Rapid Battery Exchange
For Electric Vehicles

By

Michael Cocchi

Adam Rizkalla

Senior Project

ELECTRICAL ENGINEERING DEPARTMENT

California Polytechnic State University

San Luis Obispo

2012

TABLE OF CONTENTS

LIST OF TABLES AND FIGURES

Tables Page

Figures

Abstract

The range of most electric vehicles is 100-200 miles. When their batteries are drained, it will take at least 30 minutes at best to charge their batteries to 80% capacity. This does not compare with a gas powered vehicle that can go 300 miles and then fill up in a matter of minutes. A possible answer to this problem is to exchange the batteries in the car. Completing a 5 minute exchange would be the equivalent of a gas car filling up. There are no battery exchange systems for electric vehicles today. Our senior project aims to correct this problem. By proving that a battery exchange is possible for electric cars, they become a more viable alternative to gas powered cars.

We hope to take a G-Van and perform a battery exchange. The van is currently powered with a string of 12V batteries that produce 216V. When the pack has been discharged to a reasonably low level, the van will drive up on the ramp and complete the battery exchange. With a set of charged batteries, the van will be able to continue driving. The pack that was left in the ramp will be charged so that it can be exchanged later.

I. Introduction

The most commonly used fuel used today is gasoline. But, with the price ever increasing, it is time to start looking for alternative methods. Electric vehicles seem to be the obvious alternative, but their range is extremely limited. Battery technology is currently unable to compete with a tank of gas. The range of most gasoline powered cars is in excess of 300 miles. When the tank is empty, they can just drive into the nearest gas station and fill up.

The range of most electric vehicles is 100-200 miles. In order to get that range, it takes about 8 to 14 hours to charge the batteries to full capacity during level one charging. Level two charging can cut that time in half. Some quick charge stations can even charge the batteries in as little as 30 minutes. Each of these charging options will be unable to sustain long range driving trips. With quick charging out of the question, the next viable option would be to exchange dead batteries for charged ones.

A battery exchange would consist of taking charged batteries and exchanging them discharged ones. The batteries will be removed from the vehicle using a hydraulic lift. Once removed from the vehicle, a new set will be moved into place and lifted into the vehicle. Once inside the battery will be secured into the vehicle. With the new battery in place, the vehicle will be able to continue driving. The old battery will be charged so it can be used in another exchange.

The packs will be put into the carts by the lift. Once the battery pack is in the cart it will be moved along a set of rails by an electric motor. The carts will either place a battery under the chargers to be charged or over the lift to accept a depleted battery. The batteries will be charged in three 72v sections to speed up charge times.

The Rapid Battery Exchange project was designed and completed by members of the electric vehicle club of Cal Poly. We chose this project in order to increase the viability of electric vehicles. With a working RBX, electric vehicles will be able to better compete with gas powered vehicles.

This report will discuss the hardware and software implementations of the RBX system. This includes state diagrams of the van and the ramp, microcontroller code, external inputs to the microcontroller and reading of the current state of the battery packs.

II. Background

In the past few years, the use of battery powered electric vehicles (EVs) has grown in popularity and feasibility. The first types of EVs to become commercially available were hybrid vehicles, such as the Toyota Prius, which combined the use of fossil fuels and some form of electric propulsion [1]. The gasoline engines in these vehicles are typically smaller than their conventional counterparts, and contain at least one auxiliary electric motor. These vehicles still require filling up at a gas station, but the MPG is significantly increased due to the added electric motor(s) (the Prius averages 50 MPG [2]). The Toyota Prius accomplishes this by utilizing the added power of the electric motor in some driving cases, such as climbing hills and accelerating from rest. The engine is always kept running at an efficient speed and load, and the Prius can accelerate to 15mph before even switching on the engine [3].

With the increasing technology of today's society, fully electric vehicles have been commercially developed. One of the more prominent electric vehicle manufacturers is Tesla Motors. Tesla was founded in 2003, and has grown to 21 dealerships worldwide with vehicles sold in 32 different countries [4]. The first model released was the Tesla Roadster, which was in production from 2008 to 2012. This was followed by the Model S sedan, set to come on the market sometime this year, and the Model X SUV, expected in 2014 [4]. The Roadster advertises 245 miles per charge, with a fast recharge time of 4 to 10 hours, depending on the type of connector used. This recharge time is typical of most electric vehicles, causing a huge amount of down time between drivable periods of the vehicle. With such an issue of increased recharge time of electric vehicles compared to the refueling time of conventional combustion engine vehicles, the need for a quicker alternative to battery charging has become evident.

This is where battery exchanging units and stations has come into play as a viable solution to this proposed problem. The battery pack of an electric vehicle, in theory, can easily be swapped out for a fully charged battery pack in a much more efficient time than it requires to fully charge a dead battery pack. This also allows the dead battery pack to be charged elsewhere while the electric vehicle is still able to be driven with a separate battery pack. Battery swapping stations could become the new replacement for gasoline stations, where electric vehicles could come "refill their tanks" by simply swapping in a new battery pack. A company known as Better Place has already constructed a prototype battery exchange system which they advertise can replace a dead battery in an electric vehicle in less time than it takes to fill a gas tank [5]. With this new innovation, electric vehicles can become not only more sustainable and efficient, but also more practical to use in replacement of gas-powered vehicles.

III. Requirements and Specifications

1. The van will notify the driver when the ramp is ready to begin and has completed an exchange.
2. The microcontroller (MC) will read and adjust the position of the linear actuators.
3. The MC will communicate wirelessly through the XBee module with the apparatus.
4. The apparatus software will correctly operate the lift and cart motor controls.
5. The apparatus software will communicate with the van side wirelessly through the XBee module.
6. The flow or the RBX will follow the state diagrams.
7. Both sides will contain safety checks and precautions, such as watchdog timers, and error statuses to report any malfunctions.
8. The software will be tested using manual overrides for the necessary switches in order to reach full functionality of the software implementation.
9. Switches integrated with the ramp that will notify the microcontroller of the van and cart position.
10. A set of switches that will detect when the lift is raised and lowered.
11. The MC will be notified when a pack is on the lift.
12. An external control box that will display the voltage of the pack, as well as contain the emergency stop button and manual overrides for the system.
13. A driver button in the van to begin the battery exchange.
14. The routing of all of the switches in the van to the microcontroller.
15. An interface between the chargers and the microcontroller.

IV. Design

The design for this system consists of two major components: the ramp apparatus and the electric vehicle (GMC G-Van). The ramp apparatus and the van both contain a circuit board with a microcontroller and a wireless communication module which allows the ramp and van to communicate with one another during the battery exchange. The ramp microcontroller takes input from switches and buttons and controls a lift housed in the ramp and a motor which moves the physical carts which hold the battery packs. The van contains a receptacle under it which holds the battery pack using linear actuators to support it. The linear actuators have built in potentiometers which are used to determine the current position. The van microcontroller controls the linear actuators, takes input from a switch in the receptacle and the driver, as well as notifies the driver of the current state of the battery exchange.

The general design flow of the system is as follows:

- The van will drive up onto the ramp to a known position.
- The van will notify the driver that the battery exchange is ready.
- The driver will send some input to the van microcontroller to initiate the battery exchange.
- A lift will come up from the ramp to support the battery pack under the van.
- The linear actuators in the van's receptacle will disengage from the battery pack.
- The battery pack will be lowered on the lift into a cart.
- A motor will move the carts, moving the old battery over and a new battery above the lift.
- The lift will come up again to support the battery pack in place under the van.
- The linear actuators in the van's receptacle will engage on the new battery pack.
- The lift will lower back into the ramp, and the battery exchange process will be completed.

The microcontrollers chosen for this project are the ATmega32U4. Two of the breakout boards for this chip made by Adafruit Industries were purchased. The wireless communication modules used are the XBee antennas placed in the XBee Adapter also provided by Adafruit. The breakout board for the ATmega32U4 contains a 3V and 5V rail, two ground pins, an analog voltage reference pin, and 25 GPIOs, some of which can be used for other functionality (such as ADCs and external interrupts).

Figure 1: ATmega32U4 Breakout Board

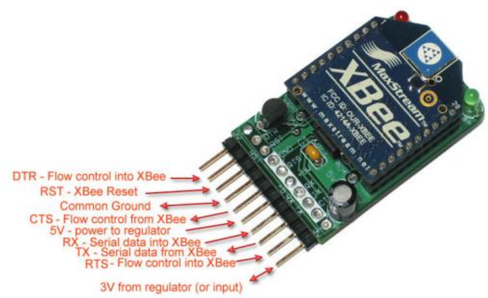


Figure 2: XBee Adapter Kit with XBee

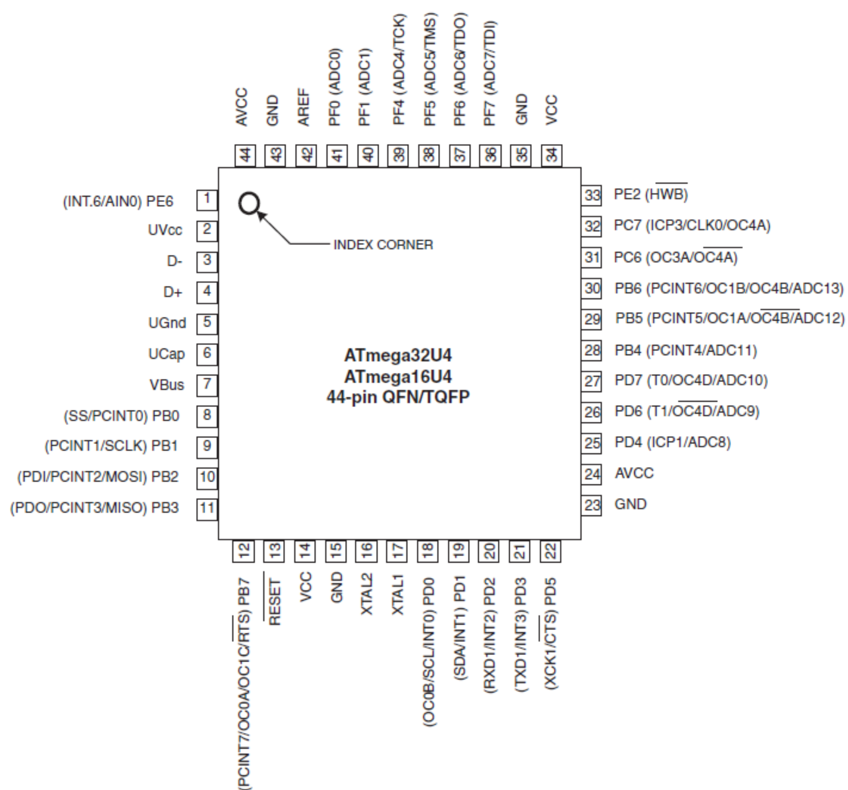


Figure 3: ATmega32U4 Pin Diagram (from Atmel datasheet)

The following tables show the pin diagrams for the microcontroller breakout boards for the ramp and the van. Some pins are still unused and can be used for future additions, though the unused pins are very limited. The decisions for pin assignments were mostly based on the various pin functionalities, such as interrupts and the available analog to digital converters (ADCs).

