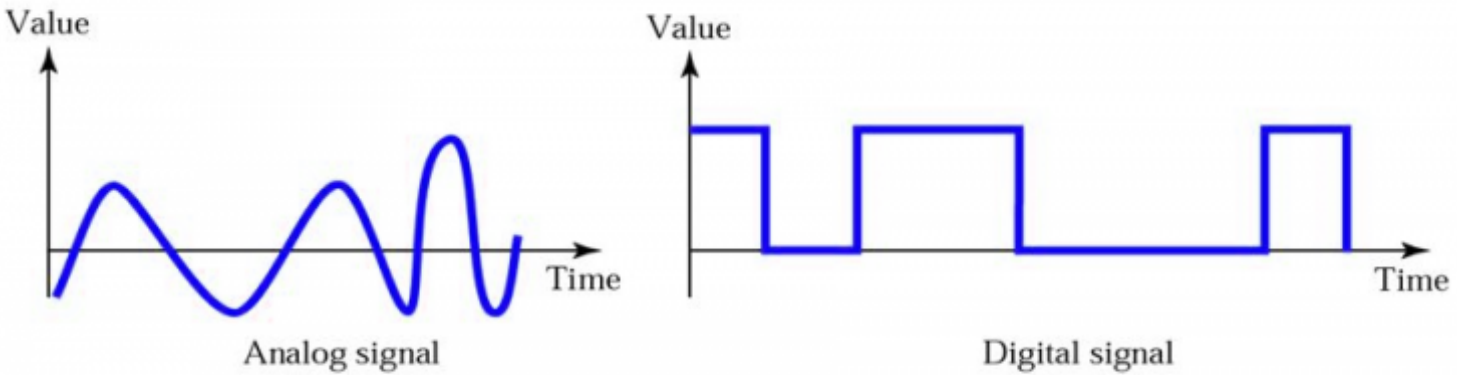
Digital and Analog pins

Digital Pins:

- Used to transmit and receive binary signals,
- Two voltage levels: HIGH (logic 1) or LOW (logic 0).

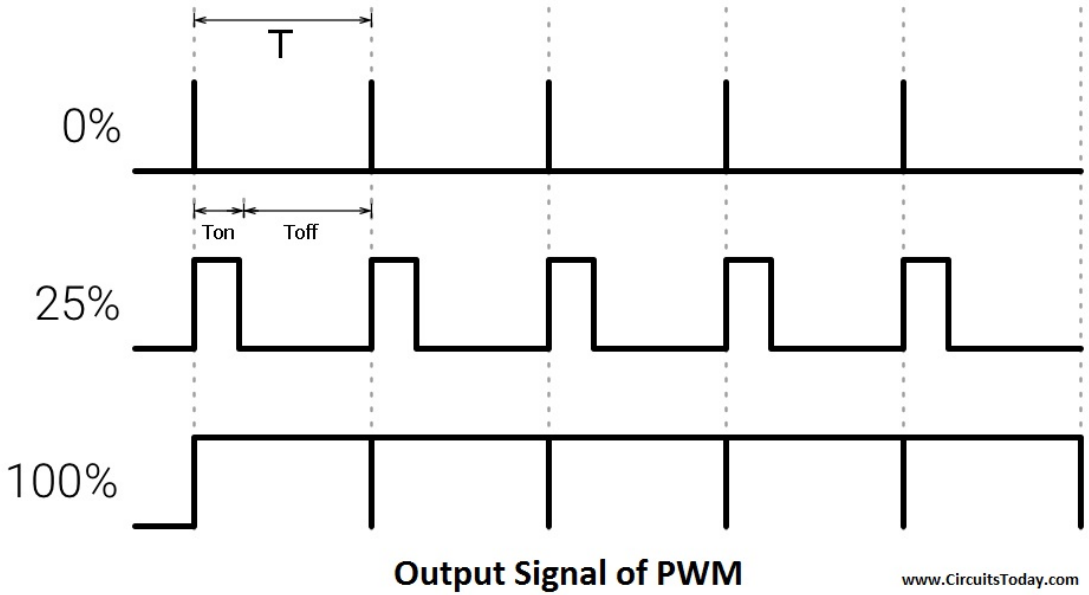
Analog Pins:

