Roadmap

8. Meet the Afternoon Teams
9. Interface Control Review
10. Vehicle Supervisory Control and Data Acquisition (VSCADA)
   a. Daemon
   b. Interfacing
   c. User Applications
   d. Data Storage
11. Dynamometer (DYNO)
   a. Decomposition and Definition
   b. Integration and Recomposition
Roadmap

8. **Meet the Afternoon Teams**

9. Interface Control Review

10. Vehicle Supervisory Control and Data Acquisition (VSCADA)
    a. Daemon
    b. Interfacing
    c. User Applications
    d. Data Storage

11. Dynamometer (DYNO)
    a. Decomposition and Definition
    b. Integration and Recomposition
Meet the Afternoon Teams

- Vehicle Supervisory Control and Data Acquisition (VSCADA)
  1. Yiming Chen
  2. Bikram Shrestha
  3. Rameel Sethi
  4. John Gehrig
  5. Sam Cesario
  6. Adam Cornwell

- Dynamometer (DYNO)
  1. Steve Mazich
  2. Brendan Malone
  3. John Bloore
  4. Nate Hand
  5. Alex Hytha
Roadmap

8. Meet the Afternoon Teams
9. Interface Control Review
10. Vehicle Supervisory Control and Data Acquisition (VSCADA)
   a. Daemon
   b. Interfacing
   c. User Applications
   d. Data Storage
11. Dynamometer (DYNO)
   a. Decomposition and Definition
   b. Integration and Recomposition
Interface Control

An Interface Control Document was created to accurately and completely define all (electrical, mechanical, semantic) aspects of top-level interfaces to allow different designers to coordinate with each other successfully.

Next, we will discuss these top-level interfaces.
Physical Interfaces Layout - Side View
System Assemblies Layout
Roadmap

8. Meet the Afternoon Teams
9. Interface Control Review
10. **Vehicle Supervisory Control and Data Acquisition (VSCADA)**
   a. Daemon
   b. Interfacing
   c. User Applications
   d. Data Storage
11. Dynamometer (DYNO)
   a. Decomposition and Definition
   b. Integration and Recomposition
Embedded Computer: VAB-820
VSCADA

VSCADA Computer Linux system

Config/sys file

SocketCAN

D I/O

Data Input Control

customized CAN library

Sys state control

Event Watcher

Syslog

SQL

RRDTool

Sensor List

Program Data

WiFi

Ethernet

USB

HDMI

LVDS

SSH

TCP/IP

demo

maintain

Power
Roadmap

8. Meet the Afternoon Teams
9. Interface Control Review
10. Vehicle Supervisory Control and Data Acquisition (VSCADA)
   a. **Daemon**
   b. Interfacing
   c. User Applications
   d. Data Storage
11. Dynamometer (DYNO)
    a. Decomposition and Definition
    b. Integration and Recomposition
Logging/Daemons - Main Program

- VSCADA uses systemd to initially launch the main program
  - systemd has most major Linux distributions support
- The main program will run in the background as the server with PID registered
- The main program will start by doing system startup procedures
Logging/Daemons - Startup

● VSCADA uses systemd to initially launch the main program
  ○ systemd has most major linux distributions support
● The main program will run in the background as the server with PID registered
● The main program will start by doing system startup procedures
System Startup

Linux System
- VSCADA Off
  - GLV Turned On
    - Boot OS
      - Account Auto Login
        - Systemd starts the Main

Main Program
- Main program is initiated
  - Setup VSCADA
    - Setup Vehicle
      - Rdy

Sub programs
- Setup the logger
- Setup the Server
- Query database
- Read config files
  - Start CAN and socketCAN program
    - Run the Server

Sub programs
- Read start up procedures
- Perform start up procedures
- Listen and check sensors
- Read config files
System States and Exceptions

Startup Stage
- When system boots up and will go to one of the following stages

Good to Go Stage
- No error or warning and is ready to be driven

Driving Stage
- The car is driving

Vehicle Boot Up Error Stage
- VSCADA is functional but other subsystems are not

VSCADA Boot Up Error Stage
- VSCADA is not functional

Failed Stage
- VSCADA failed to boot up

★ You are not allowed to drive if in these error stages
System States and Exceptions

System Errors:
Snytax error ➔ Failed Stage
VehicleStartupConfigLoadException ➔ Other Boot Up Error Stage
DatabaseLoadException ➔ VSCADA Boot Up Error Stage
RRDFileNotFoundException ➔ VSCADA Boot Up Error Stage
VehicleStartupTimeoutException ➔ Other Boot Up Error Stage
SensorCheckingTimeoutException ➔ Other Boot Up Error Stage
SystemFailureError ➔ Failed Stage
OtherCommunicationException ➔ Vehicle Boot Up Error Stage
OtherSystemException ➔ VSCADA Boot Up Error Stage

Sensor Errors:
Logic:
Errors are configurable and specific
If happens before driving, then the car is disabled from driving;
else take actions according to sensor configuration

Possible Errors:
OverHeating
UnderCharged
Logging/Daemons - Logging

- Have 5 levels, in their respective order:
  - Debug: detailed information, mainly used for debugging
  - Info: general information, should contain important data
  - Warning: Need user’s attention
  - Error: Need user to check the source of the error
  - Critical: Opps.

