REFERENCES

[1] Advanced Maintenance Solutions. "Successful Belt Maintenance." Internet: http://www.theadvancedteam.com/cases/sevensteptbeltmaintenance.html, Oct. 28, 1999 [Dec. 21, 2013].

[2] Allen Collier. "The math involved in figuring weight distribution."Internet: http://www.truckview.com/weight_distribution.html, [Nov. 25, 2013].

[3] Ashutosh Dubey and Vivek Dwivedi. "Load Cases & Boundary Conditions for Stress Analysis." *Journal of Vehicle Chassis Analysis,* vol.111, pp. 230, Jan.2004.

[4] Carlisle Power Transmission Products. "V Belt tensioning." Internet: http://www.c-rproducts.com/Industrialvbeltdrivesdesignguide.html, [Oct. 10, 2013].

[5] Department of Motor Traffic Sri Lanka. "Statistics." Internet: https://www.motortraffic.gov.lk.statistics.html, Feb 28, 2011 [Jan. 05, 2013].

[6] D.G Alciatore. "Multipulley Belt Drive Mechanics, Creep Theory vs. Shear Theory." *Mechanical Design,* vol. 117, pp. 509, Dec.1995.

[7] Dike N. Ahanotu. "Measurement Of Class-Specific Temporal And Spatial Variability." *Heavy-Duty Vehicle Weight and Horsepower Distributions*, vol. 29, pp. 26, July. 1999.

[8] Guang Tong and Xiaoxiong Jin. "Simulation of Radial Tire Wear Characteristics." *Journal of Study on the Automotive Engineering*, vol.16, pp. 23, Oct. 2003.

[9] H.A Lupker and F. Montanaro."Truck Tyre Wear Assessment and Prediction." presented at the 7th International Symposium on Heavy Vehicle Weights & Dimensions Delft, Netherlands, 2002.

[10] I.I.T. Kharagpur. "Belt drives." Internet: http://www.nptel.iitk.ac.in/courses/-Webcourse-contents/IITKharagpur/Machinedesign.html, [Apr. 13, 2013].

[11] "India ousts Japan as largest vehicle supplier to Sri Lanka." *Business Times* (Feb. 17, 2013), sec.A p.2.

[12] Informal Manual. "The flat belt drive compared to the V – Belt drive." Internet: http://www.lipsia-componentry.ru/content/files/ploskie.html, Jan 02, 1992 [Jul. 18, 2013].

[13] J. Phys. "Measurement of Vehicle Load using Capacitance and Acceleration Transducers." *Journal of the Vehicle Load Distribution,* vol31, pp. 19, Nov.2007.

[14] Loren Cook Company. (1999). *Engineering Cook Book*. (2nd edition). [On-line]. Available: http://www.lorencook.com/catalogs/cookbook.html [May 21, 2013].

[15] LIN Steering Group. "LIN Specification Package." Internet: http://www.lin-subbus.org, Nov 24, 2006 [Dec. 24, 2013].

[16] Mikel P. Groover. *Automation, Production Systems, and Computer – Integrated Manufacturing*. New Jersey, U.S.A. Pearson Education, 2010, pp. 128-33.

[17] O.Reynolds. "Creep Theory of Belt Drive Mechanics." *The Engineer*, vol. 38, pp. 396, Jan. 1990.

[18] Roads and Traffic Authority, Australia. "Guide Lines for Heavy Vehicle Mass, Loading and Access." Internet: https://www.docstoc.com/Heavy-Vehicle-Mass-Loading-and-Access-National Heavy-Vehicle.html, Feb 02, 2010 [Mar. 12, 2012].

[19] Service Manual. *MTU marine engines.* Friedrichshafen GmbH, Germany. Motoren Turbinen Union Company, 1994, pp 41-45.

[20] Stephen I. Dains. "Completed Vehicle Weight Distribution Analysis." Internet: https://www.fleet.ford.com/TRUCKBBAS.html, Mar. 05, 2004 [Sept. 25, 2012].

