Design Review
Quadpod Transform Vehicle

ECE 445 – Senior Design

<u>Date</u>

Oct 5, 2012

Team 6:

Jiwon Park

Zenon Son

Kee Woong Haan

TA: Rajashi Roy

Contents

Introduction

Quadpod transform vehicle

We believe this project is important because this scaled version of vehicle can access places (mountain and bumpy roads) that the vehicle cannot get through with typical wheels. This project merely shows the basic concept of the transformable vehicle but it may help the vehicle to be used in more various situations in the future.

Objectives:

The main goal is to make a miniature vehicle that can be transformed into a quadpod when it meets an obstacle that can't be overcome with typical wheels. The obstacle could be any place difficult to move by vehicle but we're mainly focusing on rough unpaved road or hill. The project will consist 4 legs (2 on each side), which will only be activated when the vehicle is in "quadpod" mode and use it to overcome obstacle.

Benefits:

- Vehicle can overcome obstacles such as bumpy road, mountain, jungle and forest, which typical car can't get through.
- Vehicle automatically transforms when it meets obstacle.
- Both car mode and Quadpod mode are fully controllable by user interface.

Features benefit:

- The vehicle can sense obstacles by using touch sensor in the bumper.
- Multifunctional vehicle for multipurpose use.
- Both vehicle mode and Quadpod mode easily controllable by using user interface.

Schematics, flowcharts, and mechanical frame work:

See next 7 pages for schematics, flowcharts, and frame work.

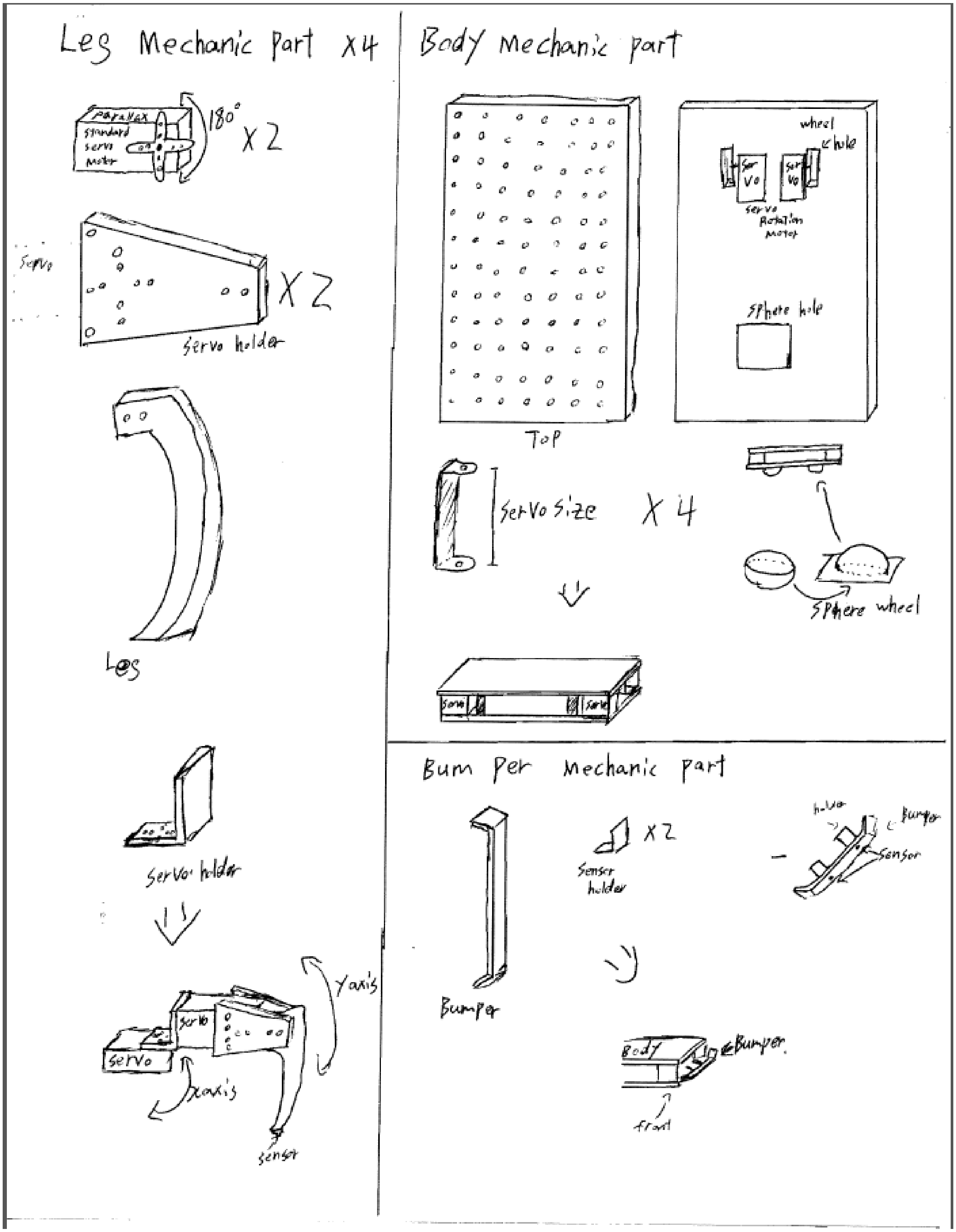

Leg Mechanic part X4
Body mechanic part
Parallax standard servo Motor
180°
X 2
Servo
X 2
servo holder
Leg
servo holder
Y axis
servo
servo
X axis
sensor
wheel hole
Ser vo
Servo Rotation Motor
sphere hole
Top
servo size
X 4
sphere wheel
Bum per mechanic part
X 2
Sensor holder
holder
Bumper
Sensor
Bumper
Body
Bumper
front

Figure 1.