- Logs are stored in syslog of Linux
  - syslog handles storage, update, filter, etc.
  - Python and other library support for syslog

- Can be viewed by clients over the TCP protocol
- Levels can be set by configuration. Info level by default
Roadmap

8. Meet the Afternoon Teams
9. Interface Control Review
10. Vehicle Supervisory Control and Data Acquisition (VSCADA)
   a. Daemon
   b. **Interfacing**
      c. User Applications
      d. Data Storage
11. Dynamometer (DYNO)
    a. Decomposition and Definition
    b. Integration and Recomposition
Interfaces

- **TSV Communication**
  - Send/Receive Packets of data from the four PACMAN via CAN
  - will follow API

- **GLV Communication**
  - JGB act as a hub for groups of sensors via CAN

- **DYNO Communication**
  - Control Throttle via JGB board
  - Motor Controller CAN
  - Dynamometer - USB interface to read RPM and torque, set valve
CAN Interface

- **SocketCAN -Linux Drivers**

```
sam@bull3:~/Desktop/vscada/can-lib$ cansend vcan0 123#11.02.fe.fe.ee.ee.95.33
```

```
sam@bull3:~/Desktop/vscada/can-lib$ candump vcan0
vcan0  123  [8]  11 22 33 44 55 66 77 88
vcan0  123  [8]  11 22 33 44 55 66 77 88
vcan0   001  [3]  11 12 13
```

- **Python-CAN**

```
sam@bull3:~/Desktop/vscada/can-lib$ python3 CANexample.py vcan0
Received: can id=123, can dlc=8, data=b'\x01\x02\xff\xff\xe8\xe8\xe8\xe8\x952'
```
## TSV Pack LevelCAN Frame

### Data frame Diagram

#### Byte 0 - 1, Pack#(1-4)
#### Byte 1 - Voltage (High)
#### Byte 2 - Voltage (Low)
#### Byte 3 - Current (High)
#### Byte 4 - Current (Low)
#### Byte 5 - SOC (High)
#### Byte 6 - SOC (Low)
#### Byte 7 - Fuse Temp
TSV(1) AMS Level CAN Frame

<table>
<thead>
<tr>
<th>BYTE 0</th>
<th>BYTE 1</th>
<th>BYTE 2</th>
<th>BYTE 3</th>
<th>BYTE 4</th>
<th>BYTE 5</th>
<th>BYTE 6</th>
<th>BYTE 7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Byte 0 - x1, Pack#(1-4)</td>
<td>Byte 1 - AMS#(1-7)</td>
<td>Byte 2 - Voltage (High)</td>
<td>Byte 3 - Voltage (Low)</td>
<td>Byte 4 - Current (High)</td>
<td>Byte 5 - Current (Low)</td>
<td>Byte 6 - Temperature (High)</td>
<td>Byte 7 - Temperature (Low)</td>
</tr>
</tbody>
</table>
GLV(2) Cockpit CAN Frame

<table>
<thead>
<tr>
<th>BYTE 0</th>
<th>BYTE 1</th>
<th>BYTE 2</th>
<th>BYTE 3</th>
<th>BYTE 4</th>
<th>BYTE 5</th>
<th>BYTE 6</th>
<th>BYTE 7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Byte 0 - x21 (Cockpit)</td>
<td>Byte 3 - Failure_LED (High)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Byte 1 - Ambient_temp (High)</td>
<td>Byte 4 - Failure_LED (High)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Byte 2 - Ambient_temp (Low)</td>
<td>Byte 5 - Warning_LED (High)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Byte 3 - Failure_LED (High)</td>
<td>Byte 6 - Ok_LED (High)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Byte 7 - Ok_LED (Low)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
GLV(2) TSI CAN Frame

<table>
<thead>
<tr>
<th>BYTE 0</th>
<th>BYTE 1</th>
<th>BYTE 2</th>
<th>BYTE 3</th>
<th>BYTE 4</th>
<th>BYTE 5</th>
<th>BYTE 6</th>
<th>BYTE 7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Byte 0 - x22 (TSI)</td>
<td>Byte 4 - x00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Byte 1 - Temperature (High)</td>
<td>Byte 5 - x00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Byte 2 - Temperature (Low)</td>
<td>Byte 6 - x00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Byte 3 - x00</td>
<td>Byte 7 - x00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
GLV(3) GLV_Power CAN Frame

<table>
<thead>
<tr>
<th>BYTE 0</th>
<th>BYTE 1</th>
<th>BYTE 2</th>
<th>BYTE 3</th>
<th>BYTE 4</th>
<th>BYTE 5</th>
<th>BYTE 6</th>
<th>BYTE 7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Byte 0 - x13</td>
<td>Byte 4 - Current (Low)</td>
<td>Byte 5 - Temperature (High)</td>
<td>Byte 6 - Temperature (Low)</td>
<td>Byte 7 - SOC</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Byte 1 - Voltage (High)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Byte 2 - Voltage (Low)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Byte 3 - Current (High)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Motor Controller CAN Frames (1 / 2)

<table>
<thead>
<tr>
<th>BYTE 0</th>
<th>BYTE 1</th>
<th>BYTE 2</th>
<th>BYTE 3</th>
<th>BYTE 4</th>
<th>BYTE 5</th>
<th>BYTE 6</th>
<th>BYTE 7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Byte 0 - RPM (High)</td>
<td>Byte 4 - RMS Current (High)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Byte 1 - RPM (Low)</td>
<td>Byte 5 - RMS Current (Low)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Byte 2 - Motor Temp</td>
<td>Byte 6 - Capacitor V (High)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Byte 3 - Controller Temp</td>
<td>Byte 7 - Capacitor V (Low)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Motor Controller CAN Frames (2 / 2)