[21] Stephen I. Dains. "Vehicle Weight Distribution Analysis." *Heavy Vehicle suspensions,* vol. 33, pp.17, April. 2002.

[22] T.Mohammad. "Using Ultrasonic and Infrared Sensors for Distance Measurement." *Journal of world academy of science*, vol.55, pp. 27, Dec.2009.

[23] Traffic Monitoring Guide. "Truck Weight Monitoring." Internet: https://www.ewu.edu/groups/trafficmonitoringguide/TMGsection5.html, May 01, 2001 [Jan. 25, 2012].

[24] U. Kahangamage. BSc Class Lecture, Topic: "Machine Elements and Design." Faculty of Engineering, University of Moratuwa, Katubedda, Sri Lanka, Apr. 10, 2005.

Annex A

RESEARCH DATA SHEET

Vehicle Model –Prime mover
Vehicle Make – TATA
Engine Model – TATA Cummins BJS 130-70
Estimated Hose Power – 180 at 2900 rpm
Venue -- Navy Camp Walisara

Date of commence 25th April 2012
Date of terminated 28th August 2012

Practical done by Chathura Sirimanna Type of Belt used – 8 PK Belt 1422

Height of the belt (a)
Width of the belt (w)
Normal length (l)
Tensioned length (L)
Angle of belt adjuster (α)

Table A1: Variation of parameters in fan belt with its increasing running millage

Practical No	Mileage driven by the belt	Height(a)	Width (w)	Length Normal	Length Tensioned	Angle	Total Force on the belt
01	0-10000 Km	5.03 mm	28.46 mm	1450 .00mm	1453.00 mm	30	39 Kg
02	10000-20000Km	5.00 mm	27.88 mm	1452 mm	1456.00 mm	20	36 Kg
03	20000-30000Km	4.95 mm	27.70mm	1453.00mm	1457.00 mm	19	35Kg
04	30000-40000Km	4.90 mm	27.58 mm	1454 .00mm	1457.50 mm	18.5	34.7 Kg
05	40000-50000Km	4.85mm	27.49mm	1454.50mm	1458.00mm	18	34.2Kg
06	50000-60000Km	4.80mm	27.40mm	1455.00mm	1459.00mm	17	34Kg
07	60000-70000Km	4.70mm	27.30mm	1455.50mm	1560.00mm	16	33.3Kg
08	70000-80000Km	4.65mm	27.23mm	1456.00mm	1561.00mm	15.5	32.8Kg
09	80000-90000Km	4.60mm	27.14mm	1456.50mm	1562.00mm	15	32.3Kg
10	90000-100000Km	4.56mm	27.10mm	1457.00mm	1463.00mm	14	32Kg

Annex B

RESEARCH DATA SHEET

Vehicle Model –Truck
Vehicle Make – TATA
Engine Model – TATA B 59 160 10 2516 C
Estimated Hose Power – 160 Hp
Venue Navy Camp Walisara
Practical done by Chathura Sirimanna

Date of commence 30th April 2012
Date of terminated 28th September2012

Height of the belt (a) Normal length (l) Angle of belt adjuster (α)
Width of the belt (w) Tensioned length (L)

Table B1: Research Data of TATA 2516 Ten Wheeler Truck

Practical No	Mileage driven by the belt	Height(a)	Width (w)	Length Normal	Length Tensioned	Angle	Force on the belt
01	0-10000 Km	5.30mm	28.46 mm	1450mm	1452mm	30	40Kg
02	10000-20000Km	5.10mm	28.40 mm	1452mm	1453mm	29	39Kg
03	20000-30000Km	4.80mm	28.35 mm	1453mm	1455mm	25	39Kg
04	30000-40000Km	4.82mm	28.33 mm	1454mm	1456.8mm	20	38Kg
05	40000-50000Km	4.80mm	28.10 mm	1455mm	1457mm	19	36Kg
06	50000-60000Km	4.75mm	27.39 mm	1456mm	1457.2mm	17	33Kg
07	60000-70000Km	4.73mm	27.35 mm	1457mm	1459mm	16	31Kg
08	70000-80000Km	4.71mm	27.30 mm	1459mm	1461mm	15	30Kg
09	80000-90000Km	4.70mm	27.25 mm	1460mm	1464mm	14	29Kg
10	90000-100000Km	4.65mm	27.10 mm	1461mm	1467mm	13	27Kg

