# CAN Capabilities with Jaguar

Scott McMahon
Texas Instruments (formerly Luminary Micro)
Mentor, Team 2468

# Why?

- Why is this potentially important?
  - The features in Jaguar that are available via CAN may simplify your robot design
  - At the end, we'll discuss a couple of applications to tie this material together so you can see how the features described can be used in a system

# Overview

- The material presented is based on the activities of the 2010 beta testers and what they were able to experiment with CAN
- Access to the CAN features on Jaguar may or may not be allowed in the contest; we will find out tomorrow

# Overview

- Features and Capabilities
  - 2010 “Black” Jaguar
  - Differences between “Grey” and “Black” Jaguar

# Overview

- CAN network topology/wiring
  - Basic wiring (4 and 6 conductor)
  - cRIO Bridges (Black Jag, 2CAN, etc.)
  - IDs

# Overview

- Using the bdc-comm utility
  - Assigning IDs
  - Updating firmware
  - Testing connections
  - Tuning values

# Overview

- Operating modes (the Jaguar API)
  - Voltage
  - Position
  - Speed

# Overview

- Jaguar configuration and status
  - Faults
  - Limit switches
  - Sensor connections

# Overview

- Application 1: Servo mode for car style steering
  - Position mode, analog feedback, soft limit switches
  - Or, position mode, digital feedback, soft and hard limit switches

# Overview

- Application 2: Drive system
  - Speed mode, encoder feedback, synchronous updates

# Resources

- Places to look for help
  - TI will be active on the FIRST forums to provide open community technical support
  - If you wish to contact TI support www.ti.com/support
  - TI will also monitor Chief Delphi, but most technical support will be on the FIRST forums

# Features & Capabilities

- Jaguar history
  - Provided in 2009 KoP
  - CAN not allowed in 2009, but the Jaguar firmware contained a fully functional CAN API

# “Black” Jaguar

- Redesign of “Grey” Jaguar was necessary since some of the components were no longer available for purchase which allowed for some enhancements, the most important being
  - More TI components to keep costs low
  - New serial interface allows cRIO to access CAN features (*may* be useful)

# Differences

| Feature | Grey | Black |
|---|---|---|
| Voltage Range | 6-13 | 6-30 |
| Control Interfaces | Servo-style PWM<br>CAN (dual 6P4C) | Servo-style PWM<br>CAN (6P4C and 6P6C)<br>RS232 serial port |
| HBridge | Not synchronous rectification | Synchronous rectification |

# Networking

- Wiring characteristics
  - Wiring is low cost
    - 100 ft 4 conductor cable + 100 6P4C connectors is < $20
    - Crimp tool is < $25 (ratcheting!)
  - Connectors have locking tab and secure in place
  - Making wires is easy, far easier than making a PWM cable

# Networking

- CAN-to-CAN
  - 2 6P4C connectors
  - 4 wire modular cable
  - The same wire goes to the same pin on both ends (pin 1 to pin 1, pin 2 to pin 2, etc.)
  - Cable has one end with tab up and one with tab down

# Networking

- RS232 Serial
  - 2 6P6C connectors
  - 6 wire modular cable
  - Resistor, shrink tube
  - Modular adapter (RJ12 to DB-9 female)
  - The same wire goes to the same pin on both ends (pin 1 to pin 1, pin 2 to pin 2, etc.)

# Networking

- CAN uses termination for proper network operation
  - CAN spec says two 120 ohm terminators, but TI has found that for reliable robot use, 100 ohm terminators are better
- Terminators typically crimped into a connector and inserted at the end of the CAN segment
- 2CAN integrates a 120 ohm terminator (jumper)