<table>
<thead>
<tr>
<th>BYTE 0</th>
<th>BYTE 1</th>
<th>BYTE 2</th>
<th>BYTE 3</th>
<th>BYTE 4</th>
<th>BYTE 5</th>
<th>BYTE 6</th>
<th>BYTE 7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Byte 0 - Stator Freq (High)</td>
<td>Byte 4 - Throttle Input</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Byte 1 - Stator Freq (Low)</td>
<td>Byte 5 - Brake Input</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Byte 2 - Controller Fault P</td>
<td>Byte 6 - System Bits</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Byte 3 - Controller Fault S</td>
<td>Byte 7 - (UNUSED)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Automotive AVR
- CAN Bus
- LIN
- UART (RS-232)

Board Inputs/Outputs
- Internal Temperature
- 5 ADC Channels
- 3 PWM Channels
- 1 DAC Channel
- 6 GPIO/SPI
- 2 Differential ADC
- USB - UART
Microcontroller Firmware Design

UART
Send/Receive
Test/Debugging Information

CAN
SCADA Communication

TIMER
PWM, Sensor Timing

WATCHDOG TIMER
Crash Prevention

I/O
System Control Interface
Microcontroller Prototype Hardware

WORKING:
- ADC
- D2A
- PWM
- GPIO

NOT WORKING:
- CAN
- UART
Microcontroller CAN Frames

<table>
<thead>
<tr>
<th>BYTE 0</th>
<th>BYTE 1</th>
<th>BYTE 2</th>
<th>BYTE 3</th>
<th>BYTE 4</th>
<th>BYTE 5</th>
<th>BYTE 6</th>
<th>BYTE 7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Byte 0 -</td>
<td>Byte 1 -</td>
<td>Byte 2 -</td>
<td>Byte 3 -</td>
<td>Byte 4 -</td>
<td>Byte 5 -</td>
<td>Byte 6 -</td>
<td>Byte 7 -</td>
</tr>
</tbody>
</table>
Microcontroller Code/Toolchain

{INSERT CODE HERE/TOOLCHAIN IMAGES}
Request - Response Model

- Client Initiates Request
- Server Responds to Request
- JSON Object Passing
- Unix-Style Commands
- Modular, Flexible, Expandable
# Server Command Architecture

## Server Command Syntax:

<table>
<thead>
<tr>
<th>CMD NAME</th>
<th>FLAGS</th>
<th>OPTIONS</th>
<th>ARGUMENTS</th>
</tr>
</thead>
</table>

**CMD NAME** -
Unique command name, identifies specific server task to carry out.

**FLAGS** -
Enables or disables specific command functionality or output.

**OPTIONS** -
Utilized to pass data from the client to server.

**ARGUMENTS** -
End objects affected by the server command.

## Syntax Notes:

All command Options, Flags, Arguments space separated. Flags begin with the “-” character. Options are followed by a string containing no spaces.
Server - Client Demonstration

{INSERT VM/Host Images Here}
Server - Client Code Review

{INSERT PYTHON TCP SERVER CODE HERE}
Roadmap

8. Meet the Afternoon Teams
9. Interface Control Review
10. Vehicle Supervisory Control and Data Acquisition (VSCADA)
   a. Daemon
   b. Interfacing
   c. **User Applications**
   d. Data Storage
11. Dynamometer (DYNO)
   a. Decomposition and Definition
   b. Integration and Recomposition
Front-End User Application

- **VSCADA Maintenance Application**
  - Contains all required user functionality in one program
  - Runs on remote PC (pit station PC) and vehicle embedded computer with touchscreen
  - Demo mode can be selected in the maintenance application
  - Password is used to protect maintenance mode from unauthorized access
Maintenance App - Measurands/Input

The image shows a screenshot of a software interface for managing meausrands and inputs. The interface is part of a VSCADA Maintenance application. The main table displays various battery pack parameters:

- **TSV**:
  - Battery Pack 0:
    - Cell0: Voltage (Analog In) 3.3, Adj Value 3.3, Units V, Current (Analog In) 10, Adj Value 10, Units A, SOC (Analog In) 700, Adj Value 70, Units %, Temperature (Analog In) 70, Adj Value 35, Units C
  - Cell1
  - Battery Pack 1
  - Battery Pack 2
  - Battery Pack 3

- **GLV**

The interface also includes options for adding, modifying, and deleting vehicle sensors and a message box that reads: "This is a message."
Maintenance App - Add/Edit Sensor Window
Maintenance App - Measurand
Graph Window
Maintenance App - Hardware/Output

- Throttle: Analog Out, Value: 78%
- SL VSCADA Relay: Digital Out, Value: On
- SL TSI Relay: Digital Out, Value: On

Add or Modify Vehicle Sensor:

This is a message.
## Maintenance App - Rules

![VSCADA Maintenance - [Preview]](image)

<table>
<thead>
<tr>
<th>Measurand</th>
<th>Priority</th>
<th>Operator &amp; Threshold</th>
<th>Output</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>TSV</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Battery Pack 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cell0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Voltage</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Warning</td>
<td>&lt; 3.0</td>
<td>Nothing</td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>Warning</td>
<td>&gt; 3.6</td>
<td>Indicator</td>
<td>On</td>
<td></td>
</tr>
<tr>
<td>Error</td>
<td>&lt; 2.8</td>
<td>Indicator</td>
<td>On</td>
<td></td>
</tr>
<tr>
<td>Error</td>
<td>&gt; 3.8</td>
<td>Indicator</td>
<td>On</td>
<td></td>
</tr>
<tr>
<td>Failure</td>
<td>&lt; 2.6</td>
<td>VSCADA relay</td>
<td>Open</td>
<td></td>
</tr>
<tr>
<td>Failure</td>
<td>&gt; 4.0</td>
<td>VSCADA relay</td>
<td>Open</td>
<td></td>
</tr>
</tbody>
</table>

