“TEAM ARYANS RACING”

DATA-ACQUISITION REPORT

"In Pursuit Of Technical Excellence"

PROJECT REPORT ON

"DATA ACQUISITION OF ATV (ALL TERRAIN VEHICLE)"

Omkar Jadhav - DAQ Lead
Shantanu Kingaonkar – DAQ Member
Apurva Joshi – DAQ Member
Renukadas Bharaswadkar – DAQ Member

"TEAM ARYANS RACING"

GOVERNMENT COLLEGE OF ENGINEERING AURANGABAD
(An Autonomous Institute Of Government of Maharashtra, India)

(2017 –2019)

Goal: -

CVT Tuning setup to get practical values of engangement RPM and shift RPM.
Obtaining combined graph of RPM of primary and secondary pulley with distance covered simultaneously from wireless transceiver using MATLAB.

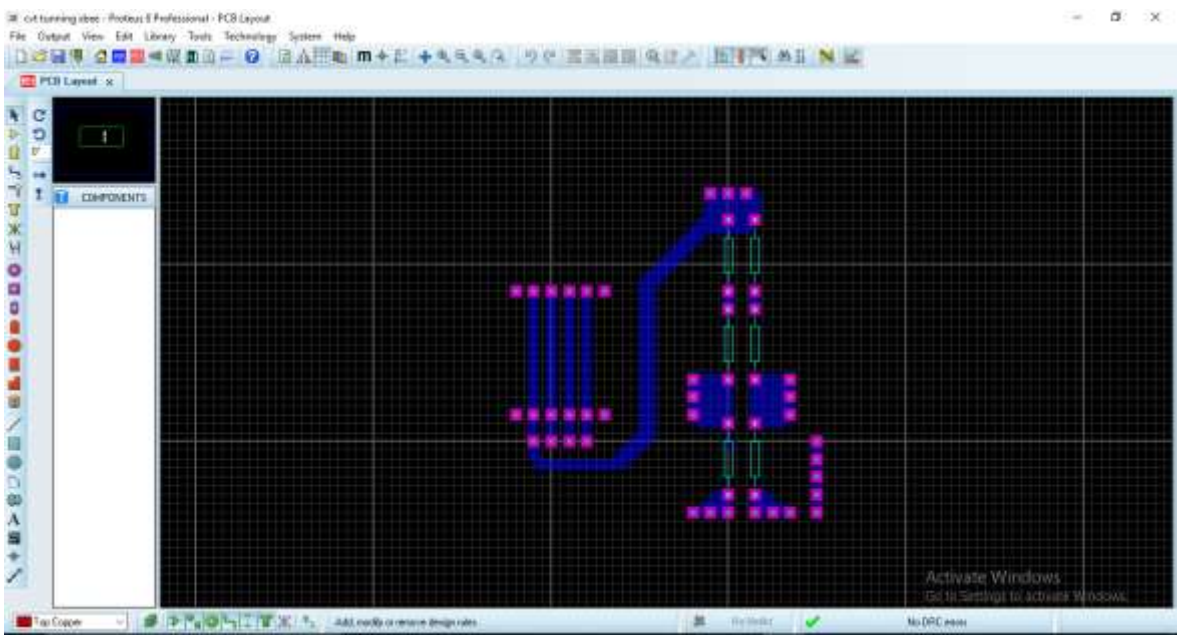
Hardware: -

Components:-

1) Pair of Xbee
2) Arduino due
3) Pair of antenna
4) Reed switch X 2
5) Arduino uno
6) Resistors
7) Connecting wires
8) Battery (12V)

We used long range wireless transceiver for data transmission and used a pull-down resistor to discard tri-stated logic. For faster processing we used ARM based processor. A 12V battery is used to power the processor for uninterrupted CVT tuning.

