MICROMOUSE

what've you gotten yourself into?

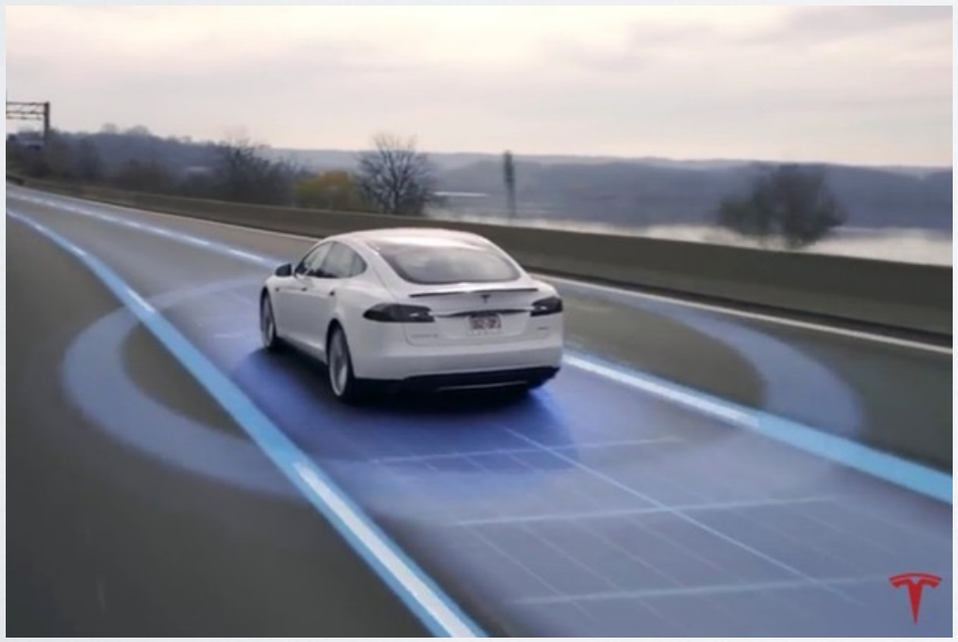
Slides adapted from Dylan Knutson (15-16MM)

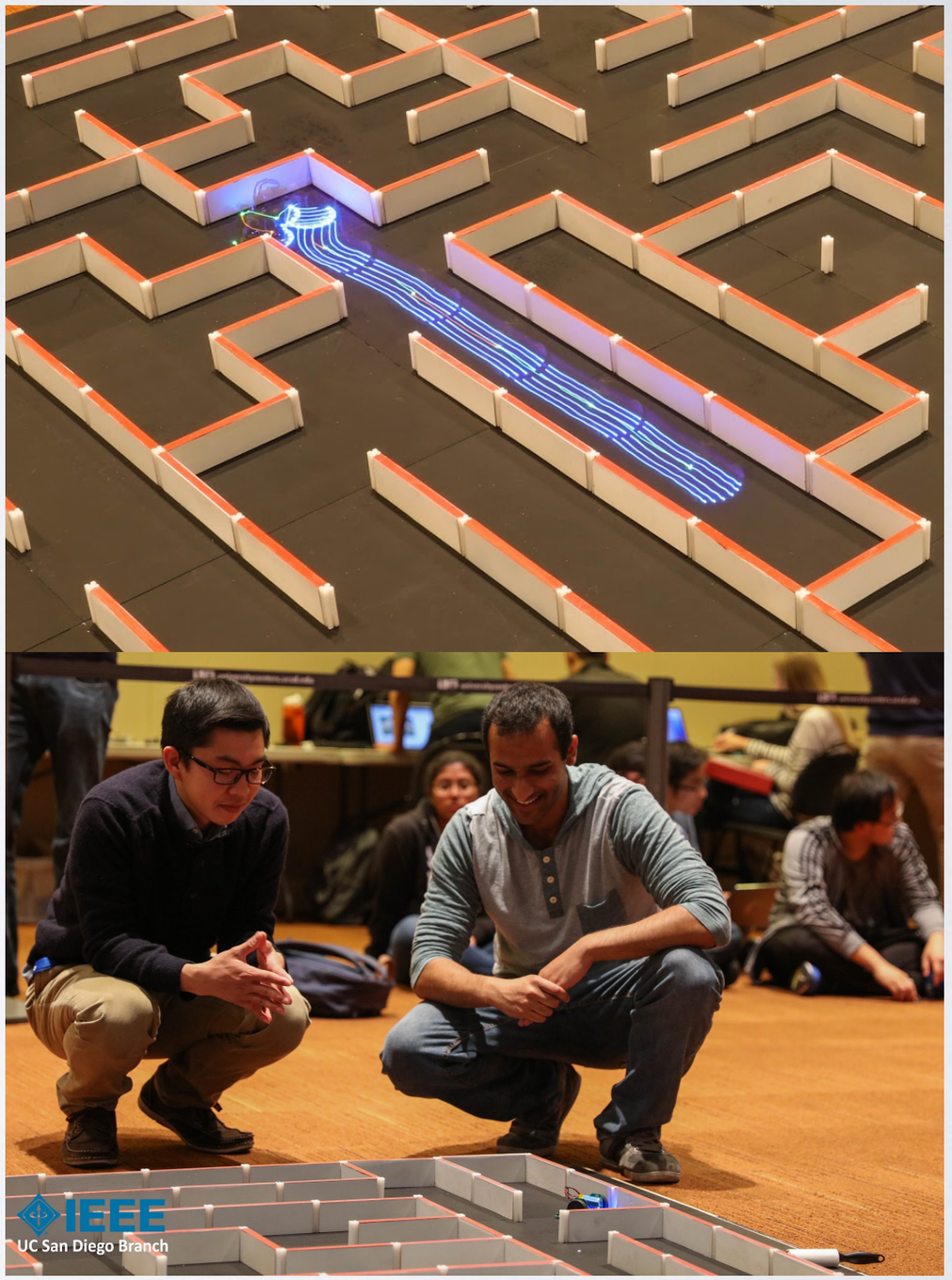
GET AN IEEE NUMBER

- Shoot me an email with it once you've got it
  - matthewyu@ucsd.edu
- Required for Micromouse Project

WHY MICROMOUSE

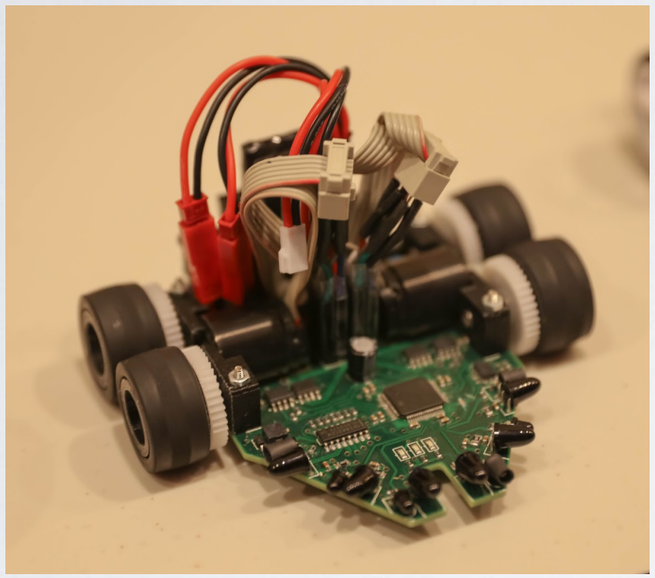
- Apply theories learned in classes to real-world problem like MM
- Internships Gateway
- Autonomous Future