Annex C

RESEARCH DATA SHEET

Vehicle Model –Prime mover
Vehicle Make – TATA
Engine Model – TATA Cummins BJS 130-70
Estimated Hose Power – 180 at 2900 rpm
Venue -- Navy Camp Walisara

Date of commence 25th April 2012
Date of terminated 28th August 2012

Practical done by Chathura Sirimanna Type of Belt used – 8 PK Belt 1422

Table C1: Fan belt ware details for separate mileages

Belt Categories	Height(a)	Width (w)	Length Normal	Length Tensioned	Angle	Force
Belt I (New)	5.03 mm	28.46 mm	1450 mm	1453 mm	30	39 Kg
Belt II (15000Km)	5.00 mm	27.88 mm	1452 mm	1456 mm	20	36 Kg
Belt III (50000Km)	4.80 mm	27.40 mm	1455 mm	1459 mm	17	34 Kg
Belt VI (100000Km)	4.56 mm	27.10 mm	1457 mm	1463 mm	14	32 Kg

Table C2: Fan belt ware details for separate mileages

Difference	Height	Width	Length Normal	Length Subjected T	Angle	Force	Distance
Belt I, Belt II	0.03 mm	0.58 mm	2 mm	3 mm	10	3 Kg	15000Km
Belt II, Belt III	0.2 mm	0.48 mm	3 mm	3 mm	3	2 Kg	35000Km
Belt III, Belt VI	0.24 mm	0.30 mm	2 mm	4 mm	3	2 Kg	50000Km

Table C3: Fan belt ware details for separate mileages

Belt Category	Height(a)	Width (w)	Length Normal	Length Tensioned	Angle	Force
Belt I (New)	5.30mm	28.46 mm	1450 mm	1452mm	30	40 Kg
Belt II (15000Km)	5.10mm	28.40 mm	1452 mm	1453mm	25	38 Kg
Belt III (50000Km)	4.75mm	27.35 mm	1456mm	1457.2mm	17	33 Kg
Belt VI (100000Km)	4.65mm	27.10 mm	1461mm	1467mm	13	27 Kg

Table C4: Fan belt ware details for separate mileages

Difference	Height	Width	Length Normal	Length	Angle	Force	Distance Km
Belt I, Belt II	0.2 mm	0.06 mm	2 mm	1 mm	10	2 Kg	15000
Belt II, Belt III	0.35 mm	1.05 mm	4 mm	4.2 mm	3	5 Kg	35000
Belt III, Belt VI	0.1 mm	0.25 mm	5 mm	10.2 mm	3	6 Kg	50000

Annex D

RESEARCH DATA SHEET

Vehicle Model –Prime mover
Vehicle Make – TATA
Engine Model – TATA Cummins BJS 130-70
Estimated Hose Power – 180 at 2900 rpm
Venue - Navy Camp Walisara

Date of commence 25th April 2012
Date of terminated 28th August 2012

Practical done by Chathura Sirimanna

Type of Belt used – 8 PK Belt 1422

The practical values are taken from under mentioned vehicle to check the behavior of voltage with the temperature.

Engine details

Vehicle Make – TATA

Vehicle Type - Prime Mover

Vehicle Model – LPS 4018C/32

Engine Model – TATA Cummins BJS 130-70

Estimated Hose Power – 180 at 2900 rpm

Firing 1 5 3 6 4 2

Valve Lash – 0.25 In, 0.51ext

Table D1: Voltage variation responding to the engine cooling temperature

Temperature sensor voltage (when the supply voltage is given)	**Engine Temperature (practical value)**
4.00 V	60^0 C
3.41V	70^0 C
3.40V	80^0 C
3.39V	90^0 C