Add or Modify Vehicle Sensor:

This is a message.
Maintenance App - Settings

IP Address
Demo Mode
Password

Save Settings

This is a message.
Roadmap

8. Meet the Afternoon Teams
9. Interface Control Review
10. Vehicle Supervisory Control and Data Acquisition (VSCADA)
    a. Daemon
    b. Interfacing
    c. User Applications
    d. **Data Storage**
11. Dynamometer (DYNO)
    a. Decomposition and Definition
    b. Integration and Recomposition
Round Robin Database (RRD)

- High performance data logging and graphing system for time series data

- Uses circular buffer to store data
  - Data size does not expand with time.
  - Overwrite the data once it reach the starting point

- Framework for storing measurement averages, min, max and derivative

- Graphical presentation for both stored and archived data.
RRD Creation

- Size of the database can be determined at creation time.

- Specify the step time (rate at which the database update the data)

- Specify the step time for archives too. For different archives, different time step can be applied.
Round Robin Archives (RRA)

- Average
- Minimum
- Maximum
- Last

Sample Data

Primary Data Points (PDPs)

Archives (RRDs)
RRD for this project

- Monitor the time series data.
- Take care of time and space complexity.
- Very simple in structure.
- Manipulate the stored data and archived the data.
- Graphing tools.
Database and Configuration

- Database for sensor list and most of the configurations
- Text files for logic related (startup procedures, logic switches, etc)
Database

- SQL is used for the followings:
  - Table 1: restoring sensor information
    - sensor hierarchy
    - CAN id
    - sampling rate
    - rrd file reference
    - This means, all data is going to be stored in RRD, but a reference is kept in SQL as a cleaner solution
  - Table 2: type of sensor
    - analog in, analog out, digital in, digital out
    - need to know this for sending out data on CAN
Database

○ Table 3: warning/error threshold
  ■ High and low values for warnings, errors and failures
  ■ Reaching these values will trigger some certain actions, which is referred in the next table

○ Table 4: warning/error actions
  ■ Each of the actions here is generic and configurable

○ Table 5: calibration
  ■ have slop and offsets
### SQL DB: ‘Sensor_Table’

<table>
<thead>
<tr>
<th>Name</th>
<th>id_CAN</th>
<th>id_Sensor</th>
<th>Type</th>
<th>Sample_Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overwrite_Period</td>
<td>Units</td>
<td>Factor</td>
<td>Offset</td>
<td>RRD_DB</td>
</tr>
</tbody>
</table>
### SQL DB: ‘Sensor_Type’

<table>
<thead>
<tr>
<th>id</th>
<th>Description</th>
<th>Type</th>
<th>Direction</th>
</tr>
</thead>
</table>

## SQL DB: ‘Sensor_Levels’

<table>
<thead>
<tr>
<th>id</th>
<th>Warning_Low</th>
<th>Warning_High</th>
<th>Error_Low</th>
<th>Error_High</th>
<th>Fail_Low</th>
<th>Fail_High</th>
</tr>
</thead>
</table>


SQL DB: 'Sensor_Actions'

<table>
<thead>
<tr>
<th>id</th>
<th>Action_Name</th>
<th>Priority_Level</th>
<th>Effector_Name</th>
<th>Effector_State</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Name</td>
<td>id_CAN</td>
<td>id_Sensor</td>
<td>Type</td>
</tr>
<tr>
<td>----</td>
<td>-----------------------------</td>
<td>--------</td>
<td>-----------</td>
<td>---------</td>
</tr>
<tr>
<td>1</td>
<td>TSV/Pack1/Voltage</td>
<td>0</td>
<td>1</td>
<td>Analog</td>
</tr>
<tr>
<td>2</td>
<td>TSV/Pack1/Current</td>
<td>1</td>
<td>2</td>
<td>Analog</td>
</tr>
<tr>
<td>3</td>
<td>TSV/Pack1/SOC</td>
<td>2</td>
<td>3</td>
<td>Analog</td>
</tr>
<tr>
<td>4</td>
<td>TSV/Pack1/Fuse_Temperature</td>
<td>3</td>
<td>4</td>
<td>Analog</td>
</tr>
<tr>
<td>5</td>
<td>TSV/Pack1/AMS1/Temperature</td>
<td>4</td>
<td>5</td>
<td>Analog</td>
</tr>
<tr>
<td>6</td>
<td>TSV/Pack1/AMS1/Voltage</td>
<td>5</td>
<td>6</td>
<td>Analog</td>
</tr>
<tr>
<td>7</td>
<td>TSV/Pack1/AMS1/Current</td>
<td>6</td>
<td>7</td>
<td>Analog</td>
</tr>
<tr>
<td>8</td>
<td>TSV/Pack1/AMS2/Temperature</td>
<td>7</td>
<td>8</td>
<td>Analog</td>
</tr>
<tr>
<td>9</td>
<td>TSV/Pack1/AMS2/Voltage</td>
<td>8</td>
<td>9</td>
<td>Analog</td>
</tr>
<tr>
<td>10</td>
<td>TSV/Pack1/AMS2/Current</td>
<td>9</td>
<td>10</td>
<td>Analog</td>
</tr>
<tr>
<td>11</td>
<td>TSV/Pack1/AMS3/Temperature</td>
<td>10</td>
<td>11</td>
<td>Analog</td>
</tr>
<tr>
<td>12</td>
<td>TSV/Pack1/AMS3/Voltage</td>
<td>11</td>
<td>12</td>
<td>Analog</td>
</tr>
<tr>
<td>13</td>
<td>TSV/Pack1/AMS3/Current</td>
<td>12</td>
<td>13</td>
<td>Analog</td>
</tr>
<tr>
<td>14</td>
<td>TSV/Pack1/AMS4/Temperature</td>
<td>13</td>
<td>14</td>
<td>Analog</td>
</tr>
<tr>
<td>15</td>
<td>TSV/Pack1/AMS4/Voltage</td>
<td>14</td>
<td>15</td>
<td>Analog</td>
</tr>
</tbody>
</table>
Configuration