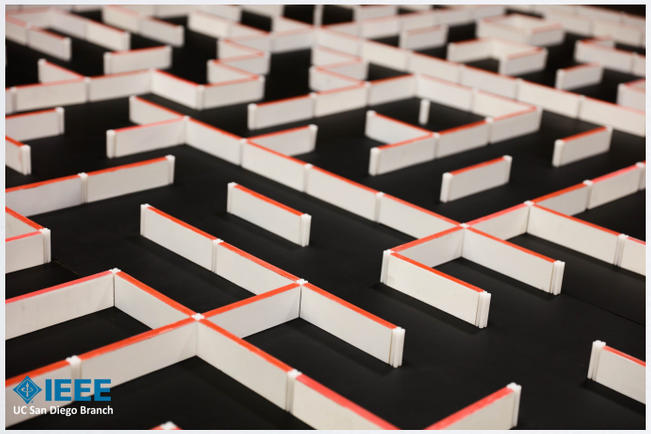


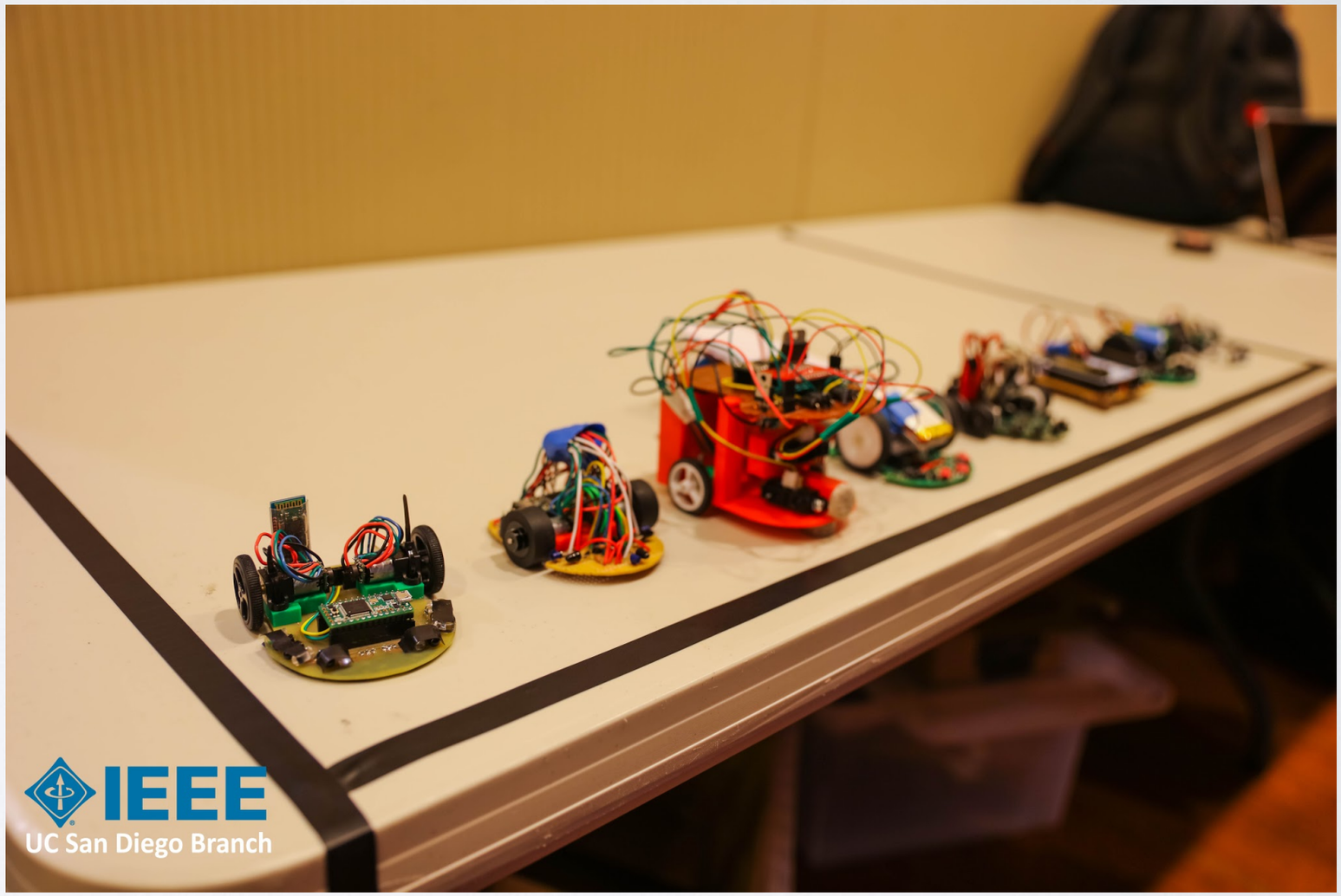


WHAT'S THIS ALL ABOUT

Micromouse is a team project with the goal of making a maze solving robot that's frickin' awesome

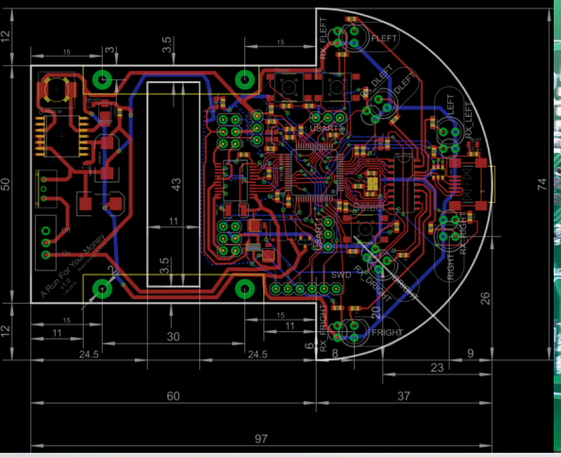
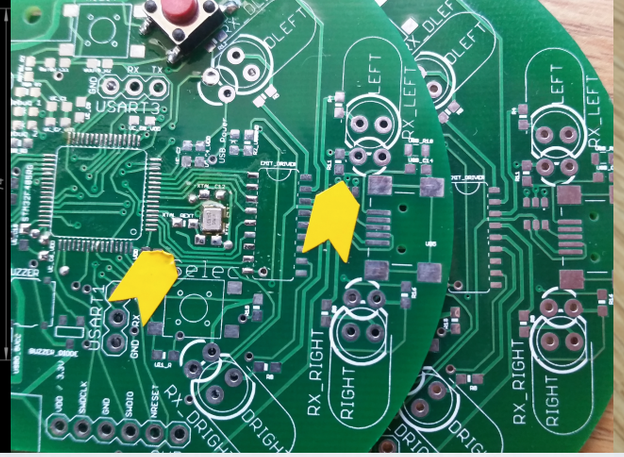


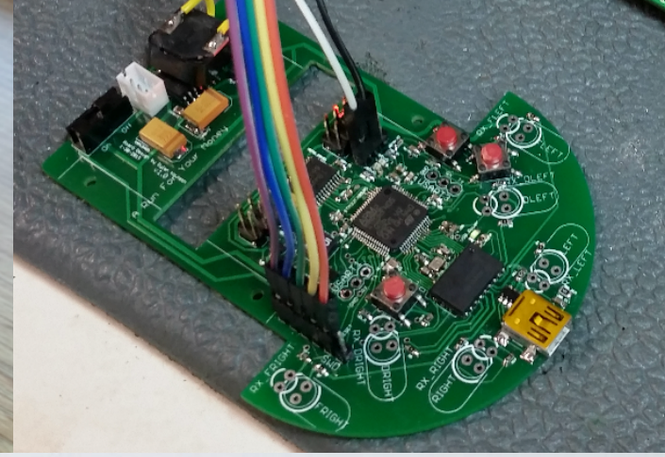




What should you expect….

MOTIVATION: MM EXPLORE



Do not underestimate the amount of time it'll take to build a working autonomous maze solving robot

- 7 months is a long time, but it's not a lot of time
- Schematic Layout, Data Sheets
- PCB Design
- Controller Pseudocode
- Debugging (so much debugging) + Testing
- Getting used to embedded systems engineering
- Mechanical design