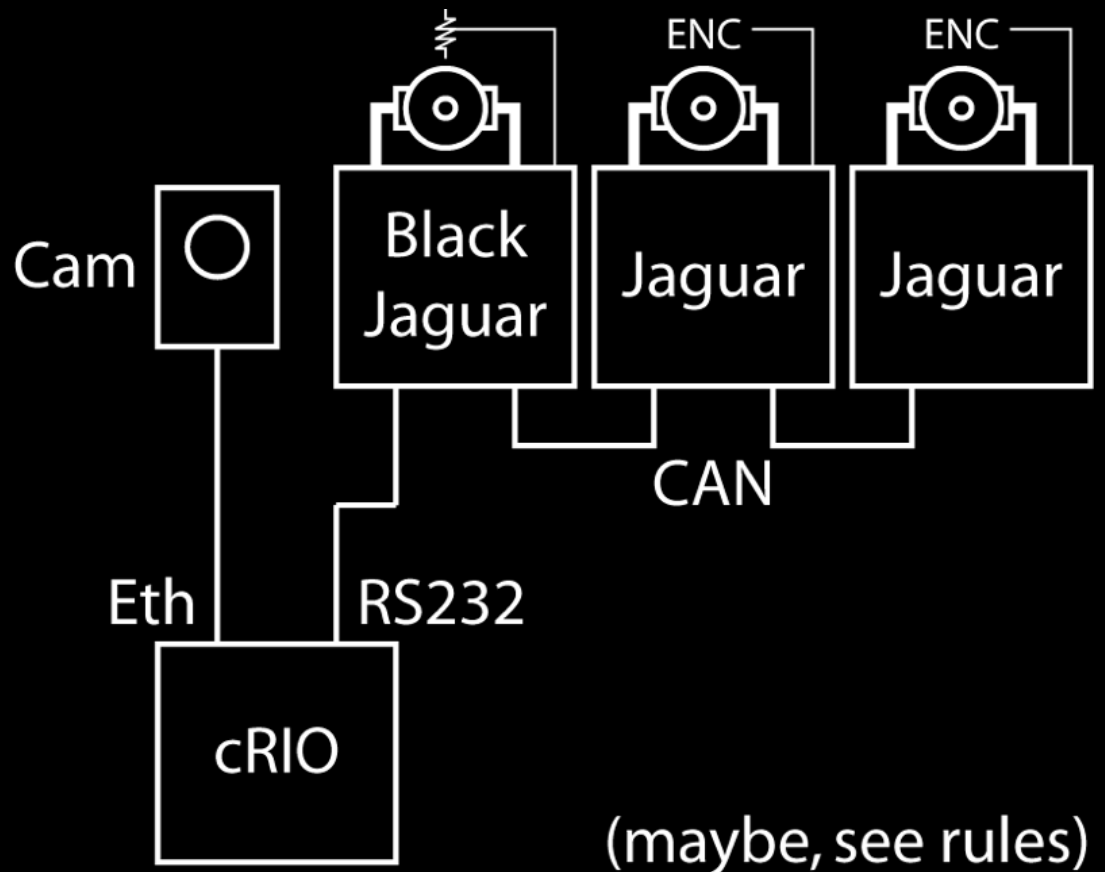
# Networking

- When using RS232 Black Jaguar bridge, termination is put inside DB-9 to modular adapter
- Note that for the super paranoid, you could wire a CAN network in a loop, in which case, termination requires a team-supplied 2-way T connection or similar custom circuit (we’re not recommending this)

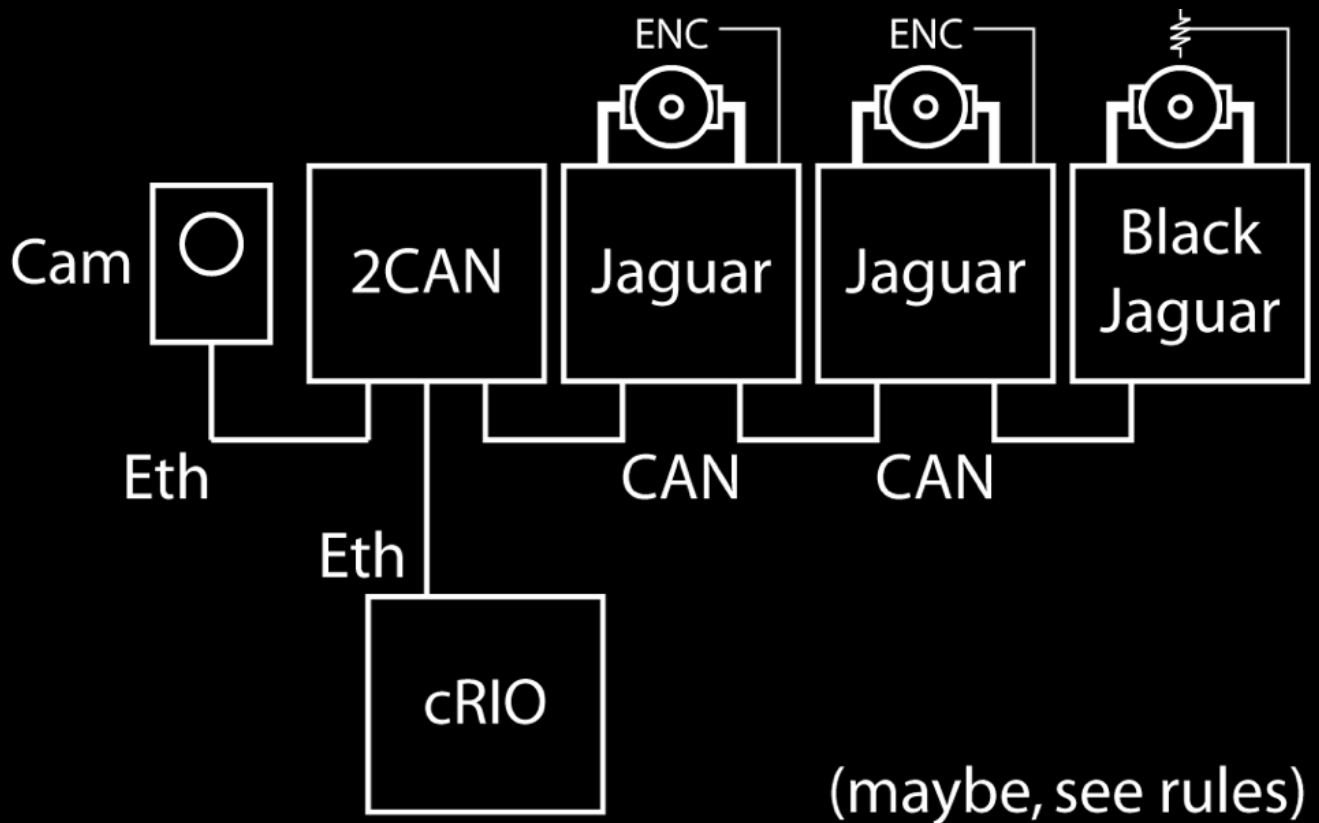
# Networking

- Bridging cRIO to a CAN bus has multiple options
  - “Black” Jaguar using the cRIO RS232 port
  - Cross the Road Electronic’s 2CAN ethernet to CAN adaptor
  - Soon-to-be released software for TI’s LM3S8962 evaluation board (source)

# Networking


ENC
ENC
Cam
Black Jaguar
Jaguar
Jaguar
CAN
Eth
RS232
cRIO
(maybe, see rules)

# Networking


ENC
ENC
Cam
2CAN
Jaguar
Jaguar
Black Jaguar
Eth
CAN
CAN
Eth
cRIO
(maybe, see rules)