Connection	Pin	Pin	Connection
–	3V	5V	–
Carts Forward Switch	E6	A	–
Manual Lift Control (D)	B0	F0	–
Manual Lift Control (U)	B1	F1	–
Manual Motor Control (F)	B2	F4	–
Manual Motor Control (R)	B3	F5	Battery Check (String 1)
Lift Down Switch	D0	F6	Battery Check (String 2)
Carts Rear Switch	D1	F7	Battery Check (String 3)
XBee TX	D2	C7	Motor Power
XBee RX	D3	C6	Move Carts (F/R)
–	D4	B7	Ready LED
Error LED	D5	B6	Left Wheel Switch
Lower Lift	D6	B5	Right Wheel Switch
Raise Lift	D7	B4	Emergency Stop Button
–	GND	GND	–

Table 1: ATmega32U4 Breakout Board Pin Assignments for Ramp

Connection	Pin	Pin	Connection
–	3V	5V	–
Driver Start Button	E6	A	–
–	B0	F0	Done LED
–	B1	F1	Error LED
Actuator 4 Out	B2	F4	Actuator 1 ADC
Actuator 4 In	B3	F5	Actuator 2 ADC
Actuator 3 Out	D0	F6	Actuator 3 ADC
Actuator 3 In	D1	F7	Actuator 4 ADC
XBee TX	D2	C7	In Progress LED
XBee RX	D3	C6	Ready LED
Actuator 2 Out	D4	B7	–
Actuator 2 In	D5	B6	–
Actuator 1 Out	D6	B5	–
Actuator 1 In	D7	B4	Lift Raised Switch
–	GND	GND	–

Table 2: ATmega32U4 Breakout Board Pin Assignments for Van

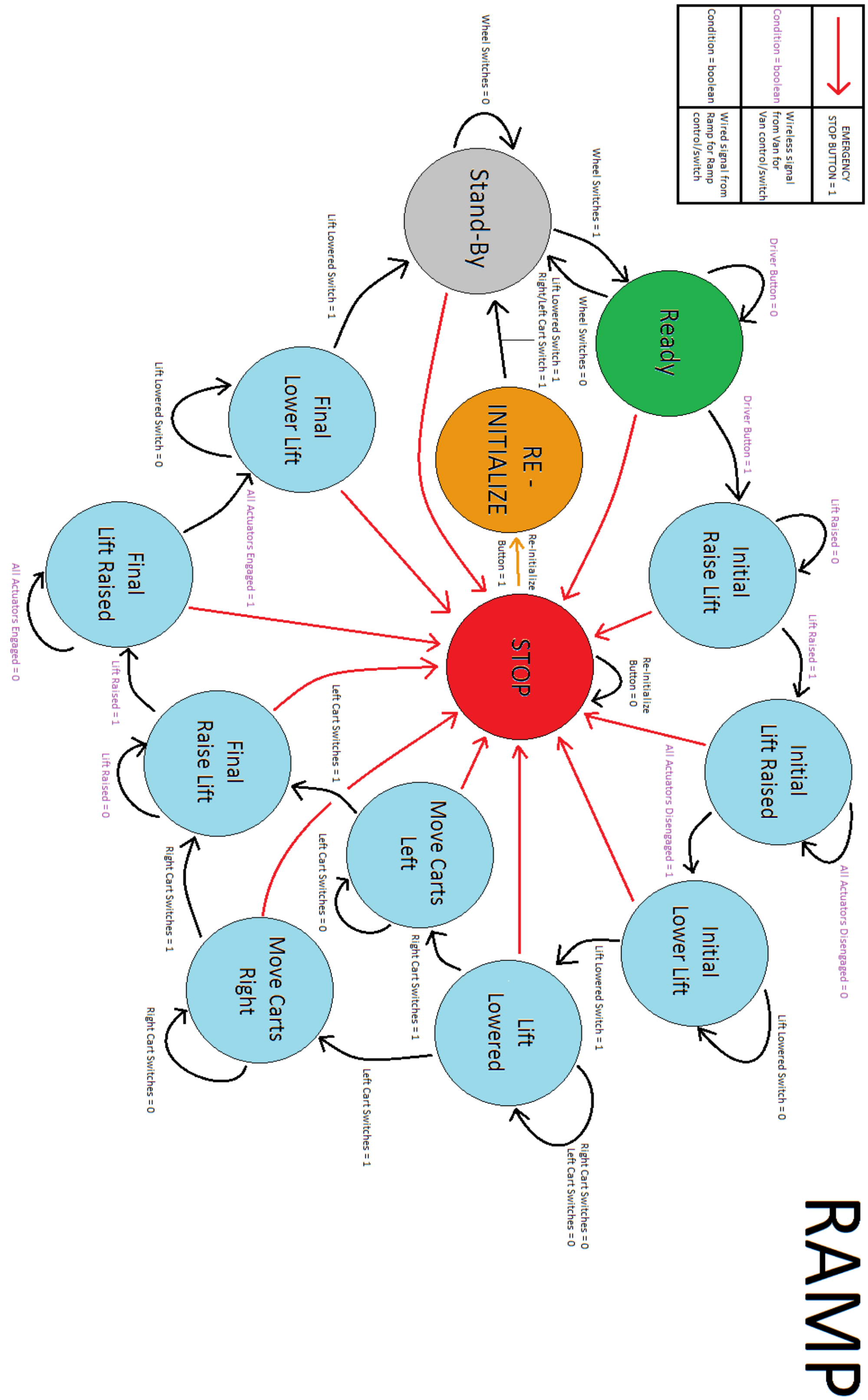


Figure 4: Ramp State Diagram (Rev. 1)

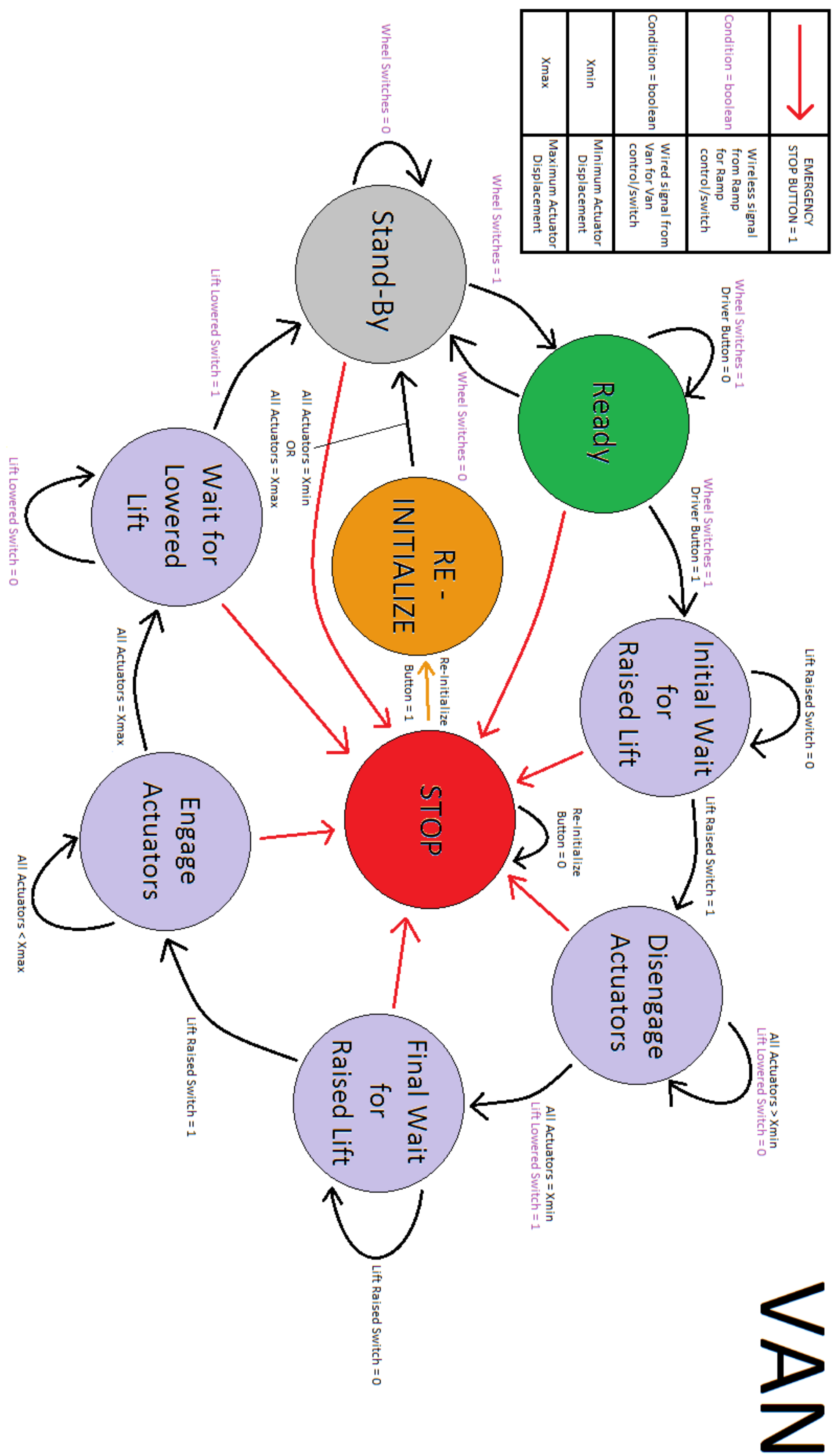


Figure 5: Van State Diagram (Rev. 1)

The state diagrams on the previous pages outline the original revision of the control flow for both the ramp and the van. This was the initial design for the embedded software on the microcontroller for the ramp and the van, respectively, which was modified in another revision (discussed later in this report). State changes in black text are information from signals contained within the system, while state changes in purple text are information received through wireless communication with the opposing component of the RBX system (i.e. wireless information from the ramp to the van and vice versa). The red arrows represent the emergency stop button being engaged, which will immediately change the current state to the Emergency Stop state on both the ramp and the van (the ramp receives the signal from the button being engaged, goes to the Emergency Stop state, and then tells the van to go into Emergency Stop as well), regardless of the current state the system is in. Once the system is in Emergency Stop, a re-initialize button (or reset button) could be pushed to put the system back into Stand-By. The control flow as well as details about each state is outlined in more detail here.

Ramp Control Flow (Revision 1)

The system will begin in stand-by upon powering up.

