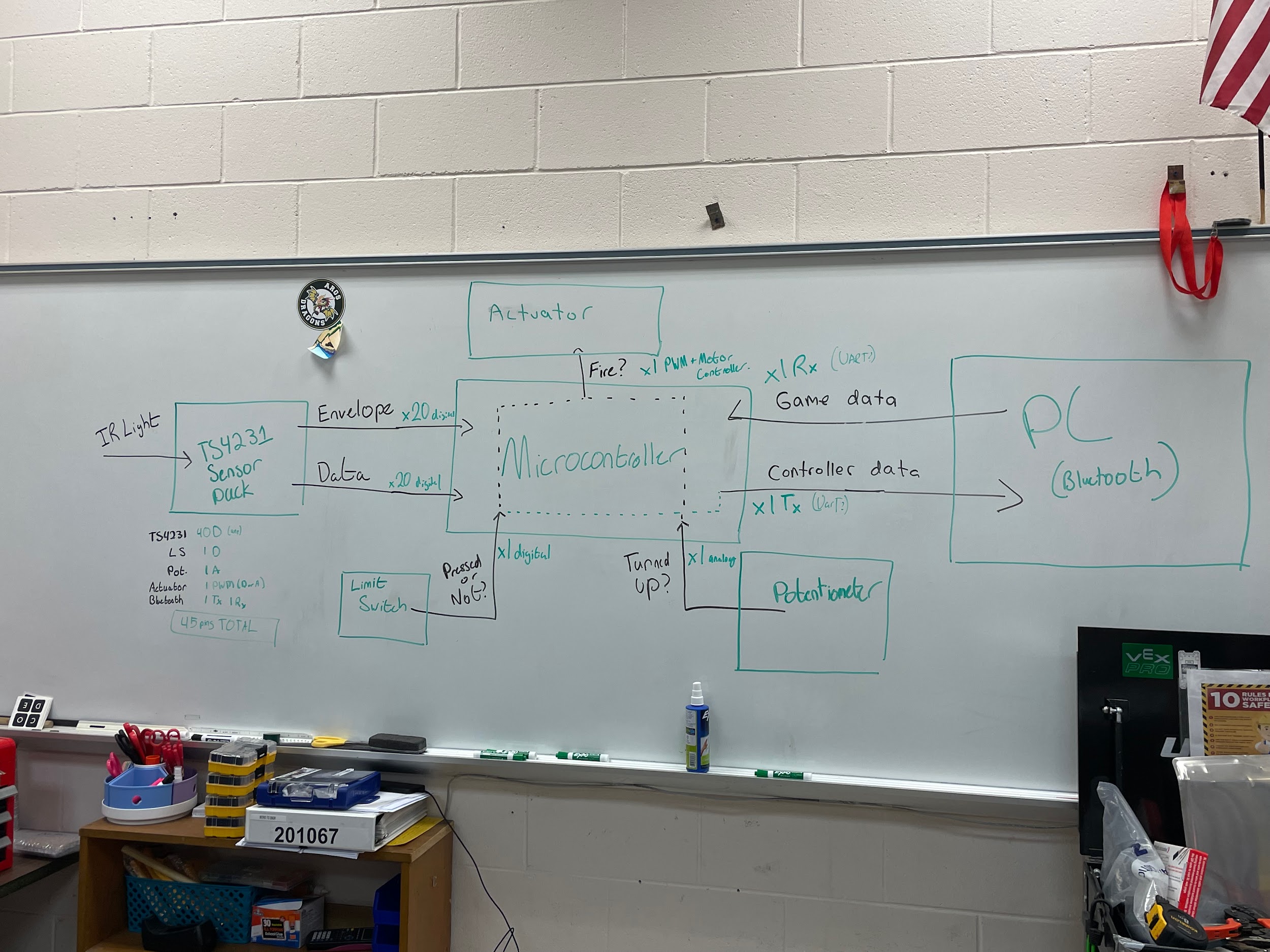
~Parts Overview

* [TS4231](https://triadsemi.com/product/ts4231/)
  + Program:
    - Configuration: Bases data off “driving” the **CFGCLK** and **CFGDATA** pins
    - Runtime: it takes IR light from **Photodiode** and outputs it into “Position-Indicating” digital **Envelope** signals. Also includes a digital representation of the IR light coming in through the **Data** signals
  + Input:
    - Photodiode[Configuration only] CFGCLK + CFGDATA
    - 3.3VCC (From Microcontroller), GND
  + Output:
    - Envelope Stream(CFGCLK) + Data Stream(CFGDATA)
* Microcontroller {[Teensy LC](https://www.pjrc.com/teensy/teensyLC.html) + [Teensy 4.1](https://www.pjrc.com/store/teensy41.html), or Arduino Due
  + Program:
    - Configuration: Through [premade library](https://github.com/TriadSemi/TS4231), will configure each **TS4231** using **CFGCLK** and **CFGDATA**
    - Runtime:
      * Controller Data
        + Is **limit switch** pressed? If so, send over signal bluetooth
        + Detect **TS4231**(envelope) input, send over bluetooth.
      * Actuator Data
        + How high is the **potentiometer dial** turned up, correspond that to strength of actuator ON TEENSY
        + Is the **limit switch** pressed? If so, activate the actuator
  + Inputs:
    - X# TS4231 (Envelope AND Data)
    - Limit Switch(trigger)
    - Potentiometer Dial (Controlling strength of knockback)
    - Game data (From bluetooth)
    - 5VCC, GND
  + Outputs:
    - [Configuration only] CFGCLK + CFGDATA (To sensor)
    - Controller Data (To Bluetooth)
      * Sensor Data (32-bit number, refer to : [this @ 7.3](https://hackaday.io/project/19570-htc-vive-lighthouse-custom-tracking/details))
      * Each sensor has its own 32-bit number.
    - Actuator data (To Actuator)
* Bluetooth LE Shield.
  + Program:
    - Will transfer controller data to PC, and transfers PC data to Controller.
  + Input
    - Teensy sends CONTROLLER DATA wired.
    - PC sends GAME DATA wirelessly.
  + Output:
    - Controller Data (to computer), Game Data (To Teensy System)

~Diagram of System **During Runtime**

Here’s some help for getting started :) Actually draw this please!

TS4231 Sensor -> Microcontroller <-> Bluetooth

* 

~Runtime Pseudocode for Teensy System

Sensors:

Mark time when Sensor Pin goes high. (Millis())

Mark time when Sensor Pin goes low. (Millis())

Time elapsed when Sensor PIn goes high vs. low

Should have 9 values roughly (Nskip, Skip, Sweep)

Time elapsed between Start of Nskip and Sweep = Angle

Variables:

* + Pins:
    - Int *LIMITSWITCH*
    - int[] *IR\_SENSOR\_LIST*
    - int *POTENTIOMETER*
  + bool *isTriggerPressed*
  + int *poteniometerValue*

Code:

* + TS4231 Sensor Input
    - For(int i = 0; i < IR\_SENSOR\_LIST.length; i++)
  + Limit Switch Input
    - L = DigitalRead(*LIMITSWITCH*)
    - If *L* is high.
      * Set *isTriggerPressed* to true
    - Else:
      * Set *isTriggerPressed* to false
  + Potentiometer Input
    - *potentiometerValue* = AnalogRead(*POTENTIOMETER*)
  + Controller Output
  + Actuator Output

~Sending Protocol

* Sensor ID (int)
* Lighthouse ID (bool)
* X or Y Axis (bool)
* Validity (bool)
* Angle (int)

SSSSSLAV

ΘΘΘΘΘΘΘΘ