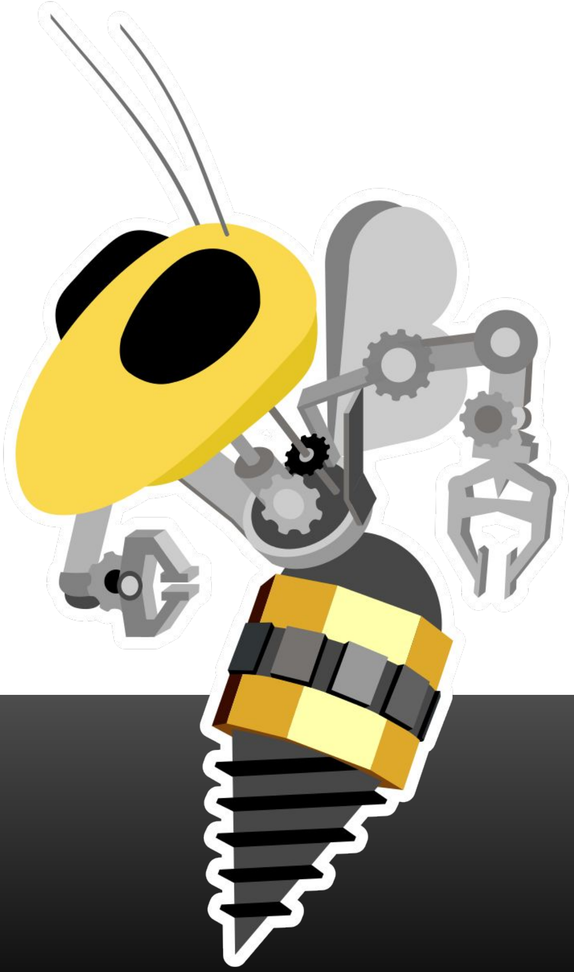
Welcome!

Electrical/Firmware Training
Week 5



ROBOJACKETS
COMPETITIVE ROBOTICS AT GEORGIA TECH

www.robojackets.org

Last Week!

- Electrical Training
  - Board Layouts
- Firmware Training
  - I2C Communications

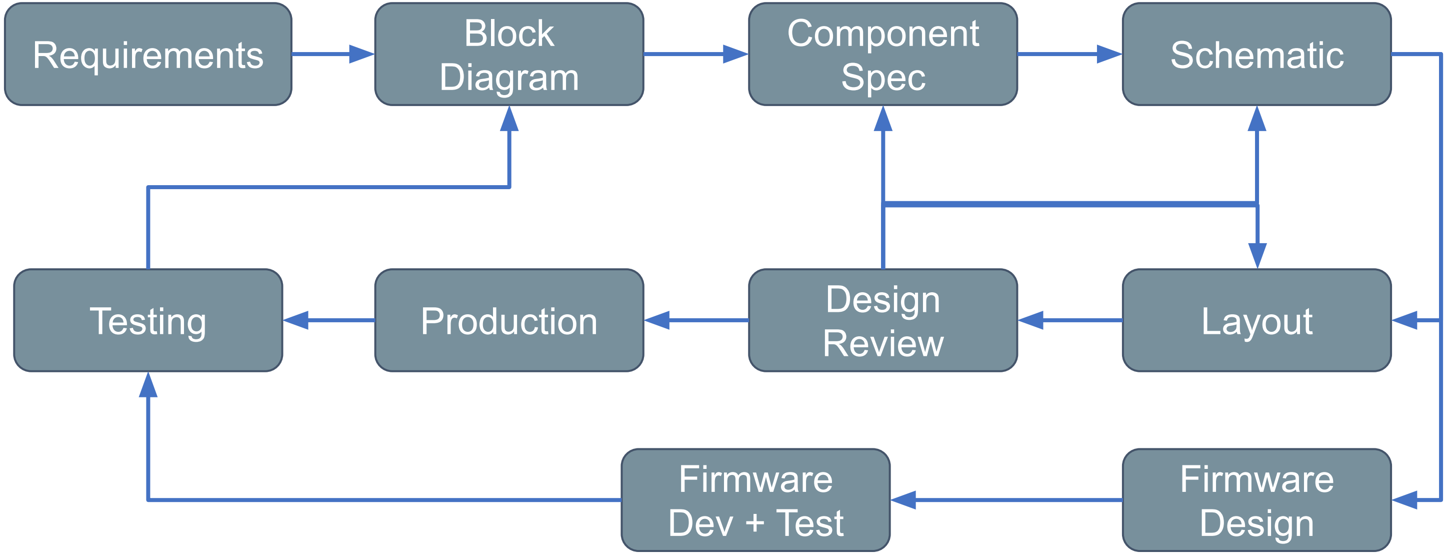
This Week!

- Design Decisions
- Requirements
- Planning a Board Design

The Design Process

Feedback Loops IRL

Timeline


Requirements
Block Diagram
Component Spec
Schematic
Testing
Production
Design Review
Layout
Firmware Dev + Test
Firmware Design

Requirements and Constraints

- What does the board do for the system
- External devices the board interacts with
- Inputs and Outputs
  - Communication
  - Power
- SWaP - Size, Weight, and Power

Inputs and Outputs

- Power System
  - What is the power source of the board?
    - Battery, Power regulator
  - How much power does the component use?
- Communications
  - What information does the board need to do its job?
  - How does the information arrive (protocols)?
  - What logic levels are coming in/out of the board?

Microcontroller?

