CS8 – Basic Robot Electronics

Why do programmers need to understand electronics?

- Object Oriented Programming languages represent real world objects
- These real world objects have characteristics and behaviors that we need to “interact” with in code
- Without an understanding of what a “thing” is and does, we can’t expect to be able to write software to control that thing

RoboRIO/RoboRIO 2.0

- The brain of the robot
- Uses a Real Time Operating System (NI Real-Time Linux)
- Has Underwhelming Specs

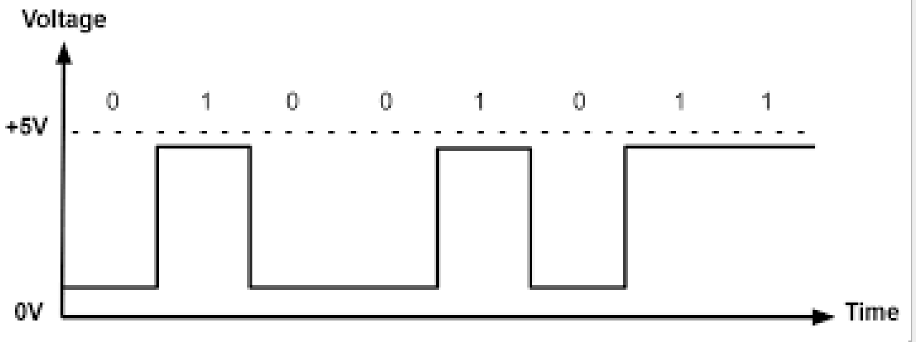
Aspect	RoboRIO	RoboRIO 2.0
CPU	667 MHz x2	866MHz x2
RAM	256MB	512MB
Storage	512MB	16MB + uSD

RoboRIO/RoboRIO 2.0 Continued

- The RoboRIOs CPU is more than just your everyday CPU.
- The CPU has a Field Programmable Gate Array (FPGA) that enables things like nanosecond timing and PWM generation that wouldn’t be possible on your everyday CPU.
- Even with the FPGA, the RoboRIO’s are limited in there processing capacity
- Other devices can be put on the robot to assist with compute heavy tasks, but that is an advanced topic not covered here

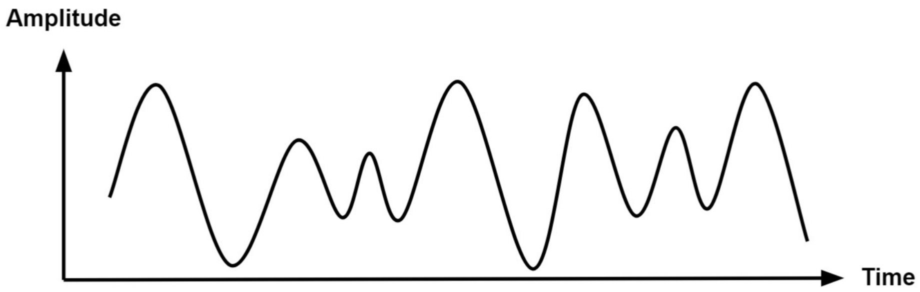
Digital I/O and Pulse Width Modulation (PWM)

- Digital Input/Output is used for things like switches and buttons.
- It can also be used for Encoders
- PWM can be used to capture sensor information, or control motor controllers



Analog In

- Digital signals represent on and off, 1 and 0, true and false. Analog can represent a wide range of values
- Analog In can take in values from sensors like Time of Flight (ToF) and Potentiometers (for measuring angles)



I2C and SPI

- Inter-Integrated Circuit (I2C) and Serial Peripheral Interface (SPI) are two types of communication mechanisms integrated into the RoboRIO.
- Both serve to communicate with sensors like color sensors and gyroscopes
- Each interface has it’s benefits and shortcomings, and most sensors will implement one or the other communication mechanism

CAN

- Controller Area Network (CAN) was originally developed for the automotive industry for sensor communication
- We use CAN for interfacing with motor controllers and sensors.
- CAN is a bus, meaning many devices can be connected with the same two wires but be “addressed” individually by assigning each connected device it’s own ID number.
- As teams are beginning to find out, the CAN bus on the RoboRIO has a hard limit on how much data it can handle (about 1Mbps)

MXP

- The myRIO eXpansion Port (MXP) is used by add on boards to provide additional functionality to the RoboRIO
- It includes additional DIO, PWM, and Analog I/O. As well one more of each of the following: I2C Port, SPI Port, Serial Port (TTL)
- A list of MXP devices that are approved for use with the RoboRIO can be found in the game manual each year. New devices are added periodically.