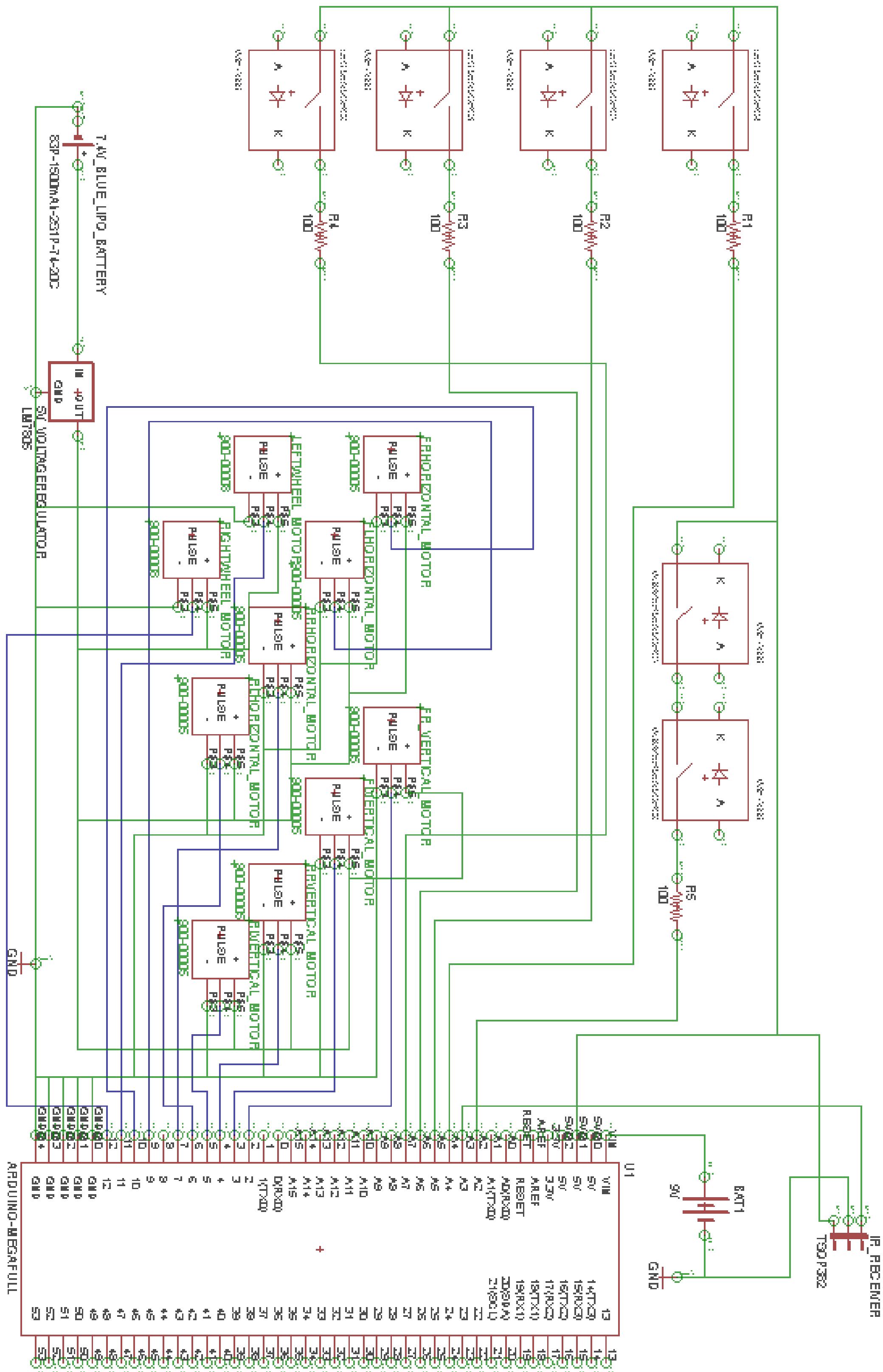

7.4V_BLUE_LIPO_BATTERY
5V_VOLTAGE_REGULATOR
LM7805
ARDUINO-MEGAFULL
U1
BAT1
9V
GND
IR_RECEIVER
TSOP382
R1
R2
R3
R4
R5