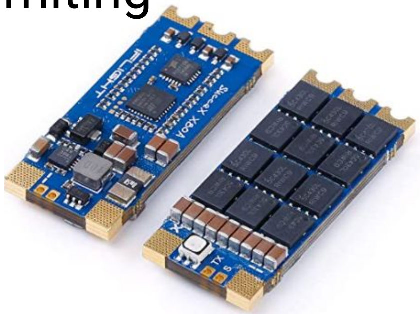
- Use case
  - When you have to decode any complex digital protocols
  - When you have to calculate
  - If the board function has to change in any way
- Specs
  - Peripherals: I2C, SPI, etc
  - CPU, FPU, etc
- Alternatives
  - Application-specific Integrated Circuit (ASIC)
  - Discrete digital or analog circuit
  - Field-Programmable Gate Array (FPGA)

Motors

- Collaboration with mech team
- Choose based on speed, torque, current, efficiency, and voltage

Motor Controller

- Motor Controller directly follows from motor specs
  - Different motor controllers for different types of motors
  - Continuous current and peak current limits
  - There is room for “fudge factor” if there is current limiting software



Types of Motors

Brushed

- Constant DC Voltage
- Cheap and simple to control

Brushless

- Sinusoidal voltage
- Complex control, high efficiency

Stepper

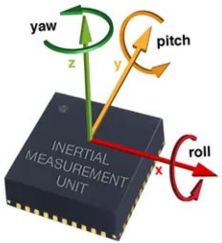
- Alternating voltage pulses
- High torque and holding power

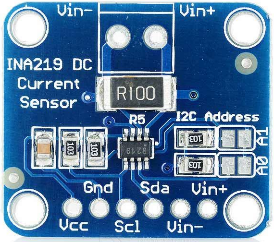
Servo

- Digital input signal
- Precise position or velocity control

Sensors

- Collaborate with software team to choose sensors that gather all required information
  - Based on algorithms/functionality software is aiming to create
- Consider dynamic range, accuracy, sample rate, and communication protocol

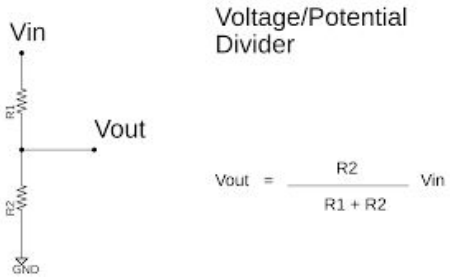




Circuit Planning

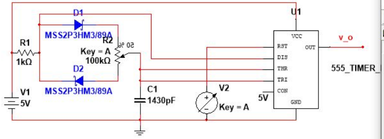
Hand Calculations

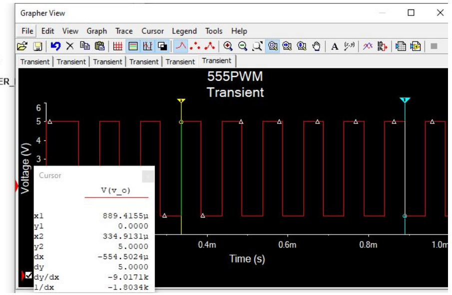
- Data transmission/communication
- Voltage levels
- Power



Simulation

- Multisim (SPICE) for complex analog systems





System Block Diagram

- Figure out required sub-circuits/systems for each aspect of functional requirement
  - Application-specific ICs
  - Communication
  - Power
- Block diagram should show relationships between sub-circuits/systems

Firmware Development

MCU Peripheral Setup → Device Driver → Subsystem Control → Full System Control

MCU Peripheral Setup

Make sure all required MCU features (memory, PWM, Comms, etc) work

Device Driver

Write interface between MCU and other devices on board/robot
Abstract any hardware level calls