# Networking

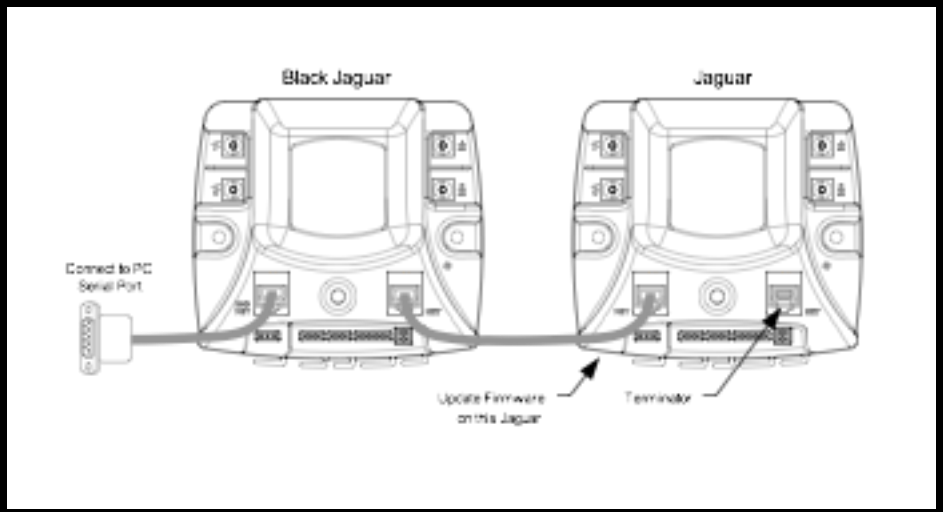
- CAN IDs
  - CAN 2.0B uses a 29-bit message identifier field in the packet sent across the bus
  - To direct a packet at a specific CAN device (Jaguar or other) Luminary Micro defined a field in the 29-bit MesgID for a device ID
  - 0 means all (broadcast) and 1 is used for the device as sent from the factory

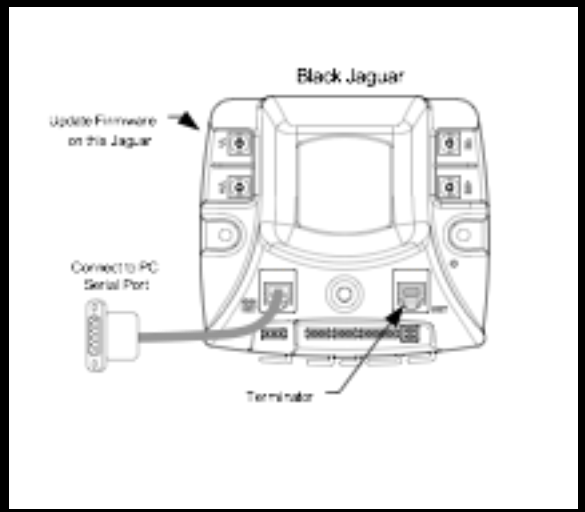
# Networking

- You need to assign an ID to each control position
- Recommend using 2-63
  - e.g. 2 is left chassis drive, 3 is right chassis drive, and 4 is turret rotation drive
- Please, avoid using 1 !!
- These are statically assigned using bdc-comm application (other methods available, but this is the easiest and most reliable)

# Networking

- When assigning CAN IDs, it is best to have as few Jaguars on the network, so the following are recommended