Stand-By

The ramp will remain idle in Stand-By.
NS -> *Ready*: Ramp receives signal from both wheel switches (right and left) being engaged.

Ready

Ramp continually sends wireless signal to van indicating that it is ready for a battery exchange.
NS -> *Stand-By*: Ramp receives signal from either wheel switch being disengaged.
NS -> *Initial Raise Lift*: Ramp receives wireless signal from van indicating driver has pressed the start button.

Initial Raise Lift

Ramp apparatus raises lift until it is supporting the weight of the battery pack, allowing van actuators to disengage. Ramp will receive signal from van when lift is fully raised.
NS -> *Initial Lift Raised*: Ramp receives signal from van that lift raised switch has been engaged.

Initial Lift Raised

Ramp waits for signal from van that all actuators have been disengaged.
NS -> *Initial Lower Lift*: Ramp receives signal from van that all (4) actuators are disengaged.

Initial Lower Lift

Ramp apparatus lowers lift until it is fully lowered.
NS -> *Initial Lower Lift*: Ramp receives signal from the lift lowered switch being engaged.
Lift Lowered

Ramp apparatus checks cart switches to determine carts positioning. Determines which direction to move the carts based on current positioning.
NS -> *Move Carts Left*: Right cart switch engaged, left cart switch disengaged.
NS -> *Move Carts Right*: Left cart switch engaged, right cart switch disengaged.

Move Carts Left/Move Carts Right

Ramp apparatus signals cart motor to move the battery packs.
NS -> *Final Raise Lift*: Cart switch on opposite side (right or left) is engaged.

Final Raise Lift

Ramp apparatus raises lift until the battery pack into position. Ramp will receive signal from van when lift is fully raised.
NS -> *Final Lift Raised*: Ramp receives signal from van that lift raised switch has been engaged.

Final Lift Raised

Ramp waits for signal from van that all actuators have been engaged.
NS: Ramp receives signal from van that all (4) actuators are disengaged.

Final Lower Lift

Ramp apparatus lowers lift until it is fully lowered. Ramp will send wireless signal to van when lift is lowered to verify battery exchange is complete.
NS -> *Stand-By*: Ramp receives signal from the lift lowered switch being engaged.

Van Control Flow (Revision 1)

Stand-By

Van remains idle in stand-by.
NS -> *Ready*: Van receives wireless signal from ramp that the front wheels are in place and the battery exchange is ready to take place.

Ready

Van signifies to driver that battery exchange is ready to commence. Van waits for driver to initiate battery exchange by pressing the start button.
NS -> *Stand-By*: Van receives wireless signal from ramp to go back to Stand-By.
NS -> *Initial Wait for Raised Lift*: Driver pushes start button to initiate battery exchange process.
Initial Wait for Raised Lift

Van waits for the lift to be in position. Van will notify ramp when lift is fully raised.
NS -> *Disengage Actuators*: Van receives signal from lift raised switch being engaged. Van sends wireless signal to ramp to notify that lift raised switch has been engaged.

Disengage Actuators

Van disengaged actuators to their minimum displacement in the receptacle. Van sends wireless signal to ramp for each actuator that is successfully disengaged.
NS -> *Final Wait for Raised Lift*: Van receives signal from each actuator that they are fully disengaged.

Final Wait for Raised Lift

Van idles while the battery packs are swapped in the ramp.
NS -> *Engage Actuators*: Van receives signal from lift raised switch being engaged.
**The Van will be waiting in this state longer than most states while the battery packs are swapped in the ramp

Engage Actuators

Van engages actuators to their maximum displacement in the receptacle. Van sends wireless signal to ramp for each actuator that is successfully engaged.
NS -> *Wait for Lowered Lift*: Van receives signal from each actuator that they are fully engaged.

Wait for Lowered Lift

Van reconnects new battery pack and waits for the lift to be lowered.
NS -> *Stand-By*: Van receives wireless signal from ramp that the lift lowered switch has been engaged.

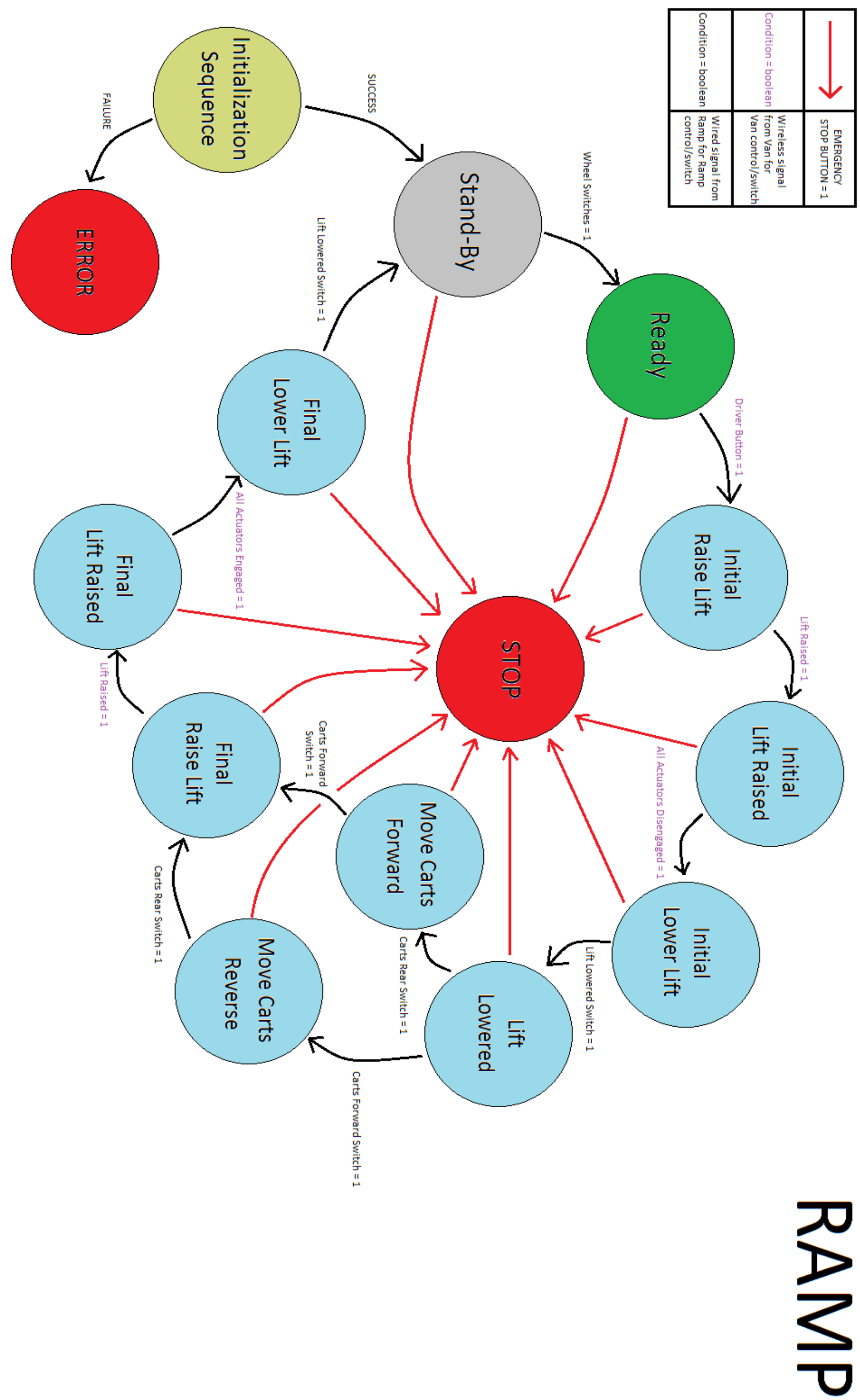


Figure 6: Ramp State Diagram (Rev. 2)

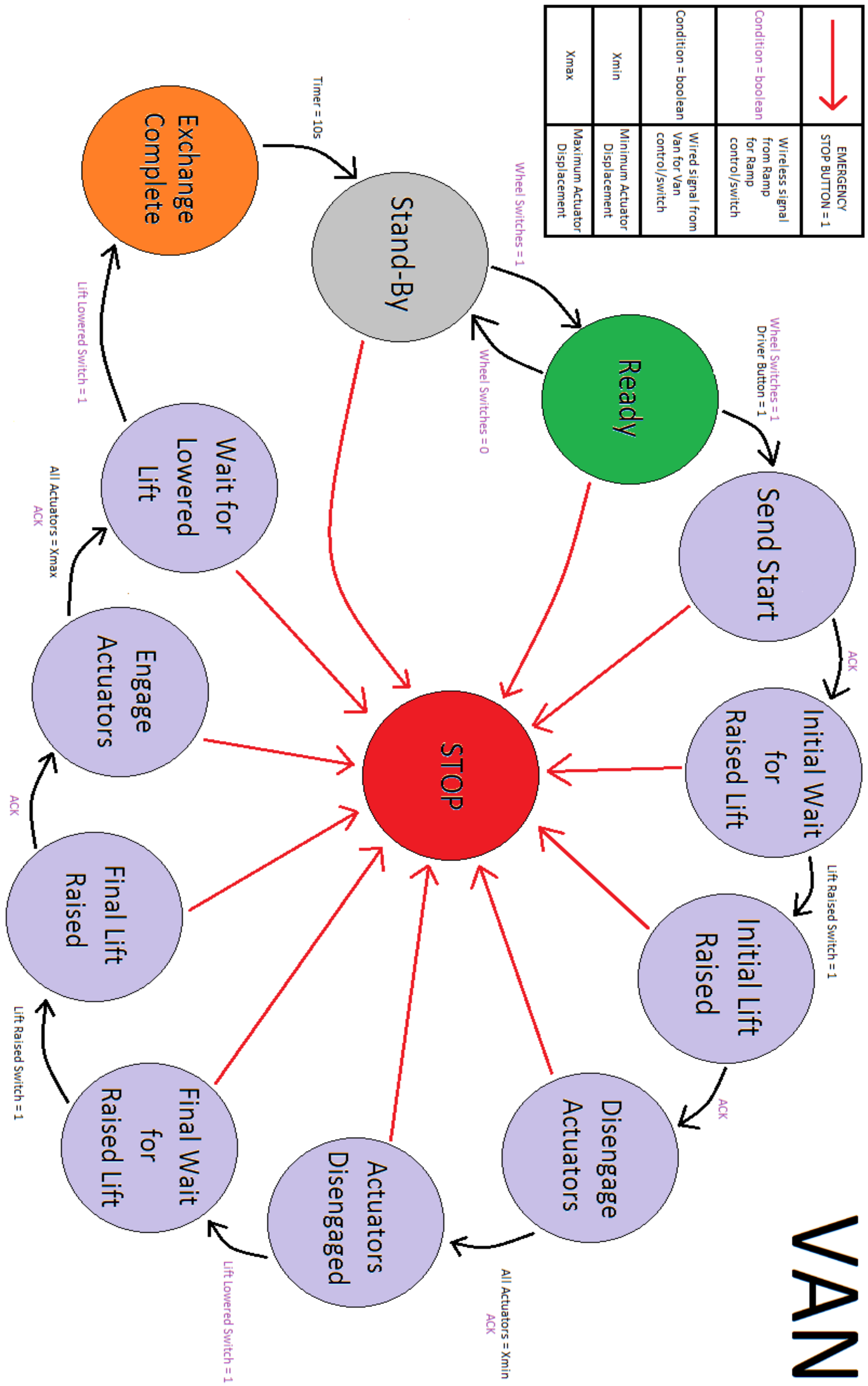


Figure 7: Van State Diagram (Rev. 2)

The state diagrams on the previous pages outline the second revision of the control flow for both the ramp and the van. This design for the embedded software on the microcontroller for the ramp and the van, respectively, evolved from the original design to account for race conditions due to wireless transmission signals and add implementation for an initialization and battery sanity check in the ramp. The re-initialize state was deprecated due to an addition of manual override switches in the ramp. The manual override switches will be contained in an external box on the driver's side of the ramp, which will also contain the emergency stop button, indicator LEDs and pack voltage displays.