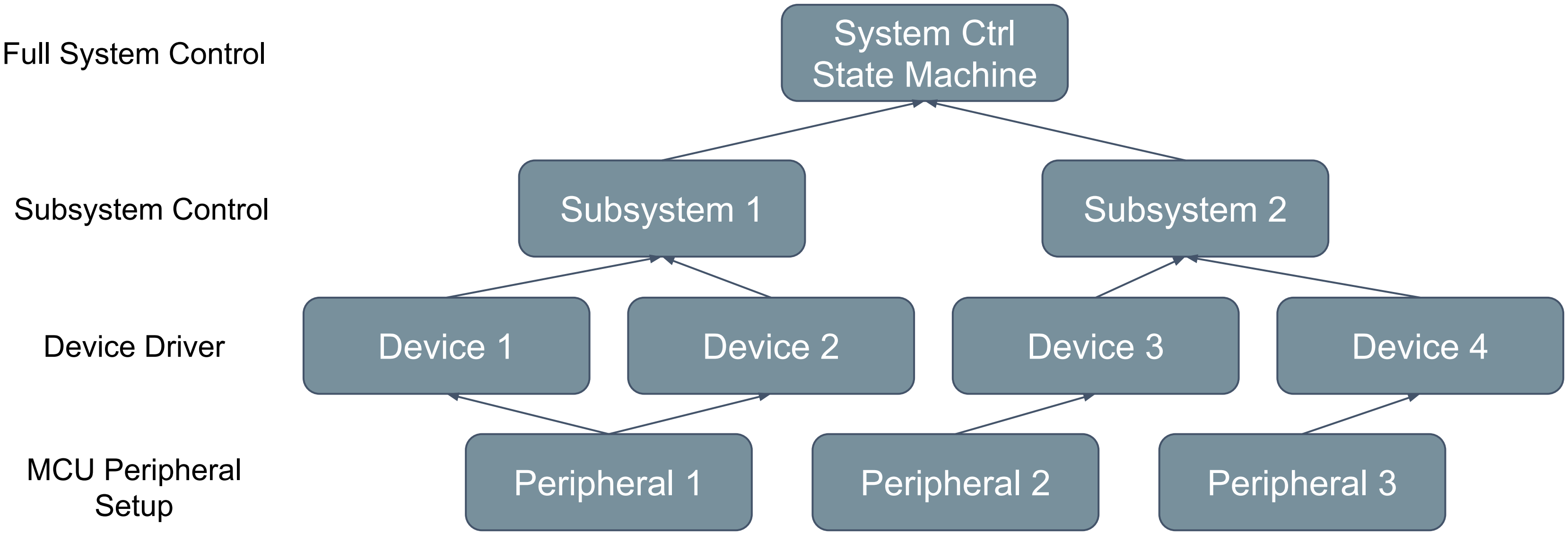
Subsystem Control

Develop robot functionality that integrates multiple devices, e.g. motor control, E-stop, etc

Full System Control

Integrate and test all desired robot functions together - State Machine

Firmware Hierarchy

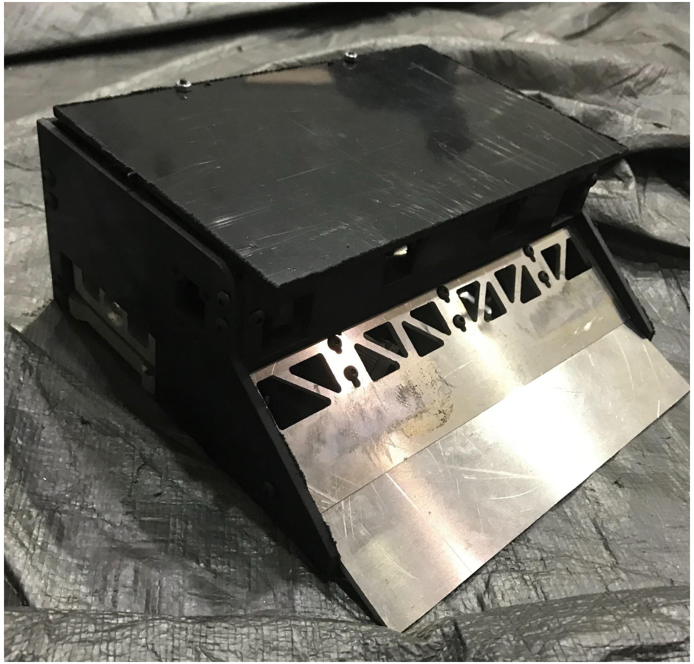


Case Study

Gucciii

What is Gucciii

- 3kg Autonomous Sumo Robot
- Top Speed: 2.8 m/s
- Magnetic Downforce: 415 lbs



Drivetrain

- Maxon DCX 32L Brushed Motor
  - 8270 RPM, 70W max power
  - 72A stall current
- Motor Controller - RoboClaw 60A
  - Max voltage 34VDC
  - 60A continuous, 100A Peak current
  - Communicates with TTL Serial protocol

Sensors

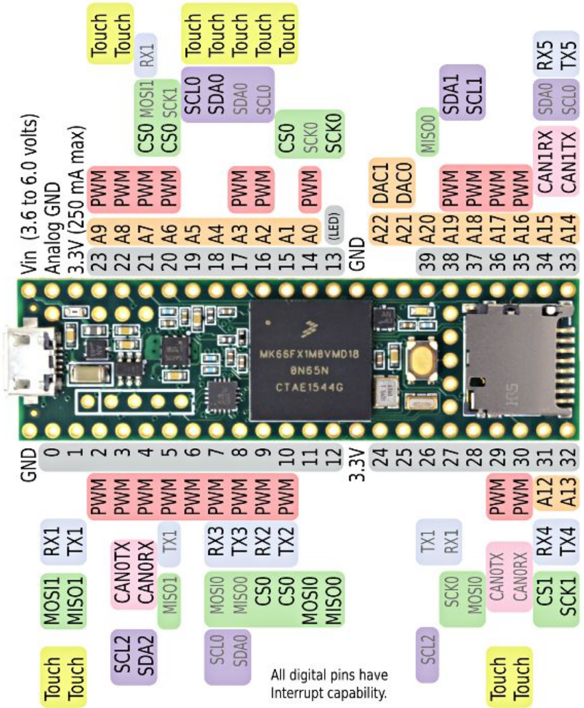
- LiDARs
  - Measure distance to opponent; communicate through UART
- Line Sensors
  - Output analog voltage based on light intensity
- Start Module
  - IR board that receives ready, start, stop commands from a control remote
- Radio
  - Allows for wireless debugging and reporting system status
- IMU
  - Used to detect contact with the opponent

Choosing a LiDAR/ToF Sensor

Parameter	TFMini Plus	VL53L0X
Range	12m	1m
Communication	UART (115200 baud)	I2C (up to 400KHz Clock)
Field of View	3.6 degrees	25 degrees
Refresh Rate	Up to 1kHz	15 or 30Hz
Size	18.5x21x35	17x20x2mm
Cost	$44.95	$14.95