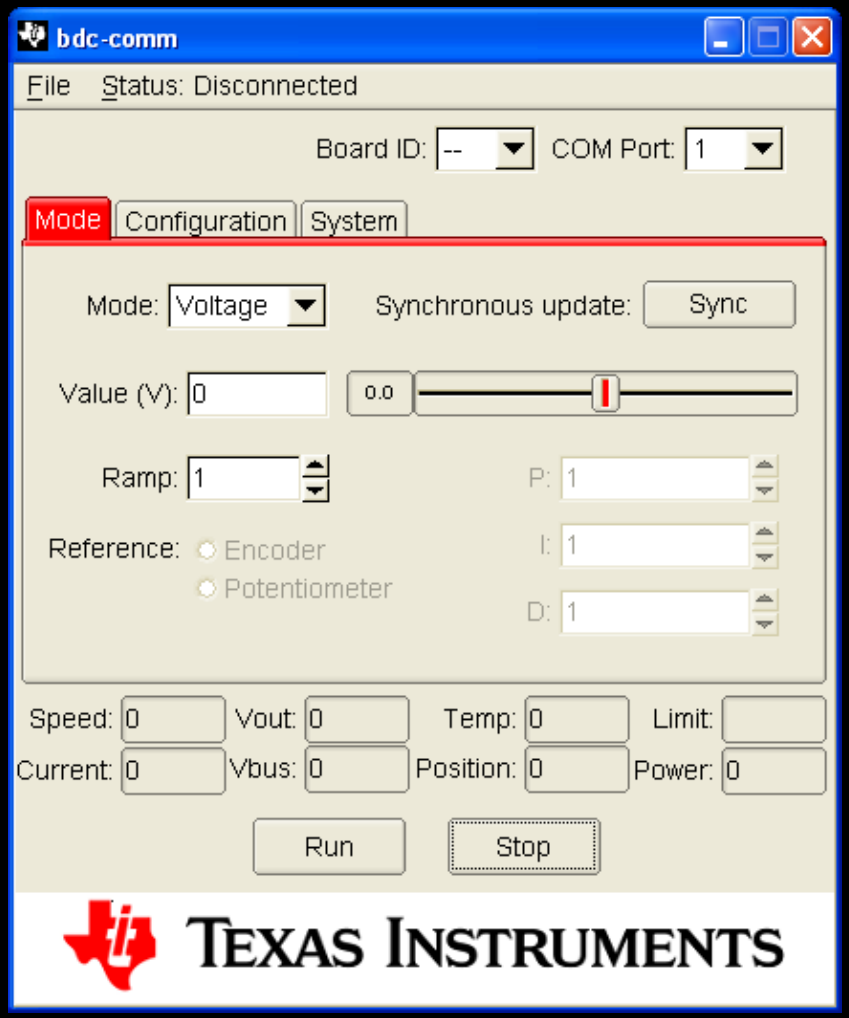


# bdc-comm

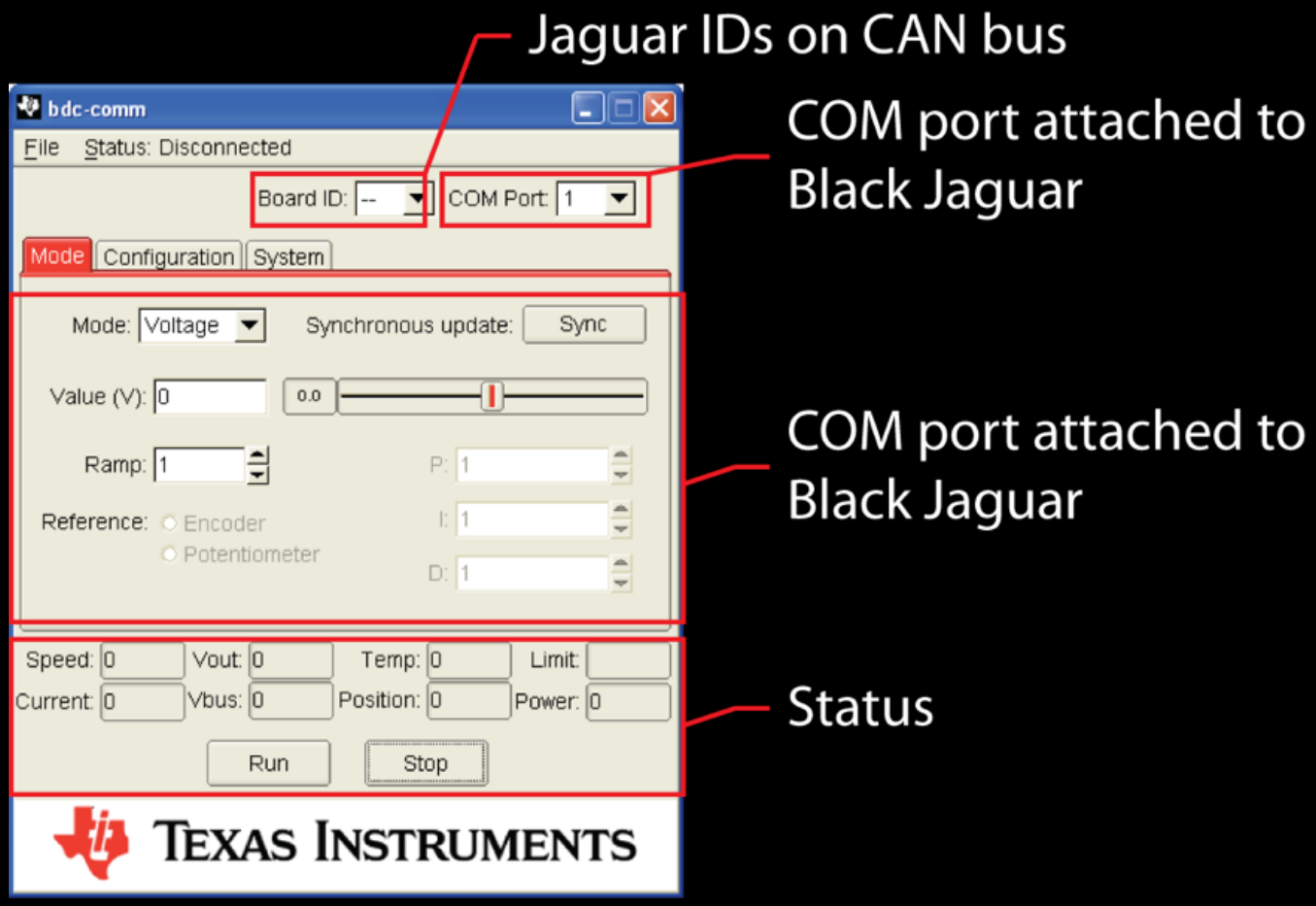
- bdc-comm is a Windows application that provides access to Jaguars via COM port using a Black Jaguar bridge.
- bdc-comm is a GUI and command-line utility
  - Run with -c N for CLI (N is COM port number)

# bdc-comm

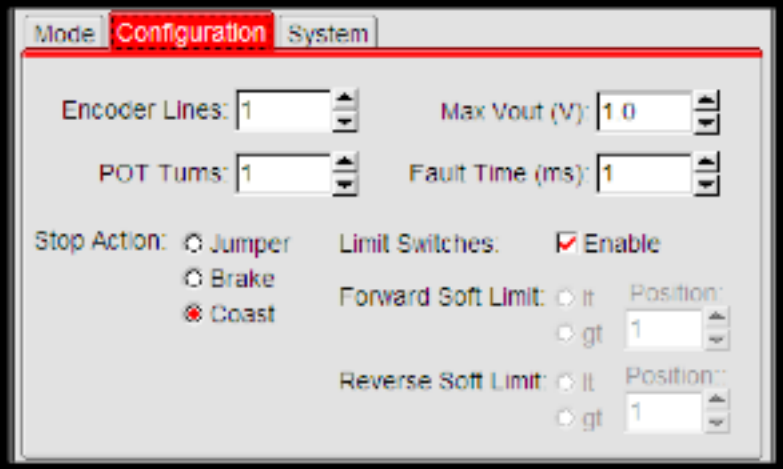



- With bdc-comm you can:
  - Set ID
  - Update firmware
  - See status
  - ...