- Bash style
- Read during startup, and bad syntax will raise exceptions and the car will be disabled from driving
- Switches can be updated and modified by maintenance app
- Will be stored under same directory and database in a separate folder
- Is accessible from debug port
Acceptance Testing

● Show that VSCADA meets all requirements as both:
  ○ part of integrated LFEV system
  ○ standalone software system
● Strive for maximum amount of test automation/avoid recompiling software
● Main criteria:
  ○ Exception handling
  ○ Automated hardware detection/configuration
  ○ Logging, plotting and storing of measurands
  ○ Controlling system state
Acceptance Testing (cont.)

Test configurations:

- Config A: VSCADA powered by 12 V power source
- Config B: VSCADA interfaced with GLV
- Config C: VSCADA interfaced with GLV and TSV
- Config D: VSCADA interfaced with GLV, TSV and DYNO
Acceptance Testing (cont.)

T000 - System Startup/Shutdown and GLV Data Logging

- Config B
- Tests:
  - Automatic startup without user interaction once GLV power is provided
  - Logging of GLV measurands
  - Keeping of backup in case of unexpected shutdown
Acceptance Testing (cont.)

T001 - Safety Checking/Exception Handling

- Config D
- Tests:
  - Lighting of Ready LED on cockpit if all subsystems are in a safe state when Ready-to-Drive button pressed
  - Lighting of warning LEDs to warn user and prevent drive mode being activated by Ready-to-Drive button if unsafe condition occurs (exception handling)
  - Examples are open safety loop, voltage threshold exceeded, temperature threshold exceeded, missing config file for sensors
Acceptance Testing (cont.)

T002 - Maintenance App Operation

- Config D
- Tests:
  - Requirement of proper user credentials to login to maintenance mode
  - Logging and storing of all subsystem measurands (TSV pack/cell voltages, currents, temperatures, GLV voltage, current, Dyno torque, RPM)
  - Allowing user to control all aspects of VSCADA such as disabling safety checks, disabling data logging, and programming individual shutdown rules
Acceptance Testing (cont.)

T003 - Drive Mode Operation

- Config D, then repeat with Config A (simulated throttle)
- Tests:
  - Accurate reporting of measurands while driving
  - Logging of exceptions should unsafe condition occur while driving
  - Demo operation of vehicle through software throttle if other subsystems not available
Acceptance Testing (cont.)

T004 - Pack Charging/Discharging

- Config C
- Tests:
  - Displaying that accumulator is charging
  - Displaying that accumulator is discharging
Acceptance Testing (cont.)

T005 - Reliability Test

● Config D

● Tests:
  ○ System can run through series of drive modes/simulations and maintenance configuration changes over period of 24 hours without failure
Acceptance Testing (cont.)

T006 - Maintainability Test

- Config D
- Tests:
  - Novice user can solve frequently occurring problem
  - Expert maintenance individual can solve unexpected problem
  - New sensors can be added to system without software recompilation
  - VSCADA software can be installed easily using “make/install” on different computer
Schedule

Week 9

**Demonstration System Integration & Debugging**
System parts designed in the past six weeks will be integrated into a cohesive system demonstration for CDR, and for displaying system capabilities to other groups.

**CAN Communication PCB Fabrication**
The General Sensor CAN Communication PCB GERBER files will be ready for fabrication and sent out for production.

Week 10

**Preliminary Demonstration System**
A primitive scada system will be functioning, and ready for demonstration to other groups. This system should be capable of allowing groups to test communications between themselves and the SCADA system in the future.

Week 11

**SCADA Server Maintenance Mode**
The main system server will be capable of performing all 'Maintenance Mode' tasks, and interfacing with all 'Maintenance Mode' client interfaces.

**QA Report Submitted**
Deliverable **D006** (QA Report) will be submitted.
Schedule cont’d

Week 12

**System Integration & Debugging**
- Any remaining components not added to the SCADA system will be added at this time. Debugging and integration into other vehicle sub-systems.

**SCADA Server Demo Mode**
- The main system server will be capable of performing 'Demo Mode' tasks.

Week 13

**Final ATR Report Submitted**
- Deliverable **D005** (ATR Report) will be submitted.

**System Integration & Debugging**
- Any remaining components not added to the SCADA system will be added at this time. Debugging and integration into other vehicle sub-systems.

**Dynamometer Communication Library**
- The main system is capable of sending messages to the Huff Box over serial ports.
Schedule cont’d

Week 14

**System Integration & Debugging**
Any remaining components not added to the SCADA system will be added at this time. Debugging and integration into other vehicle sub-systems.

**System Documentation**
All project documentation will be finalized and completed.

Completed Maintenance **Manual Submitted**
A VSCADA Maintenance Manual Working Draft will be submitted.