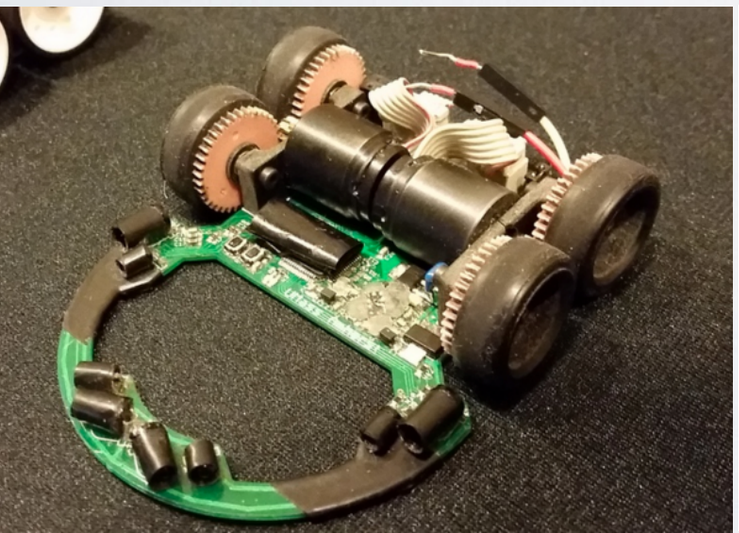
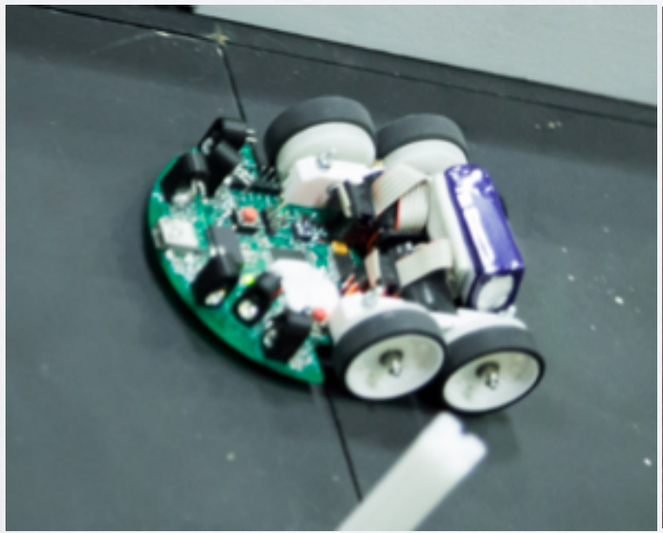
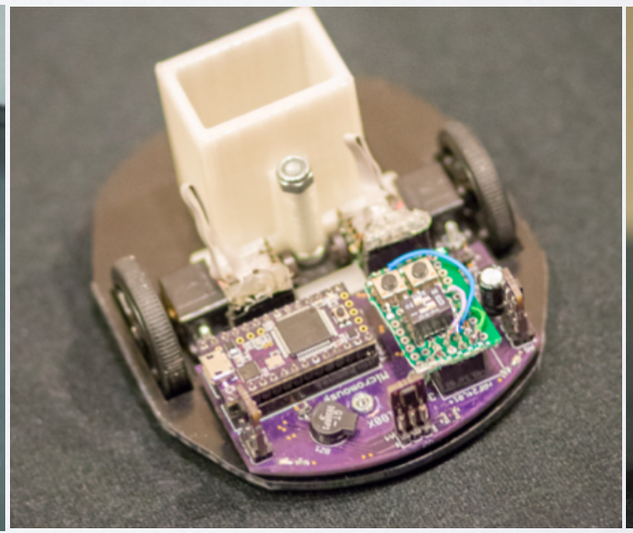
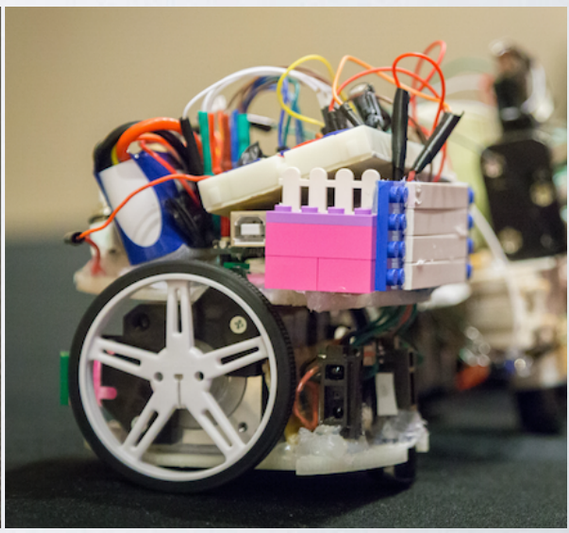
- Movement
  - Motors
  - Motor driver (H-bridge, purpose built driver chips)
  - PWM (microcontroller functionality)
- Navigation
  - Distance sensors
  - Encoders
  - Gyros and Accelerometers
- The Chassis
  - Two big ol' wheels
  - Fast Four wheel design
  - "The Pillar"
- Microcontrollers
  - Arduinos
  - Roll your own STM32F{1,3,4}
  - Teensy

INGREDIENTS

for a happy mouse

MECHANICAL DESIGN

- Smaller competitive mice (the 4-wheeled race car)
- Larger, easier to design mice (the cylinder designs)
- Encoder based motors: Higher resolution, faster
- Steppers: Slow, easier to measure distance

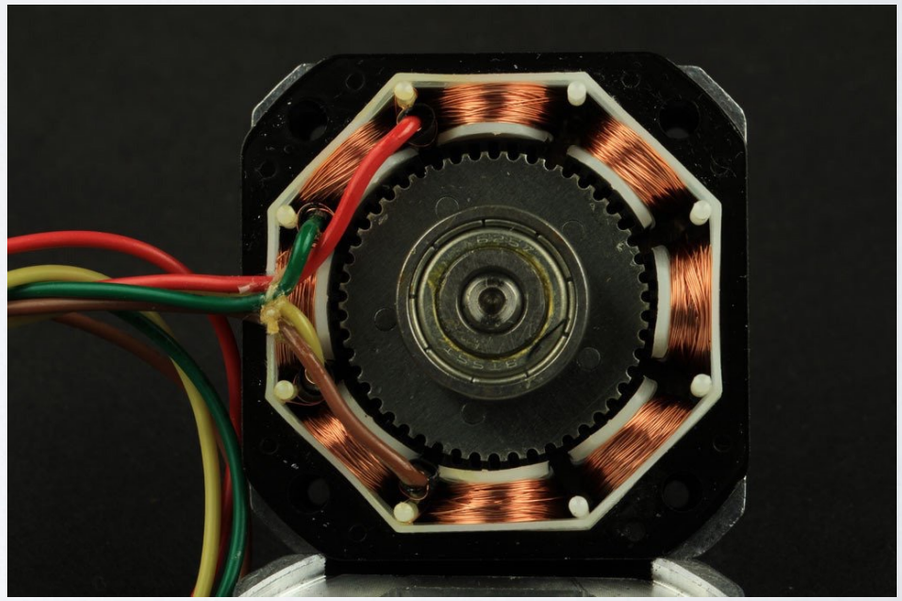





MOTORS

- Brushed DC motors
- Brushless DC motors
- Stepper Motors
- Servos

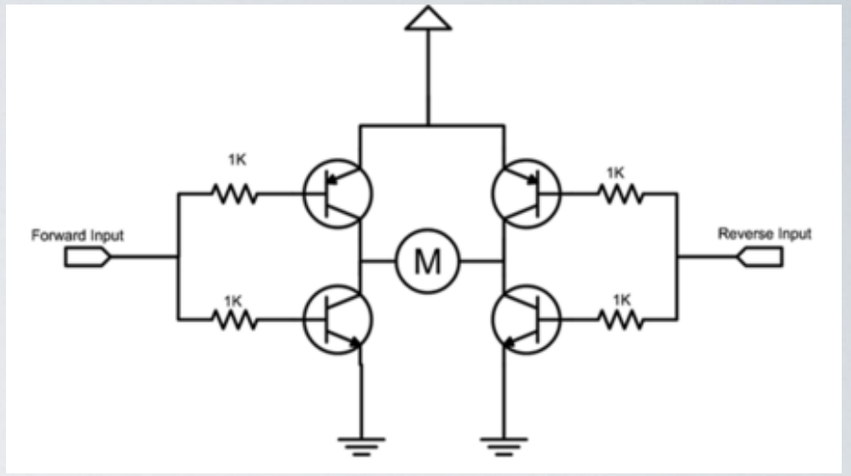
Faulhaber 1717SR Brushless

- Does your motor have encoders built in?
- Power requirements of your motor?
- Fast enough (After gearing down, max RPM high enough?)
- Torque: A motor is no good if it can't move your bot
- Mounting: How will you fit this into your chassis?