- Used to measure continuous or analog signals, such as voltages or sensor readings
- Convert analog signals into digital values
- Often labeled with an "A" followed by a number (e.g., A0, A1, A2)



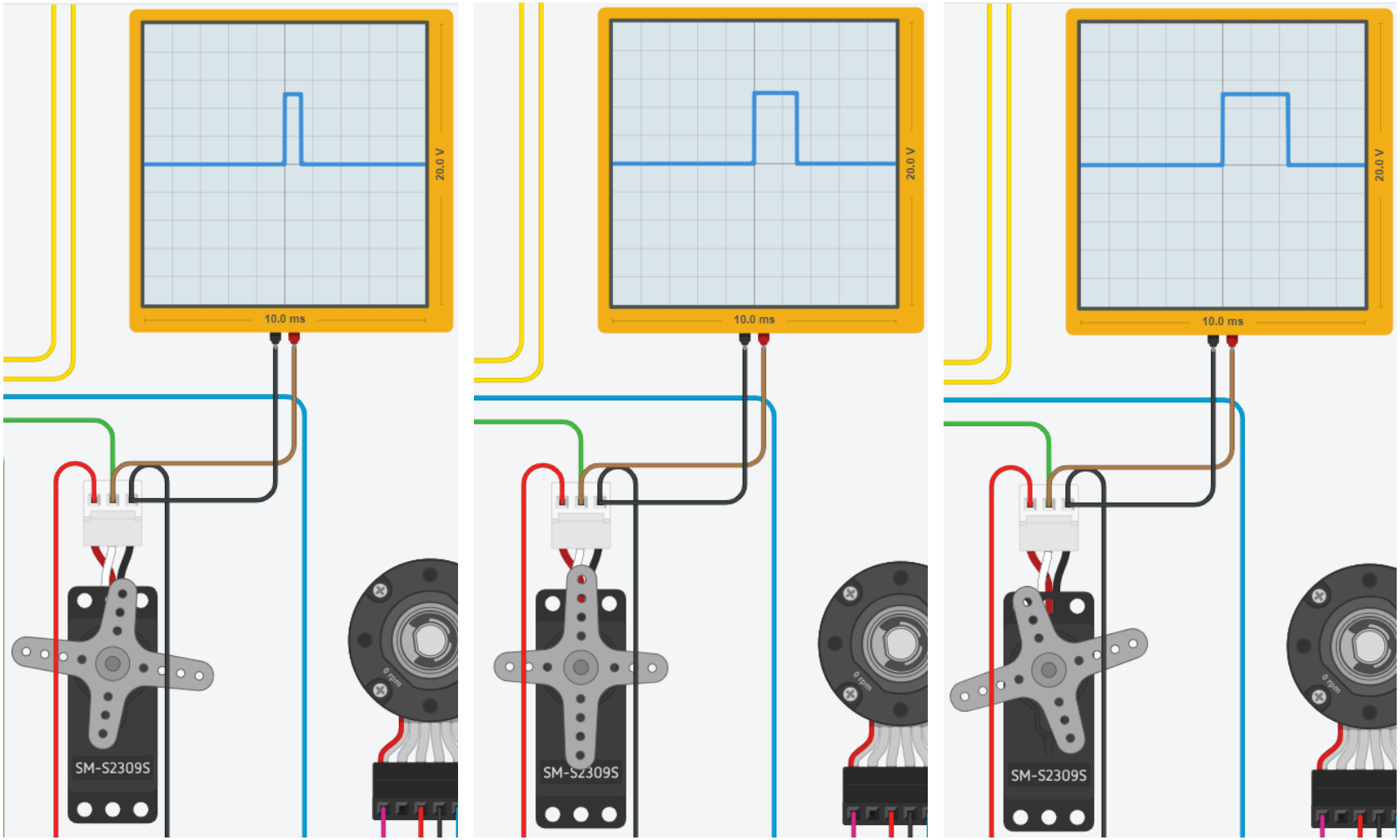
PWM (Pulse Width Modulation) Signal

- Encode analog information in a digital signal by varying the width of the pulses
- The output analog value is the ratio between the time the signal is HIGH (=1) and the period T
- Useful in our case to drive the servo and motor easily
- Only certain Digital pins support PWM, refer to the pinout of the micro-controller