# bdc-comm

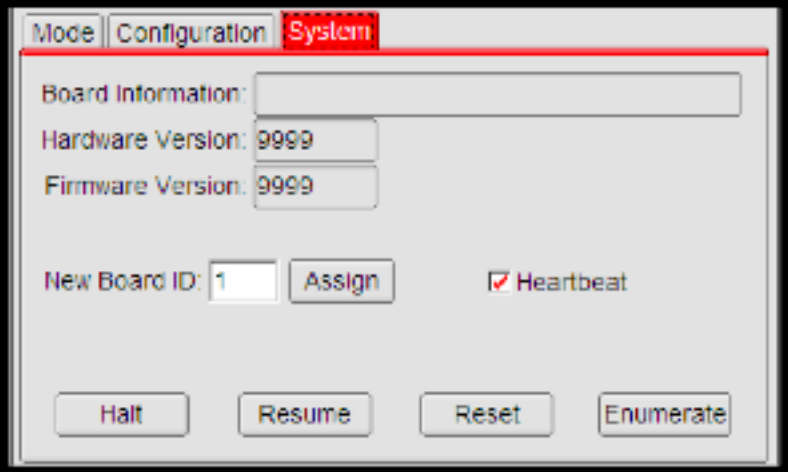

Jaguar IDs on CAN bus
COM port attached to Black Jaguar
COM port attached to Black Jaguar
Status
bdc-comm
File
Status: Disconnected
Board ID:
COM Port:
Mode
Configuration
System
Mode:
Voltage
Synchronous update:
Sync
Value (V):
Ramp:
Reference:
Encoder
Potentiometer
P:
I:
D:
Speed:
Vout:
Temp:
Limit:
Current:
Vbus:
Position:
Power:
Run
Stop
Texas Instruments

# bdc-comm




- Configuration tab
  - Sensor config
  - Brake/Coast
  - Soft limits enable and config
  - Fault time

# bdc-comm



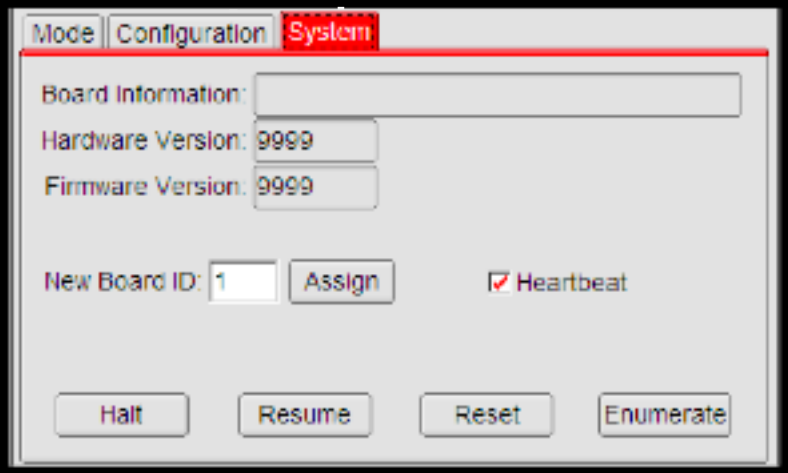

- System tab
  - Check version
  - Check/set CAN ID

# bdc-comm

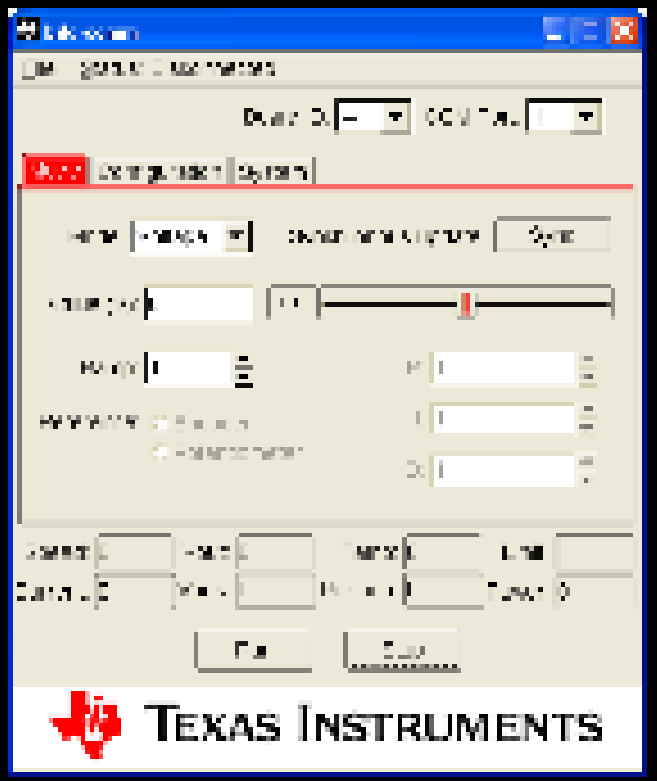
- Update CAN IDs then update firmware
  - Assign and update your RS232-connected Black Jaguar first without connecting any other Jaguar devices (i.e. bridging device first)
    - CAN ID > 1
  - Then assign and update other Jaguars by attaching them to the bridging device’s CAN port one at a time (i.e. no more than 2 Jaguars connected together at the same time)

# bdc-comm

- Updating ID
  - Enter ‘New Board ID’ value and click




- Updating firmware
- File > Update Firmware



# bdc-comm CLI

```
$ ./bdc-comm -c 2
# help
help      - display a list of commands
h         - alias for help
?         - alias for help
id        - set the target ID
heartbeat - start/stop the heartbeat
volt      - voltage control mode commands
cur       - current control mode commands
speed     - speed control mode commands
pos       - position control mode commands
stat      - status commands
config    - configuration commands
system    - system commands
update    - update the firmware
boot      - wait for boot loader to request update
exit      - exit the program
quit      - alias for exit
q         - alias for exit
```

# Fixed-Point Arithmetic

- The Jaguar MCU does not provide a floating point unit, and therefore most parameter values are encoded using fixed point representation
- For example, 16.16 means 16 bits of whole and 16 bits of fractional data
  - Smallest value of a 16.16 is 1/65536 ~ 0.000015
  - 1.0 is 0x010000, 0.5 is 0x08000,
    0. 0625 is 0x01000, 3.1415 is ~0x32439