MCU

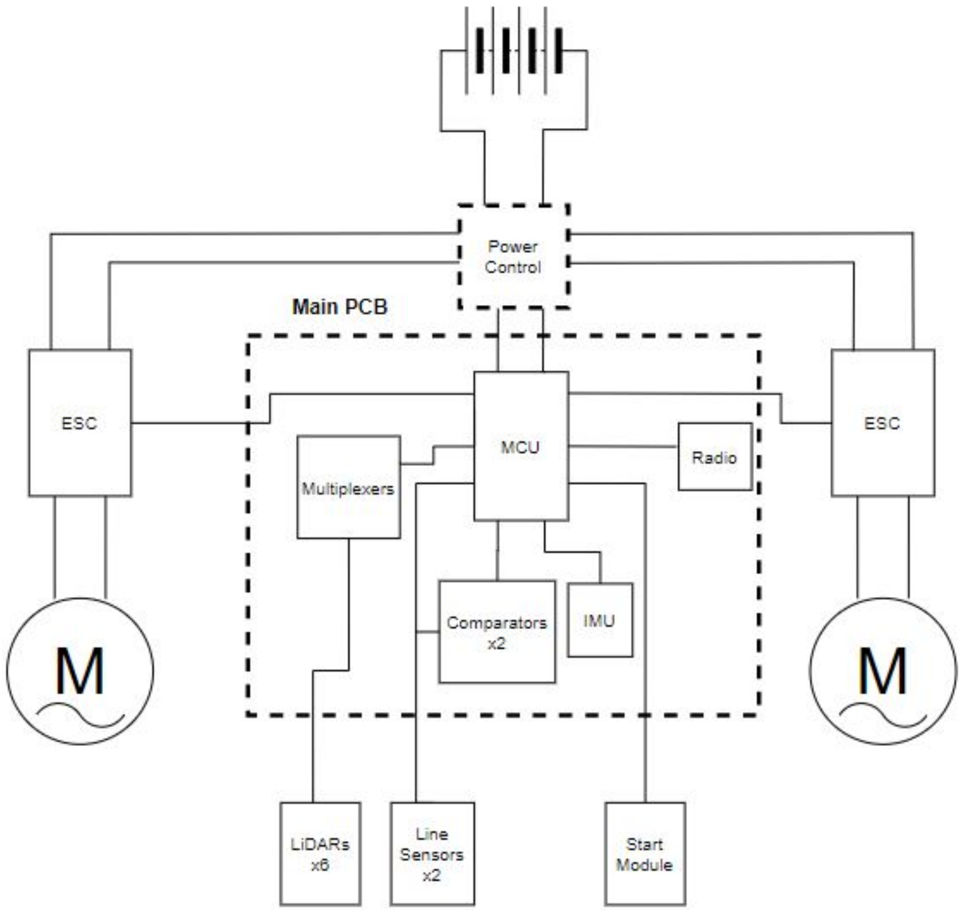
- Teensy 3.6
  - Pros
    - Good Arduino IDE support
    - Has ***a lot*** of pins - flexibility
    - ~~Fastest~~ clock development board
  - Cons
    - Pins not 5V tolerant



Choosing a Microcontroller

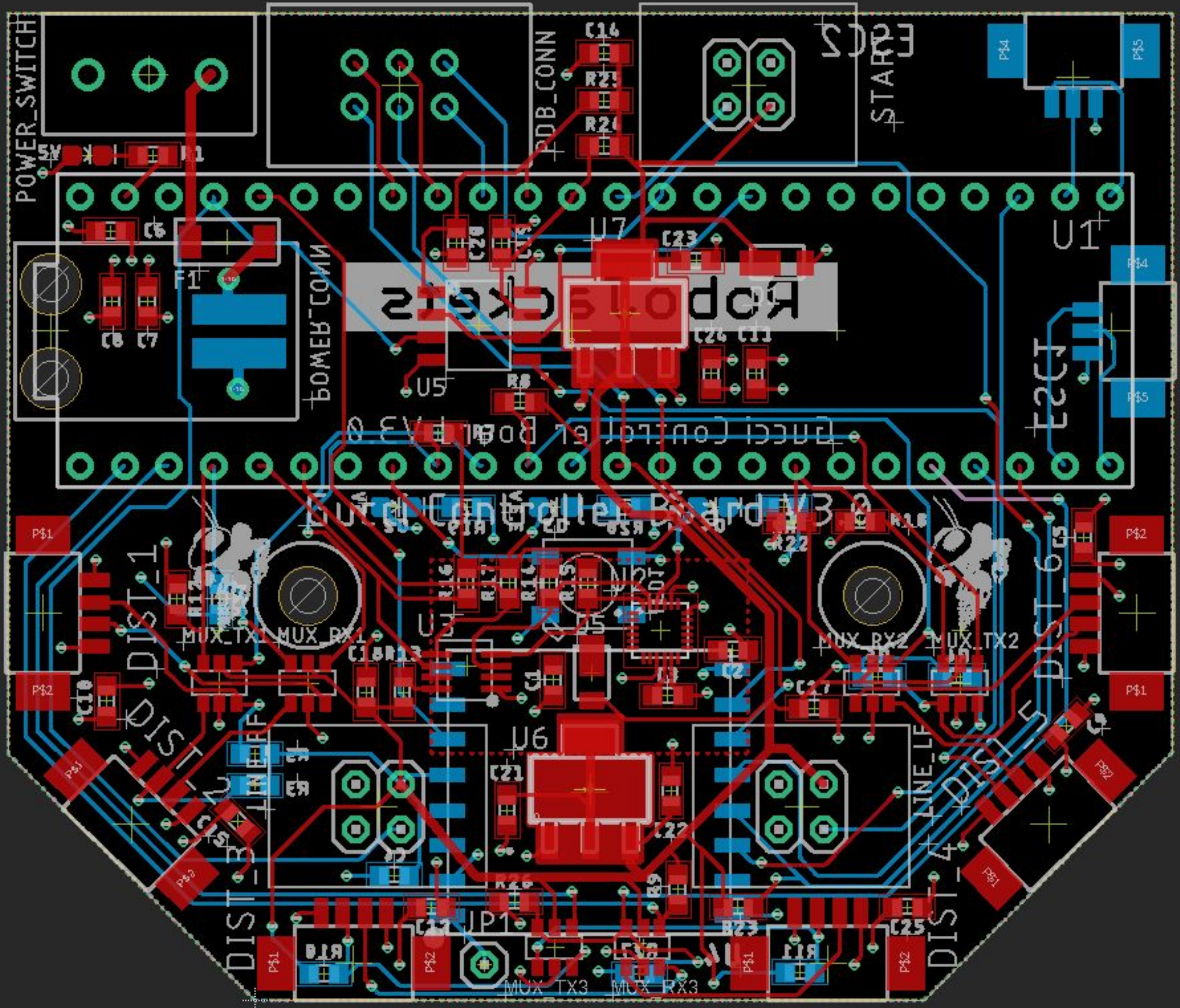
Parameter	Teensy 3.6	Arduino Pro Micro
Processor Type	ARM Cortex M4F (NXP)	AVR ATMega32u4
RAM	256 kB	2.5 kB
Floating Point Unit	Has	Does not have
Pins	48	24
Operating Voltage	5V (3.3V logic)	5V
Communication Ports	5x UART, 2x SPI, 2x I2C	1x UART, 1x SPI, 1x I2C
External Interrupts	44	6