Week 15

**Final Report & Maintenance Manual Submitted**
Deliverable **D003** (Final Report) will be submitted.

**System Errata Documentation**
Any known bugs, and system errata will be documented for use by future students.
## Budget

<table>
<thead>
<tr>
<th>Item/Group</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SCADA</strong></td>
<td></td>
</tr>
<tr>
<td>Embedden Computer System</td>
<td>270</td>
</tr>
<tr>
<td>Dashboard LCD display and controller</td>
<td>80</td>
</tr>
<tr>
<td>Wireless Radio</td>
<td>50</td>
</tr>
<tr>
<td>Slave Sensor Micro Controller Hardware</td>
<td>100</td>
</tr>
<tr>
<td>Debugger</td>
<td>80</td>
</tr>
<tr>
<td>Programmer</td>
<td>10</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>590</td>
</tr>
<tr>
<td><strong>Budget:</strong></td>
<td>715</td>
</tr>
</tbody>
</table>
Roadmap

8. Meet the Afternoon Teams
9. Interface Control Review
10. Vehicle Supervisory Control and Data Acquisition (VSCADA)
   a. Daemon
   b. Interfacing
   c. User Applications
   d. Data Storage
11. Dynamometer (DYNO)
    a. Decomposition and Definition
    b. Integration and Recomposition
Roadmap

8. Meet the Afternoon Teams
9. Interface Control Review
10. Vehicle Supervisory Control and Data Acquisition (VSCADA)
   a. Daemon
   b. Interfacing
   c. User Applications
   d. Data Storage
11. Dynamometer (DYNO)
   a. **Decomposition and Definition**
   b. Integration and Recomposition
Concepts of Operation

- Generate a torque curve
- Develop a software simulation of the car
- Develop a hardware simulation of the car
- Determine the car gear ratio
System Requirements

- Motor/Dyno Selection
- Motor Controller
- Software
  - Data Acquisition
  - Throttle Control
- Interfaces
  - VSCADA
  - GLV
  - TSV
- Safety
Motor Selection

- HPEVS AC 50-27.28
Dynamometer Selection

- Huff HTH-100
Motor Controller

- Curtis 1238R-7601
Interfaces

- VSCADA - Interface for data acquisition and throttle control
- GLV - Interface for power and data transmission
- TSV - Interface to power supply
Software

VSCADA - Dyno

● Data Acquisition
  ○ RPM
  ○ Torque
  ○ Temp - Motor and Controller
  ○ RMS Current
  ○ Voltage

● Throttle Control
Safety

● **Emergency Shutoff**
  ○ Must be have an emergency stop
  ○ Must be shut down when GLV is down

● **Oil Temperature Shutoff**
  ○ Must shut down when temperature limit is exceeded

● **Galvanic Isolation**
  ○ Must separate high and low voltage subsystems

● **Motor Controller Contact Shield**
  ○ Prevents accidental contact with terminals
High-Level Design

- ICD Layouts
- Safety Shutoff
- Throttle
- HUFF - VSCADA Interface
- Motor Controller
  - Cooling
  - Safety
- Galvanic Isolation
ICD Layout

- Two configurations
  - Dyno Testing Configuration:
ICD Layout

- Two configurations
  - Integrated Design Configuration:
Safety Shutoff

● Requirements -
  ○ Must include emergency stop
  ○ Must include temperature shutoff

● Design -
  ○ Use the power supply control inputs. These control mechanical contactors.
Huff - VSCADA interface

- USB interface
- Utilizes serial communication
- Based on a call and response system
- Used to acquire data and set values
- Protocol is defined by the chip on data acquisition board
Motor Controller Cooling

- Must regulate MC temperature
  - Storage ambient temperature range: -40°C to 95°C
  - Operating ambient temperature range: -40°C to 50°C
  - Internal heatsink operating temperature range: -40°C to 95°C

- Utilize a Water Cooling system
  - Pump → MC → Radiator → Pump
  - Mounted Cooling Housing
  - Effectiveness to be determined upon delivery of parts
Motor Controller Safety

- Must prevent conductive injury from MC ports
  - High Voltage
- Cover all electrical hazards to prevent accidental contact
  - Use non-conductive plastic cover
Throttle

- Must control the throttle input of the MC
  - Throttle input is 0 to +5 volts
  - VSCADA must be connected
  - Must be scriptable for testing

- Use two systems:
  - Use a VSCADA CANbus node with an analog output for the integrated system
  - Use an Arduino connected over USB to control an analog output
Galvanic Isolation

- High/Low voltage CAN must be separated
- High/Low Voltage Throttle must be separated
Layout Review:
Detailed Design

- **Simulations**
  - Motor
  - Car
  - Track

- **Safety**
  - Independent shutoff
  - Insulating covers

- **Throttle**
  - Independant solution
  - VSCADA solution

- **Motor Controller**
  - Isolation
  - Parameters
  - Wiring Diagram
  - Cooling
  - Safety

- **Room Wiring**
  - Testing config
  - Integrated config
Simulations - Motor

Using IEEE circuit equivalent model developed the following torque curve
Simulations - Car/Track

Car will be reduced down to a singular body in order to simplify the static and dynamic equations required.