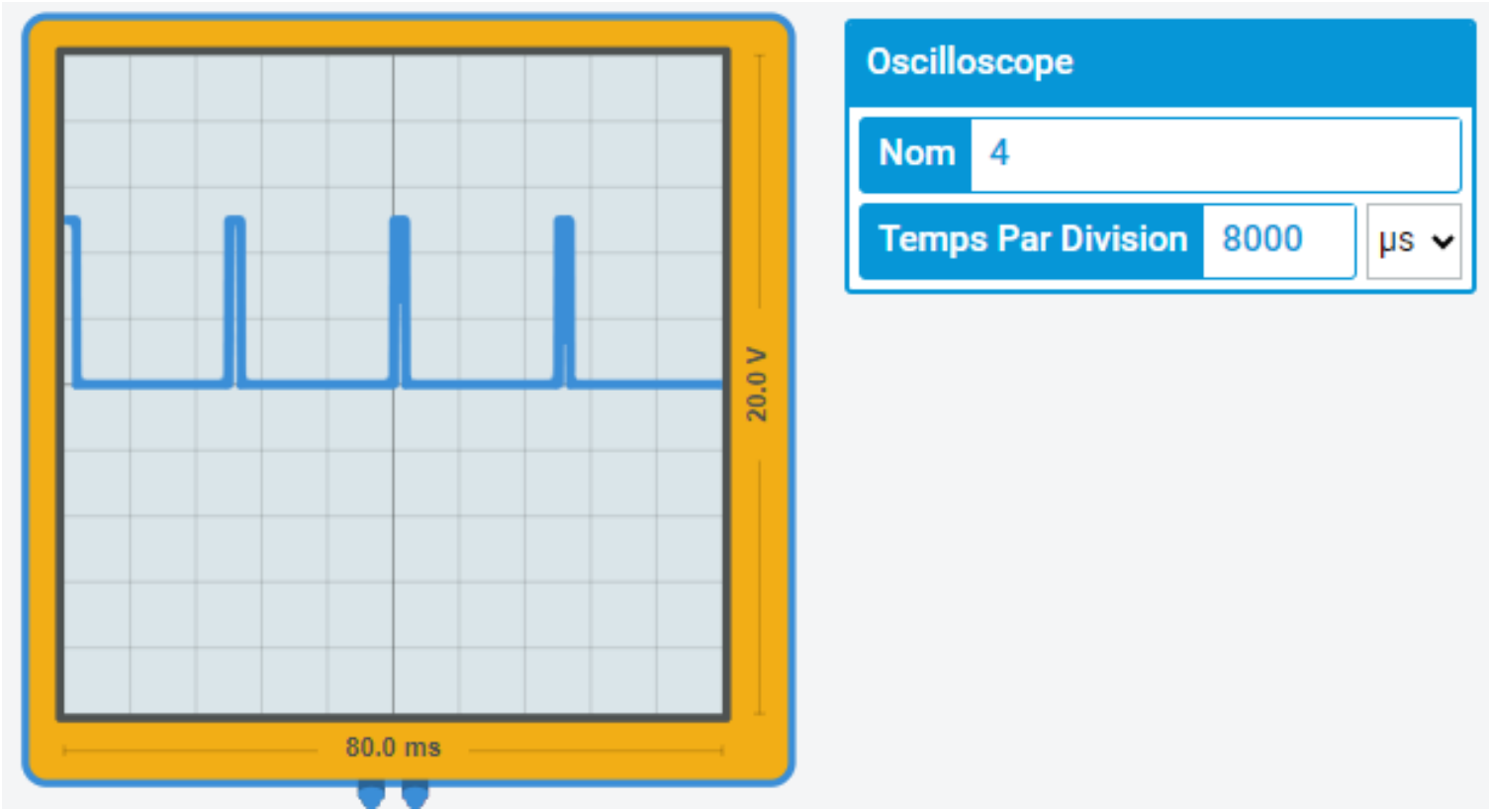
PWM (Pulse Width Modulation) Signal

left, center, right



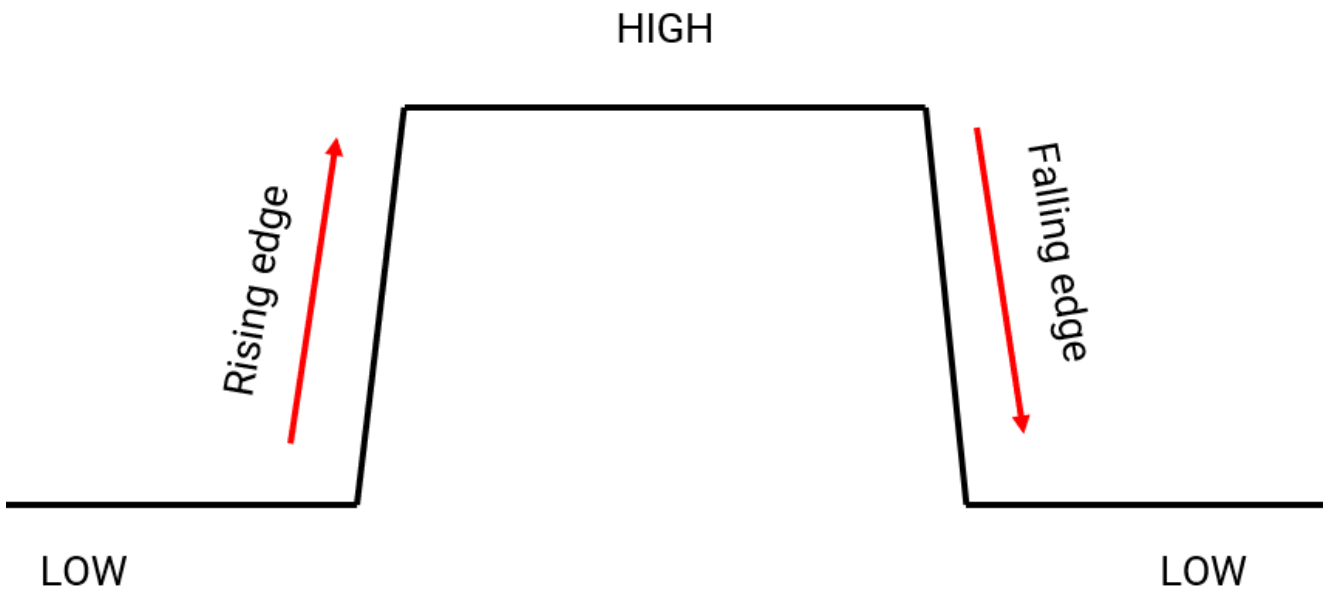
PWM (Pulse Width Modulation) Signal

- Here we observe 4 periods, so T = 80ms / 4 = 20ms
- Full left = 0.5ms HIGH time (500 microseconds). duty of 2.5%
- Full right = 2.5ms HIGH time (2500 microseconds) duty of 12.5%