Stepper Motor

MOTOR DRIVERS

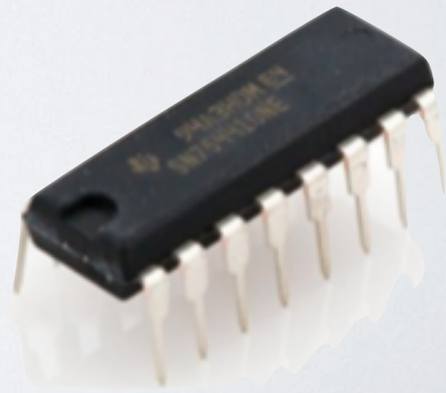


H-Bridge

- Microcontrollers can't source enough current for a motor
- Also, we want analog speed control, and most uC pins are digital
- Have to be able to reverse direction of the motor

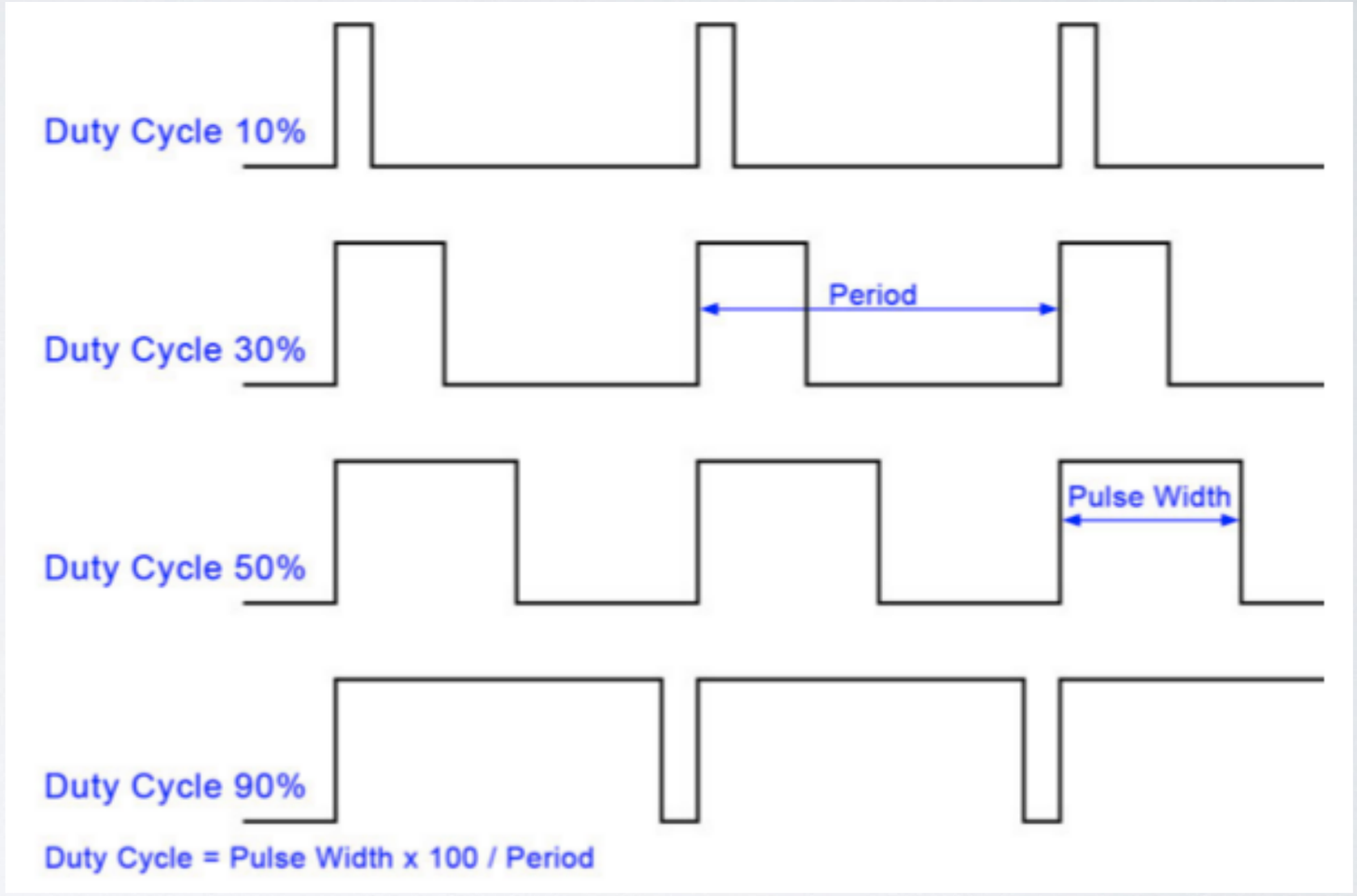
TB6612FNG
Dual Driver

- Solution
  - An H-Bridge motor driver for forward/reverse, and high current throughput
  - PWM from the micro controller to have "analog"-like speed control



TI SN754410NE
(Also a dual driver)

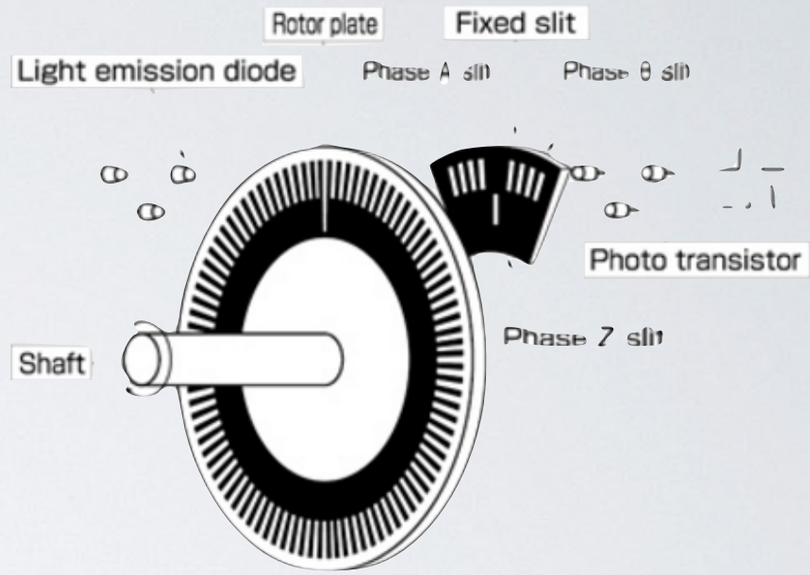
QUICK INTRO TO PWM



- Pulse Width Modulation
- Used to mimic an analog signal
- Many micro controllers have dedicated hardware to do this

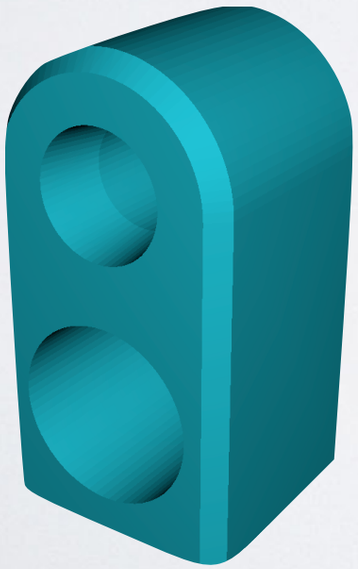
ENCODERS

- Optical or magnetic
- Keeps track of how far the motor shaft has turned
- Good approximation of how far you've gone
- 1 vs 2 channel
  - 1 channel: Only know number of ticks
  - 2 channel: Know number of ticks, and in which direction

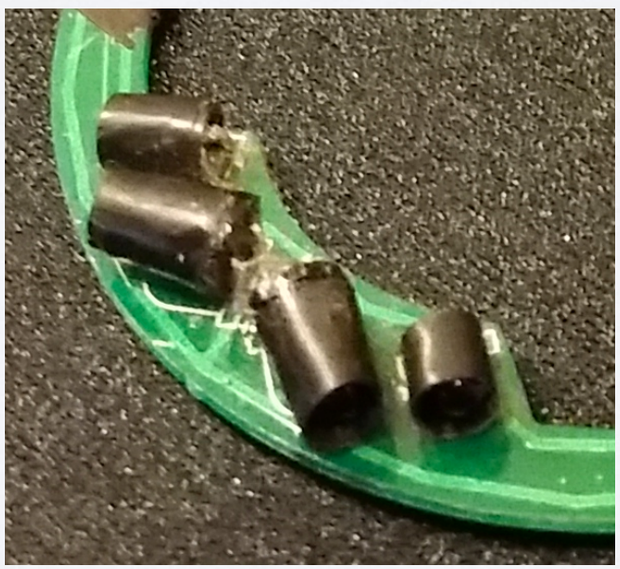


SENSORS

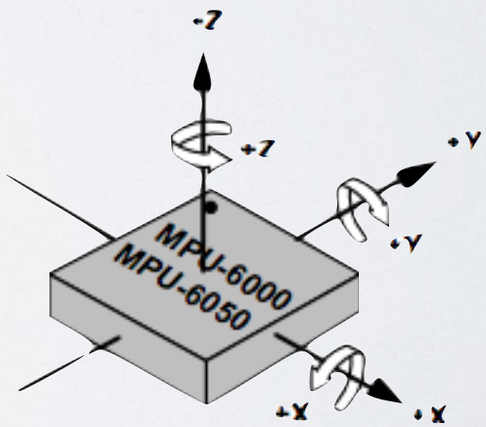
- Almost all use Infrared sensors
- Signal strength translates to object distance
- Gyroscope: High accuracy rotational direction
- Quadrature (dual channel) or single channel encoders
- Accelerometer: Colliery to using Δdistance to calculate speed/acceleration