Software Design: -
PCB design :-



Here, the program used for transmitter is as follows:-

```
double t1 = 0, t = 0, x2, x1, tp = 0, ts = 0;
double rps1 = 0, rps2 = 0;
float rpm1 = 0, rpm2 = 0;
int flag = 0;
void setup()
{
  Serial.begin(9600);
  attachInterrupt(1, PRM, RISING);
  attachInterrupt(0, ISR0, RISING);
}
void loop()
{
 Serial.println('<');
 Serial.println(rpm1);
 Serial.println(rpm2);
 Serial.println('>');
}
void PRM()
```

```
{ flag++;
  if (flag == 1)
    tp = micros();
  if (flag == 2)
  {
    detachInterrupt(1);
    t1 = micros() - tp;
    x1 = t1 / 1000000;
    rps1 = 1.0 / x1;
    rpm1 = 60.0 * rps1;
    Serial.println(rpm1);
    flag = 0;
    delay(1);
    //t1=millis();
    attachInterrupt(1, PRM, RISING);
  }
}
void ISR0()
{ flag++;
  if (flag == 1)
    ts = micros();
  if (flag == 2)
  {
    detachInterrupt(0);
    t = micros() - ts;
    x2 = t / 1000000;
    rps2 = 1.0 / x2;
    rpm2 = 60.0 * rps;
    if (rpm2 < 5000)
      Serial.println(rpm2 * 8.45);
    flag = 0;
    delay(1); millis();
    attachInterrupt(0, ISR0, RISING);
  }
}
```

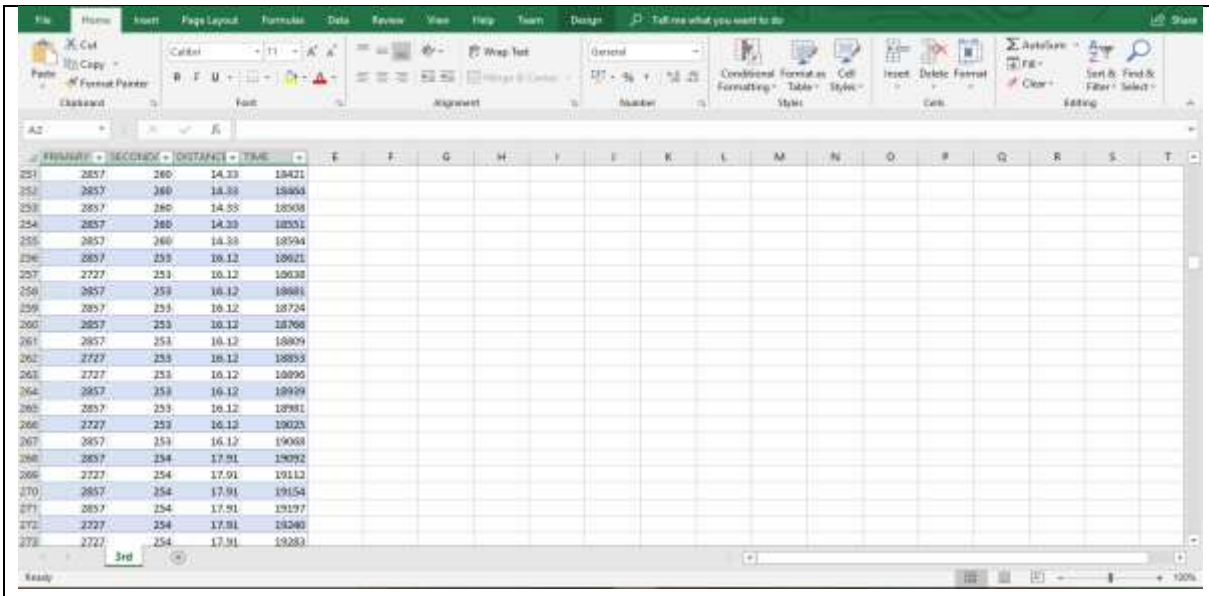
Program used for receiver is: -

```
bool started= false;
bool ended       = false;
char incomingByte ;
char msg[30];
byte index;

void setup() {
          Serial.begin(9600);
}
```