The track will be simulated by an array of values that will dictate the curvature and incline of the track based on position.
Simulations Layout
Safety - Independent Shutoff

- Use the input control lines
  - Directly linked to mechanical contactors

- Interface is a 37-pin D-Sub connector
  - Need only the start and stop inputs

Figure 3.26. Digital input control lines
Safety - Independent Shutoff

- Solution: simple rack mounted unit
Safety - Insulating Covers

- Plastic Cover
  - Non-conductive
  - Transition temperature higher than cutoff temp
- Aluminum connecting rods
Throttle - Independent Solution

- Need to control throttle from a computer
  - Arduino with USB connection
- No analog outputs
  - Low pass filter on a PWM
- Scripting
  - Write values in a python script
Python script writes PWM values, which are filtered to analog voltages.
Throttle - VSCADA Solution

- USB connection from VSCADA to Huff Box
- Serial communication
- Call and response
- Protocol dictated by DAQ chip
- Generate PWM signal
  - Relates RPM to voltage
  - PWM is low pass filtered
Motor controller - Isolation

- Isolates motor controller from GLV systems
  - Isolate CANbus
    - Using TI ISO1050DUBR
    - Voltage step down using LM7805
  - Isolate throttle
    - Using 6N135 optocoupler
    - Low pass filter PWM signal
# Motor Controller - Parameters

<table>
<thead>
<tr>
<th>Program</th>
</tr>
</thead>
<tbody>
<tr>
<td>User Settings</td>
</tr>
<tr>
<td>Speed Settings</td>
</tr>
<tr>
<td>Forward Speed 5500 rpm</td>
</tr>
<tr>
<td>Reverse Speed 5500 rpm</td>
</tr>
<tr>
<td>Econo Speed 5500 rpm</td>
</tr>
<tr>
<td>Accel Rates</td>
</tr>
<tr>
<td>Normal Accel Rate 0.4 Seconds</td>
</tr>
<tr>
<td>Econo Accel Rate 1.0 Seconds</td>
</tr>
<tr>
<td>Throttle Settings</td>
</tr>
<tr>
<td>Throttle Type 2</td>
</tr>
<tr>
<td>Deadband 0 Volt</td>
</tr>
<tr>
<td>Throttle Max 5 Volt</td>
</tr>
<tr>
<td>Mapped Throttle 50 %</td>
</tr>
<tr>
<td>Brake Pedal Settings</td>
</tr>
<tr>
<td>Brake Type 0</td>
</tr>
<tr>
<td>Brake Deadband 0.30 Volt</td>
</tr>
<tr>
<td>Brake Max 3.50 Volt</td>
</tr>
<tr>
<td>Regen Brake Light Th. 50 Ampere</td>
</tr>
<tr>
<td>Current Limits</td>
</tr>
<tr>
<td>Normal Neutral Braking 15 %</td>
</tr>
<tr>
<td>Econo Neutral Braking 25 %</td>
</tr>
<tr>
<td>Shift Neutral Braking 7 %</td>
</tr>
<tr>
<td>Normal Drive Current 100 %</td>
</tr>
<tr>
<td>Econo Drive Current 60 %</td>
</tr>
<tr>
<td>Brake Current Limit 10 %</td>
</tr>
<tr>
<td>Idle Enable</td>
</tr>
<tr>
<td>Idle Enable Off</td>
</tr>
<tr>
<td>Clutch Start Enable Off</td>
</tr>
<tr>
<td>Idle Speed 600 rpm</td>
</tr>
<tr>
<td>Idle Torque 50 %</td>
</tr>
<tr>
<td>Creep Torque 0 %</td>
</tr>
<tr>
<td>Motor Tuning</td>
</tr>
<tr>
<td>Motor Type 50</td>
</tr>
<tr>
<td>Base Speed 3000 rpm</td>
</tr>
<tr>
<td>Field Weakening 50 %</td>
</tr>
<tr>
<td>Econo Field Weakening 20 %</td>
</tr>
<tr>
<td>Weakening Rate 60 %</td>
</tr>
<tr>
<td>Main Contactor</td>
</tr>
<tr>
<td>Main Contactor Voltage 48 Volt</td>
</tr>
<tr>
<td>Main Holding % 80 %</td>
</tr>
<tr>
<td>Display Menu Items</td>
</tr>
<tr>
<td>Auto Scroll On</td>
</tr>
<tr>
<td>Scroll Delay Time 10 Seconds</td>
</tr>
<tr>
<td>Display SOC Off</td>
</tr>
<tr>
<td>Display Motor RPM On</td>
</tr>
<tr>
<td>Display Battery Amps On</td>
</tr>
<tr>
<td>Display Voltage On</td>
</tr>
<tr>
<td>Display Motor Temp On</td>
</tr>
<tr>
<td>Display Controller Temp On</td>
</tr>
<tr>
<td>Display Minimum Volt... On</td>
</tr>
<tr>
<td>Display Maximum Cur... On</td>
</tr>
<tr>
<td>BMS</td>
</tr>
<tr>
<td>BMS Installed Off</td>
</tr>
<tr>
<td>BMS Address 768</td>
</tr>
<tr>
<td>User Undervoltage 80 %</td>
</tr>
<tr>
<td>Low Cell Begin Cutback 2.800 Volt</td>
</tr>
<tr>
<td>Low Cell Full Cutback 2.300 Volt</td>
</tr>
<tr>
<td>Max Current at Full C... 59 %</td>
</tr>
<tr>
<td>Maximum Cel Voltage 3.703 Volt</td>
</tr>
<tr>
<td>Low SOC Cutback 20 %</td>
</tr>
<tr>
<td>Max Current at Low... 30 %</td>
</tr>
<tr>
<td>Dual Drive</td>
</tr>
<tr>
<td>Dual Drive Mode Off</td>
</tr>
<tr>
<td>Response Timeout 200 ms</td>
</tr>
</tbody>
</table>
Motor Controller - Wiring