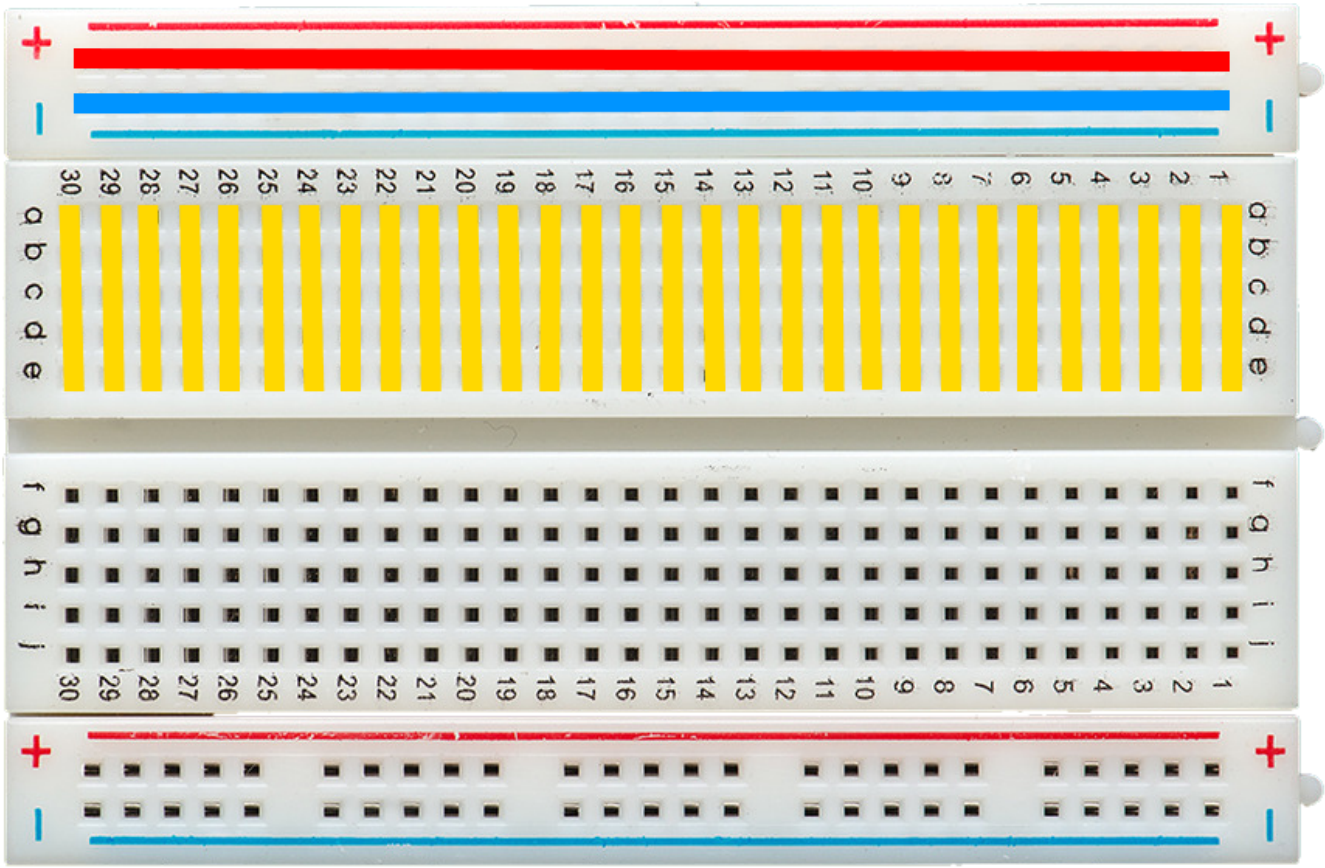


Interrupt pins

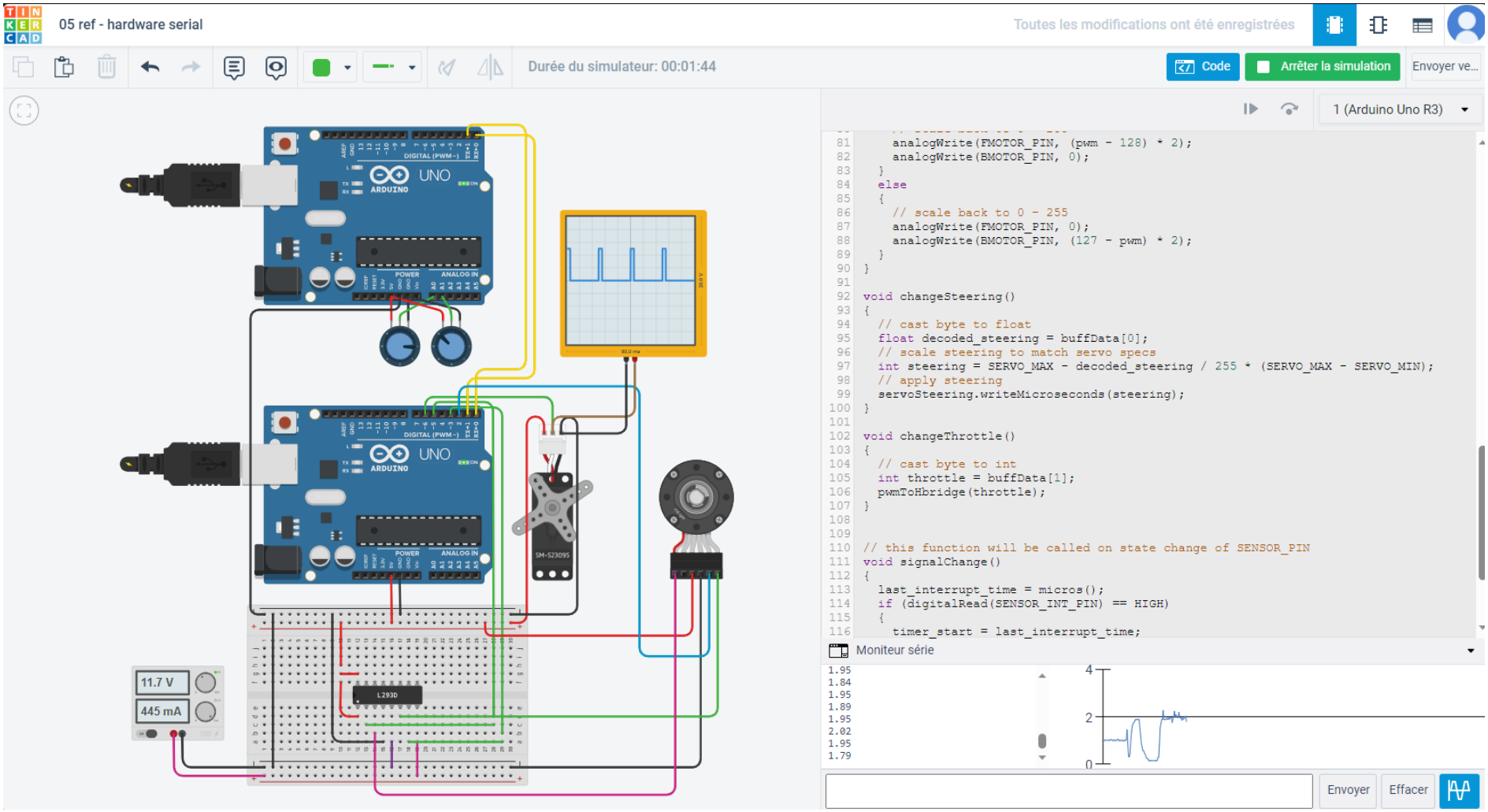
- Digital pin 2 and 3 on arduino Uno
- Trigger an interrupt request when specific event occurs.
  - **LOW** - triggers INT when pin is LOW
  - **RISING** - triggers INT when pin goes from LOW to HIGH
  - **FALLING** - triggers INT when pin goes from HIGH to LOW
  - **CHANGE** - triggers INT when pin changes state.



Breadboard wiring



Practice 05 - Tinkercad




```
    analogWrite(FMOTOR_PIN, (pwm - 128) * 2);
    analogWrite(BMOTOR_PIN, 0);
  }
  else
  {
    // scale back to 0 - 255
    analogWrite(FMOTOR_PIN, 0);
    analogWrite(BMOTOR_PIN, (127 - pwm) * 2);
  }
}

void changeSteering()
{
  // cast byte to float
  float decoded_steering = buffData[0];
  // scale steering to match servo specs
  int steering = SERVO_MAX - decoded_steering / 255 * (SERVO_MAX - SERVO_MIN);
  // apply steering
  servoSteering.writeMicroseconds(steering);
}

void changeThrottle()
{
  // cast byte to int
  int throttle = buffData[1];
  pwmToHbridge(throttle);
}


// this function will be called on state change of SENSOR_PIN
void signalChange()
{
  last_interrupt_time = micros();
  if (digitalRead(SENSOR_INT_PIN) == HIGH)
  {
    timer_start = last_interrupt_time;
```

```
1.95
1.84
1.95
1.89
1.95
2.02
1.95
1.79
```