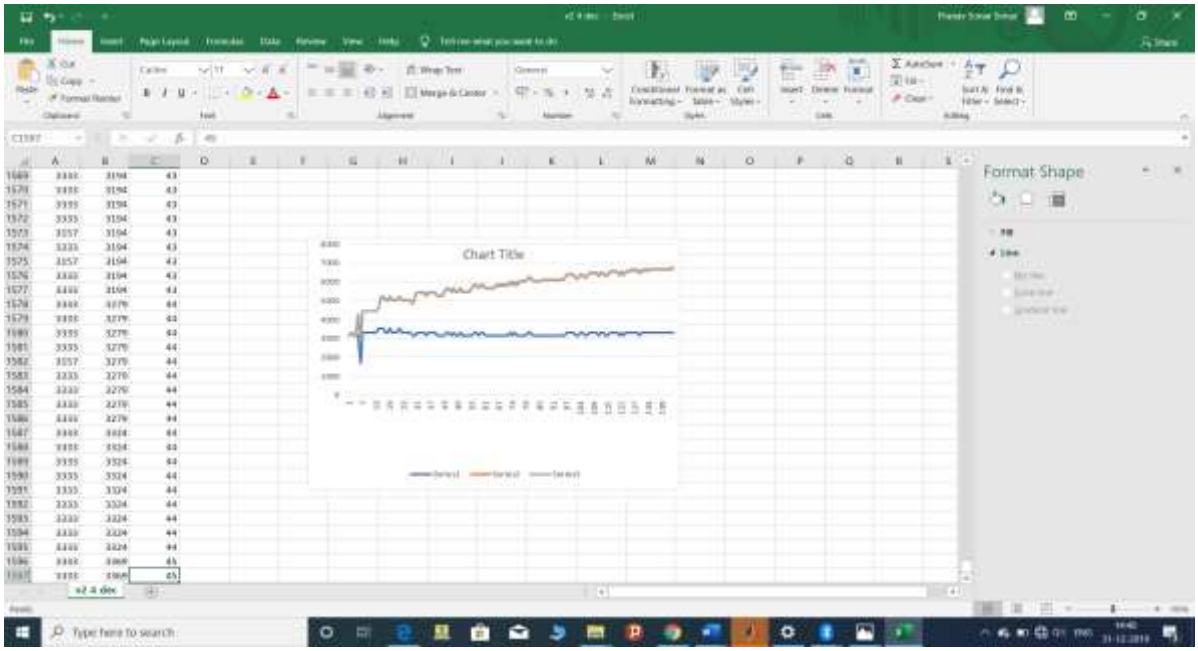
```
void loop() {

 while (Serial.available()>0){
  incomingByte = Serial.read();
  if(incomingByte == '<')
  {
   started = true;
   index = 0;
   msg[index] = '\0';
  }

  else if(incomingByte == '>')
  {
   ended = true;
   break;
  }
  else
  {
   if(index < 31)
   {
    msg[index] = incomingByte;
    index++;
    msg[index] = '\0';
   }
  }
 }

 if(started && ended)
 {
  Serial.println(msg);
  index = 0;
  msg[index] = '\0';
  started = false;
  ended = false;
 }
}
```

Results & Achievements: -

Using above, we successfully plotted the RPM graphs of primary and secondary pulley in MATLAB and based upon this values, the parameters of CVT are varied for better tunning.



PRIMARY	SECONDS	DISTANCE	TIME	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
2857	260	14.33	18421																
2857	260	14.33	18466																
2857	260	14.33	18508																
2857	260	14.33	18551																
2857	260	14.33	18594																
2857	253	16.12	18621																
2727	253	16.12	18638																
2857	253	16.12	18681																
2857	253	16.12	18724																
2857	253	16.12	18766																
2857	253	16.12	18809																
2727	253	16.12	18853																
2727	253	16.12	18896																
2857	253	16.12	18939																
2857	253	16.12	18981																
2727	253	16.12	19025																
2857	253	16.12	19068																
2857	254	17.91	19092																
2727	254	17.91	19112																
2857	254	17.91	19154																
2857	254	17.91	19197																
2727	254	17.91	19240																
2727	254	17.91	19283																



Goal: -
To measure the shock travel to increase the performance of vehicle.