IR Sensor Mount



2 Sensor Pairs



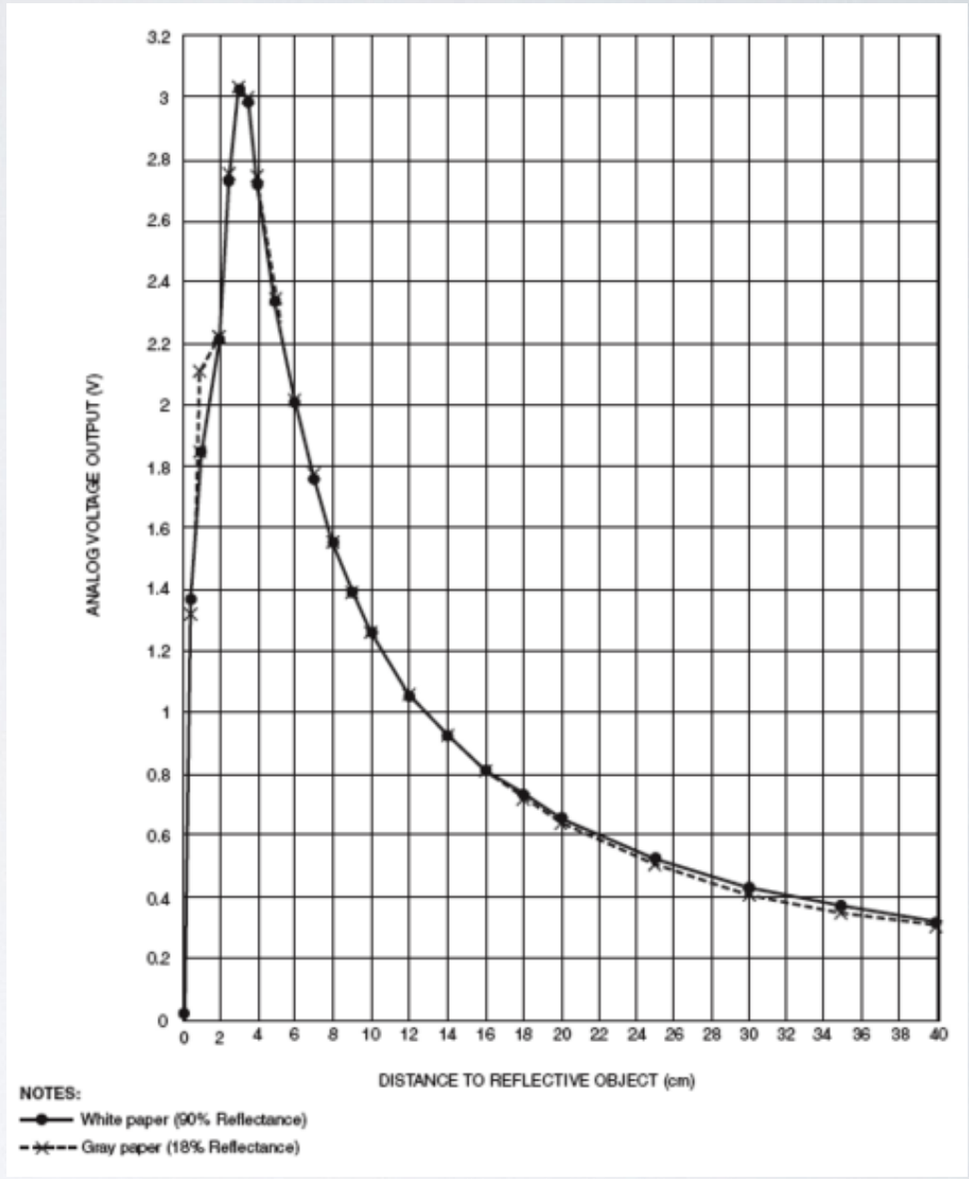
MPU-6050 Gyro+Accel

OTHER NAVIGATION

- **Accelerometer**: High precision reading of current acceleration
  - Can calculate current velocity via integration
  - Wheels can slip, so this might be a better "source of truth" than encoders when accelerating quickly
- **Gyro**: Know to a high degree of certainty the heading of the mouse
  - More wheel slippage might have you facing the wrong direction
  - Useful when turning, can get closer to exactly 90 degree turns
- Both of these devices are harder to deal with in software
- Also typically much harder to solder onto boards (have a steady hand or a reflow oven)

SENSORS

- Sensor signal averaging
- Linearize the sensor readings
- Equate the analog signal to a distance

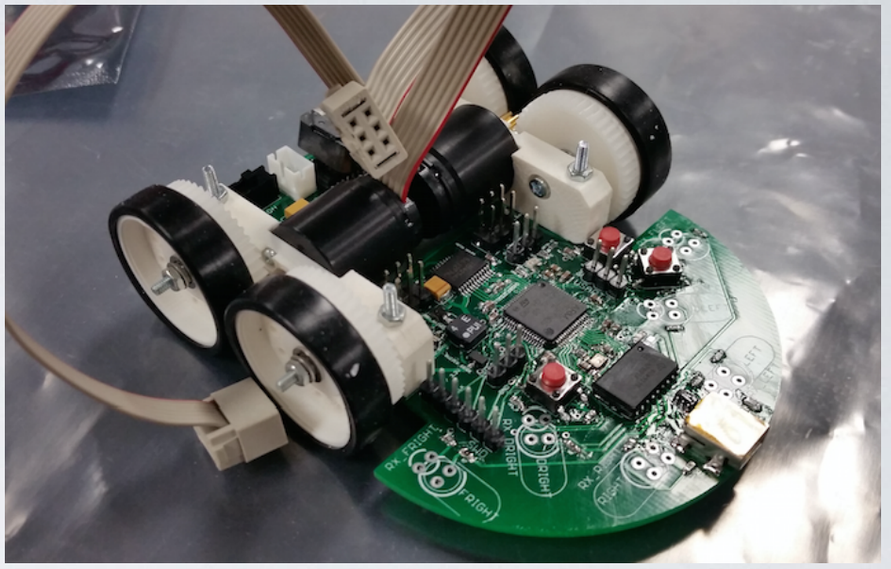


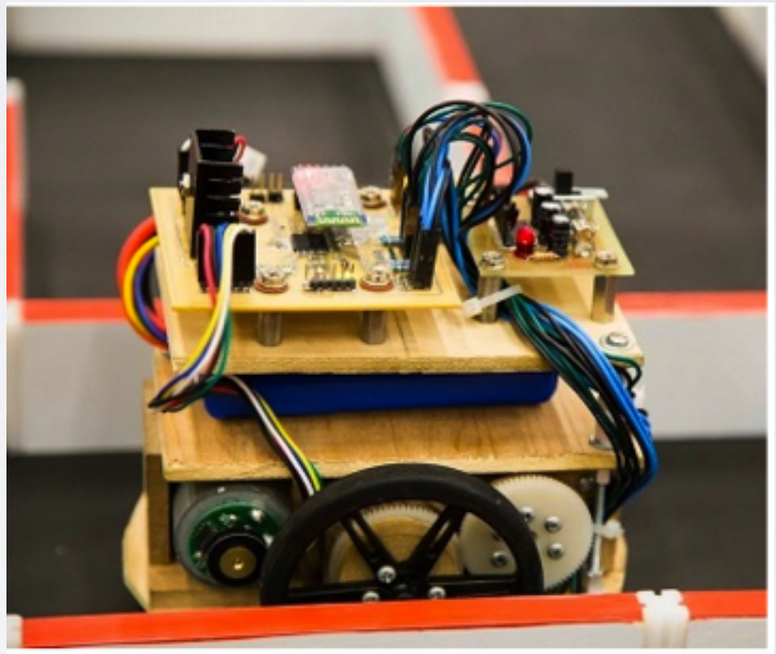
BATTERIES

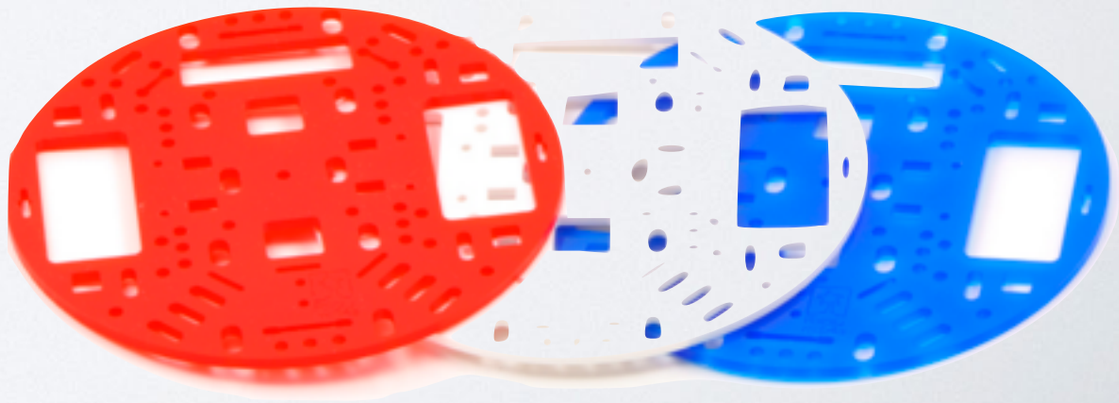
- http://micromouseusa.com/?p=1002
- LiPo is the popular choice
  - High current draw, abundant, nice voltages
- Ensure the mouse can run ~10 minutes on a battery
  - Or, get multiple batteries
- **Always, *always*, have an extra**