PDP/PDH

- The Power Distribution Panel/Hub (PDP/PDH) is used to manage and monitor the power connections for you robot.
- All of your devices must be in some way connected to the PDP/PDH
- The PDP/PDH has the ability to report power usage information to the RoboRIO to help diagnose power usage problems and monitor consumption during matches

PDP/PDH Continued

- Each PDP/PDH Channel has to have a fuse or self resetting breaker
- Power channels won't function at all without the appropriate fuse/breaker installed
- Each channel should only ever have 1 thing plugged into it receiving power at a time (with some minor exceptions)

PCM/PCH

- The Pneumatics Control Module/Hub (PCM/PCH) is used to control pnuematic cylinders and compressors.
- Each PCM/PCH can drive either 12v or 24v Solenoids
- The PCH added analog pressure sensing for more precise control of when the attached compressor turns on or off.

Robot Radio

- The Open-Mesh OM5P-AC currently serves as the robot’s WiFi Radio for many teams
- This device is like any other wireless access point you might have at home or at school, with the exception that this one is mostly garbage.
- Power can be provided through a barrel jack or via Power over Ethernet (PoE)
- I have very little good to say about this little white hockey puck.
- With the new REV control system hardware, I suspect we are not far from seeing a replacement for this device.

Batteries and The Main Breaker

- We use very specific 18(ish)Ah Sealed Lead Acid AGM batteries for FRC
- There are no other options of power in FRC, and for good reasons. The batteries we use are long battle tested
- The main breaker is used to turn the robot on and off, it has a 120Amp limit for about 2 seconds before it pops

Motors

- For the most part in FRC, we use 12v motors, of varying sizes.
- We can use both brushed and brushless motors, many teams however have been moving towards using brushless motors thanks to benefits like less weight for the same performance.
- Brushed motors still have there place however as they tend to cost less overall when you factor in the cost of the motor controller

Motors – The CIM Family

- The CIM motor family has been around for many years in FRC
- Three different sizes, the “standard” CIM, miniCIM, and BAG motor
- Because CIMs are so ubiquitous, you’ll find many gearbox solutions that are already capable of interfacing with the motors in the CIM family
- Many other motors, like NEOs use hole patterns from the CIMs to ensure compatibility
- There are other similar motors like the 775Pro and AndyMark Redline motors that fill “performance holes” that the CIM family doesn’t fill.

Motors – The NEO Family

- The NEO motors were the first general use brushless motors to enter FRC
- Offering similar performance to CIMs and BAG motors. The NEO and NEO 550 motors are lighter weight with roughly the same (slightly better) performance
- NEOs have integrated encoders due to how they function, so measuring distance or rate of travel is slightly easier to implement in terms of physical hardware
- NEOs can be cost prohibitive for some teams, due to the high cost of the associated motor controller (and the slightly higher cost of the motors themselves when compared to CIMs)

Motors – Falcon 500s

- The Falcon 500 is VEX's answer to the NEO motor
- Falcon's have become very popular in FRC due to there integrated Talon FX Motor Controller and an Encoder
- Has user replaceable output shafts, so cutting shafts no longer limits the future usability of the motor
- Has the best power to weight ratio (at 20Amp) in FRC
- Used very commonly in swerve drivetrains
- As with the NEOs, Falcon's have a relatively high cost to implement when compared to other CIM family brushed solutions