Annex E

The research details of loading weight

Table E1: The details of the heavy trucks taken for the research

Model	Type	Maximum loading capacity	Number of vehicle taken for the research
Tata open truck	LPT 1613	11 Ton	20
Tata ten wheel open truck	LPT 2516	19 Ton	05
Tata canopy truck	LPT 909	06 Ton	05

The data sheet of the research done for above 30 no's of vehicles for the 12 months of time period is as follows. The process started on January 2012 and the data collected till December 2012

RESEARCH DATA SHEET

Vehicle Model – Open Truck Date of commence January 2012
Vehicle Make – TATA 1613 LPT Date of terminated December 2012
Engine Model
Estimated Hose Power – 160 Hp
Venue Navy Camp Walisara
Recommended load – 11 Ton
Practical done by Chathura Sirimanna

Table E2: The weight loading to the vehicle type LPT 1613 during year 2012

Sr No	Naha Number	Type of vehicle	Jan	Feb	March	April	May	Jun	Jul	Aug	Sept	Oct	Nov	Dec
1	9224	LPT1613	12	07	04	04	07	09	13	07	08	11	10	11
2	9235	LPT1613	10	05	09	13	08	09	11	06	05	11	09	12
3	9236	LPT1613	10	08	08	09	06	04	09	12	09	11	09	12
4	9238	LPT1613	11	07	07	05	13	09	08	05	05	10	08	11
5	9239	LPT1613	08	08	07	07	07	11	13	09	07	09	12	11
6	9240	LPT1613	07	14	06	06	06	13	10	11	12			
7	9241	LPT1613	11	09	13	11	07	03	03	09	07	13	09	11
8	9242	LPT1613	12	07	10	08	13	03	10	07	10	08	10	09
9	9243	LPT1613	10	09	13	10	12	08	06	11	09			11
10	9245	LPT1613	07	10	12	08	06	06	07	09	13	08	08	07
11	9246	LPT1613	10	10		07	07	06	09	09	11	12	09	07
12	9249	LPT1613	11		08				09	10	11	05	09	13
13	9250	LPT1613	09	09	10	07	12	10	13		07	10	10	11
14	9254	LPT1613	09	11	01	11	10	12	08	10	09	13	06	10
15	9237	LPT1613	09	13	09	10	11	10	12	10	11	06	06	13
16	9263	LPT1613	09	08		05	05	07	09	10	08	09	07	07
17	9264	LPT1613			04	07	09	10	09	13	09	14	09	11
18	9265	LPT1613		10	04	08	04	10	06	13	03	11	05	12
19	9255	LPT1613	09	05	09	12	09	10	13	07	08		10	11
20	9251	LPT1613	11	05	10	05	04	08	10	05	05	11		

RESEARCH DATA SHEET

Vehicle Model – Open Truck
Vehicle Make – TATA 2516 LPT
2012Venue Navy Camp Walisara
Recommended load – 19 Ton
Practical done by Chathura Sirimanna

Date of commence January 2012
Date of terminated December

Table E3: The weight loading to the vehicle type LPT 2516 during year 2012

Sr No	Naha Number	Type of vehicle	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sept	Oct	Nov	Dec
1	9293	LPT2516	09	17	19	20	19	18	16	17	18	19	20	21
2	9294	LPT2516			17	19	19	21	16	19	15	17	20	19
3	9295	LPT2516			19	20	20	19	20	21	19	15	20	19
4	9296	LPT2516			18	19	19	20	21	20	09	17	17	20
5	9298	LPT2516		19	15	10	21	07	20	19		10	10	19

RESEARCH DATA SHEET

Vehicle Model – Canopy Truck
Vehicle Make – TATA 909 LPT
Venue Navy Camp Walisara
Recommended load – 06 Ton
Practical done by Chathura Sirimanna

Date of commence January 2012
Date of terminated December 2012

Table E4: The weight loading to the vehicle type LPT 909 during year 2012

Sr No	Naha No	Type	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
1	9277	LPT 909	07	06	04	04	06	03	06	05	06	04	03	07
2	9283	LPT 909	05	05	04	06		03	07	05	04	07	04	04
3	9289	LPT 909	06	03	04	03	04	04	06	05				
4	9291	LPT 909	06				07	05	08	08	06	05	04	06
5	9292	LPT 909		07	07	06	02	05	03	06	04	03	07	06