Figure 2.

Flow Chart while in vehicle mode.

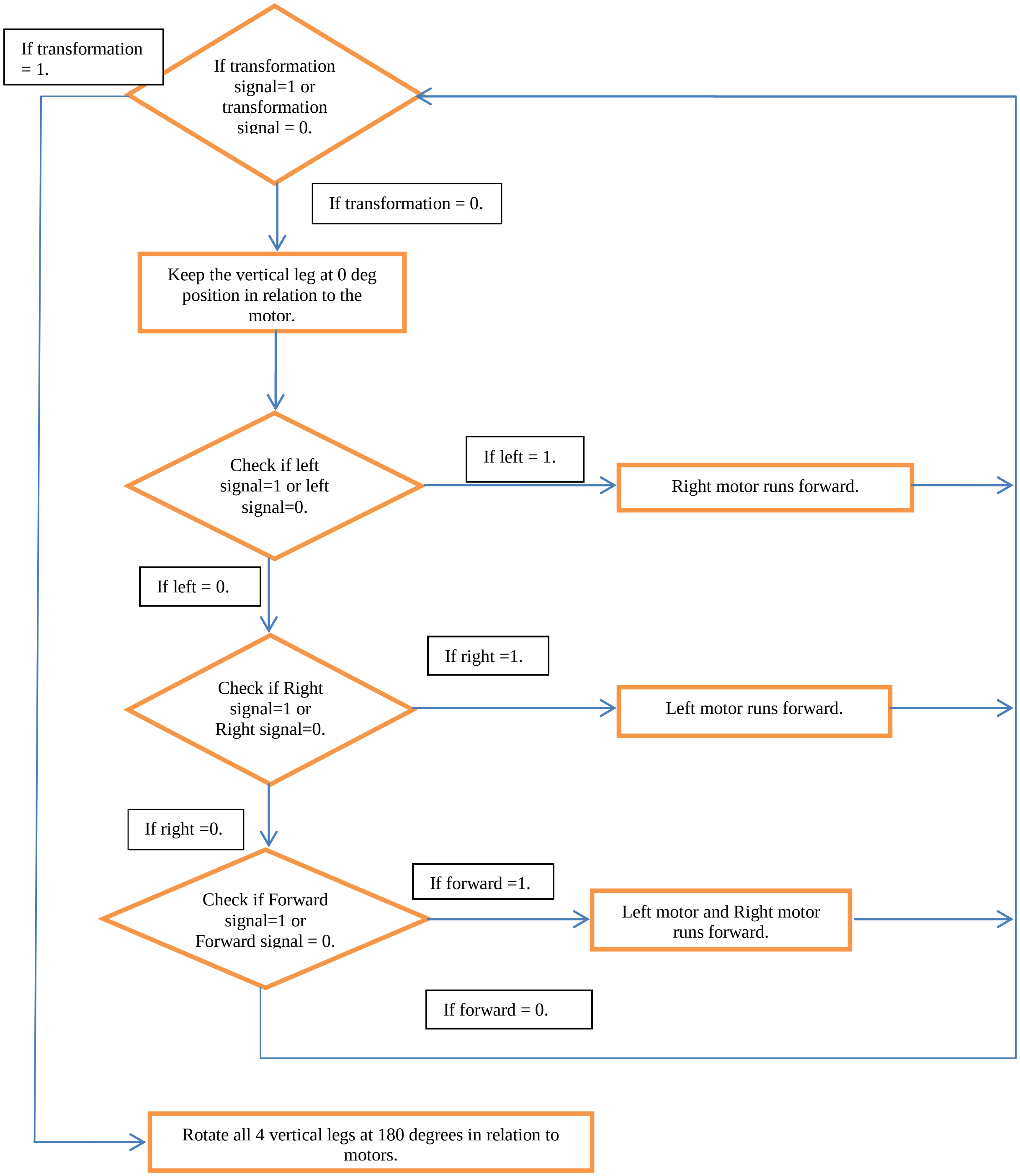
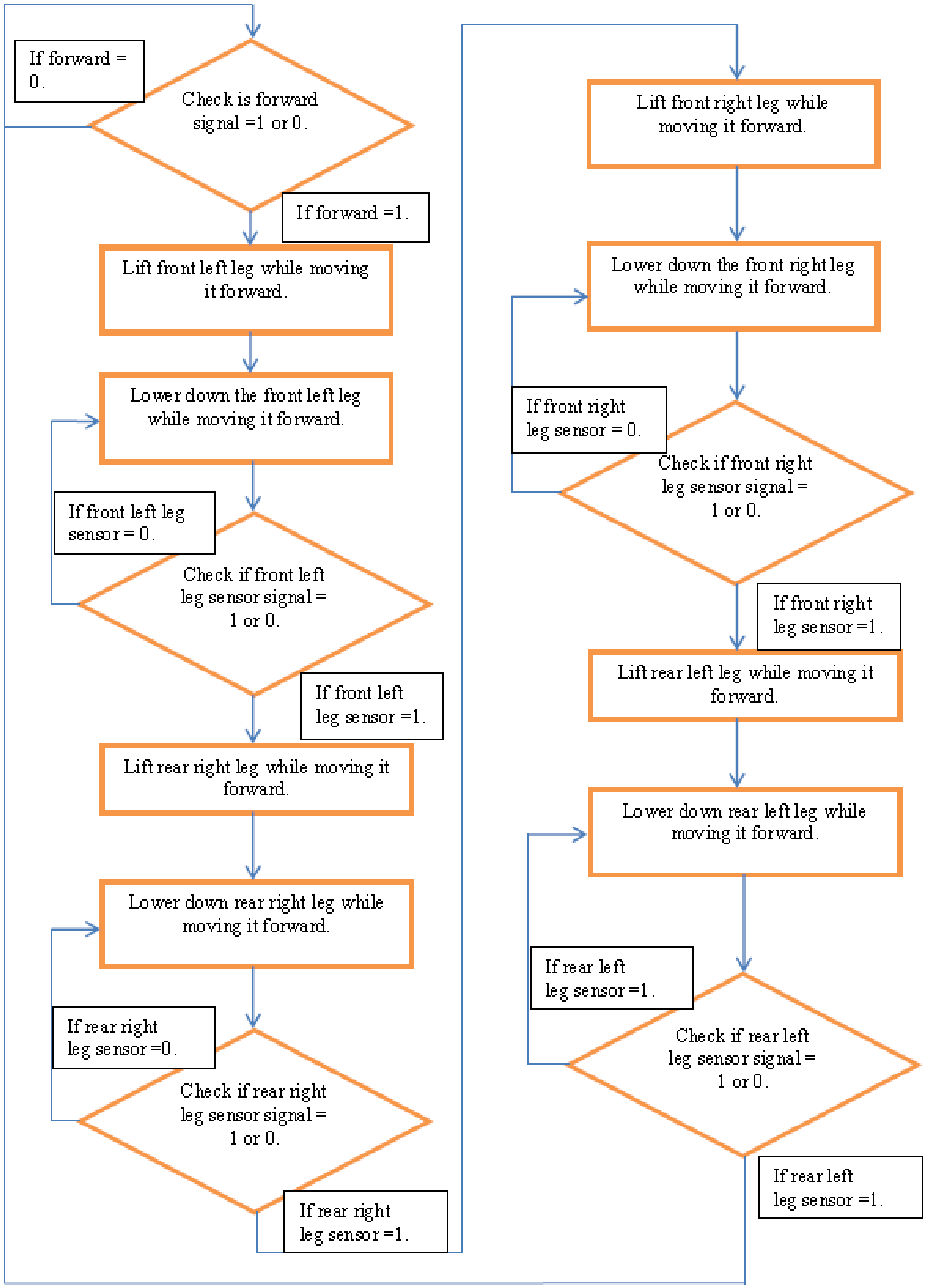
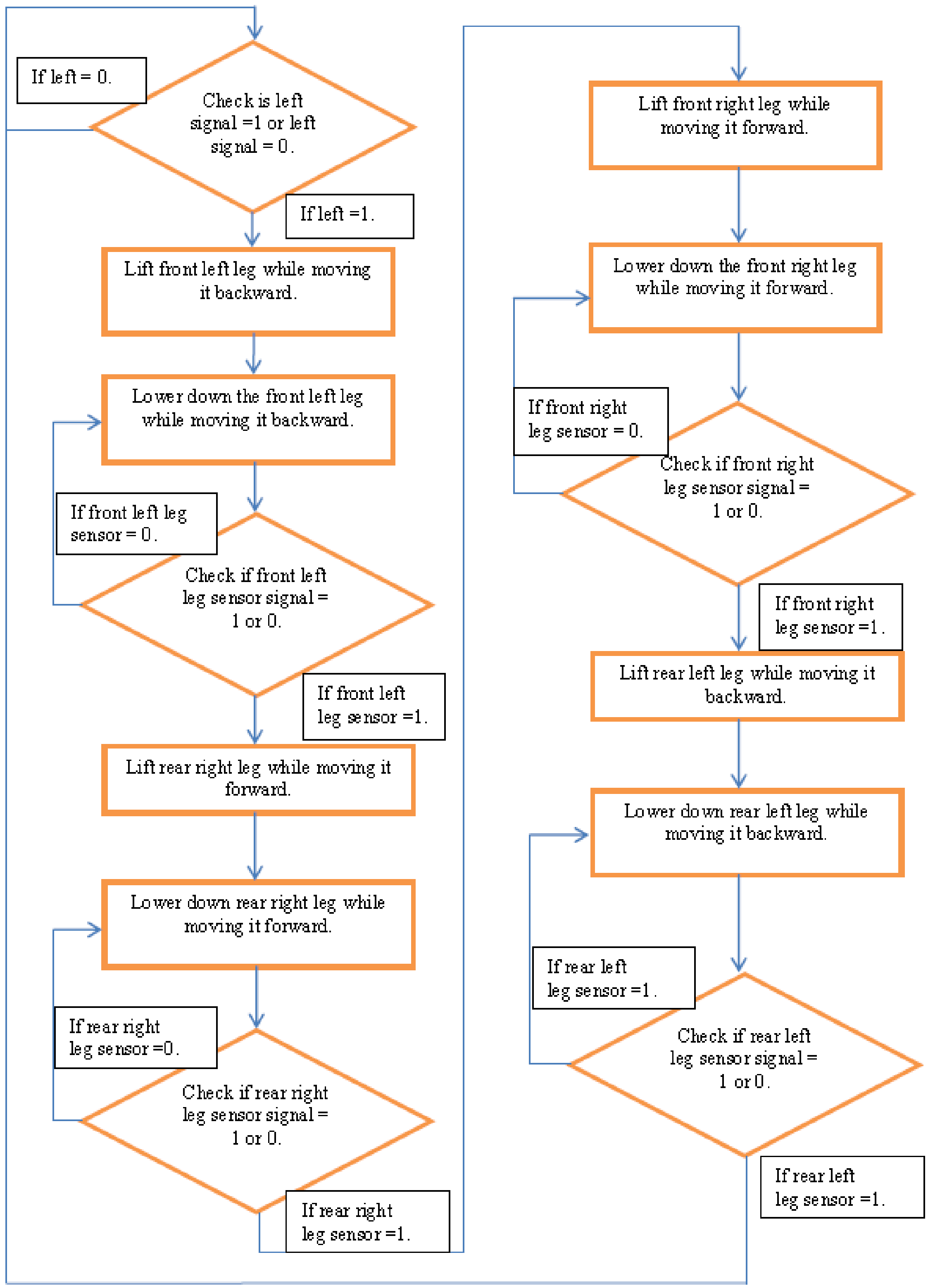