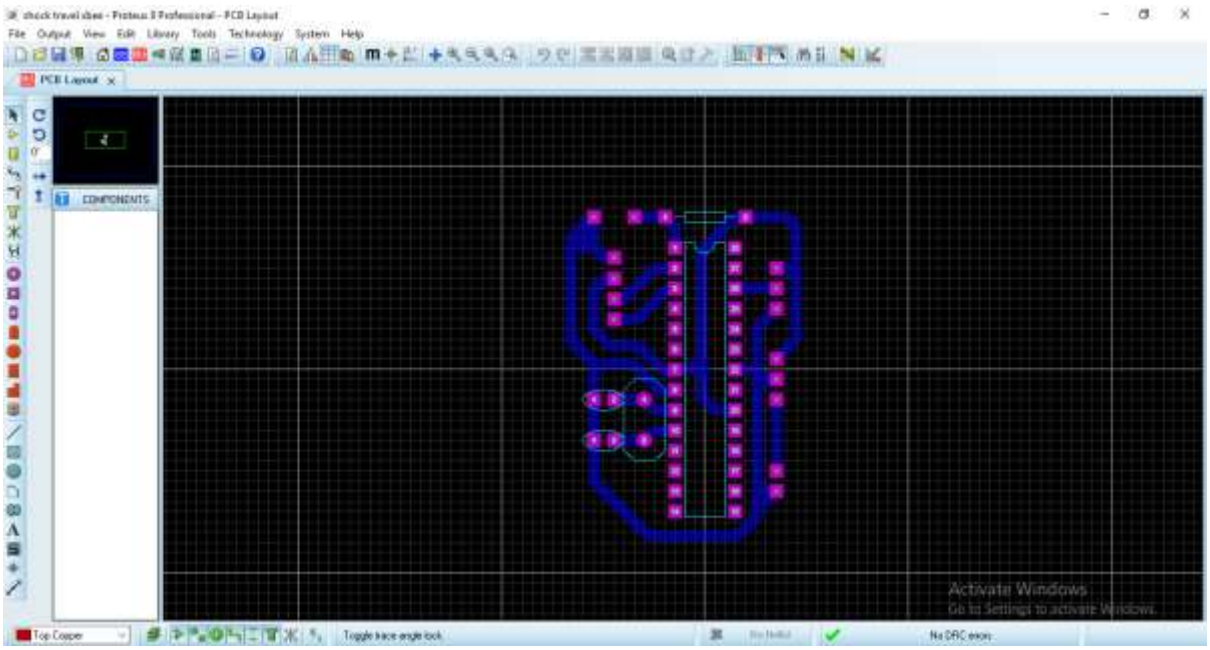
Hardware: -

Components:-
1) Linear Potentiometer.
2) Pair of Xbee.
3) Pair of antenna.
4) PCB of Arduino UNO.
5) Battery (9V).

We used customized linear potentiometer for measuring the change in length of shock. Resistance of the sensor used is 20kΩ. It works on a principle that it gives varying output voltage when its length is changed as resistance depends on the length. We calibrated the sensor by giving known length of compression, we plotted the table in which there are output voltage and respective compression lengths. After plotting the graph in MATLAB, we acquired the equation which gives travel of shock with respect to voltage change. So, mounting it with the shock we made several iterations which helped us in better shock tuning. The readings given by the sensor were received wirelessly using transceiver and we plotted the Travel vs Time graph in MATLAB.