The initialization sequence for the ramp is as follows:

- Ramp determines how the battery packs are currently oriented by checking the cart switches.
- If both switches are engaged, it will go into an error state. If neither switch is engaged, the ramp apparatus will drive the motor to orient the carts into a default position.
- The ramp now checks if there is currently a battery pack on the side of the engaged switch by checking the analog signals from the charging contacts.
- The ramp apparatus takes record of whether or not there is a battery on the current cart that was checked, then drives the motor to orient the carts into the opposite positioning.
- The ramp now checks if there is currently a battery pack on the side of the newly engaged switch by checking the analog signals from the charging contacts.
- If the ramp detects that both carts contain a battery pack (i.e. there are two battery packs in the ramp), or that neither cart has a battery pack (i.e. no battery packs in the ramp), it will go into an error state. If the ramp sees that there is only one battery pack in the whole system, the ramp will ensure that the battery pack is under the charging contacts and the empty cart is positioned above the lift in preparation for a battery exchange. If this initialization sequence is successful, the ramp will then continue into Stand-By.

In addition to the initialization sequence, the ramp control flow also notifies the van of received signals by sending an ACK byte back to the van. This notifies the van that the ramp has successfully received the currently transmitted signal, and allows the van to move forward in the state diagram. This signal can be seen in the van state diagram.

It was also considered a good idea in this revision to add an additional error check which occurs in any state after the Ready state. If at any point in the battery exchange process one of the wheel switches is disengaged, the system will go into the Emergency Stop state and report an error.

Ramp Control Flow (Revision 2)

The system will begin in the initialization sequence on power up.

Initialization Sequence

The ramp will run through the initialization sequence (described above).
NS -> *Stand-By*: Ramp successfully runs through initialization sequence, determining which cart holds a battery pack, and finally orients the carts so that the empty cart is above the lift and the battery pack is under a charging station.
NS -> *Error*: Ramp recognized a failure in some step of the initialization sequence (i.e. multiple battery packs, no battery packs, multiple cart switches engaged, etc.).

Stand-By

The ramp will remain idle in Stand-By.
NS -> *Ready*: Ramp receives signal from both wheel switches (right and left) being engaged.

Ready

Ramp continually sends wireless signal to van indicating that it is ready for a battery exchange. When the ramp has received the start signal from the van, it will reply with an ACK before continuing.
NS -> *Stand-By*: Ramp receives signal from either wheel switch being disengaged.
NS -> *Initial Raise Lift*: Ramp receives wireless signal from van indicating driver has pressed the start button. Ramp replies to van with an ACK byte before proceeding to next state.

Initial Raise Lift

Ramp apparatus raises lift until it is supporting the weight of the battery pack, allowing van actuators to disengage. Ramp will receive signal from van when lift is fully raised. Ramp replies to van with an ACK confirming it has received the signal for lift is fully raised.
NS -> *Initial Lift Raised*: Ramp receives signal from van that lift raised switch has been engaged. Ramp replies with an ACK byte before proceeding to next state.

Initial Lift Raised

Ramp waits for signal from van that all actuators have been disengaged. Ramp replies to van with an ACK confirming it has received the signal for all (4) actuators being disengaged.
NS -> *Initial Lower Lift*: Ramp receives signal from van that all (4) actuators are disengaged. Ramp replies with an ACK byte before proceeding to next state.

Initial Lower Lift

Ramp apparatus lowers lift until it is fully lowered. Ramp sends signal to van when lift is fully lowered. NS -> *Initial Lower Lift*: Ramp receives signal from the lift lowered switch being engaged. Ramp sends wireless signal to van that the lift lowered switch has been engaged.

Lift Lowered

Ramp apparatus checks internal state configuration to determine cart positioning and confirms by checking cart switches. Determines which direction to move the carts based on current positioning.
NS -> *Move Carts Forward*: Rear cart switch engaged, forward cart switch disengaged.
NS -> *Move Carts Reverse*: Forward cart switch engaged, rear cart switch disengaged.

Move Carts Forward/Move Carts Reverse

Ramp apparatus signals cart motor to move the battery packs.
NS -> *Final Raise Lift*: Cart switch on opposite side (forward or rear) is engaged.

Final Raise Lift

Ramp apparatus raises lift until the battery pack is into position. Ramp will receive signal from van when lift is fully raised. Ramp replies to van with an ACK confirming it has received the signal for lift is fully raised.
NS -> *Final Lift Raised*: Ramp receives signal from van that lift raised switch has been engaged. Ramp replies with an ACK byte before proceeding to next state.

Final Lift Raised

Ramp waits for signal from van that all actuators have been engaged. Ramp replies to van with an ACK confirming it has received the signal for all (4) actuators being engaged.
NS: Ramp receives signal from van that all (4) actuators are disengaged. Ramp replies with an ACK byte before proceeding to next state.

Final Lower Lift

Ramp apparatus lowers lift until it is fully lowered. Ramp will send wireless signal to van when lift is lowered to verify battery exchange is complete.
NS -> *Stand-By*: Ramp receives signal from the lift lowered switch being engaged. Ramp tells van that battery exchange is complete.

The van control flow required additional states to be added to the state diagram (mainly for waiting to receive an ACK from the ramp before proceeding). There was also an issue in the initial design where the van would immediately jump from disengaging the actuators to re-engaging the actuators. It was found out that this was due to the fact that the lift raised switch did not have time to become depressed before the van went into a state of waiting for the lift to raise a second time. For this reason, an additional intermediate state was added in which the van waits to receive a signal from the van notifying that the lift is all the way down before proceeding to wait for the lift to raise again. A final state signifying a completed battery exchange has been added as well, which will notify the driver for ten seconds after the battery exchange process is completed before going back to Stand-By. The dashboard of the van will be modified to contain and house the driver start button and indicator LEDs.

<u>Van Control Flow (Revision 2)</u>

Stand-By

Van remains idle in stand-by.
NS -> *Ready*: Van receives wireless signal from ramp that the front wheels are in place and the battery exchange is ready to take place.

Ready

Van signifies to driver that battery exchange is ready to commence.
NS -> *Stand-By*: Van receives wireless signal from ramp to go back to Stand-By.
NS -> *Send Start*: Driver pushes start button to initiate battery exchange process.

Send Start

Van continually sends start signal to ramp and waits to receive ACK from ramp.
NS -> *Initial Wait for Raised Lift*: Van receives ACK byte from ramp to proceed.

Initial Wait for Raised Lift

Van waits for the lift to be in position.
NS -> *Initial Lift Raised*: Van receives signal from lift raised switch being engaged.

Initial Lift Raised

Van tells ramp that the lift is fully raised and that it should stop the lift.
NS -> *Disengage Actuators*: Van receives ACK byte from ramp notifying it has received the lift raised signal.

Disengage Actuators

Van disengaged actuators to their minimum displacement in the receptacle. Van sends wireless signal to ramp for each actuator that is successfully disengaged.
NS -> *Actuators Disengaged*: Van receives signal from each actuator that they are fully disengaged. Van receives ACK byte from ramp that it has been notified about all actuators.

Actuators Disengaged

Van waits for lift to be lowered.
NS -> *Final Wait for Raised Lift*: Van receives wireless signal from ramp that the lift lowered switch has been engaged.

Final Wait for Raised Lift

Van waits for the lift to be in position.
NS -> *Final Lift Raised*: Van receives signal from lift raised switch being engaged.
**The Van will be waiting in this state longer than most states while the battery packs are swapped in the ramp

Final Lift Raised

Van tells ramp that the lift is fully raised and that it should stop the lift.
NS -> *Engage Actuators*: Van receives ACK byte from ramp notifying it has received the lift raised signal.

Engage Actuators

Van engages actuators to their maximum displacement in the receptacle. Van sends wireless signal to ramp for each actuator that is successfully engaged.
NS -> *Wait for Lowered Lift*: Van receives signal from each actuator that they are fully engaged. Van receives ACK byte from ramp that it has been notified about all actuators.

Wait for Lowered Lift

Van waits for the lift to be lowered.
NS -> *Exchange Complete*: Van receives wireless signal from ramp that the lift lowered switch has been engaged.

Exchange Complete

Van notifies driver that the exchange is complete. After ten seconds, the van will go back to the Stand-By state.
NS -> *Stand-By*: Timer has elapsed.