Figure 3.

Flow chart while in Quadpod mode (Walking forward). **(Figure 4.)**



Flow chart while in Quadpod mode (turning left). **(Figure 5.)**



Flow chart while in Quadpod mode (turning right). **(Figure 6.)**

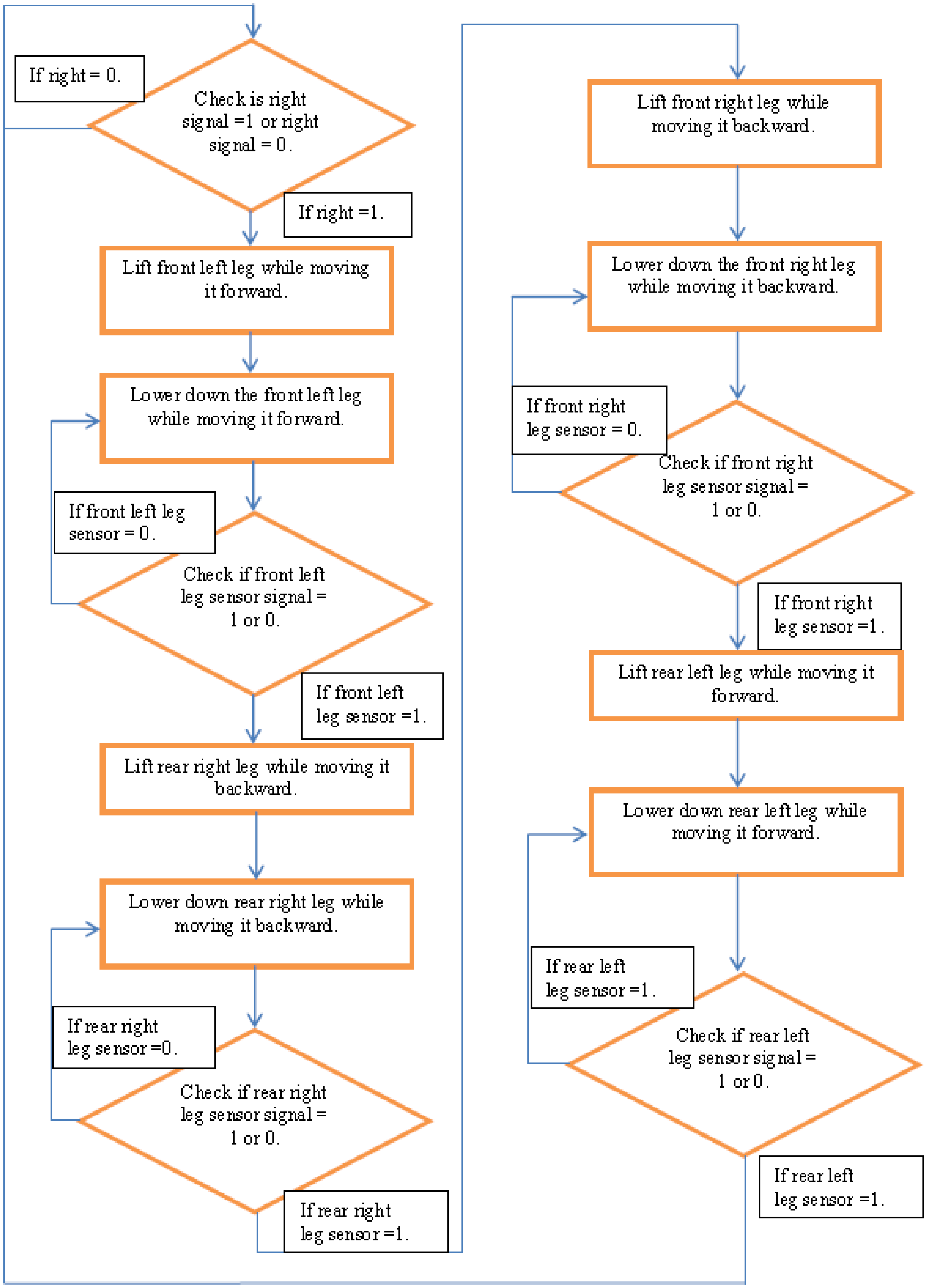


Figure 2 is schematic for the project. The project has 2 modes, vehicle mode and quadpod mode. While the project is in vehicle mode, it can go forward, left, right depends on the signal received by IR receiver. It can also transform into quadpod depend on the signal received from bumper sensor and the IR receiver. The Right and left Wheel motors are continuous rotation motor and they are in charge of controlling movement of the vehicle, while the 4 vertical movement motors are standard servo motors and they are in charge of transformation. All the motors are controlled with the length of the pulse given by the main station; thus, the motors are connected to the PWM output of the main station. While the project is in quadpod mode, it can go forward, turn left, turn right depends on the signal received by IR receiver. All these operations are done using the 8 standard servo motor, where 4 of them are in charge of vertical movement of legs and 4 in charge of horizontal movement. Again, all the motors are controlled with the length of the pulse given by the main station; thus, the motors are connected to the PWM output of the main station. There are also 4 leg sensors to send signal to main station to inform stability of each leg while the quadpod is moving. Pin 3-13 of the PWM output of the main station.

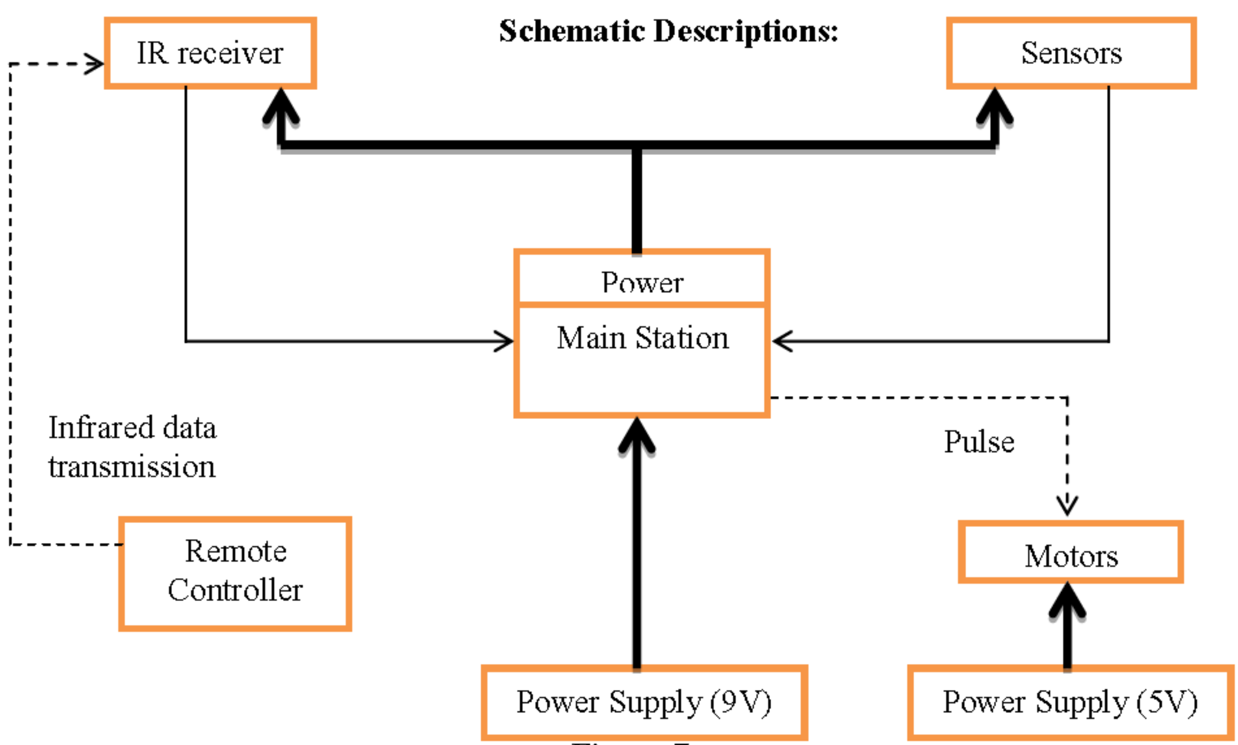


Figure 7.

Pic 1[10]. (The remote controller which will be used)

Main Station (Arduino Mega):

This unit is the network manager. It gets signal from the IR receiver which receives data from remote controller through infrared data transmission. It keeps gets user's input signals and evaluates appropriate action. The main station have 9V battery power source, however Arduino mega has built-in voltage regulator so that the voltage will be approximately 6V. This station will power up the IR receiver by using 5V output voltage pin. The maximum current that this pin can load is 40mA. By using PWM I/O pin, Pulse signal will be generated for the motors. The pulse with the length of 750us to 2250us will be generated from the PWM I/O pins.[1)]

P_{Main} = 20mA X 5V = 100mW
By using 9V battery which contains 550mAh;
Max operating Hours: 550mA X 3600Sec / 100m = 19800sec = 5.5 hours
$Pulse_{Max}$= 16 MHz > $Pulse_{Motor}$

Main station leg control unit (Leg sensor)

Each of the leg will have own sensor and will be activated when pressed. Push button will be attached on bottom of the legs so whenever the robot walks, Signal will send to the main station. This button will send analog signal to main station to verify that the leg which the button is attached to now stable so that next leg can move.
The button operate in the weight range of 100±30g. The buttons will be attached on 4 legs so we need weight of 4(100±30g) = 400 ± 120g. We need the total weight of the project to be over 520g. Since each standard servo motors weigh 44g and each continuous motor weigh 40g, (8x44) + (2x40) = 432g. Our lithium battery, which will be used to power up the motors weigh 121g. Even without adding the weight of Arduino Mega, Frame, and other miscellaneous parts, the weight of all the motors and battery is already 432g + 121g = 553g, which is over 520g.[8)]

$F_{max} = 4M_{require}g = 0.52 \text{ X } 9.81 = 5.1\text{N}$
$F_{weight} = M_{weight}g = 0.553 \text{ X } 9.81 = 5.42\text{N}$
$\therefore F(max) < F(weight)$

The button requires the current capacity 1~50mA, and the voltage capacity 5~24 VDC. This button will be powered by the 5V output pin of Arduino. Since the max current from the output of Arduino is 40mA, it should be safe to operate with the buttons. The following equation was used to determine the current values for signal main station. The signal should be strong enough to be detected by Arduino but small enough to not burn our main station.