RESEARCH DATA SHEET

Vehicle Model –Ten wheel Truck
Venue Navy Camp Walisara
Practical done by Chathura Sirimanna

Date of commence January 2012
Date of terminated December 2012

Table E5: The defect which occurred in the vehicle due to over loading in year 2012

Sr No	Naha Number	Type of vehicle	Hub oil seal defect	Leaf spring defect	Other defects
1	9224	LPT1613	01		s/r ball joint
2	9235	LPT1613	02		
3	9236	LPT1613	01		
4	9238	LPT1613	03		p/s pump r/kit
5	9239	LPT1613	02		
6	9240	LPT1613	03	Main leaf	
7	9241	LPT1613	02		
8	9242	LPT1613	01	Main & helper	
9	9243	LPT1613			
10	9245	LPT1613	01		
11	9246	LPT1613	02		
12	9249	LPT1613			
13	9250	LPT1613	01		
14	9254	LPT1613	02		
15	9237	LPT1613	03		
16	9263	LPT1613	01		
17	9264	LPT1613	02		
18	9265	LPT1613	02	5th leaf	
19	9255	LPT1613	01	2nd leaf	
20	9251	LPT1613			
21	9293	LPT2516	03		
22	9294	LPT2516	02		
23	9295	LPT2516	01	Main leaf	
24	9296	LPT2516	02		p/s pump r/kit
25	9298	LPT2516	02		
26	9277	LPT 909	01		
27	9283	LPT 909	02		
28	9289	LPT 909			
29	9291	LPT 909			
30	9292	LPT 909	02		

Annex F

The following practical were carried out to check engine oil condition. For the experiment, 30 heavy trucks of the same type have been taken and they were driven nearly the same distance of mileage. Further, the data was taken for the vehicles running within the city limits and out station separately.

RESEARCH DATA SHEET

Model –Prime mover,crane,ten wheel truck
Venue - Navy Camp Walisara
Practical done by Chathura Sirimanna

Date of commence 25th August 2012
Date of terminated 11th December 2012

Table F1: Research data of vehicle running distance and time duration taken

Sr No	Naha No	Type	From	To	Distance (Km)	Running Hours	Average mileage
01	5363	Prime mover	Walisara	Trincomalee and back	540	17Hrs & 15 Min	31.3km/hrs
02	5254	Crane truck	Walisara	Panama and back	735	21 Hrs & 25 Min	34.32km/hrs
03	5360	Prime mover	Walisara	Trincomalee and back	603	21 Hrs & 55 Min	27.51 km/hrs
04	5754	Crane truck	Walisara	Trincomalee and back	894	32 Hrs & 45 Min	27.3 km/hrs
05	9293	Ten wheel truck	Walisara	Ampara and back	736	22 Hrs & 55 Min	32.11 km/hrs
06	9296	Ten wheel truck	Walisara	Trincomalee and back	502	16 Hrs & 40 Min	30.12 km/hrs
07	9298	Ten wheel truck	Walisara	Trincomalee and back	505	15 Hrs & 35 Min	32.40 km/hrs
08	5364	Prime mover	Walisara	Ampara and back	744	28 Hrs & 15 Min	29.46 km/hrs
09	9224	Open truck	Walisara	Panama and back	735	21 Hrs & 25 Min	34.32 km/hrs
10	9250	Open truck	Walisara	Trincomalee and back	575	18 Hrs & 05 Min	31.8 km/hrs

RESEARCH DATA SHEET

Vehicle Model – Truck (Driven within Colombo area)
Vehicle Make – TATA 1613 LPT
Engine oil inspected place – Vehicle service station, Navy camp, Walisara
Date of commence January 2012
Date of terminated December 2012