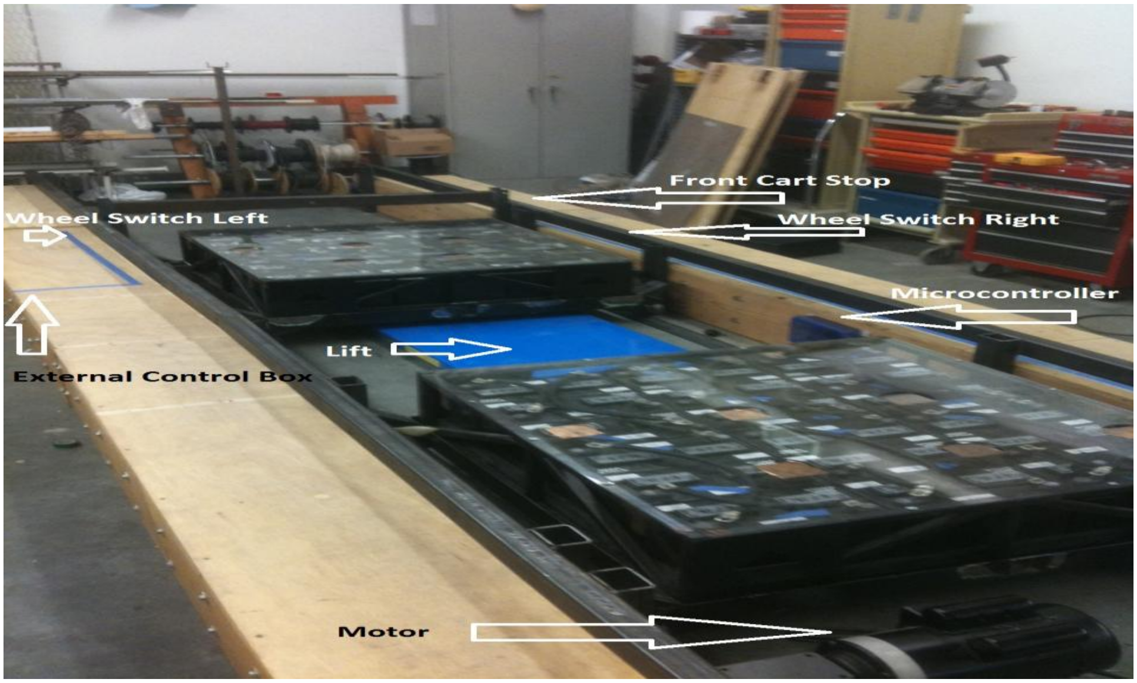


Figure 8: Ramp Picture Diagram

The microcontroller will be placed outside of the lift motor box. The other microcontroller will be mounted under the van on the right side. This ensures that the distance between the two XBees is kept at a minimum. The wheel switches will ensure that the van stops in the same spot during every exchange. To ensure that the voltage of the pack to be exchanged is higher than the voltage of the pack in the van, digital multimeters will display the voltage of the pack in the ramp. To initiate an exchange, the driver will push the driver button after the ready light is lit.

The lift is positioned directly under the receptacle in the van. A switch under the lift ensures that the lift is down when the packs are moving. The packs are moved by a 1.5 horsepower motor. By checking the voltages under the charging contacts, and checking which cart stop has been hit, the lift will not raise unless a pack is over the lift. The lift will rise until another switch is pressed under the van to verify that the pack has been properly inserted.

Once the pack has been inserted, the linear actuators will lock the pack in position. The lift will then lower until the switch under the lift has been depressed. The depleted pack will move under the ramp to be charged. The exchange complete light will illuminate to show the driver that they can now continue on their way.

All of the switches will be connected to the microcontroller using 24 gauge wires and insulated disconnects in the ramp. The wires will be connected at four different points using d-sub connectors. The connectors will be placed at key disconnect points in the ramp: at the microcontroller, at the front cart stop, at the left wheel switch and the external control box.

The switches and LEDs will all be powered by a 5v rail that is generated by the micro controller. The current draw of the digital multimeters is more than the microcontroller can supply. For their power rail, a 5v power supply will be added to the ramp. Using a LM2576 switching power supply, we will be able to provide a stable 5v rail to different portions of the ramp. The input to the circuit will come from the 12v power supply at the back of the ramp.

The ramp and van will have a series of multimeters hooked to the batteries to notify the driver of the pack voltages. Three multimeters that range from 0-100v will be placed in the external control box. Each multimeter will measure one of the 72v strings in the battery pack. They will be constantly powered by the 5v power supply at the end of the ramp. In the van, there will be another multimeter to measure the voltage of the pack during operation. It will be powered by the 12v auxiliary battery in the van.

In order for the microcontroller to measure the pack voltage, the voltage must be limited to 5v. To step the voltage down we will be using the following voltage divider circuit. This configuration was chosen so that the microcontroller could not be used as a ground for the packs.

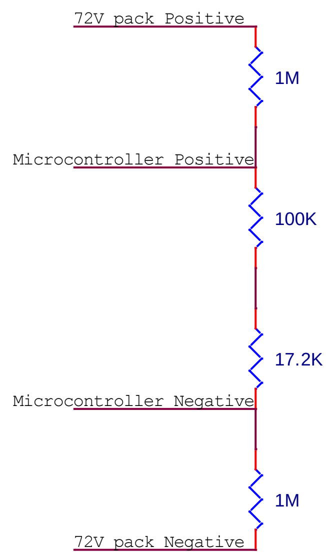


Figure 9: Pack Voltage Divider

.

A feature that could be added in the future would be a current gauge in the van. This could be implemented using another multimeter and low resistances shunt. A 500 amp shunt will produce a 50mv drop at 500 amps. This voltage drop could be measured by a multimeter hooked across the shunt. The parts to implement this have been purchased, but this is an additional feature beyond the requirements of this project.

V. Test Plans

Bread board testing

The microcontroller will be tested on a bread board with 5v signals as inputs. To verify that the microcontroller then sends the right output, LEDs will be placed on the output pins.

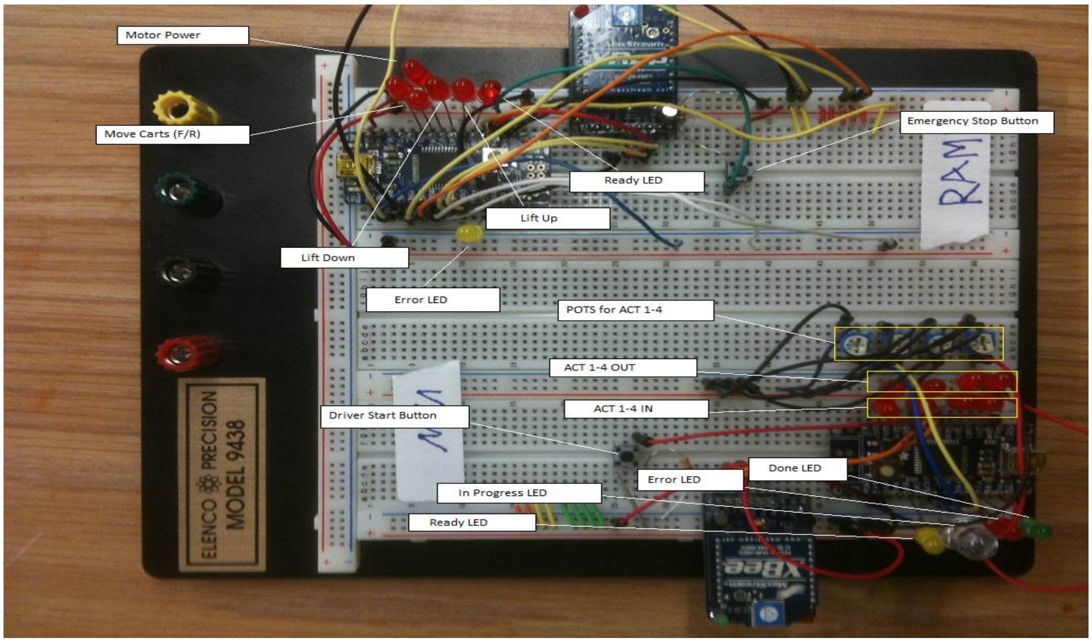


Figure 10: Breadboard Test Circuit

First the microcontrollers and XBees are powered. A solid green light means they are powered and a red light on the XBee means that it is communicating with another device. All of the switches are powered by the 5v rail of the microcontroller. The LEDs are hooked up at the output pins of the microcontroller. Due to the low output current of the microcontroller, no resistors were added.

To start an exchange the first thing that must happen is the wheel switches need to be depressed. This is done by placing the wires of B5&B6 in the 5v rail. The ready led on C6 of the van is then lit. At this point the driver would check the voltage of the pack in the ramp; this was deemed irrelevant at this stage of testing. The driver button is then depressed to start the exchange process and the C7 I progress LED is lit.

If voltage is detected by the microcontroller at the charging contacts, that means that the battery is not on the lift and the lift can start rising. B4 on the van microcontroller is then placed in the 5v rail and the signal is transmitted to the ramp side. The actuators were modeled as pots with a varying

resistance of 500-10,000 ohms hooked to F4-F7. The pots were turned to simulate them detaching from the pack. Once this is done, the D6 LED is lit to show the lift is moving down. When D0 is placed in the 5v rail the lift lowered signal is sent.

Depending on whether E6 carts forward or D1 carts back is placed in the 5v rail, the carts will move in the opposite direction as noted by the LEDs on F6 move forward or F7 move back. Once The opposite cart switch has been placed in the 5v rail, the D7 LED will light to show the lift is rising. B4 will then be placed in the 5v rail to tell the lift to stop moving. The Actuators will be extended by turning the pots. Once they are extended the D6 LED is lit to show the lift is moving down. When D0 is placed in the 5v rail the lift lowered signal is sent.

The carts will then move to the opposite side by the lighting of F6 move forward or F7 move back until E6 carts forward or D1 carts back is placed in the 5v rail. The green LED on F0 will then light to show that the exchange is done.

Wireless test

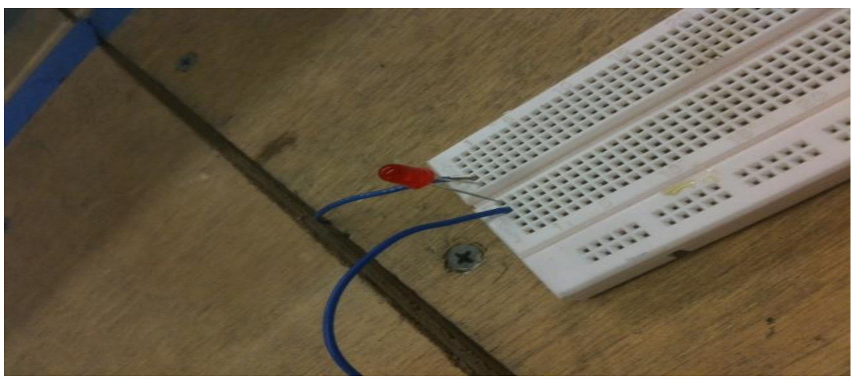


Figure 11: Wireless Testing LED

The microcontrollers will be placed in their housings and mounted; one microcontroller in the van and the other in the ramp. An LED will light if the two boards can communicate. The program will then be executed and stopped if the LED is lit. The microcontroller will send packets of information at a rate of one hertz so that if a packet is missed the LED will not light.

Hardware test

The switches used in the ramp will be connected to their corresponding ramp wires and continuity tested. Once this is complete, the switches will be checked to make sure they are securely mounted. Since most of them will be spring mounted, we must make sure that the button is depressed before the spring is. The XBee communication will be tested using LEDs again. The manual overrides will be tested to ensure proper operation.

The voltage at the external control box should be 5v. The multimeters will be hooked to the 5v supply and the supplies temperature will be measured. The digital multimeters will be calibrated using a 72V battery string. Once the multimeters are tested the step down circuit will be tested to ensure that the microcontroller is not fed too much voltage.