$$I_{out} = 5V/\ (R_s + 100)$$

Transforming control unit (Bumper sensor)

Two push buttons will be attached on each side of bumper and will be used as the bumper sensor. If two side of the bumper is pushed, it will send analog signal to the main station so that main station will send appropriate pulse to the motors in charge of transformation. The push buttons used here are the same kind used for the leg sensor and its operating force range is 100±30g.[8)]

$$F_{Max} = 2Mg = 0.26 \text{ X } 9.81 = 2.55\text{N}$$

The button requires the current capacity 1~50mA, and the voltage capacity 5~24 VDC. This button will be powered by the 5V output pin of Arduino. Since the max current from the output of Arduino is 40mA, it should be safe to operate with the buttons. The following equation was used to determine the current values for signal main station. The signal should be strong enough to be detected by Arduino. If the signal amplitude is too large, it might burn our main station.

$$I_{out} = 5V/\ (R_s + 100)$$

Parallax standard servo motor X-axis unit

1. This unit will make our quadpod to move around.
2. This unit will get power and pulse from main station.
3. Depends on the pulse input from main station, (750us~2250us), our motor will move to corresponding angle (0°~180°) [2].
4. This unit will rotate in 45° to 135°
5. This motor need repeated pulse signal in every 20ms to maintain its position.

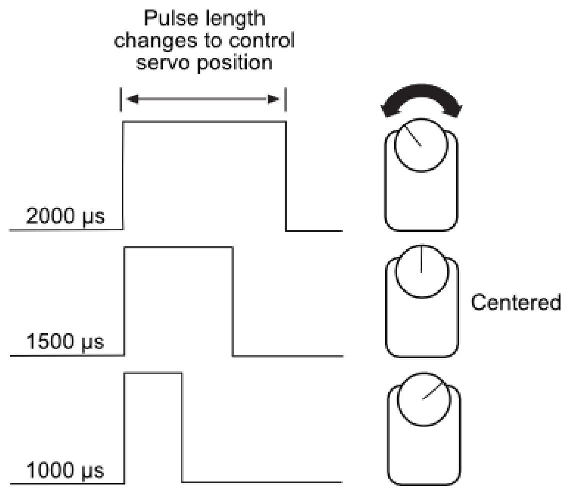


Figure 8.[2]

The angle will be calculated with the following equation:

$$T\ (us) = 750us + \{1500us\ (\theta/180)\} \qquad (45° < \theta < 135°)$$

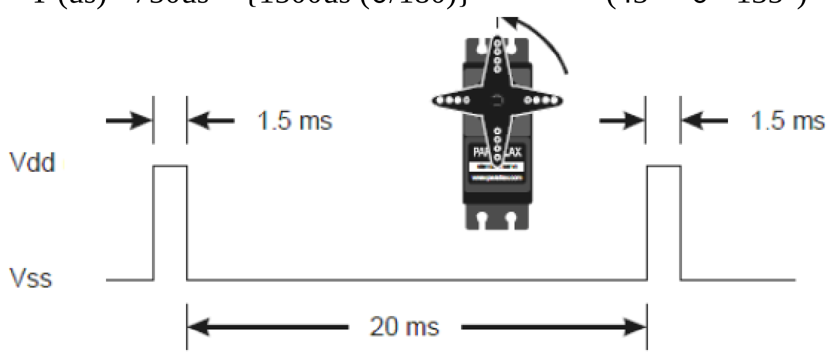


Figure 9.[2]

Parallax standard servo motor Y-axis unit, transforming unit

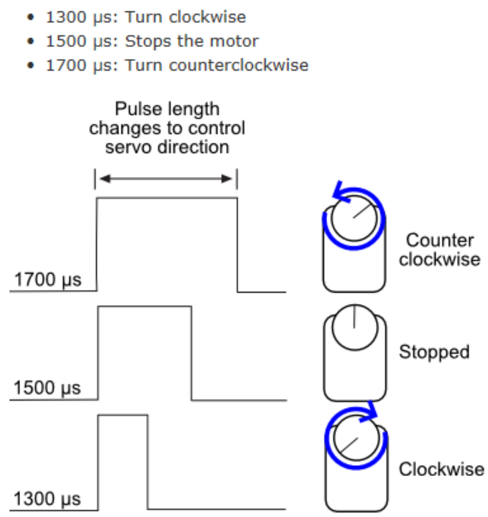
1. This unit will function in two different ways.
2. This unit will be used to transforming car to robot.
3. Also, while quadpod mode, this unit will move legs in y-axis when moving around.
4. This unit will get power and pulse from main station.
5. Depends on the pulse input from main station, (750us~2250us), our motor will move to corresponding angle (0°~180°) [2].
6. This unit will rotate in 90° to 180° while the quadpod is moving.
7. This motor need repeated pulse signal in every 20ms to maintain its position.

The angle will be calculated with the following equation:

$$T\ (us) = 750us + \{1500us\ (\theta/180)\} \qquad (45° < \theta < 135°)$$

Parallax continuous rotation servo motor unit

1. This unit will function when it is in car-mode.
2. There will be two motors on each rear wheels.
3. This unit will get power and pulse from main station.
4. Depends on the pulse input from main station,(1300us~1700us), our motor will rotate corresponding direction(Clockwise, Counterclockwise)[3].
5. This motor need1500us pulse to stop spinning and maintain the position.



d

Figure 10.[3]

IR receiver

Takes serial data from the remote transmitter and sends it to the main station. The original commend inside remote controller will not be used. Instead, we will code our own commends. There are 6 different commend and each will be different signal so that the signal will not messed up with each other's. This device need 2.5 to 5.5V to power up and it needs at least 5mA source to activate. Since our Arduino provide 40mA with 5V, it will be reliable to fully function. Since this IR receiver is wirelessly interacting with remote controller, we need to verify the distance that our IR receiver can receive the signal.[9]

Voltage Regulator

We are connecting 7.4V Li-ion battery to power up the motors. However, our motors need 4-6 voltage. Therefore, we add voltage regulator to decrease the voltage to 5V. The current larger than 1A will be flow through this regulator; therefore, we should make sure that regulator can endure it.[12]

Input voltage: 7.4V
Max Input current: 190mA X 8 = 1.5A
Voltage regulator output voltage = 5V
Voltage regulator max current output = 1.5A

Power Supply

We have two independent power sources. One of them is Blue LiPo 7.4V Battery which power up the servo motors. We will use voltage regulator to make voltage down to 5V. This voltage regulator should be capable to endure all of the current flow to the motors. Other one is 9V battery which power up the arduino Mega. Our main station requires 7 to 12V; therefore, 9V will be reliable.
As we state at the description on main station, 9Vbattery is good enough to power up main station for 5.5hours. For the motors, we will run 8 motors at most. Each motor will consume 140±50mA when applying 4-6V. The battery should be large enough to provide all the motors.

Main station power source: Voltage = 9V
Capacity = 550mAh[13)]

Servo motors power supply: Voltage = 5V
Capacity =1500mAh
Max continuous current = 20A
Power consumption by one motor = 190mA
8 X I_{motor} = 1.5A[11)]
Maximum operation Hours = 1.5A X 3600 / 1.5 = 1Hours

Performance Requirement:

<table>
<tr><th>Requirement</th><th>Verification</th></tr>
<tr>
<td><u>Arduino Mega 2560</u>
The Arduino Mega is the board we use for the circuit part. The operating voltage is 5V, but the recommended voltage is 7~12V, so we will use 9V batteries to help better performance. The current capacity for I/O pin is 40mA.[1]

The Arduino Mego should have a capability of supporting synchronized signal to the all the components connected to it.</td>
<td>1. Arudino Mega should give a synchronized signal to all the motors with a desired pulse signal. All the servo motors should work in the range of pulse width of 750us ~ 2250us[2] and Arduino Mega can provide these pulses from the PWM output. We can check this using the oscilloscope.
Once we have verified the motors and the PWM outputs, we can check the synchronization between the PWM output and the motors by checking if the motor is in a position described by the pulse width.