Motors - Others

- There are tons of other motors (mostly brushed) available to FRC teams
- To name a few, Bosch Seat Motors, Dynamo Brushless, Window Motors, Neverest Motors, PG Gearmotors, and more

Motor Controllers

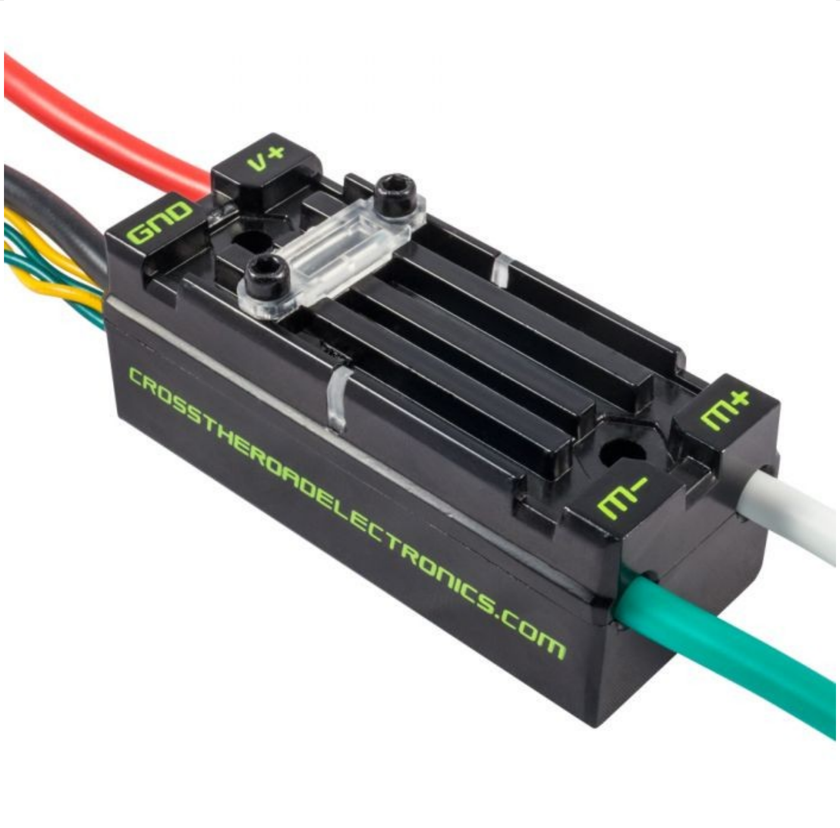
- Motor controllers are how we control the speed of motors
- Motor controllers vary the voltage given to a particular motor to change its potential amount of output speed and power
- Most motor controllers in FRC are sent instructions via CAN these days, although some still rely on PWM
- There are many types of motor controllers available to teams, the general performance is more or less the same, but additional “smart” features or support for brushless motors are often the differentiating factors

Motor Controllers – Victor SP and Victor SPX

- The Victor SP was a motor controller that only used PWM speed control, whereas the SPX can use both PWM and CAN
- Some of the cheapest motor controllers in FRC for brushed motors
- The Victor SP is no longer sold, the Victor SPX cost was lowered slightly to take up the gap that the Victor SP left
- The SPX has on-board PID control support, as well as other advanced features to remove overall load from the RoboRIO

Motor Controllers – Talon SRX

- The Talon SRX is a high end, well featured motor controller for brushed motors.
- Supports PWM, CAN, SPI, and Serial for control
- Has on-board PID control and an additional expansion port for direct connection of sensors like limit switches and encoders
- For many teams implementations, the difference between a Victor SPX and Talon SRX is simply cost, although there are high end control features on the SRX that do make it attractive for teams willing to spend the money



Motor Controllers – Spark MAX

- The Spark MAX was the first general use Brushless motor controller in FRC
- Spark MAXs have similar advanced features to the Talon SRX
- Spark MAXs can be configured to drive both brushed and brushless motors
- Spark MAXs support the NEOs encoder requirement through a port on the front, as well as a top port that similar expansion to the Talon SRXs additional port