Software Design: -

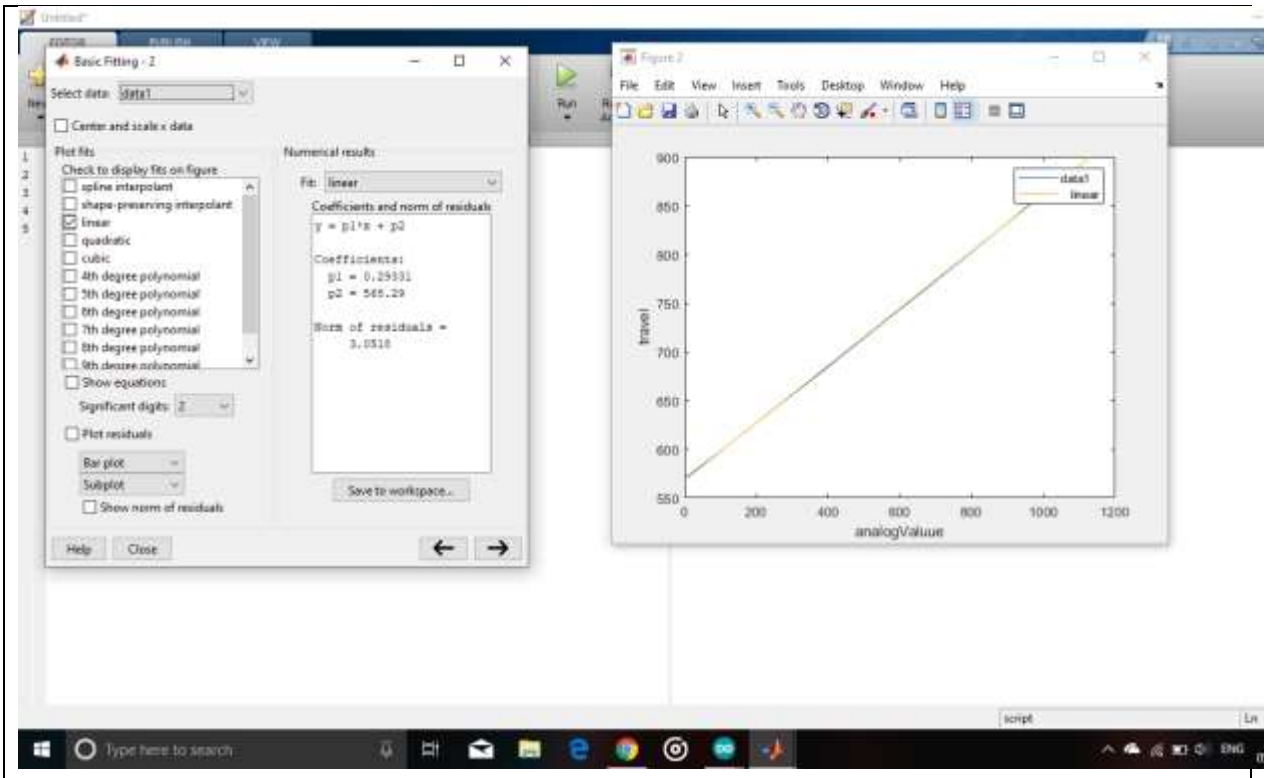
PCB design:-



For accurate and reliable readings, we calibrated the sensor by plotting the curve of travel vs voltage by applying known compression, in MATLAB to check the behaviour of graph. Using the equation of curve, we acquired the readings.

Equation: -

y= (0.29331) * X + 568.29



Program used for transmitter is shown below: -

```
int time;
void setup() {
  Serial.begin(9600);
}

void loop() {

  int sensorValue = analogRead(A3);
  float shock= 0.29331*sensorValue + 568.29;

  Serial.print('<' );
  Serial.print(shock);
  time=millis();
  Serial.print(',');
  Serial.print(time);
  Serial.print('>');
}
```

Program for receiver: -

```
bool started= false;
bool ended  = false;
char incomingByte ;
```

```
char msg[6];
byte index;

void setup() {

  Serial.begin(9600);
}

void loop() {

  while (Serial.available()>0){
    incomingByte = Serial.read();
    if(incomingByte == '<')
    {
     started = true;
     index = 0;
     msg[index] = '\0';
    }
   else if(incomingByte == '>')
   {
     ended = true;
   }
   else
   {
     if(index < 7)
     {
       msg[index] = incomingByte;
       index++;
       msg[index] = '\0';
     }
   }
 }

if(started && ended)
 {
   Serial.println(msg);
   index = 0;
   msg[index] = '\0';
   started = false;
   ended = false;
 }
}
```

Results & Achievements: -

With the help of MATLAB, we successfully plotted the graphs of multiple iterations which helped us in better shock tuning for dynamics of vehicle and also for the driver's comfort.