Gucci Block Diagram

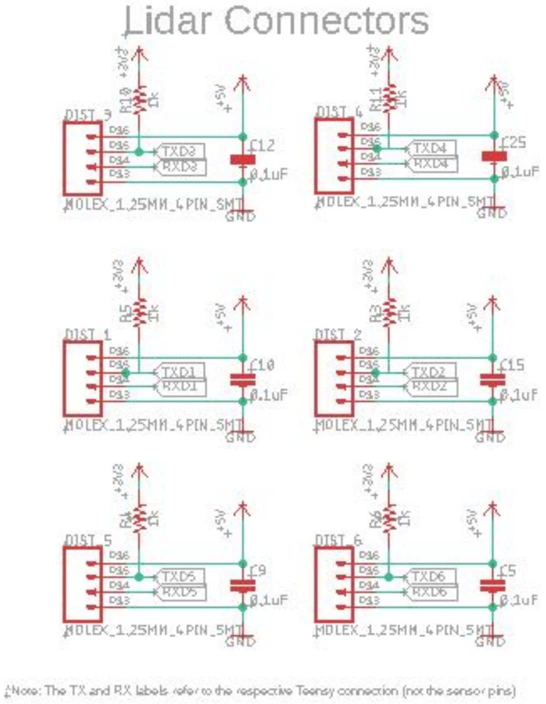


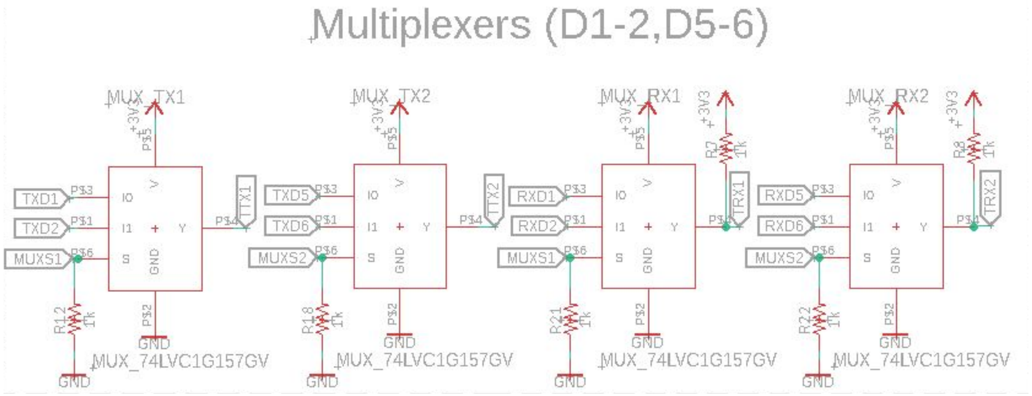
Gucci Control Board

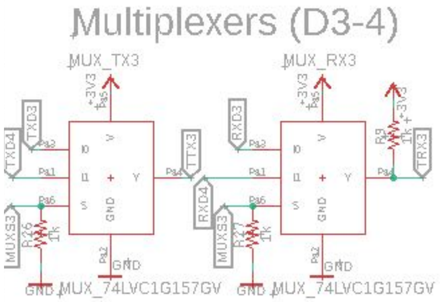
Let’s break it down



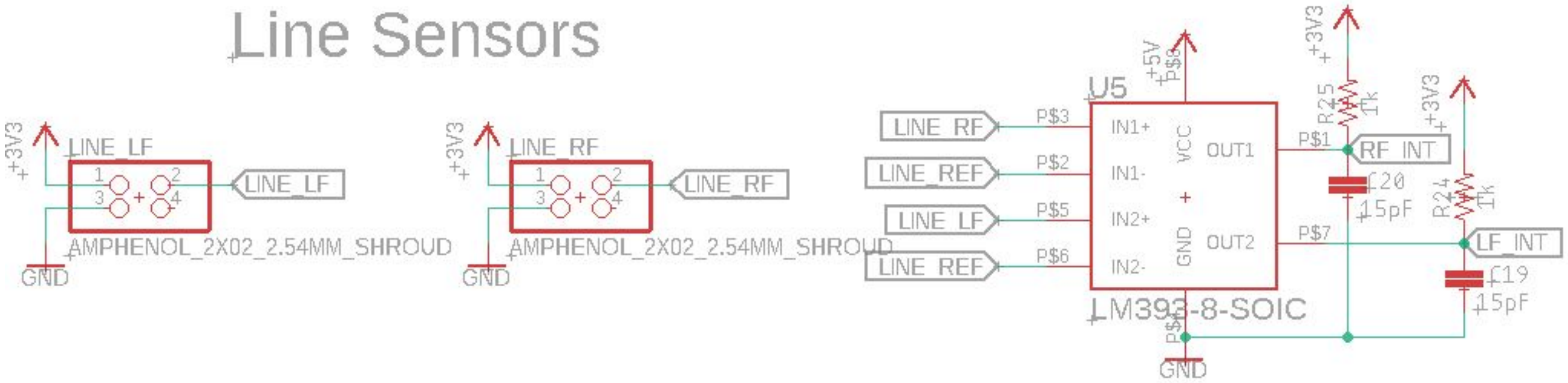
LiDARs


Lidar Connectors
DIST_3
DIST_4
DIST_1
DIST_2
DIST_5
DIST_6
TXD3
RXD3
TXD4
RXD4
TXD1
RXD1
TXD2
RXD2
TXD5
RXD5
TXD6
RXD6
C12
C25
C10