2. We can test the current value from I/O pin of Arduino Mega using resistor and some calculation. First, we connect resistor at I/O pin and Measure voltage across the resistor with voltage meter. Then we will be able to calculate the current coming out from the arduino using the following equation.
$V_R/R = I_{main} = 40mA$ (with 5% error)</td>
</tr>
<tr>
<td><u>Servo Motors</u>
The voltage capacity for the servo motors is 4 to 6V, and the current capacity is 140+/-50mA at 6V when it's idle.[6,7] The communication should be through pulse-width modulation for the motors.</td>
<td>1. By using a function generator, check how much a motor rotates by a given pulse. For the standard servo motor, the motor should rotate the amount of angle described by the table below. [2]
$\frac{(\text{Pulse width} - 750) \times 180}{2250 - 750} = \text{the angle of the motor}$

<table>
<tr><td>Pulse width</td><td>Degree</td></tr>
<tr><td>750~900µs</td><td>0~18degree</td></tr>
<tr><td>1400~1600µs</td><td>78~102degree</td></tr>
<tr><td>2100~2250µs</td><td>162~180degree</td></tr>
</table>
</td>
</tr>
</table>

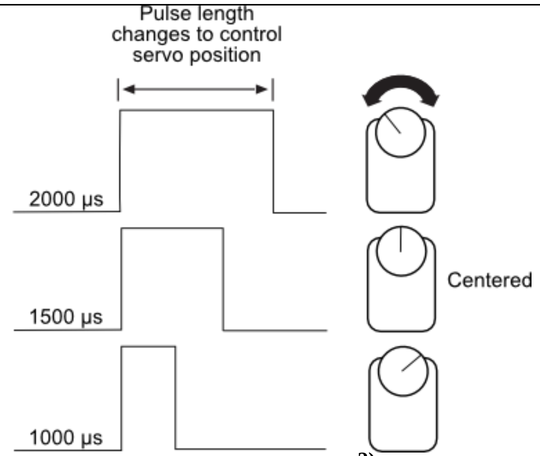


Figure 11.[2]

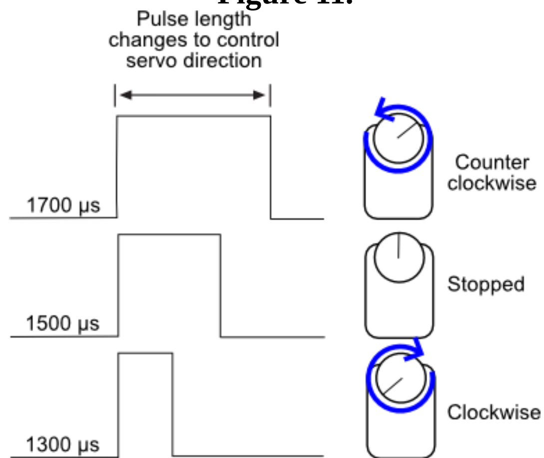


Figure 12.[3]

2. Check if the power supply is enough to provide voltage to all the motors within the effective range.
   Maximum Standard Servo:
   P_{max} = 190mA × 5V = 950mW[6]
   Maximum Continuous Rotation Servo :
   P_{max} = 190mA × 5V = 950mW[7]
   Maximum total power consumption of 10 motors :
   $P_{total,max}$ = 950mA × 10V = 9.5W

1. Check the maximum current capacity with 10mΩ resistor to prevent from overheating the motors. (I_{max} = 140mA[6),7)]) The current will be measured using 10mΩ resistor and the voltmeter by the Ohm's Law

Sensor

The button requires the current capacity 1~50mA, and the voltage capacity 5~24 VDC. The operating force for the button should be 100±30g for the proper functioning.[8]

The schematic for the verification is shown below with pull-down logic. The pull-down logic is used so that when the input voltage is not within the range that makes the digital signals 0 or 1, the logic makes the input as the digital signal 0.

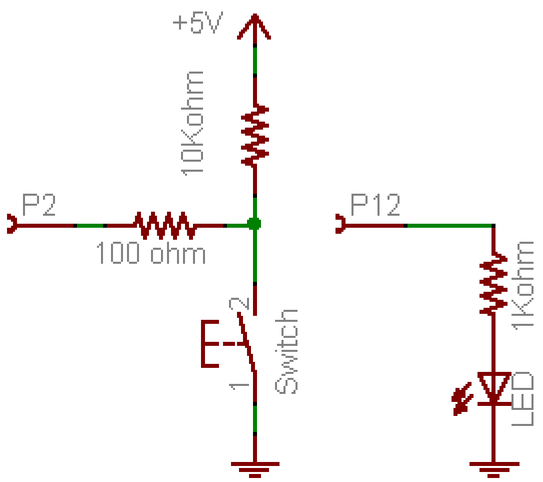


Figure 13.[5]

The picture below is the way we would like to use for the verication with Arduino Mega. The code that will be used in this case can be found in appendix.

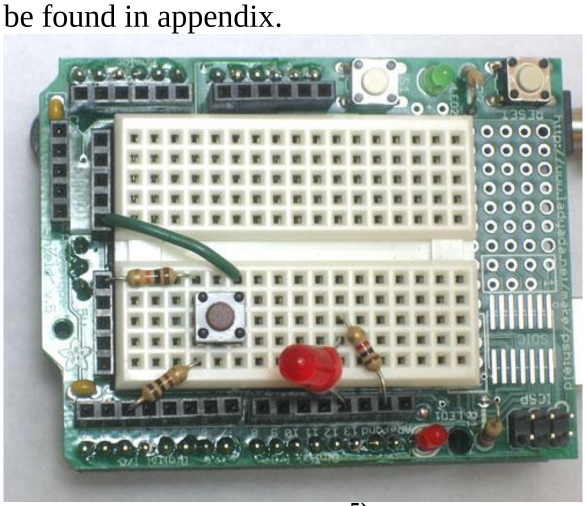


Figure 14.[5]

1. After building circuit described by the schematic on the left, check if the button gives a signal when it's pressed using a LED.
2. Try weights in the range 80~150g to check if the button is pressed within the effective range, 100±30g. We can use a tiny bucket on top of the button to test on which weight the LED will be turned on. There will be four buttons under the legs, therefore;

   Wtotal > 400± 120g.
3. The maximum current is 50mA[3], so the input current shouldn't go over the value. The current will be checked using 10mΩ resistor and a voltmeter. The measured value should be less than 50mA.
4. The voltage should be within the range between 5 and 24 V[3]. The measured voltage on the input pin of the button should be over 5V to operate. The measured value should be over 5V.

IR Receiver

The Receiver (TSOP382) only reads modulated 38kHz IR radiation, which means we need to pulse the LED at 38kHz for the data stream. For each bit of information, the LED will either be off for about 600us or on modulating at 38kHz for 600us, as shown below.[4)]

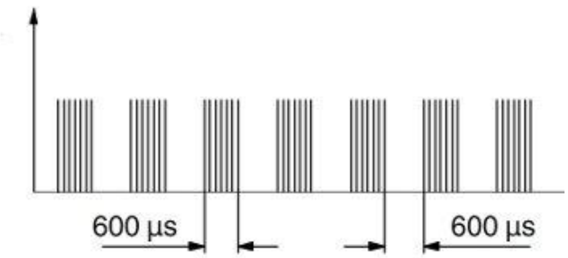


Figure 15.[4)]

The current capacity is 0.27~0.45mA, normally 0.35mA. If there is sunlight, the typical current would be around 0.45mA due to Electric field generated (E = 40klx). The voltage capacity is 2.5~5.5V, and the maximum distance for the remote control is 45m.[9)]

2. Check if the receiver receives the modulated 38 kHz pulse signal from the IR LED. The <code1>[4)] given by Sparkfun Electronics, the product company, will check if the signal is transmitted by printing ‘hit’ on the screen.