Motor Controllers – Obsolete

- There are many motor controllers in FRC that are now considered obsolete
- Many of these controllers still work just fine and are FRC legal, but have been obsoleted by technological improvements and aren’t worth the additional weight and wiring complexity they require.
- They’d still be fine for offseason projects and demo bots

Sensors

- If motors are how our robot interacts with the world, sensors are how our robot senses the world
- There are many different types of sensors, too many really to go over in this particular concept set
- Sensors tend to follow human senses, like touch, sight, hearing, balance, etc.
- The sensors we'll discuss are some of the more common ones in FRC, but remember that there are many other options that meet many different needs.

Sensors - Buttons and Switches

- Buttons and switches are the simplest way of sensing contact between two surfaces
- Buttons and switches use Digital Inputs, on or off signals, for sensing
- Some example applications include detecting contact with an object, or sensing when a mechanism, like an elevator, is at the top or the bottom of its track

Sensors - Potentiometers

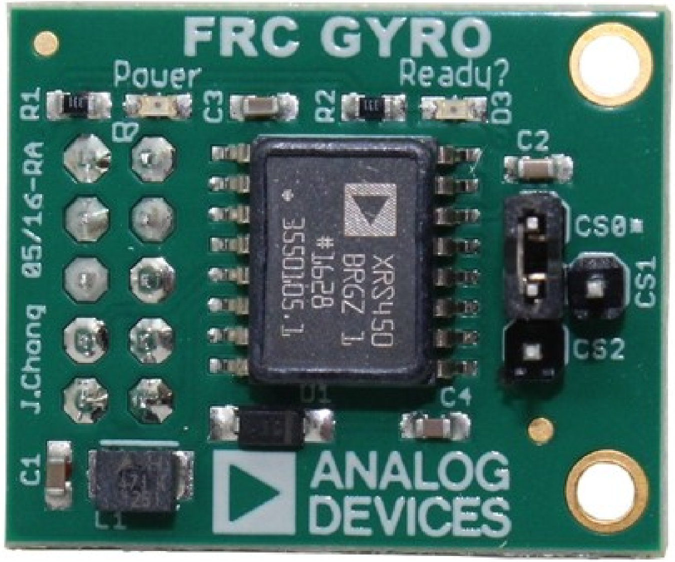
- Potentiometers are used to measure angles by measuring a change in voltage when something turns the potentiometer
- Potentiometers use analog input for sensing
- Potentiometers tend to be limited to a certain number of degrees that they are capable of turning
- An use example would be measuring the current angle of a joint on a robotic arm

Sensors - Encoders

- Encoders are used to measure the rate at which something rotates, and how far something travels
- Encoders often use 2 or 3 digital inputs to sense distance traveled, rate of travel, and direction of travel
- Encoders are great for measuring the distance traveled by a mechanism, like a drivetrain

Sensors - Gyroscopes

- Gyroscopes are used to measure rotational acceleration, from that information, you can get rotational velocity and angle.
- Gyroscopes will usually use a specific communication protocol, like I2C, CAN, or SPI
- Gyroscopes are great for when you want to know what heading your robot is facing



Sensors - Accelerometers

- Accelerometers are used to measure linear acceleration, and like gyroscopes, math can be used to get linear velocity and distance
- Accelerometers will usually use a specific communication protocol like I2C, CAN, or SPI
- The RoboRIO has a built in 16G Accelerometer
- Accelerometers aren’t used by themselves often, but in tandem with with gyroscopes and some clever math, it’s possible to create a system that tracks the robots position relative to where it started

Sensors – Cameras

- Cameras should be obvious what they are
- You can use standard USB webcams with the RoboRIO (with some limitations)
- Some cameras, can also connect to the robot over a regular Ethernet networking cable
- A lot of teams will use Cameras to track reflective tape targets on the field using vision tracking code they've written
- In recent years, the Limelight has become a ubiquitous, easy to implement vision tracking camera that is accessible to all teams, without writing complicated code

FINALLY DONE!

(╯°□°)╯︵ ┻━┻