C15
C9
C5
0.1uF
+3V3
+5V
1k
GND
MOLEX_1.25MM_4PIN_SMT
Note: The TX and RX labels refer to the respective Teensy connection (not the sensor pins)

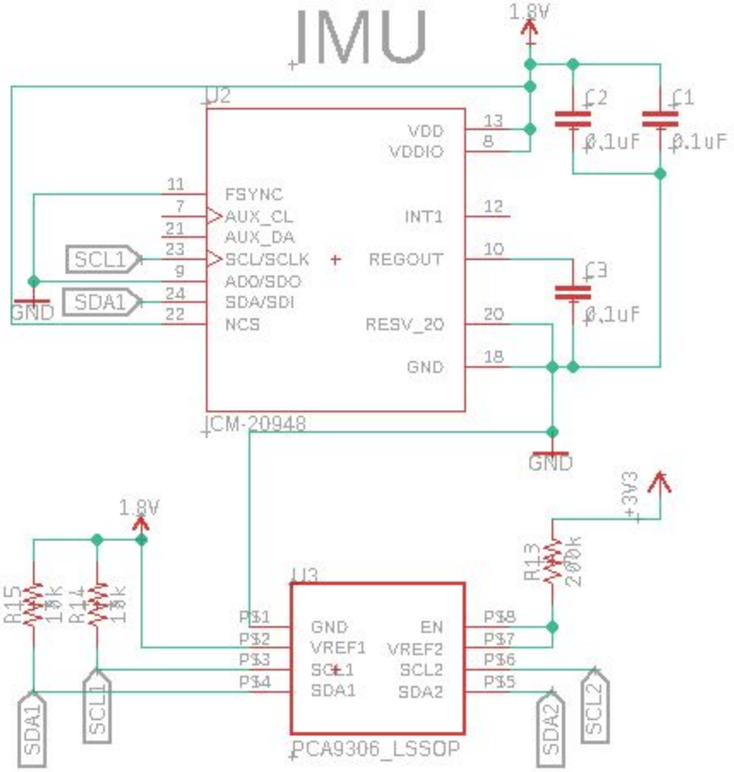

Multiplexers (D1-2,D5-6)
MUX_TX1
MUX_TX2
MUX_RX1
MUX_RX2
TXD1
TXD2
MUXS1
TXD5
TXD6
MUXS2
RXD1
RXD2
RXD5
RXD6
TTX1
TTX2
TRX1
TRX2
R12
R18
R21
R22
R7
R8
1k
+3V3
GND
MUX_74LVC1G157GV