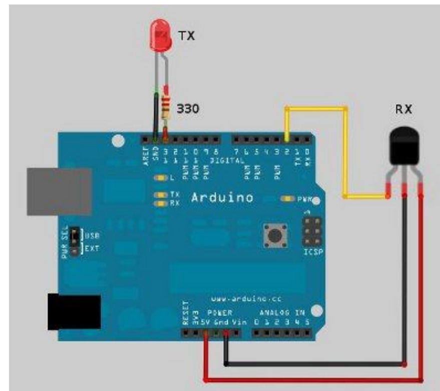


Figure 16. < an example of setup>[4)]

3. Check if the receiver receives the signal of the button on the remote control by printing the button pressed on the screen from the <code2>[4)] given by Sparkfun Electronics. Try all five buttons 10 times in the order of ←, ↑, →, Vehicle Mode, the Quadpod Mode.

4. Using the <code2>, check if the effective distance can be up to 20m. By the datasheet, the maximum distance is 45m, but practically the distance wouldn’t be over 20m. To check the effective distance, the button will be pushed to check the receiver can receive the correct signal from the remote control as moving further.

5. Maximum power consumption : 10mW[9)]

6. The typical current is 0.45mA[9)] when there is sunlight, so the current measured on the input pin for the receiver should be 0.45mA ± 10%

	error. The current range can be 0.27~0.45mA[9] if there is no strong sunlight. The current will be measured using 10mΩ resistor and the voltmeter by the Ohm's Law *The codes are included in the appendix.
Power Supply We have two independent power sources. One of them is Blue LiPo 7.4V Battery which power up the servo motors. We will use voltage regulator to make voltage down to 5V. This voltage regulator should be capable to endure all of the current flow to the motors. Other one is 9V battery which power up the arduino Mega. Our main station requires 7 to 12V; therefore, 9V will be reliable.[11) 13)]	1. To check the output voltage of the battery, the voltmeter will be used if the value is 7.4 ± 5% error. We can perform same test for the 9V battery also.
5V regulator The 5V regulator should make the7.4 V to 5V for the voltage for the motors.[12)]	1. By using voltmeter, the input node of the regulator should show 7.4V ± 5%, and the output node of the regulator should show 5V ± 5%. 2. Voltage regulator max current output = 1.5A Therefore, we check the endurance of the regulator to make sure that this device is sufficient.

Relative IEEE ethical issue

3. To be honest and realistic in stating claims or estimates based on available data
Our major technical part is the transformation between the Quadpod mode and the vehicle mode. We do not exaggerate any data that may give an impression that our transformation would be more that our real data and experience. We admit our design could have the parts that need an improvement and believe that we could get more valuable and critical review from other professional technicians if we provide our data honestly.

7. To seek, accept, and offer honest criticism of technical work, to acknowledge and correct errors, and to credit properly the contributions of others

We are trying to create new type of vehicle with simple design in regards of the practical usage. We are open to accept any honest criticism regarding technical issues by admitting our lack of knowledge for better and practical design and technical work. As we use some parts that are already made by a company, we must give a full credit to the company by mentioning their name and work in proper manner.

Cost Analysis:

Part #	Provider	Desc	For	Price	Qty	Total
900-00005	Parallax	Standard servo motor	Leg	$12.99	8	$103.92
DEV-11021	Arduino	Arduino Mega	Main	$49.99	1	$49.99
900-00008	Parallax	Continuous servo motor	Wheel	$12.99	2	$25.98
B3F-1000	ECE store	Button	Sensor	$0.50	6	$3.00
RTL-10783	Sparkfun	IR Control Kit Retail	Controller	$9.95	1	$9.95
N/A	ECE store	100ohm resistor	Sensor	$0.50	6	$3.00
LM7805	ECEstore	Voltage Regulator - 5V	Power	$1.25	1	$1.25
N/A	ECEstore	9V battery	Power	$2.80	2	$5.60
2S1P-74-20C	HobbyPartz	Blue Lipo 2-Cell 1500mAh 2S1P 7.4v 20C RC Battery w/ 4.0 banana connector	Power	$6.67	1	$6.67
Total						$209.36

Note: Cost for frame work from Machine shop is not included.

Name	Hourly Rate	Total Hours Invested	Total =Hourly Rate X 2.5 X total hours Invested
Kee Woong Haan	$30.00	125	$9,375
Jiwon Park	$30.00	125	$9,375
Zenon Son	$30.00	125	$9,375
Total	$90.00	375	$28,125

Grand total: $28,334.36

Schedule:

Week	Kee Woong Haan	Zenon Son	Jiwon Park
9/16	Project Proposal	Project Proposal	Project Proposal
9/23	Overall frame design, Research Remote Controller.	Research Arduino. Work on Design Review. Work on Flow Chart.	Research on Servo motors and Rotation Motors, Research Arduino. Work on Flow Chart
9/30	Order parts on machine shop. Work on schematic for Design Review.	Work on Schematic using eagle. Finish up Design Review and upload.	Order parts. Obtain servo motor and test them using function generator.
10/7	Vehicle Frame Work (Vehicle Structure, Sensor Bumper) Remote Controller.	Coding for Vehicle mode of the project. Debugging.	Installing Electrical components For vehicle mode (Wheel motor, power supply, arduino).
10/14	Quadpod Frame Work (Legs, Joint, Rubber footing, leg sensor)	Coding for Quadpod mode of the project. Debugging.	Installing Electrical components For quadpod mode (Vertical leg motion motors, Horizontal leg motion motors)
10/21	Individual Process Report. Make necessary adjustment on the frame work.	Individual Process Report. Combine codes for Quadpod mode and Vehicle mode.	Individual Process Report. Fixes on electrical components for both quadpod mode and vehicle mode of the project
10/28	Overall debugging of sensor part.	Overall debugging of arduino.	Overall debugging of motor part.
11/4	Mock up Demos		
11/11	Work on Presentation and Paper. (Sensors)	Work on Presentation and Paper. (Arduino)	Work on Presentation and Paper. (Motor)
11/18	Thanksgiving Break (Recover Late Work)		
11/25	Continue preparation for Demo and Presentation.	Continue preparation for Demo and Presentation.	Continue preparation for Demo and Presentation.
12/2	Demo and Presentation. Work on Final paper. (Sensors)	Demo and Presentation. Work on Final paper. (Arduino)	Demo and Presentation. Work on Final paper. (Motor)
12/9	Check out.	Check out.	Finish up final paper and turn in.

Appendix

/* Button testing code */ [5]

```
int ledPin = 12;                      // LED is connected to pin 12
int switchPin = 2;                    // switch is connected to pin 2
int val;                             // variable for reading the pin status


void setup() {
  pinMode(ledPin, OUTPUT);         // Set the LED pin as output
  pinMode(switchPin, INPUT);        // Set the switch pin as input
}


void loop(){
  val = digitalRead(switchPin);      // read input value and store it in val
  if (val == LOW) {                          // check if the button is pressed
    digitalWrite(ledPin, HIGH);     // turn LED on
  }
  if (val == HIGH) {                         // check if the button is not
pressed
    digitalWrite(ledPin, LOW);      // turn LED off
}
```

/*Code for standard servo motor verification*/[2]

```
'{$STAMP BS2}
'{$PBASIC 2.5}

counter VAR Word

Main:
  FOR counter = 1 TO 100 ' Loop for 2 seconds
    PULSOUT 0, 1000      ' Servo counterclockwise
    PAUSE 20
  NEXT

  FOR counter = 1 TO 100 ' Loop for 2 seconds
    PULSOUT 0, 500       ' Servo clockwise
    PAUSE 20
  NEXT

  FOR counter = 1 TO 100 ' Loop for 2 seconds
    PULSOUT 0, 750       ' Servo center
    PAUSE 20
  NEXT

  GOTO Main
```

/*Code for continuous rotation servo motor verification*/[3)]