CHASSIS

- Ideally
  - Small
  - Lightweight
  - Easy to work wit
- Options
  - 3D Printed Chassis
  - Metal
  - Wood
  - PCB

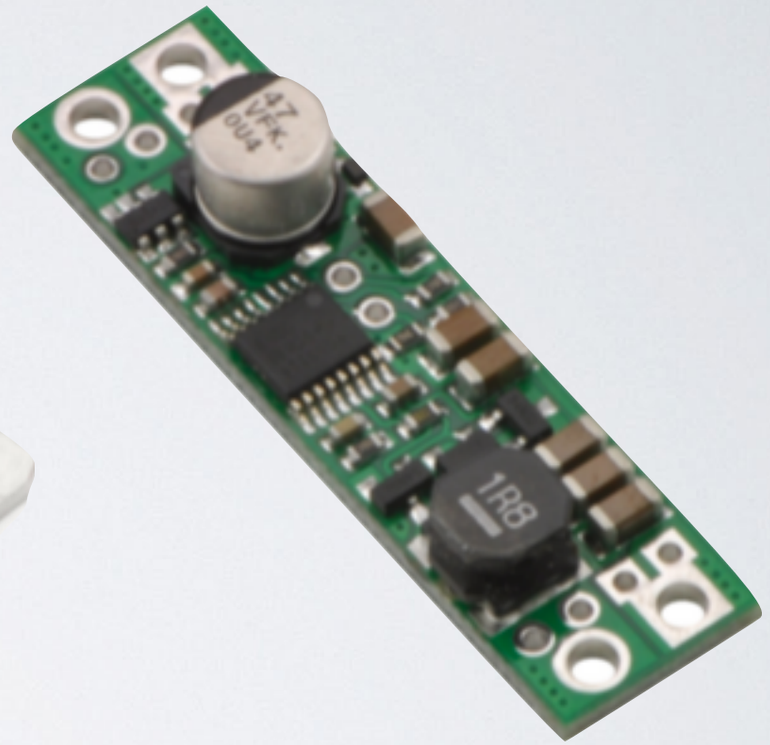


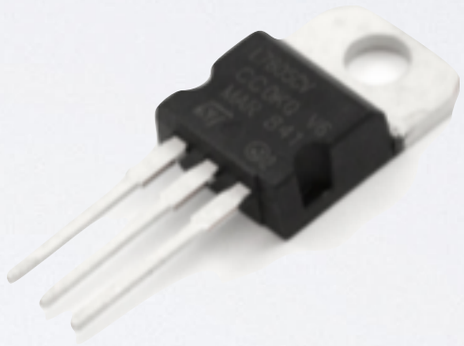


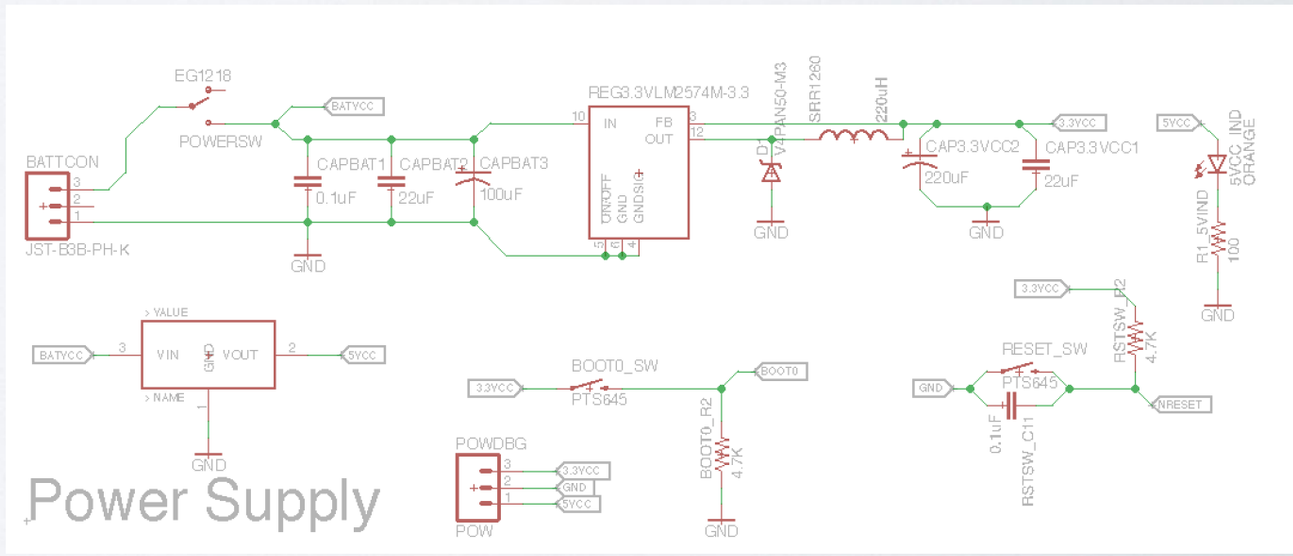


POWER SUPPLY

- A voltage divider is not enough
- V-Regs provide consistent power to the MCU, sensors, drivers
- Ensure the V-Reg can supply enough current to the mouse, minus the motors
  - (Motors typically draw directly from the battery through the driver chip or h-bridge)