# Fixed-Point Arithmetic

- Internal arithmetic performed using higher precision representation
  - 16.16 x 16.16 generates a 32.32 result (64 bit)
- Embedded programmers don’t worry about conversions, we let the compiler do them for us

```
#define PI 3.1415
#define ONE_16DOT16  0x010000

Foo(PI * ONE_16DOT16);
```

# Operating Modes

- Grey and Black Jaguars have numerous operating modes
  - Voltage
  - Position
  - Speed

# Voltage

- Analogous to PWM mode, but with enhancement
- Voltage in -1 to 1 (-12 V to 12 V) range
- Optional ramping applied to output voltage
- Limit switches function in voltage mode

# Position

- Turns your Jaguar into a servo controller, just add feedback and motor
- The position value is expressed in ‘turns’ of the feedback sensor
- Feedback is either a quadrature encoder or a 10 kohm potentiometer (multi-turn optional)

# Position

- Program the P, I, and D constants (all 16.16 FPC)
- Program the type of feedback sensor used and any characteristics (encoders require the number of pulses per revolution and the value of the current position)
- Program the desired position, with optional synchronous update group
- Soft limits are optional

# Speed

- The speed value is expressed in ‘revolutions per minute’ of the feedback sensor
- Feedback is a quadrature encoder
- Program the P, I, and D constants (all 16.16 FPC)
- Program the encoder characteristics (encoders require the number of pulses per revolution)
- Program the desired speed, with optional synchronous update group

# Synchronous Updates

- Allows the application to load the next voltage/position/speed value into the Jaguar for delayed activation
- Multiple Jaguars are updated and then they're sent a broadcast 'Synchronous Update' packet and all activate the posted value
  - It is like a PWM splitter cable, but better since you can update both sides of a robot

# Limit Switches

- Two limit switch inputs
- Each switch controls a direction of travel (rotation) for the controlled motor
- Uses normally closed switches
  - Closed = direction enabled
  - Open = direction disabled (Vout = 0)
- When not used, the jumpers must be in place

# Limit Switches

- State of the limit switches is available to software

# Soft Limits

- Software enabled
- The value returned by the sensor is compared against a limit value and comparison criteria (e.g. greater than 2 turns)
  - Unused direction disabled using unsatisfied expression
- Works in parallel with physical limit switches
- State of soft limit is available to software

# Configuration

- Position sensor is Pot or Encoder
- Potentiometer turns
- Encoder pulses per rotation (lines)
- Brake/Coast jumper override
- Soft limit switch enable, forward and reverse position limit and condition (less than/greater than)
- Maximum output voltage

# Configuration

- Fault time duration (default is 3 s, but can be reduced to 0.5 s)

# Status

- Output voltage
- Bus voltage
- Fault status
- Current status
- Temperature
- Position (pot or encoder position)
- Speed

# Status

- Limit switch/soft limit state

# Safety

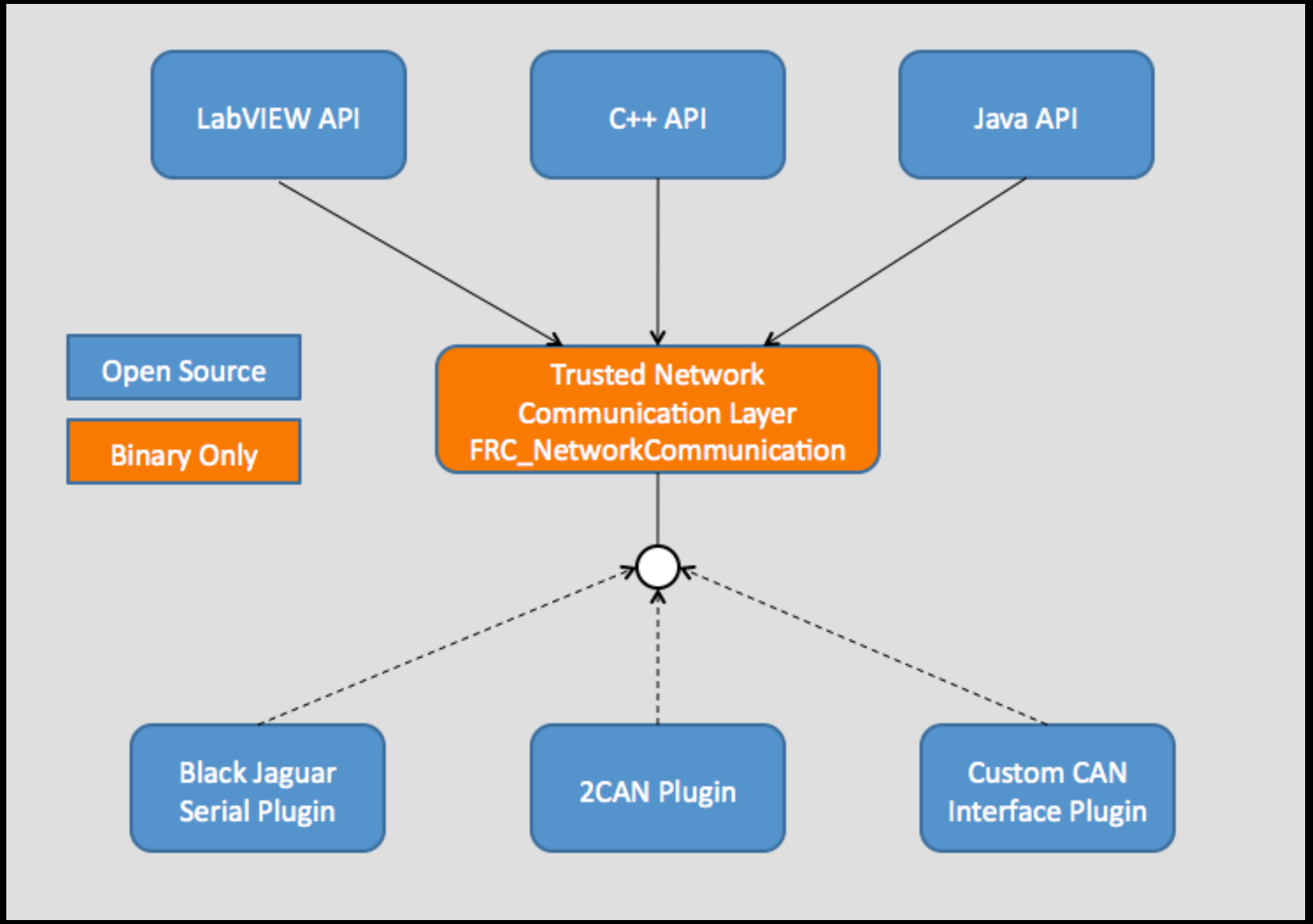
- Due to safety concerns, the methods of operation of the Jaguars will *likely* be limited by FIRST (see rules tomorrow!)
- It is very likely that the PWM mode used during last year's competition will be supported
  - This is because the underlying software/FPGA can disable the PWM signals connected to the motor controllers automatically