Goal: -

Designing a wireless kill switch for driver's safety while testing in hilly and rough areas.

Hardware: -

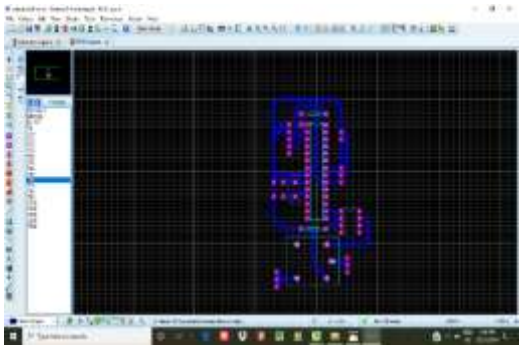
Components:-

1) Pair of Xbee
2)Pair of antenna
3)Relay operating on 5V
4) Optocoupler EL 817
5) PCB of Arduino X 2
6) Battery (12V)

We used wireless transceiver Xbee S2C PRO for signal transfer. A 5V operating relay is used which is triggered using optocoupler EL 817 and the complete circuit is powered using an external 12V power supply. The optocoupler gets digitally high supply to the anode when "5678" code comes from Xbee and the cathode is grounded through a resistor. Thus, the circuit gets completed and the phototransistor is triggered. Hence the voltage through emitter terminal is received by coil of relay whose another terminal is already grounded. The kill switch wire from engine connected to the normally open terminal of relay gets a lower potential from common terminal of relay which is grounded and hence is turned off.

Software Design: -

PCB design :-



The program used for transmitter is: -

```
void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);
  pinMode(10, INPUT);
}

void loop() {
  // put your main code here, to run repeatedly:
  if (digitalRead(10) == 1)
  {
    Serial.print('<');
    Serial.print("5678");
    Serial.println('>');
  }
  else
  {
    Serial.print('<');
    Serial.print("0000");
    Serial.println('>');
  }
}
```

Program of receiver:-

```
const int opto = 13;

bool started = false;
bool ended         = false;
char incomingByte ;
char msg[4];
byte index;

void setup() {
  Serial.begin(9600);
  pinMode(opto, OUTPUT);
  pinMode(10, OUTPUT);
}

void loop() {
    digitalWrite(10, HIGH);
  while (Serial.available() > 0) {
    incomingByte = Serial.read();
    if (incomingByte == '<')
    {
```

```
    started = true;
    index = 0;
    msg[index] = '\0';
   }
   else if (incomingByte == '>')
   {
    ended = true;
   }
   else
   {
    if (index < 5)
    {
     msg[index] = incomingByte;
     index++;
     msg[index] = '\0';
    }
   }
  }

  if (started && ended)
  {
   int value = atoi(msg);
   Serial.println(value);
   if (value == 5678)
   {
    digitalWrite(opto, HIGH);
   }
   if (value==0)
   {
    digitalWrite(opto, LOW);
   }
   value = 0;
   index = 0;
   msg[index] = '\0';
   started = false;
   ended = false;
  }
}
```

Results and Achievements: -

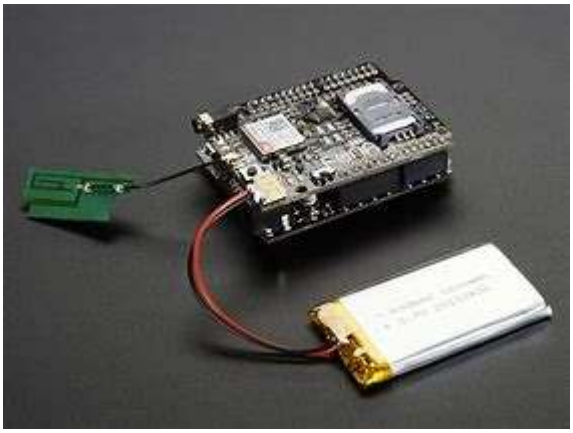
We successfully designed a wireless kill switch which can operate in 1km range practically and conducted our testing in various hilly areas.