Testing w/ Power Supply
Motor Controller - Wiring

Testing w/ TSV Packs
Motor Controller - Wiring

Suggested in-vehicle wiring
Motor Controller - Cooling
Room Wiring - Testing Config
Room Wiring - Integrated Config
System Integration

- Two configurations
  - Dyno Testing Configuration
  - Integrated Design Configuration
Dynamometer

● Dyno Testing Configuration
  ○ Utilizes Windows PC with software
  ○ Records RPM and Torque
  ○ Controls dynamometer

● Integrated Design Configuration
  ○ Utilizes VSCADA computer
  ○ Records RPM and Torque
  ○ Controls dynamometer
System Wide

● VSCADA
  ○ CAN data acquisition
  ○ Dyno data acquisition and control
  ○ Throttle

● TSI
  ○ Galvanic isolation from TSV packs

● TSV
  ○ Supplies power
Budget

- **Wires/Tubing**: $203.92
  - Water cooling tubes - $15.30
  - Power supply cables - $188.62

- **Transistor/Electrical Parts**: $43.22
  - Button switches - $15.76
  - Transistors - $27.46

- **Connectors**: $108.52
  - Panels - $107.37
  - Pin Connector - $1.15

- **Total**: $362.19
Roadmap

8. Meet the Afternoon Teams
9. Interface Control Review
10. Vehicle Supervisory Control and Data Acquisition (VSCADA)
   a. Daemon
   b. Interfacing
   c. User Applications
   d. Data Storage
11. Dynamometer (DYNO)
   a. Decomposition and Definition
   b. Integration and Recomposition
Unit/Device Testing

- Throttle Testing
- Safety Shutoff
- Galvanic Isolation
Throttle Testing

● Independant Solution - T001-2
  ○ Using Arduino system
  ○ Sweep the throttle in increments of 100 RPM
  ○ Value must be within 5 RPM with 90% confidence

if complete:

● VSCADA Solution - T002-3
  ○ Using VSCADA system
  ○ Sweep the throttle in increments of 100 RPM
  ○ Value must be within 5 RPM with 90% confidence
Safety Shutoff

● Emergency Stop - T000-1
  ○ Press the stop, check that the system powers down

● Oil Temp. Shutoff - T000-3
  ○ Heat up sensor, check that the system powers down
Subsystem Verification

- Data Acquisition
  - Verification tests for:
    - Torque
    - Velocity
    - Current
    - Voltage
    - Temperature
    - Load control

- Simulation Results Comparison
Data Acquisition

- Verify sensor accuracy - T001-1
  - Torque - verified with first principles
    - Calibrate with weights on the arm
    - Verify calibration with different weights
Data Acquisition

- Verify sensor accuracy - T001-2
  - Motor Velocity - redundant measurements
    - Dynamometer encoder
    - Motor encoder
    - Handheld tachometer
  - Verified statistically
Data Acquisition

- Verify sensor accuracy - T001-3
  - Motor Current - redundant measurements
    - Motor controller output
    - Clamp sensor
  - Verified statistically

http://www.hiokiusa.com/images/products/m9709.gif
Data Acquisition

- Verify sensor accuracy - T001-4
  - Motor Voltage - redundant measurements
    - Motor controller output
    - Power supply reading
  - Verified statistically

http://www.magna-power.com/products/programmable-dc-power-supplies/ts-series
Data Acquisition

- Verify sensor accuracy - T001-5
  - System Temperature - redundant measurements
    - Motor controller output (Motor/Controller temp.)
    - Handheld sensor
  - Verified statistically
Data Acquisition

• Verify sensor accuracy - T001-6
  ○ Load Variance - check torque response
    ■ Use steady motor RPM
    ■ Vary load and check that torque varies

http://curriculum.vexrobotics.com/sites/default/files/7.7.1%20Torque%20vs.%20Speed.PNG
Simulation Results Comparison

Create scripts to mimic the MATLAB simulations in hardware. This will be done in automatically controlling the valve opening in the dynamometer.
System Verification

● VSCADA Tests  
  ○ Data acquisition  
  ○ Throttle control

● TSI Tests  
  ○ Checks for galvanic isolation

● TSV Pack Tests  
  ○ With 4 packs  
  ○ With 1 pack
VSCADA Tests

- VSCADA data acquisition via CAN
- VSCADA data acquisition from Huff Box
- VSCADA throttle control
  - Uses data from CAN and Huff Box
  - Control Dynamometer valve via Huff Box
TSI Tests

- Run the dynamometer with the TSI attached
  - T002-4 (assumes the TSI works properly)
  - If the system powers down:
    - It is not galvanically isolated
  - If the system runs:
    - It is properly isolated
TSV Tests

- Run the dynamometer with TSV power
  - If 4 packs have been completed:
    - Connect the packs to the TSI
  - If only 1 pack has been completed:
    - Connect the pack in series with the power supply
    - Power supply will make up voltage difference
System Validation

- Gear Ratio
- Torque Curve
- Final Demonstration
Gear Ratio

Knowing the maximum rpm and our desired maximum velocity we can solve for the gear ratio
Torque Curve

- At minimum:
  - Must show velocity vs. torque at an estimated load

- Goal:
  - Will show velocity vs. torque at several load points
  - Will show power consumption

- Ideal:
  - 3D graph of velocity, torque, and load
Budget

- Initially allocated money:
  - Dyno - $148
  - SCADA - $715
  - GLV - $1397.90
  - TSV - $2739.10
Budget

- Money Spent so far
  - Dyno - $362
  - TSV - $1026.03
  - SCADA - $304.26
  - GLV - $10.24
Budget

Current status of money:

![Budget Status Graph](image)