Multiplexers (D3-4)
MUX_TX3
MUX_RX3
TXD3
TXD4
TTX3
RXD3
RXD4
TRX3
MUXS3
R26
R27
R9
1k
+3V3
GND
MUX_74LVC1G157GV

Line Sensors

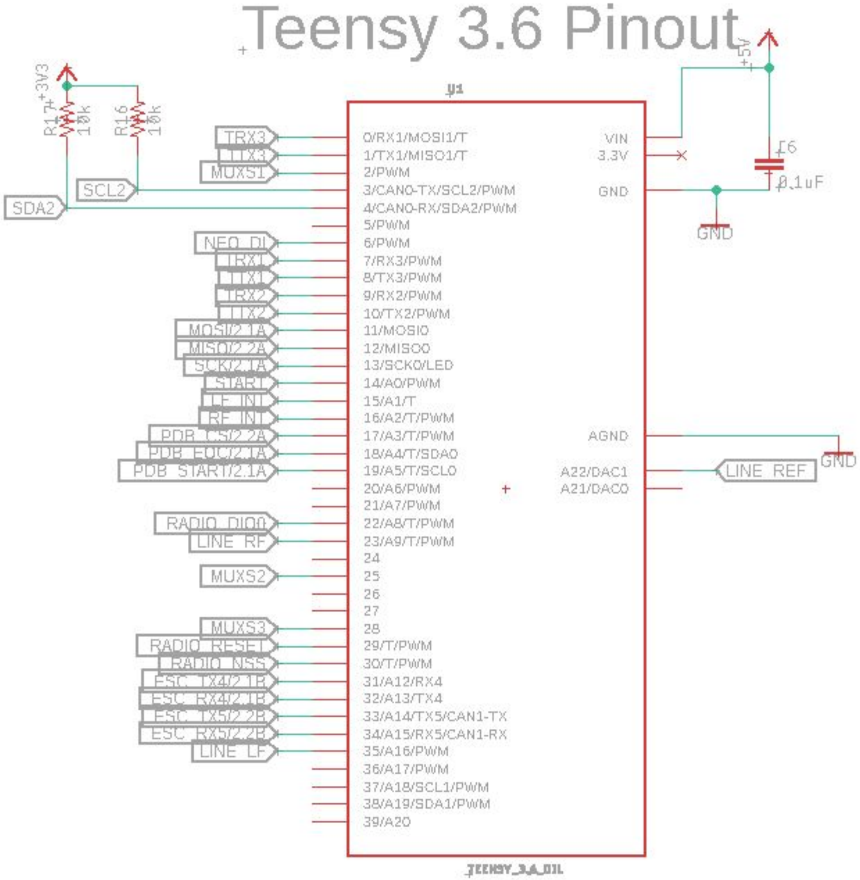


IMU Level-shifting

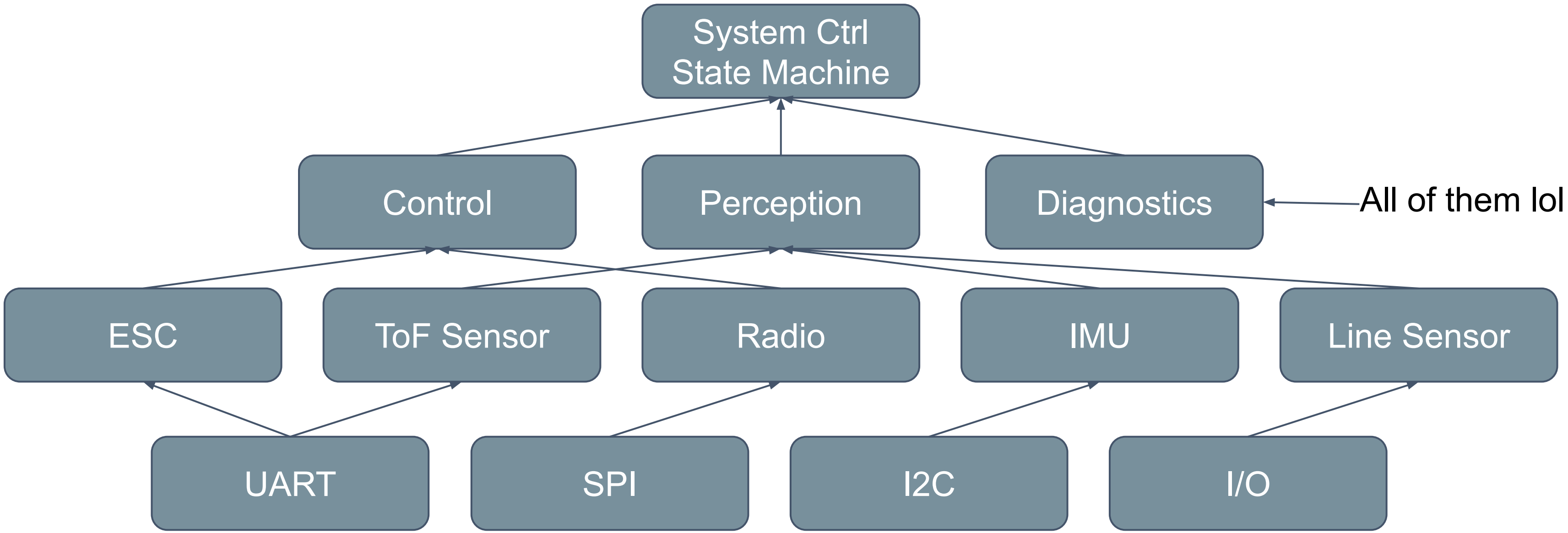


- IMU operates at 1.8V
- Need to communicate at 3.3V with Teensy

Teensy

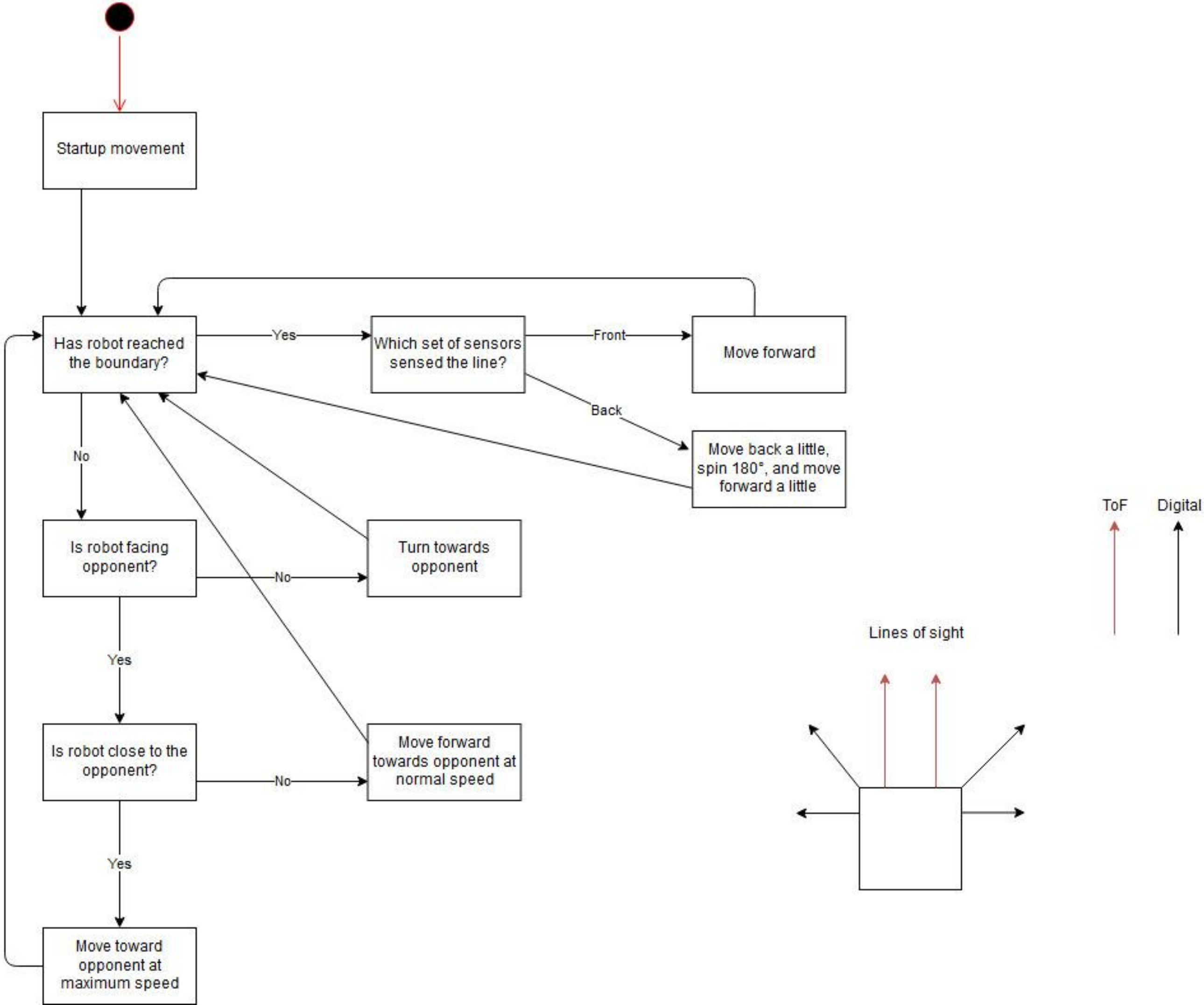

Teensy 3.6 Pinout
U1
+3V3
R17
10k
R16
10k
+5V
C6
0.1uF
GND
TRX3
TTX3
MUXS1
SCL2
SDA2
NEO_DI
TRX1
TTX1
TRX2
TTX2
MOSI/2.1A
MISO/2.2A
SCK/2.1A
START
LF_INT
RF_INT
PDB_CS/2.2A
PDB_EOC/2.1A
PDB_START/2.1A
RADIO_DIO0
LINE_RF
MUXS2
MUXS3
RADIO_RESET
RADIO_NSS
ESC_TX4/2.1B
ESC_RX4/2.1B
ESC_TX5/2.2B
ESC_RX5/2.2B
LINE_LF
0/RX1/MOSI1/T
1/TX1/MISO1/T
2/PWM
3/CAN0-TX/SCL2/PWM