# Safety

- For CAN connections, the same safety goals need to be met
- A *trusted* mode of CAN operation may be supported which requires the use of an API
- In trusted mode the cRIO still must be capable of sending a halt message to the Jaguars, and so the Jaguars need to trust that the commands are coming from and endpoint capable of inserting a halt command

# Safety

- Therefore a new protocol was developed that permits the command stream to authenticate that the source is the cRIO using the API
- Third party developers, like Cross The Road Electronics, will require simple changes to their provided APIs so that their devices may utilize the trusted API

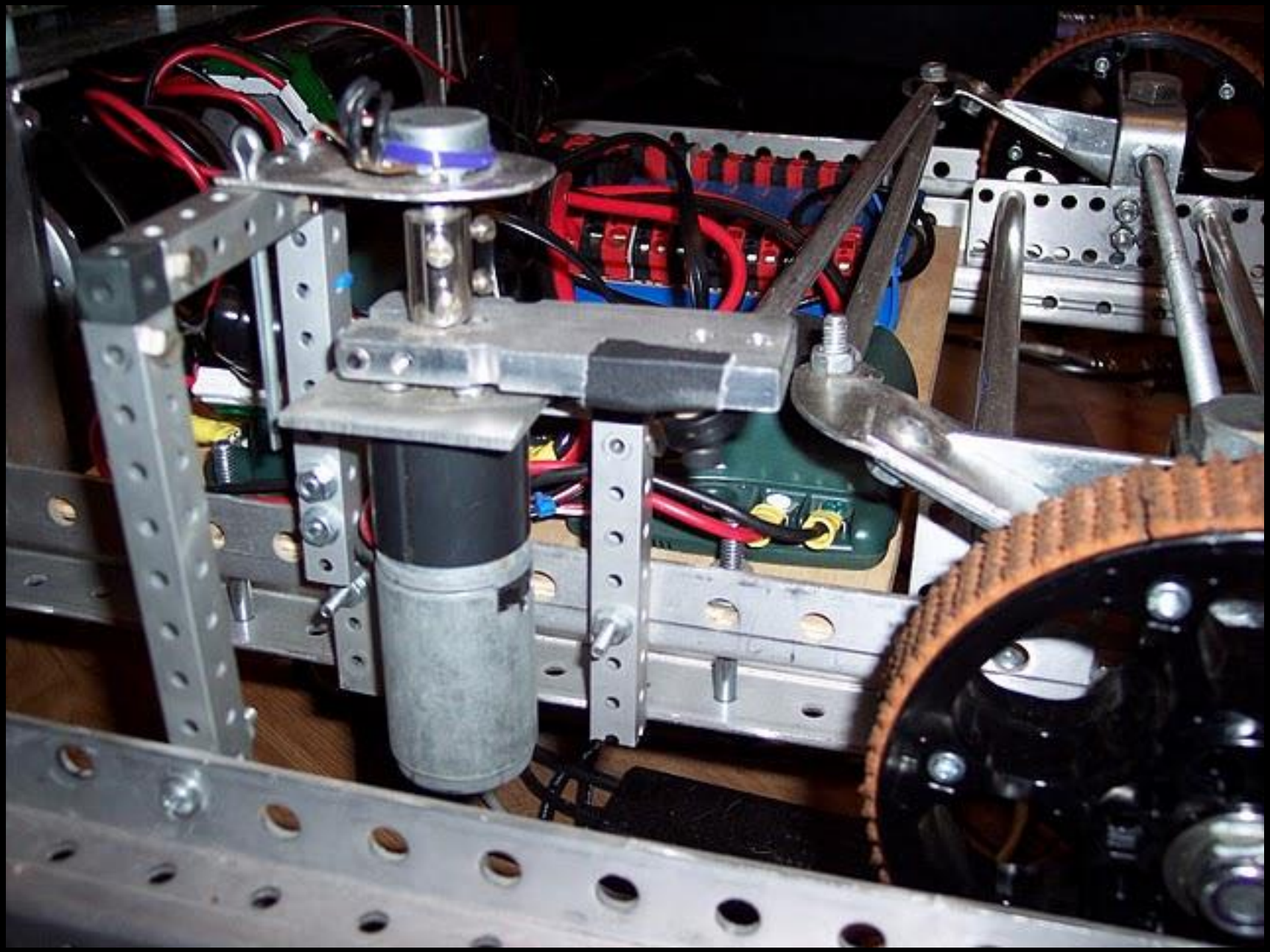

LabVIEW API
C++ API
Java API
Open Source
Binary Only
Trusted Network
Communication Layer
FRC_NetworkCommunication
Black Jaguar
Serial Plugin
2CAN Plugin
Custom CAN
Interface Plugin