Goal:-

GSM for driver- pit communication during race.

Hardware:-

1) Adafruit GSM shield
2) Arduino UNO
3) Headset
4) Sim card
5) UFL Antenna
6) Li-ion battery



Arduino program for GSM

```
// FONA Incoming Call Number Example
// Listens for a call and displays the phone number of the caller (if available).
// Use this example to add phone call detection to your own FONA sketch.
#include "Adafruit_FONA.h"

// Pins which are connected to the FONA.
// Note that this is different from FONAtest!
#define FONA_RX          2
#define FONA_TX          3
#define FONA_RST         4
int pushButton=8;
// Note you need to map interrupt number to pin number
// for your board.  On an Uno & Mega interrupt 0 is
// digital pin 2, and on a Leonardo interrupt 0 is
// digital pin 3.  See this page for a complete table:
//   http://arduino.cc/en/Reference/attachInterrupt
// Make sure this interrupt pin is connected to FONA RI!
#define FONA_RI_INTERRUPT  0

// We default to using software serial. If you want to use hardware serial
// (because softserial isnt supported) comment out the following three lines
// and uncomment the HardwareSerial line
#include <SoftwareSerial.h>
```

```
SoftwareSerial fonaSS = SoftwareSerial(FONA_TX, FONA_RX);
SoftwareSerial *fonaSerial = &fonaSS;

// Hardware serial is also possible!
//  HardwareSerial *fonaSerial = &Serial1;

Adafruit_FONA fona = Adafruit_FONA(FONA_RST);

void setup() {
  Serial.begin(115200);
    pinMode(pushButton, INPUT);
  Serial.println(F("FONA incoming call example"));
  Serial.println(F("Initializing....(May take 3 seconds)"));

  fonaSerial->begin(4800);
  if (! fona.begin(*fonaSerial)) {
    Serial.println(F("Couldn't find FONA"));
    while(1);
  }
  Serial.println(F("FONA is OK"));

  // Enable incoming call notification.
  if(fona.callerIdNotification(true, FONA_RI_INTERRUPT)) {
    Serial.println(F("Caller id notification enabled."));
  }
  else {
    Serial.println(F("Caller id notification disabled"));
  }
}

void loop(){
  // Create a small string buffer to hold incoming call number.
  char phone[32] = {0};
  int buttonState = digitalRead(pushButton);

    Serial.println(F("RING!"));
 fona.pickUp();
}
```

Results and Achievements:-

With the help of this, we successfully communicated the driver in dynamic state.