The voltage from the 72v to 5v voltage divider will be measured at various levels to make sure that the microcontroller does not see more than 5v. The ramp process will then be stepped through without the van to make sure that things function properly. The emergency stop button will be hit at various points in the exchange to make sure that everything stops. Once the sequence is completed without the van, the van will be placed on the ramp and the exchange executed.

VI. Development and Construction

The development code for the ATmega32U4 breakout boards was written in C. Although Atmel provides a free IDE for development on their chips called AVR Studio, it was unable to be used with the breakout boards. The reason the IDE was unable to connect to the boards was because the bootloader software that the microcontroller is programmed with on the breakout board before shipping requires a button to be pressed to open the serial communication port on the board, which times out after ten seconds. For this reason, AVR Studio was used for development, but a command line tool known as avrdude was used to actually program the microcontrollers.

In Windows, a bootloader INF driver file for the AVR109 family needed to be installed before the board would show up as a serial/COM port. After this was installed, avrdude could be used from the command line to connect to and program the microcontroller after the button was pressed.

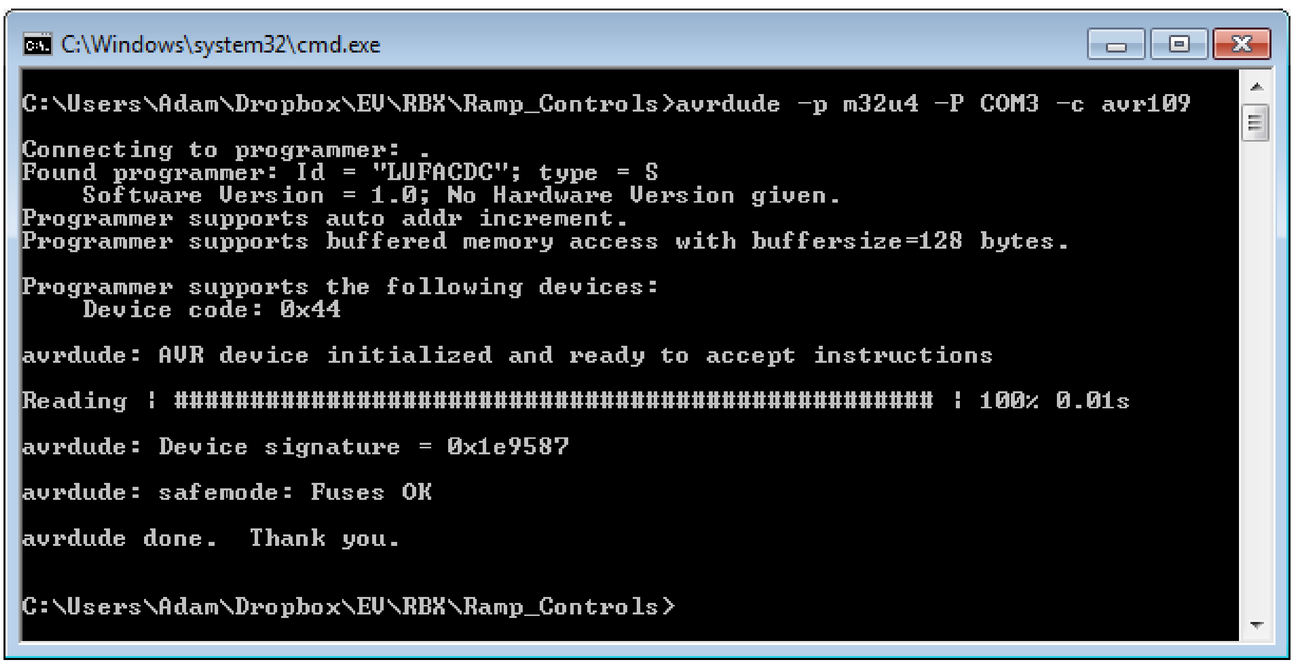


Figure 12: avrdude Initializing Bootloader

Once the HEX file was compiled in AVR Studio, it could be programmed to the device using avrdude in a command similar to the one shown here:

```
avrdude -c avr109 -p m32u4 -P COM4 -U flash:w:filename.hex
```

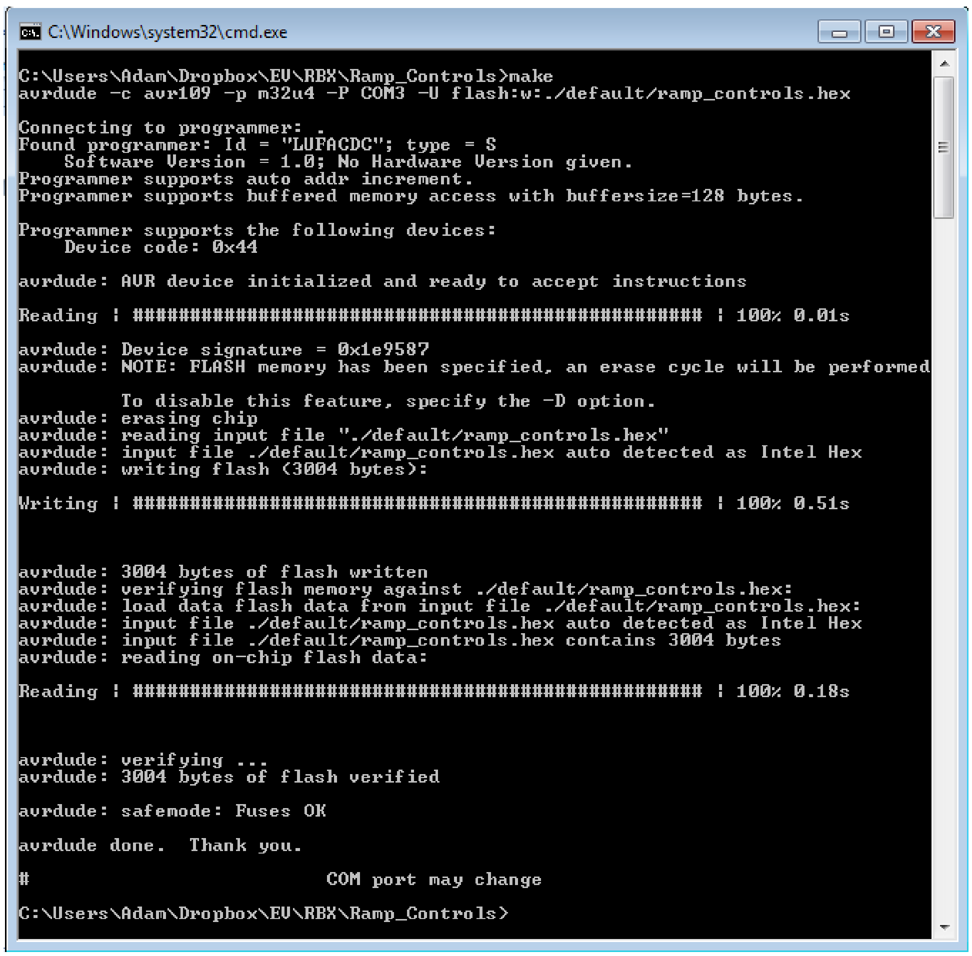
A Makefile was placed in the root each project folder created by AVR Studio to allow for easy programming of the boards. The Makefiles look something like this:

```
################################################################################
# Makefile for the project EV (to program ATMega32u4)
################################################################################

program:
	avrdude -c avr109 -p m32u4 -P COM4 -U flash:w:./default/ramp_controls.hex
	#                          COM port may change
```

Figure 13: Makefile for using avrdude

With a Makefile, all that needs to be changed is the HEX filename for each project. The COM port also changes depending on which USB port is being used. Now, the boards could be programmed using make:



```
C:\Windows\system32\cmd.exe

C:\Users\Adam\Dropbox\EV\RBX\Ramp_Controls>make
avrdude -c avr109 -p m32u4 -P COM3 -U flash:w:./default/ramp_controls.hex

Connecting to programmer: .
Found programmer: Id = "LUFACDC"; type = S
    Software Version = 1.0; No Hardware Version given.
Programmer supports auto addr increment.
Programmer supports buffered memory access with buffersize=128 bytes.

Programmer supports the following devices:
    Device code: 0x44

avrdude: AVR device initialized and ready to accept instructions

Reading | ################################################## | 100% 0.01s

avrdude: Device signature = 0x1e9587
avrdude: NOTE: FLASH memory has been specified, an erase cycle will be performed

         To disable this feature, specify the -D option.
avrdude: erasing chip
avrdude: reading input file "./default/ramp_controls.hex"
avrdude: input file ./default/ramp_controls.hex auto detected as Intel Hex
avrdude: writing flash (3004 bytes):

Writing | ################################################## | 100% 0.51s



avrdude: 3004 bytes of flash written
avrdude: verifying flash memory against ./default/ramp_controls.hex:
avrdude: load data flash data from input file ./default/ramp_controls.hex:
avrdude: input file ./default/ramp_controls.hex auto detected as Intel Hex
avrdude: input file ./default/ramp_controls.hex contains 3004 bytes
avrdude: reading on-chip flash data:

Reading | ################################################## | 100% 0.18s



avrdude: verifying ...
avrdude: 3004 bytes of flash verified

avrdude: safemode: Fuses OK

avrdude done.  Thank you.

#                          COM port may change

C:\Users\Adam\Dropbox\EV\RBX\Ramp_Controls>
```

Figure 14: Programming with avrdude

The XBees were programmed with free software from Digi that allows you to change the wireless configuration and the destination/source addresses. Using this software, X-CTU, the two XBees were set to the same PAN ID (4556) and configured so that their source and destination addresses corresponded to one another.