4/CAN0-RX/SDA2/PWM
5/PWM
6/PWM
7/RX3/PWM
8/TX3/PWM
9/RX2/PWM
10/TX2/PWM
11/MOSI0
12/MISO0
13/SCK0/LED
14/A0/PWM
15/A1/T
16/A2/T/PWM
17/A3/T/PWM
18/A4/T/SDA0
19/A5/T/SCL0
20/A6/PWM
21/A7/PWM
22/A8/T/PWM
23/A9/T/PWM
24
25
26
27
28
29/T/PWM
30/T/PWM
31/A12/RX4
32/A13/TX4
33/A14/TX5/CAN1-TX
34/A15/RX5/CAN1-RX
35/A16/PWM
36/A17/PWM
37/A18/SCL1/PWM
38/A19/SDA1/PWM
39/A20
VIN
3.3V
GND
AGND
A22/DAC1
A21/DAC0
LINE_REF
GND
TEENSY_3.6_DIL

Firmware Development



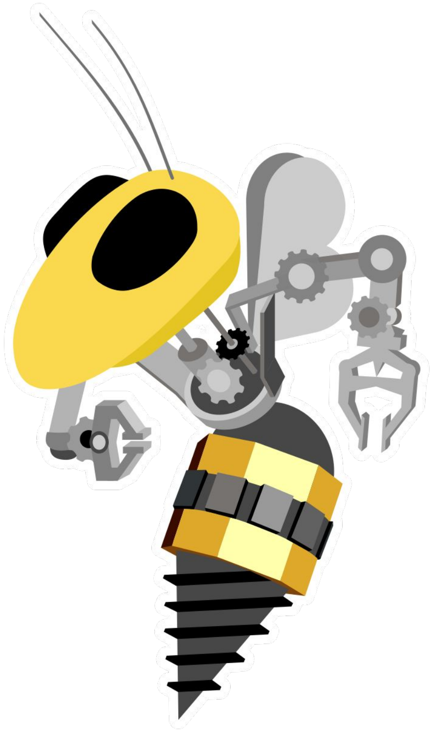
Final State Machine

Or is it?...



In a Nutshell

- Planning planning planning
- Subteam Collaboration is Key
- Incremental Complexity
  - Build it up one subsystem at a time



Feedback Time!

https://forms.gle/CfDN7SfjpWLiAHJE7