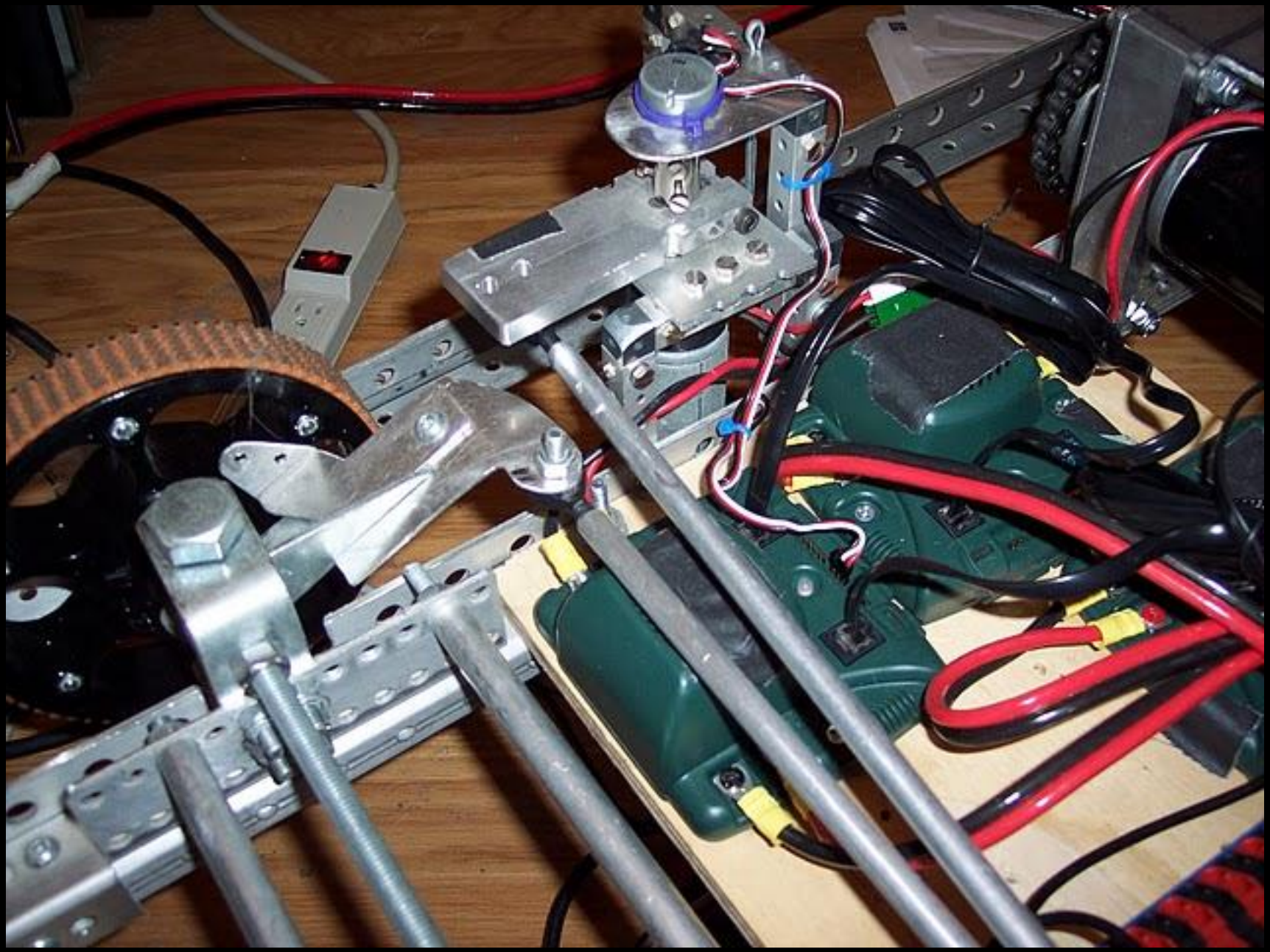
# Safety

- This is in addition to the standard features like the detection of loss of signal due to entanglement

# Application 1

- Jaguar and Globe motor used for a steering servo
- Potentiometer is attached to motor shaft through coupler
- Motor shaft moves Ackerman steering linkage





# Application 1

- Jaguar (ID 4) configured as follows:
  - Configuration Pot Turns = 1
  - Position Reference = Pot
  - Position Mode Enable, initial position = 0.5
  - Position P Value = 768, I Value = 0, D Value = 16
  - Configuration Limit Forward > 0.63 turns
  - Configuration Limit Reverse < 0.37 turns
  - Configuration Limit Mode enable

# Application 1

- How were the soft limit values determined?
  - Monitor the position value as you manually move the wheels
- The same basic configuration could be used to control the steering motor of each corner of a swerve drive (e.g. WildSwerve module), just tune for different P, I, and D values

# Application 2

- Jaguar and CIM motor used in chassis drive
- Encoder is attached to gearbox output shaft through encoder

# Application 2

- Jaguars ID=2 (left) and 3 (right) are configured as follows:
  - Configure Encoder Lines = 250
  - Configure Brake/Coast = coast
  - Speed Mode Enable
  - Speed P Value = 0, I Value = 0.005, D Value = 0
  - Speed Reference Set = Encoder

# Application 2

- How were the P, I, and D values determined?
  - Empirically
- The same basic configuration could be used to control the drive motor of each corner of a swerve drive (e.g. WildSwerve module), just tune for different P, I, and D values

# Application

- Combine application 1 and application 2 and you have the demonstration robot we took to Atlanta last year
- Combining the updates to position and speed are done synchronously
  - Position Set ID = 4 Value = pos, Group = 1
    Speed Set ID = 2 Value = lspeed, Group = 1
    Speed Set ID = 3 Value = rspeed, Group = 1
    Synchronous Update Group = 1

# Questions?

- Do we actually have any time left?  If so, I apologize for talking too quickly, if not I apologize for talking slow :)

# End

- Thanks for your feedback; good feedback makes products better
- Check out the Jaguar website from time to time and look for firmware updates
  www.luminarymicro.com/jaguar
- I hope everyone gets a good dose of ‘I’ (inspiration)
- Best of luck in the competition season