<u>* Calibration</u>

```
'{$STAMP BS2}
'{$PBASIC 2.5}

DO  ' Repeat forever
  PULSOUT 0, 750 ' Stop
  PAUSE 20
LOOP
```

<u>* Demonstration Code</u>

```
'{$STAMP BS2}
'{$PBASIC 2.5}

counter VAR Word

DO  ' Repeat forever
  FOR counter = 1 TO 100 ' Loop for 2 seconds
    PULSOUT 0, 850 ' Servo counterclockwise
    PAUSE 20
  NEXT

  FOR counter = 1 TO 100 ' Loop for 2 seconds
    PULSOUT 0, 650 ' Servo clockwise
    PAUSE 20
  NEXT

  FOR counter = 1 TO 100 ' Loop for 2 seconds
    PULSOUT 0, 750 ' Servo stop
    PAUSE 20
  NEXT

LOOP
```

/*Code for Remote control verification*/

Code1[4)]

```
/*
  SparkFun Electronics 2011
  OSHW License http://freedomdefined.org/OSHW

  IR TX(LED) and RX(Receiver) demo at 38k Hz
  -Outputs a 38KHz wave on pin 11
  -takes input from TSOP382 on pin 2

  The IR LED carrier wave of 38kHz is turned on and off
  to blink and LED.

*/

//define your square wave frequency
#define IR_CLOCK_RATE    38000L

int ledPin = 13;       // the pin that the LED is attached to

void setup() {
  // toggle pin 11 on compare
  TCCR2A = _BV(WGM21) | _BV(COM2A0);
  TCCR2B = _BV(CS20);

  // 38kHz timer
  OCR2A = (F_CPU/(IR_CLOCK_RATE*2L)-1);
  pinMode(11, OUTPUT);

  //endable an interrupt on pin 2, when there is a falling edge
  //jump to the blink function
  attachInterrupt(0, blink, FALLING);

  //initialize serial
  Serial.begin(9600);
}

void loop() {
  //turn the 38kHz carrier wave off and on
  TCCR2B = 0;
  delay(500);
  TCCR2B = _BV(CS20);
  delay(500);
}
void blink() {
    //blink the LED and print 'hit'
    digitalWrite(ledPin, HIGH);
    Serial.println("hit");
    delay(800);
    digitalWrite(ledPin, LOW);
}
```

Code2[4)]

```
      /*
  SparkFun Electronics 2010
  Playing with IR remote control

  IR Receiver TSOP382: Supply voltage of 2.5V to 5.5V
  With the curved front facing you, pin 1 is on the left.
  Attach
    Pin 1: To pin 2 on Arduino
    Pin 2: GND
    Pin 3: 5V

  This is based on pmalmsten's code found on the Arduino forum from 2007:
  http://www.arduino.cc/cgi-bin/yabb2/YaBB.pl?num=1176098434/0

  This code works with super cheapo remotes. If you want to look at the
individual timing
  of the bits, use this code:
  http://www.arduino.cc/playground/Code/InfraredReceivers

  This code clips a lot of the incoming IR blips, but what is left is
identifiable as key codes.

*/

int irPin = 2; //Sensor pin 1 wired to Arduino's pin 2
int statLED = 13; //Toggle the status LED every time Power is pressed
int start_bit = 2200; //Start bit threshold (Microseconds)
int bin_1 = 1000; //Binary 1 threshold (Microseconds)
int bin_0 = 400; //Binary 0 threshold (Microseconds)

void setup() {
  pinMode(statLED, OUTPUT);
  digitalWrite(statLED, LOW);

  pinMode(irPin, INPUT);

  Serial.begin(9600);
  Serial.println("Waiting: ");
}

void loop() {
  int key = getIRKey();           //Fetch the key

  if(key != 0) //Ignore keys that are zero
  {
    Serial.print("Key Recieved: ");
    switch(key)
    {
      case 144: Serial.print("CH Up"); break;
      case 145: Serial.print("CH Down"); break;
      case 146: Serial.print("VOL Right"); break;
      case 147: Serial.print("VOL Left"); break;
      case 148: Serial.print("Mute"); break;
      case 165: Serial.print("AV/TV"); break;
      case 149:
```

```
        Serial.print("Power");
        if(digitalRead(statLED) != 1) //This toggles the statLED every time
power button is hit
          digitalWrite(statLED, HIGH);
        else
          digitalWrite(statLED, LOW);
        break;

      default: Serial.print(key);
    }

    Serial.println();
  }
}

int getIRKey() {
  int data[12];
  int i;

  while(pulseIn(irPin, LOW) < start_bit); //Wait for a start bit

  for(i = 0 ; i < 11 ; i++)
    data[i] = pulseIn(irPin, LOW); //Start measuring bits, I only want low
pulses

  for(i = 0 ; i < 11 ; i++) //Parse them
  {
    if(data[i] > bin_1) //is it a 1?
      data[i] = 1;
    else if(data[i] > bin_0) //is it a 0?
      data[i] = 0;
    else
      return -1; //Flag the data as invalid; I don't know what it is! Return
-1 on invalid data
  }

  int result = 0;
  for(i = 0 ; i < 11 ; i++) //Convert data bits to integer
    if(data[i] == 1) result |= (1<<i);

  return result; //Return key number
}
```

Reference

1. “Arduino – ArduinoMega2560.” *Arduino – ArduinoMega2560*.N.P., n.d. Web. 30 Sept 2012. < http://arduino.cc/en/Main/ArduinoBoardMega2560>

2. “Parallex Standard Servo.” *Parallex inc – Parallex Standard Servo.*NP.,n.d. Web. 28 Sept 2012. < http://learn.parallax.com/kickstart/900-00005>

3. “Parallex Continuous Rotation Servo.” *Parallex inc – Parallex Continuous Rotation Servo.*NP.,n.d. Web. 28 Sept 2012. < http://learn.parallax.com/KickStart/900-00008>

4. “IR Control Kit Landing Page.” *SparkfunElectronics – IR Control Kit Landing Page*.NP.,n.d. Web. 30 Sept 2012. < http://www.sparkfun.com/tutorials/291>

5. “Arduino Tutorial” *ladyada.net– Arduino Tutorial – Lesson 5*.NP.,n.d. Web. 28 Sept 2012. < http://www.ladyada.net/learn/arduino/lesson5.html>

6. “Parallax Standard Servo” *Standard Servo.* NP.,n.d. Web. 28 Sept 2012. < http://www.parallax.com/Portals/0/Downloads/docs/prod/motors/900-00005-StdServo-v2.2.pdf>

7. “Parallax Continuous Rotation Servo” *Continuous Rotation Servo.* NP.,n.d. Web. 28 Sept 2012. http://www.parallax.com/Portals/0/Downloads/docs/prod/motors/900-00008-CRServo-v2.2.pdf>

8. “Tactile Switch 3BF”. NP.,n.d. Web. 28 Sept 2012. <http://www.adafruit.com/datasheets/B3F-1000-Omron.pdf>

9. “IR Receiver Modules for Remote Control Systems” *Datasheet (TSOP38238)*. NP.,n.d. Web. 29 Sept 2012. < http://www.sparkfun.com/datasheets/Sensors/Infrared/tsop382.pdf>

10. “IR Control Kit Retail” *SparkfunElectronics – IR Control Kit Retail* NP.,n.d. Web. 29 Sept 2012. < https://www.sparkfun.com/products/10783 >

11. “7.4V battery” HobbyPartz –Blue Lipo 2-Cell 1500mAh 2S1P 7.4v 20C RC Battery w/ 4.0 banana connector, Web. 3 Oct. 2012. < http://www.hobbypartz.com/83p-1500mah-2s1p-74-20c.html>

12. “Voltage Regulator - 5V” – *Positive voltage regulator*. Web. 03 Oct. 2012. < https://www.sparkfun.com/datasheets/Components/LM7805.pdf>

13. “Duracell 9V” –*Duracell coppertop*. Web. 03 Oct. 2012. < http://datasheet.octopart.com/MN1604-Duracell-datasheet-5306548.pdf>