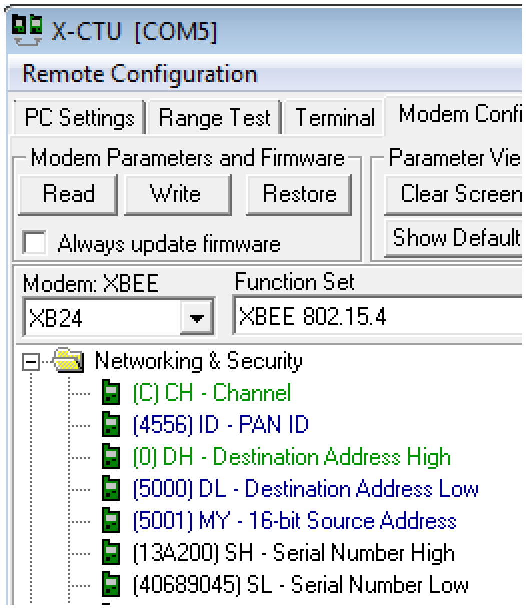


Figure 15: XBee 1 Configuration

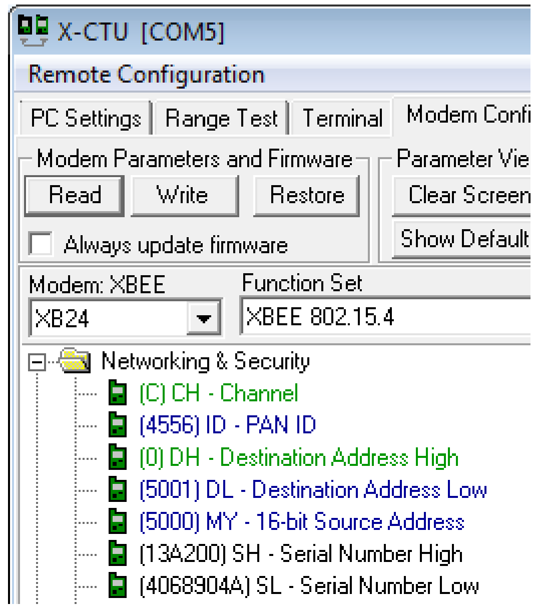


Figure 16: XBee 2 Configuration

The microcontrollers were also designed to contain device parameters stored in EEPROM which could be updated at run-time without recompiling and reflashing the chips. This is accomplished by editing and loading an EEP file onto the microcontrollers. A program known as PonyProg simplifies this task, allowing a user to download and see the current values of the memory addresses in EEPROM and edit them. However, using the software to connect to the breakout board was not possible using the USB connection since the software uses its own serial communication protocol. The software could be used with an ISP programmer connected to the breakout board since there are 6 pins for the ISP header available on the breakout board. This is something that can be done in future implementations.

The switches used are Sierra MP39310 (figure 17) Starter / Horn Push Button Switches. They are used as wheel switches, lift down, lift up and cart stops. Everything but lift down is spring mounted to account for addition movement after the button is pressed.

Figure 17: Sierra MP39310

The connectors are 25pin D-subs (figure 18). They have solder terminals on the back, so they can be easily added to for further development. The wire used throughout the ramp is 24 gauge for signals and 22 for the 5v power rail that is not generated by the microcontroller. The 5v rail is generated by a LM2576 (figure 19) High Input (7V~40V) Switching 5V Power Module/Regulator. It can supply up to 1.5 amps without a heat sink.

Figure 18: 25pin D-subs

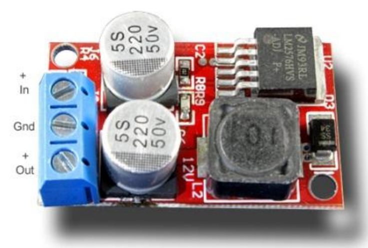


Figure 19: LM2576

The external control box on the ramp has three 100V multimeters (figure 20) with a resolution of 0.1V. The van has a 1000V multimeter (figure 21) with a resolution of 0.1V and a 20V multimeter with a resolution of 0.01V.

LEDs are used to show errors, emergency stop, ready to exchange, exchange in progress and exchange complete. Since these LEDs are powered by the microcontroller, there is no need to add a resistor due to the current output of the microcontroller. LEDs are used on the ramp and in the van.

Figure 30: 100V Multimeter

Figure 21: 1000V Multimeter

The cart stop bridges the two sides of the ramp, it has a spring mounted switch that hits the carts and then signals the microcontroller to stop the motor. The wires run from one side of the ramp to the other in it.

Figure 22: Cart Stop

VII. Integration and Test Results

Breadboard testing worked perfectly. We were able to step through from state to state and see that the micro controller was giving the right output. The XBees were able to transmit the data perfectly when they were on the same breadboard. The XBees were unable to transmit very well through walls. This was proven by having an LED blink when a packet of information was received at a one hertz rate. To improve data transmission we placed one microcontroller outside of the ramp and the other under the van.

The switches worked perfectly, the button depresses before the spring. Continuity was maintained throughout the ramp with the addition of the D-sub connectors and the insulated disconnects. The LEDs lit when the van went into the appropriate states. The buttons performed their various functions.

The 5V power rail and the multimeters did not arrive in time to test them. The receptacle for the van was not completed by the time this testing was completed. Also the receptacle could not be test fitted because one of the actuators could not be retracted. As a result, the following was unable to be tested: lift raised switch, van multimeter, van LEDs, driver button, 5V power rail and pack multimeters.

VIII. Conclusion

In conclusion, the rapid battery exchange has turned into an ongoing project which can be continued by other club members or other senior project students interested in electric vehicles. The full system could not be completed in time, but our individual parts were completed and our requirements met. This senior project laid a key foundation for future club members to finish the RBX. The hardware and software needed for both the ramp and the van are now in place to begin testing the system as a whole using the actual physical components of the RBX. There are still many improvements that can be made to the system, but what has currently been accomplished will provide a good basis for expanding upon and adding additional features.

With the growing impact and commercial availability of electric vehicles, it is only a matter of time until battery exchange stations will be as needed and as common as conventional gas stations. Electric vehicles will become a more integral part of our society in the future. They are more efficient and sustainable than their gas-powered counterparts. This senior project aims to help solve one of the key issues that electric vehicles face, which is long down times between charges.

IX. Bibliography

1. "What is a Hybrid Vehicle?" Mar. 2012. [Online]. Available: http://what-is-what.com/what_is/hybrid_vehicle.html. [Accessed Mar. 14, 2012].

2. "Toyota Prius-2012 Performance & Specifications," Mar. 2012. [Online]. Available: http://www.toyota.com/prius-hybrid/specs.html. [Accessed Mar. 14, 2012].

3. "How Hybrid Cars Work," Mar. 2012. [Online]. Available: http://auto.howstuffworks.com/hybrid-car6.htm. [Accessed Mar. 14, 2012].

4. "2012 Models," Mar. 2012. [Online]. Available: http://www.teslamotors.com/. [Accessed Mar. 14, 2012].

5. "Better Place Unveils an Electric Car Battery Swap Station," May 2009. [Online]. Available: http://www.wired.com/autopia/2009/05/better-place/. [Accessed Mar. 14, 2012].

6. "Electric Vehicles," Mar. 2012. [Online]. Available: http://www.wired.com/autopia/2009/05/better-place/. [Accessed Mar. 14, 2012].

7. "Chevrolet Volt," Mar. 2012. [Online]. Available: http://www.chevrolet.com/volt-electric-car/. [Accessed Mar. 14, 2012].

8. "Better Place to install 20 battery swap stations in Denmark by spring 2012," Mar. 2012. [Online]. Available: http://green.autoblog.com/2011/07/05/better-place-to-install-20-battery-swap-stations-in-denmark-by-s/. [Accessed Mar. 14, 2012].

9. "Hybrid Cars: Auto alternatives for the 21st century," Mar. 2012. [Online]. Available: http://www.hybridcars.com/index.php. [Accessed Mar. 14, 2012].

10. "Battery-Boosted Quick-Charger Doesn't Shock System," Mar. 2012. [Online]. Available: http://www.wired.com/autopia/2012/03/battery-boosted-rapid-ev-charging-station-wont-shock-the-electric-grid/. [Accessed Mar. 14, 2012].

Appendices A: Senior Project Analysis

Students' Name: Michael Cocchi Signature:____

Adam Rizkalla Signature:____

Advisor's Name: Art McCarley Advisor's Initials:_____________ Date:________________

Summary of Functional Requirements

The rapid battery exchange system will take the dead battery in a G-Van and replace it with a charged one. The dead battery will then be charged and used in another exchange.

Primary Constraints

To build the RBX it took about $10,000. That includes two 216V battery packs, a hydraulic lift, a 1.5HP motor, wood and metal. With more money we could have gotten components that are easier to use and have more features. One company spent in excess of $500,000 to build an RBX.

Financial Capital EVEC

Original Estimated Time:

Development 40 Hours

Research 10 Hours

Trouble Shooting 40 Hours

Writing the Report 30 Hours

Total Estimate Time 120 Hours

Actual Development Time:

Development 40 Hours

Research 15 Hours

Trouble Shooting 60 Hours

Writing the Report 50Hours

Total Estimate Time 165 Hours

The main cost of this project is in the engineering. It took a whole club four quarters for a semi working model, it would be hard for a company to devote this kind of man power to a project. This project requires a massive amount of power. Using four different electrical circuits we can use 110V outlets, this could be done with one 220V plug. This would require homeowners to plug in where their dryer it to power the apparatus.

Our original cost estimate for our portion of the project was $200.00. We were able to come close to this budget, but there were more components needed than we thought.

BOM of RBX Project For
Michael Cocchi & Adam Rizkalla

Number	Name	Quantity	Price
1	ATMEGA32u4 breakout board	2	20
2	XBee kit	1	100
3	Matching Printed Circuit Board	1	3.19
4	SPST Switch	2	2.19
5	Resistors	4	1.19
6	Wire	700FT	FREE
7	D-sub connectors	8	2.29
8	Push button SPST	1	2.19
9	100V Multimeters	3	9.5
10	1000V Multimter	1	14.5
11	LM2576 H Switching 5V Power Module/ Regulator	1	5.95
		Total	221.79

This project does save the user the difference between filling up a tank of gas and charging a battery. However, the initial cost of the unit may make this impossible to pay back over the life of the system.

This project could be produce after a few years of research and development. This produce will last if the apparatus is kept in an enclosure that safeguards it from the elements. Also the battery pack cannot be changed in size or voltage to keep using the apparatus.

Once this apparatus is finished, it can be upgraded to have better features. It could be completely automated and it could also have a display that would allow the user to choose when to charge the batteries at the lowest cost. This project will be upgraded for years to come by thee Cal Poly Electric Vehicle Club.

Financial Capital If Manufactured

This project would be in the millions of dollars if a company tried to make one. This process cannot be automated and the number of man hours makes it not feasible. The number of units produced would be one, if that. The purchase price would be at least a million, so the company could cover costs after the

first couple of units. The initial profit would be negative, but after a few years, the profit margin might increase. The cost to operate the device depends on whether or not the user needs to charge their batteries during peak hours.

Environmental

This project is great for the environment because it will cut down the need for fossil fuels. There would also be a reduced need for transporting petroleum which would greatly help the environment. This would decrease the demand on oil reserves and as a result, the number of oil spills in the ocean would decrease. This project improves the ocean ecosystem and the desert ecosystem with the decreased demand. It would not harm any ecosystems. Land dwelling animals will also be less impacted by the decreased need for drilling.

Manufacturability

This project would be hard to manufacture in its current state. It took a Cal Poly club 4 quarters of work to get a semi-working model. Then cost for a company to make the tools and dedicate that much man power to one project does not seem feasible.