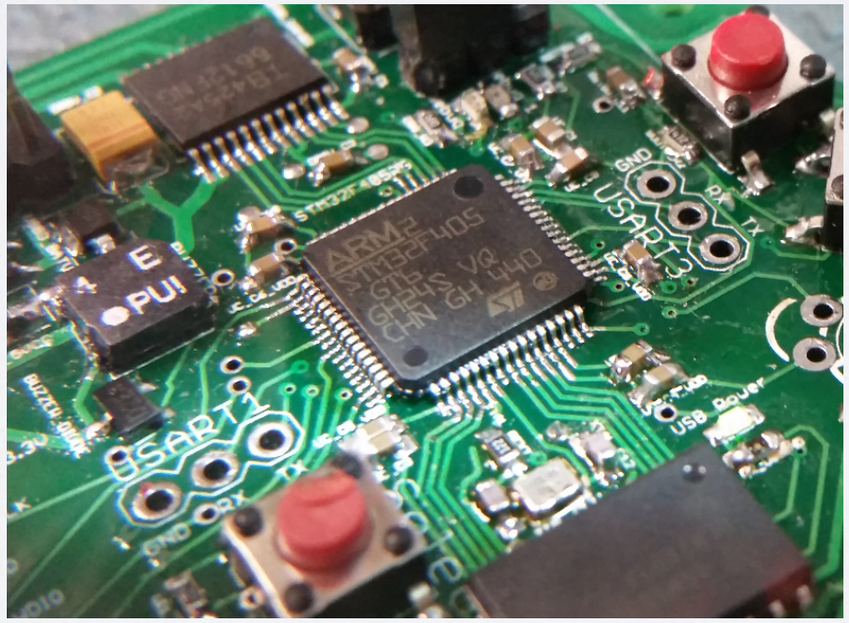


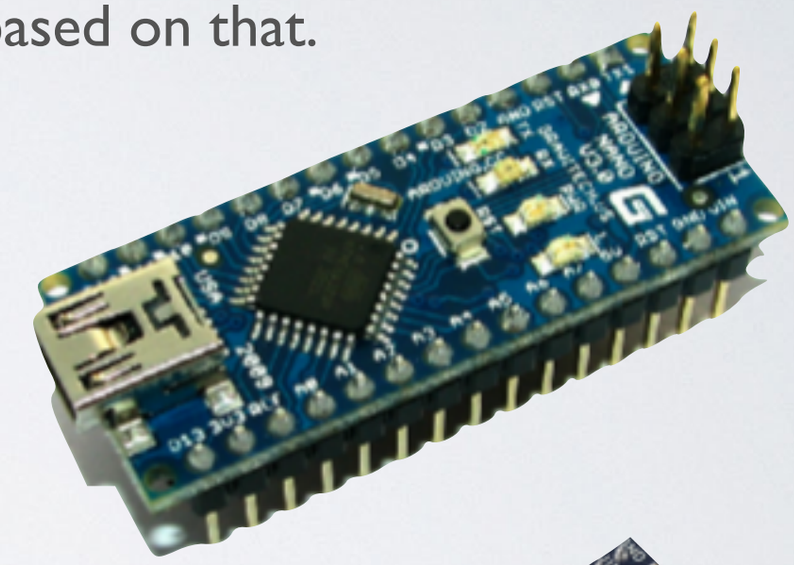
MICROCONTROLLERS

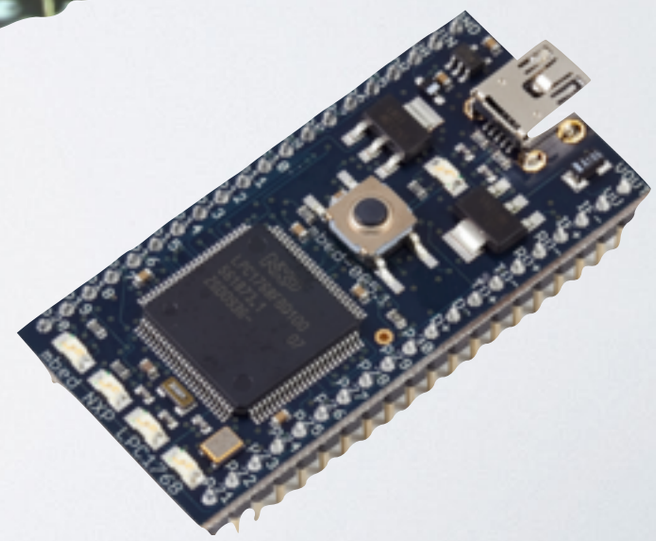
This mini-computer is the brain of the mouse. Nearly all components feed straight into a pin on the micro controller, which means mouse schematics aren't insanely complex. The uC reads and records sensory data, and moves the mouse based on that.

- Arduino
- ATMegas
- STM32F{1,3,4}
- Teensy

- PIC
- MBED





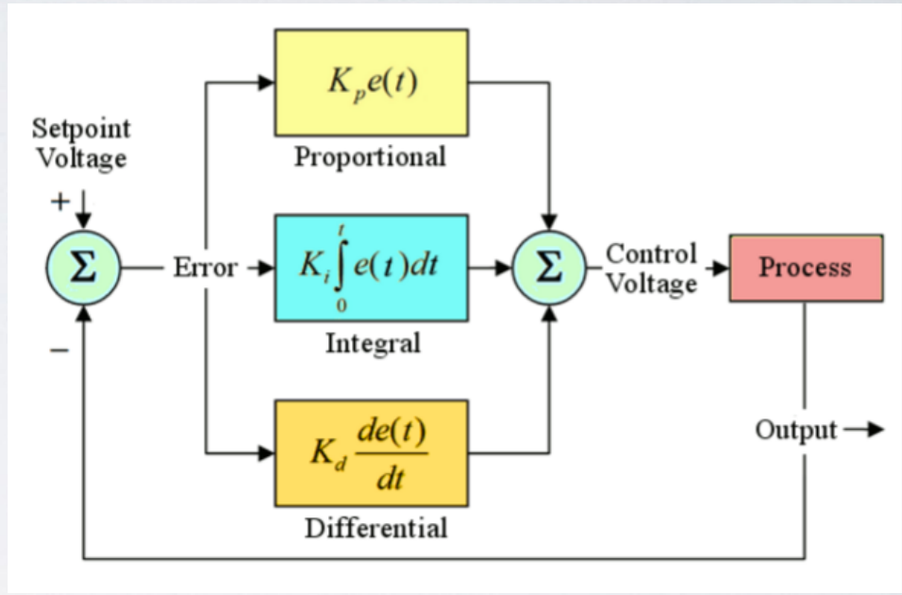


SELECTING A MICROCONTROLLER

- **I/O:** Does it have enough inputs and outputs for my needs. Are they analog or digital pins? PWM outputs? Quadrature encoder inputs?
- **Speed:** How fast is it going to process the information (Typical controllers range from 16Mhz to168Mhz)?
- **Memory:** Do I have enough memory for my algorithm and to store the map? (Typical, 32-128KB)
- **Toolchain**: What software do I need to program it? (IDEs, cross compilers)
- **Support:** How easy is it to find documentation on the controller? Are there library files for common tasks?

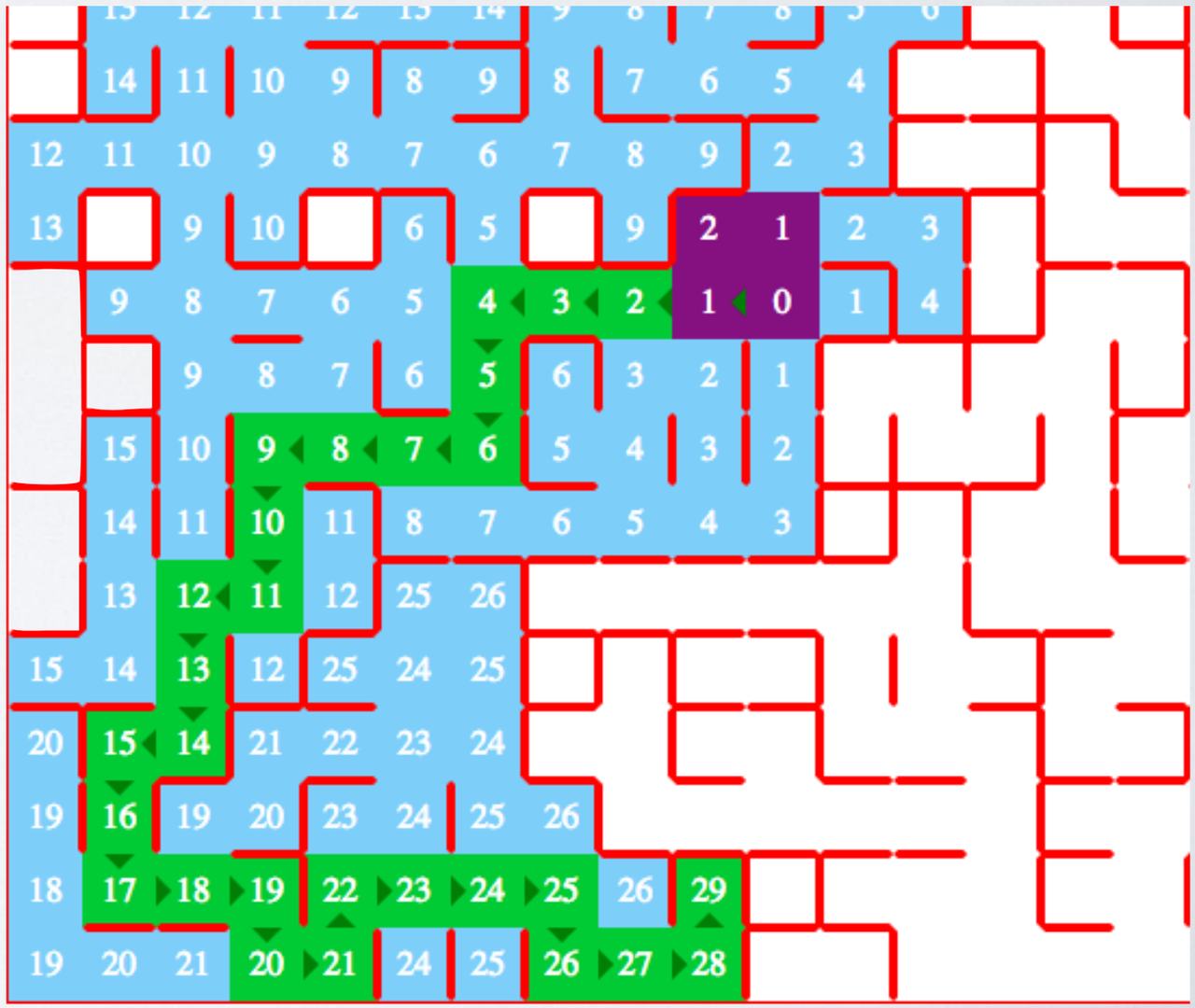
SOFTWARE

- PID: Keeping your mouse going in a straight line
  - (if you have sensor data)
- Sensor sampling
  - How often?
  - How do I decide if there's a wall or not?
  - Integrating accelerometer or gyro data?
- Maze navigation
  - Keeping track of where the mouse is
  - Deciding on which cell to go to next
  - Finding the shortest path to a cell
  - How much of the map have I explored?