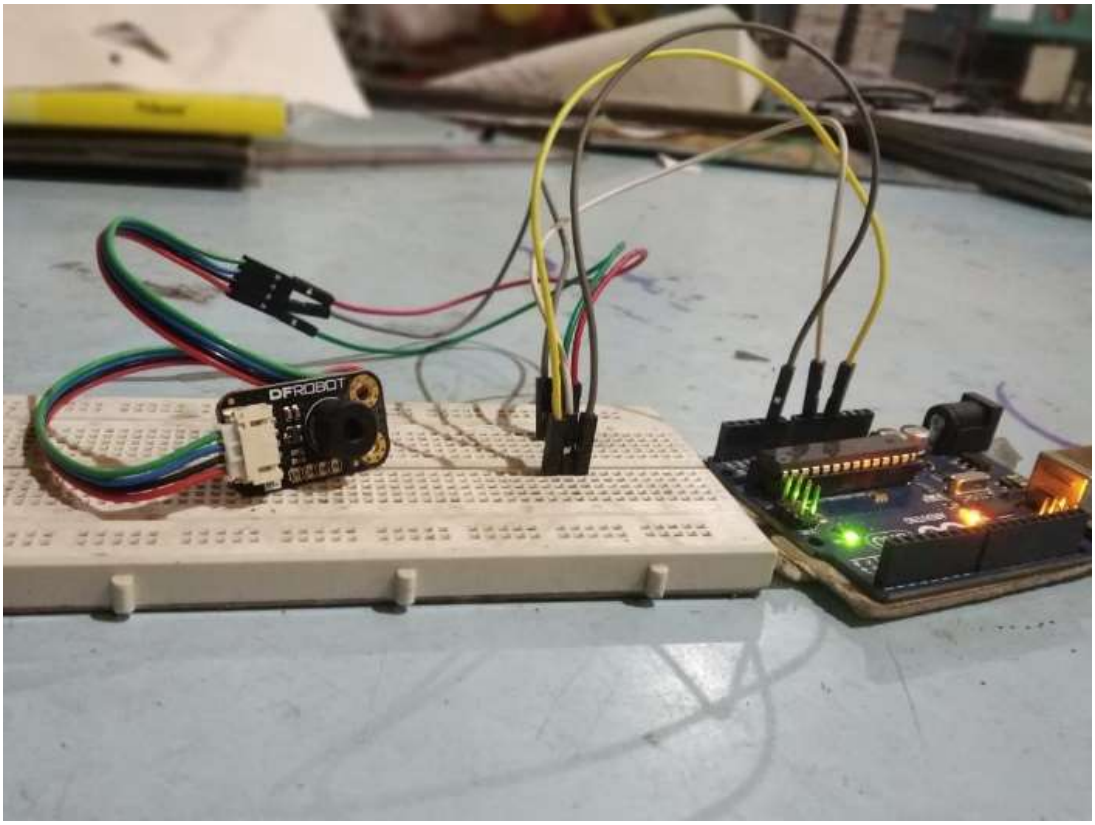
Goal: -

Temperature measurement of different heating components to prevent its mechanical losses and increasing its sustainability.

Hardware: -

Components: -
1) Arduino UNO.
2) IR Temperature Sensor MLX90614.
3) Connecting wires.

We used an infrared non-contact temperature sensor to measure temperature. Holding it at a particular distance from the target, it senses the temperature and displayed on serial monitor.



Software Design: -

Program used for this purpose is as follows: -

```
#include <Wire.h>
#include <SparkFunMLX90614.h>

IRTherm therm;
```

```
const byte LED_PIN = 8;

void setup()
{
  Serial.begin(9600);
  therm.begin();
  therm.setUnit(TEMP_F);

  pinMode(LED_PIN, OUTPUT);
  setLED(LOW);
}

void loop()
{
  setLED(HIGH);

  if (therm.read())
  {

    Serial.print("Object: " + String(therm.object(), 2));
    Serial.write('°');
    Serial.println("F");
    Serial.print("Ambient: " + String(therm.ambient(), 2));
    Serial.write('°');
    Serial.println("F");
    Serial.println();
  }
  setLED(LOW);
  delay(500);
}

void setLED(bool on)
{
  if (on)
    digitalWrite(LED_PIN, LOW);
  else
    digitalWrite(LED_PIN, HIGH);
}
```

Results and Achievements: -

Using this sensor, we acquired the following data as follows: -

SR NO.	PART	TEMPERATURE	
		FAHRENHEIT	CELSIUS
1.	ENGINE	104	40
2.	MUFFLER	570	298.88
3.	GEARBOX CASING	130	54.44
4.	CVT SHIELDING	130	54.44
5.	PRIMARY PULLEY	140	60
6.	SECONDARY PULLEY	150	65.66
7.	BELT	160	71.11
8.	FIREWALL(UP)	84	28.88
9.	FIREWALL (NEAR ENGINE)	134	56.66

From these values, we tried to execute a better cooling arrangement for CVT and its components.