Sustainability

To maintain the RBX, it needs to be in an enclosed building, so that it is not exposed to the elements. It would also need to be constantly plugged in to charge the batteries. With renewable energy, the RBX could become completely sustainable. The RBX is currently powered by the grid, but with solar panels the cost of using the system would decrease. With the addition of a display that would allow the user to choose when the RBX would charge batteries. The microcontroller would need to control the chargers in order to do this.

Ethical

If used improperly it could kill someone due to the high voltage and current. But, we have tried to decease this risk by putting safeguards in the microcontroller code.

Health and Safety

As stated before, if used improperly it could kill someone due to the high voltage and current. But, we have tried to decease this risk by putting safeguards in the microcontroller code.

Social and Political

This device could potentially put other similar devices out of the market such as the 30 minute quick charge system, thus harming the people involved with making and selling that device. Many people might not agree with the system, because of the risk of electrocution due to exposed charging contacts. This project will impact the people involved in creating it, the users, and ecosystems it protects. The direct stake holders are the people who created it and the people using the device. The indirect

stakeholders are the people/companies who the components are purchased form in order to make the device and the ecosystems it protects. The project will benefit the stakeholders financially, but it may harm the people who create similar products but decreasing their sales.

Development

This project allowed us to become more fluent in teamwork. With a project this big, multiple parts were needed to make a whole. Some parts required others to be done first; as a result, everyone was helping no matter what the task was. This in turn, really helped us to plan things out ahead of time.

Appendices B: Specs & Requirements

1. The van will notify the driver when the ramp is ready to begin and has completed an exchange.
2. The microcontroller (MC) will read and adjust the position of the linear actuators.
3. The MC will communicate wirelessly through the XBee module with the apparatus.
4. The apparatus software will correctly operate the lift and cart motor controls.
5. The apparatus software will communicate with the van side wirelessly through the XBee module.
6. The flow or the RBX will follow the state diagrams.
7. Both sides will contain safety checks and precautions, such as watchdog timers, and error statuses to report any malfunctions.
8. The software will be tested using manual overrides for the necessary switches in order to reach full functionality of the software implementation.
9. Switches integrated with the ramp that will notify the microcontroller of the van and cart position.
10. A set of switches that will detect when the lift is raised and lowered.
11. The MC will be notified when a pack is on the lift.
12. An external control box that will display the voltage of the pack, as well as contain the emergency stop button and manual overrides for the system.
13. A driver button in the van to begin the battery exchange.
14. The routing of all of the switches in the van to the microcontroller.
15. An interface between the chargers and the microcontroller.

Appendices C: Parts List & Cost

BOM of RBX Project For Michael Cocchi & Adam Rizkalla			
Number	Name	Quantity	Price
1	ATMEGA32u4 breakout board	2	20
2	XBee kit	1	100
3	Matching Printed Circuit Board	1	3.19
4	SPST Switch	2	2.19
5	Resistors	4	1.19
6	Wire	700FT	FREE
7	D-sub connectors	8	2.29
8	Push button SPST	1	2.19
9	100V Multimeters	3	9.5
10	1000V Multimter	1	14.5
11	LM2576 H Switching 5V Power Module/ Regulator	1	5.95
		Total	221.79

Table 3: BOM of RBX Project

Appendices D: Wire List

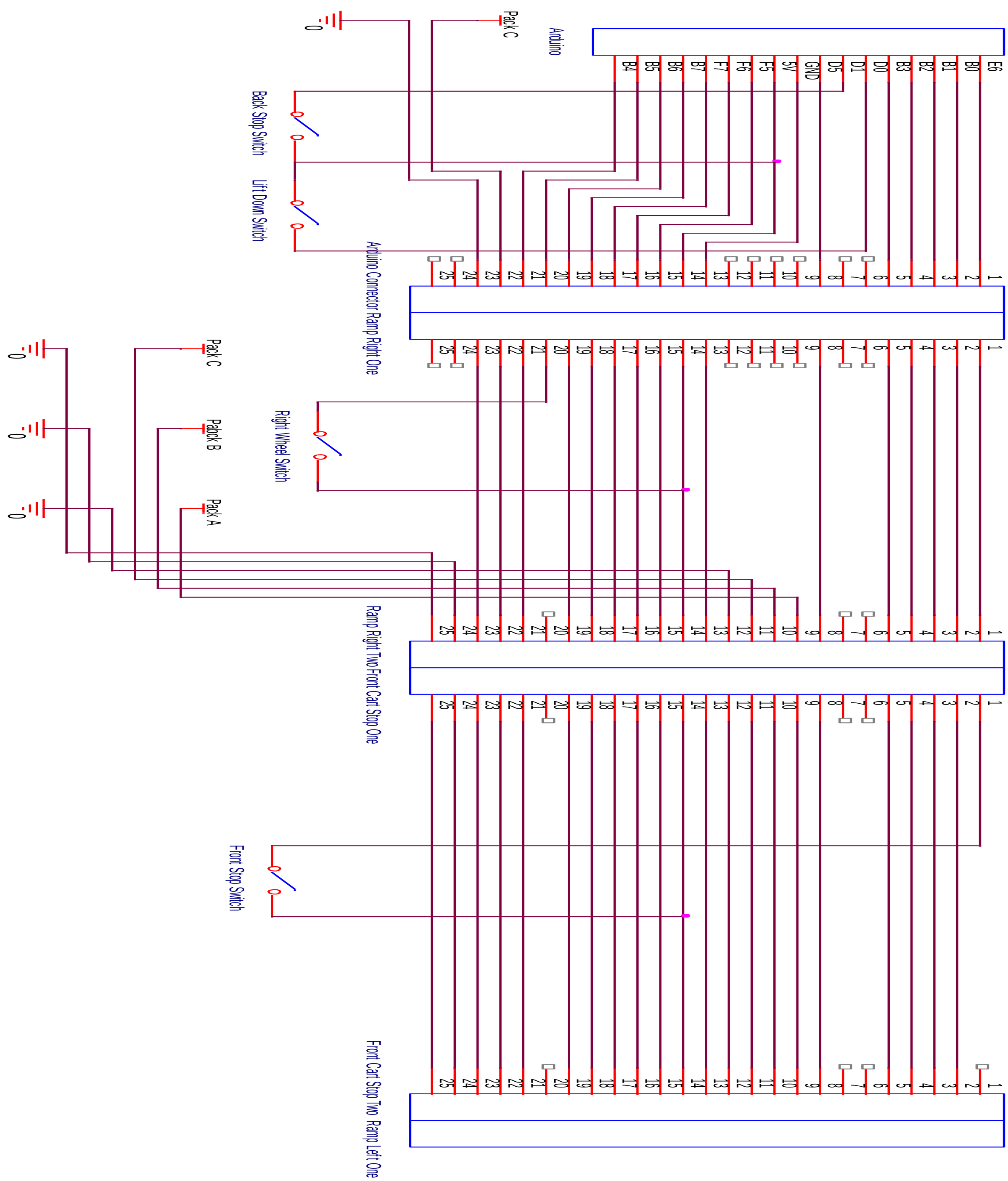


Figure 23: Wireless 1

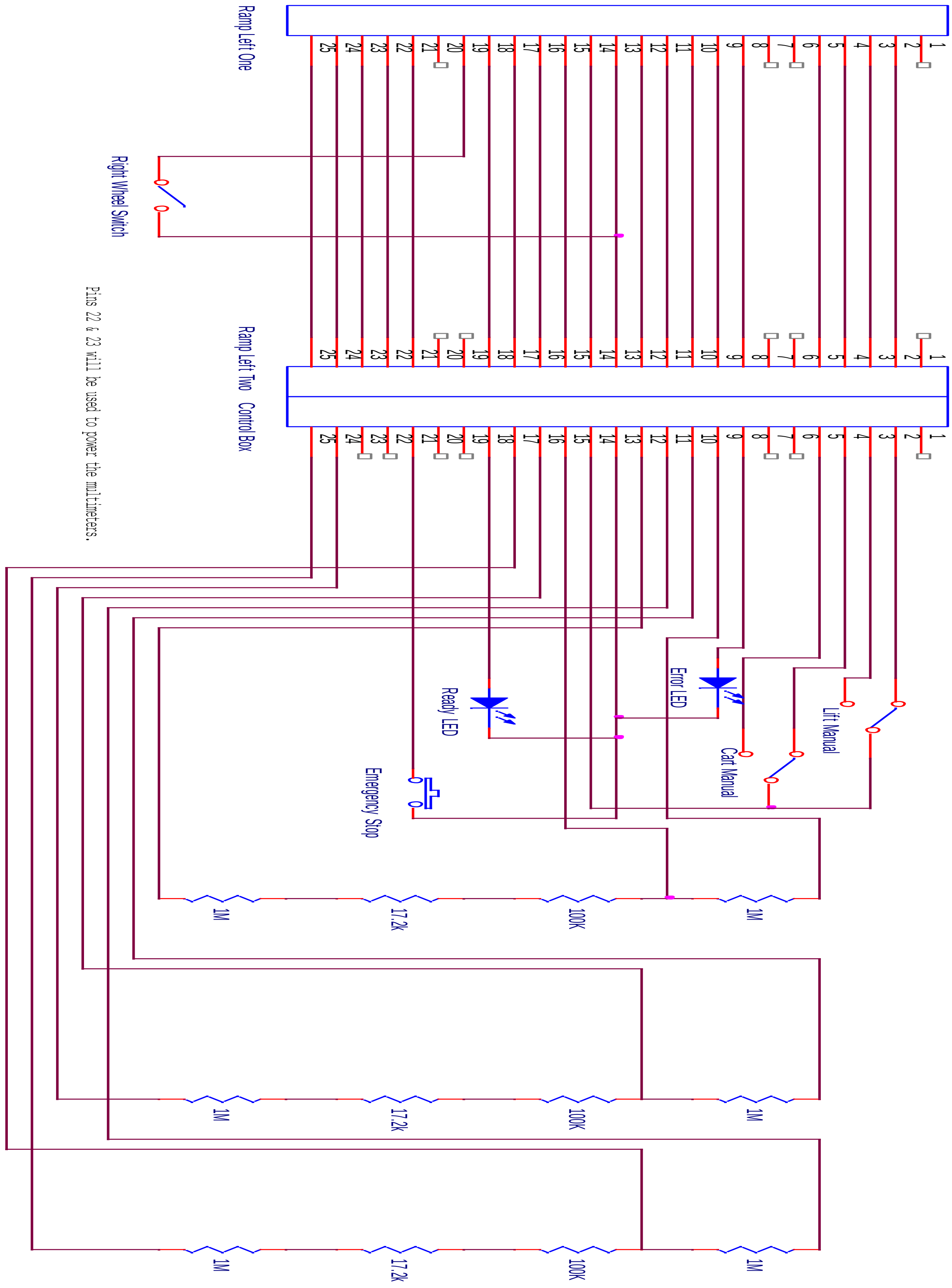


Figure 24: Wireless 2