MAPPING ALGORITHMS

- Floodfill (Only finds routes between cells, very common)
- Wall Follower
- Termaux's Algorithm
- Gaston Tarry Method
- Standard BFS

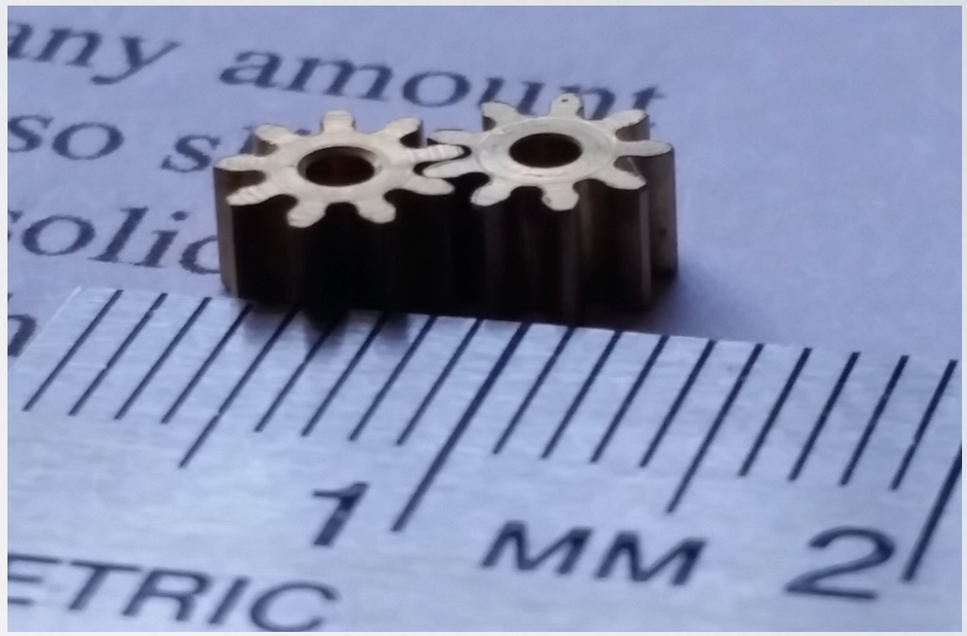


END RESULT

Green Giant 5.10V Run at 2016 California Micromouse Competition

SOURCING PARTS

- 3D Printers on Campus
  - Digital Media Lab @Geisel
  - Price Center's Imprints
  - UPS Store
  - Anyone taking a ME class (probably)
- Electronics
  - Digi-Key
  - Mouser
  - Sparkfun



- Gears, Li-Po Batteries
  - homefly.com
  - venompower.com
  - AliExpress, Ebay
  - KDP Gears

CHECKPOINT/DELIVERABLES TIMELINE

week	Fall	winter	spring
1		(Checkpoint) Board Layout	MM Competition @ UCR
2	Sign up begins / IEEE GBM	(Checkpoint) Boards submitted to shop	(Checkpoint) Mouse can do basic maze mapping
3	Sign up ends	(Waiting for boards) (Checkpoint) Mechanical parts designed	
4	MM Intro Workshop	(Waiting for boards) (Checkpoint) Floodfill Pseudocode (Checkpoint) Mechanical parts purchased/ printed	AAMC Competition @ UCLA
5		(Boards arrived, begin assembly)	(Tentative: Competition at other University)
6	(Checkpoint) Block diagram of micromouse	(Checkpoint) Board assembled	(Tentative: Competition at other University)
7	(Checkpoint) PID Pseudocode for chosen sensors	(Checkpoint) Flashing code to PCB	CAMM Competition @ UCSD Mandatory for ALL TEAMS
8		(PCB revisions) (Checkpoint) Working Demo	
9	(Checkpoint) Schematic		
10			

Deadlines for teams listed in the schedule are due at the end of that week, Friday, at midnight.

For instance, for week 6, the Micromouse Block Diagram would be due Friday, Nov. 4th.

USEFUL RESOURCES

- micromouseusa.com (Green Ye's site, MM God)
- micromouseonline.com
- Sparkfun EagleCad Tutorial: https://learn.sparkfun.com/tutorials/using-eagle-schematic
- My past MM resources: https://drive.google.com/open?id=0B2tnA6hzalcZQXNUUzBKWE4zY2M

USEFUL RESOURCES CONT

- $500 Budget (IEEE)
- Maker Space (http://jacobsschool.ucsd.edu/envision/access.shtml)
- IEEE Project Space (EBU1-4309)
- Facebook Group

Any questions?

GET TO KNOW YOUR TEAM

Team 1	Team 2	Team 3	Team 4
Javier Urquizu	**Jessica Chen**	**Nathan Apolonio**	**Moulik Solanki**
Eddie Calva	Farimah Pirahmadi	**Alex Ng**	Josh Kim
Christopher Lew	Sandeep Raghunandhan	Aykan Fonseca	Erick Soto
	Haoyu Dong	Michael Lindarto	Tiffany Pan
		Hongtao Zhang	

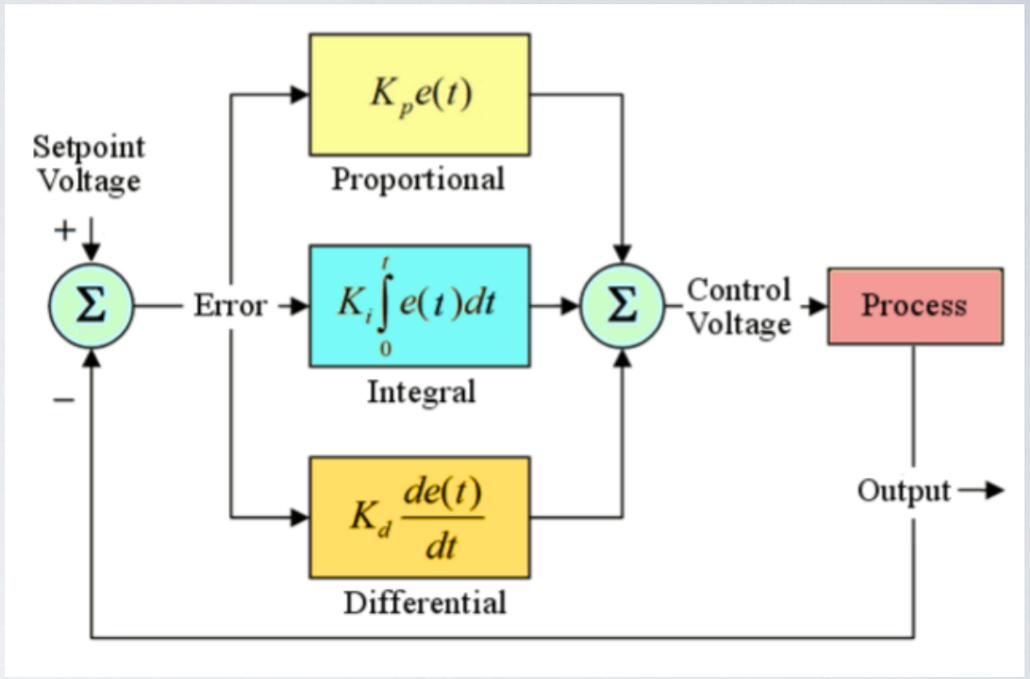
Team 5	Team 6	Team 7	Team 8
Fengjunyan Li	**Nguyen Bui**	**Jeffrey Yeung**	**Anh Mai**
Hongda Xiao	Austin Hermida	Steven Apodaca	Lisa Chang
Haoran Wang	Jocelyn Tran	Gates Zeng	David Chang
Zhouhang Shao	Achint Singh	Luke Thomas	Tara Shamblin
Haocheng Liang	Andy Zhang		Jiaxiao Zhou

NEXT STEPS

- Be prepared to meet next week with your team
- Send a Doodle/WhenIsGood for team schedule
- Exchange Contact Info/Facebook

BACKUP SLIDES

PID



- Proportional Integral Derivative
- Keeps your mouse going in a straight line
- Mice usually only use the P and the I
- Both are multiplied by a constant (P and I constants)
- Integral is an accumulation of how far off you've been
- Derivative is how fast the error is changing
- Useful: Do PID based on velocity, not position
  - Lets you change velocity at a steady pace to control acceleration
  - Set an explicit velocity for the mouse to travel, instead of having P and I constants determine the speed of the mouse