Table F2: Research data of vehicle driven mileage inside Colombo area

Sr No	**Naha No**	**Jan**	**Feb**	**Mar**	**Apr**	**May**	**Jun**	**Jly**	**Agst**	**Sep**	**Oct**	**Nov**	**Dec**	**Total**
1	9235	2092	1878	2429	2143	2189	2963	1938	2071	1891	2092	1876	2730	26272

2	9238	2200	2036	2964	1959	3019	2092	1834	1980	2973	1201	1367	2164	25789
3	9239	3718	1902	2089	1893	2836	1962	2846	2546	1189	1839	1054	2173	26047
4	9249	1182	2089	2199	2106	2526	2198	1076	2108	2073	2146	2127	2497	24327
5	9254	2091	2983	2045	1986	2012	1936	2075	1534	2073	1947	2017	2831	25530
6	9277	1783	2964	2910	1710	2195	2358	1937	2073	1965	2984	2197	1073	26149
7	9283	1673	978	1783	2917	3217	1456	1257	2916	1382	2691	2472	2018	24760
8	9289	2326	819	2018	1879	1582	2819	2639	1973	2910	2194	1036	2516	24711
9	9291	2235	1892	2891	1928	2017	2891	1836	2017	1290	2103	1893	1783	24776
10	9292	2617	1738	1998	2344	1891	1920	1938	2718	1893	2718	2813	1928	26516
11	9293	2182	2891	2891	1892	1783	1854	2190	2978	1783	1873	1639	2017	25973
12	9294	2091	1896	2210	1837	2974	1973	2938	1839	2281	910	1891	2311	25151
13	9296	1267	2893	1820	2138	2183	1953	2351	991	2913	2192	2181	2132	25014
14	9255	2190	2937	1222	2109	2167	1889	2571	2037	2193	1738	2910	1732	25695
15	9251	819	1820	2718	1937	2914	2588	1739	1989	2130	2937	1939	2316	25846

Table F3: Research data of monthly service intervals for heavy vehicles

Sr No	Naha No	Jan	Feb	Mar	Apr	May	Jun	Jly	Aug	Sep	Oct	Nov	Dec	Total	Svc Int	condition of oil
1	9235		03				28				31			26272	9114	Unserviceable
2	9238	17			25				21					25789	9217	Unserviceable
3	9239					23						14		26047	9263	Unserviceable
4	9249											06		24327	8893	Unserviceable
5	9254			27				07					20	25530	9291	Unserviceable
6	9277	20				21					14			26149	9204	Unserviceable
7	9283	11					15						11	24760	8976	Oil change immediately
8	9289			02										24711	9125	Oil change immediately
9	9291		17				28							24776	9352	Unserviceable
10	9292								25				03	26516	8940	Unserviceable
11	9293			12			06			13				25973	9352	Unserviceable
12	9294						20		20			01		25151	9354	Oil change immediately
13	9296							04		10			11	25014	8798	Oil change in near future
14	9255				09					13		16		25695	9310	Unserviceable
15	9251	26			30									25846	9426	Unserviceable

RESEARCH DATA SHEET

Vehicle Model – Truck (Driven long distance)
Vehicle Make – TATA 1613 LPT
Engine oil inspected place – Vehicle service station, Navy camp, Walisara
Date of commence January 2012
Date of terminated December 2012

Table F4: Research data of vehicle driven mileage outside Colombo area

Sr No	Naha No	Jan	Feb	Mar	Apr	May	Jun	Jly	Agst	Sep	Oct	Nov	Dec	Total
1	9224	2176	1529	2409	1993	2342	2022	1978	1520	1736	1618	2534	2609	24466
2	9236	1618	1981	2089	2889	1539	2780	1986	2014	2176	1548	2449	2308	25377
3	9240	2160	2242	2189	1986	2751	2134	3002	1548	2367	1652	2094	1159	25284
4	9241	2317	1962	2190	2019	2557	1406	1253	2300	2983	2814	1564	2631	25996
5	9242	2099	3245	1441	2614	1618	2823	2413	1253	1537	2924	1791	2345	26103
6	9243	2669	2691	2577	2639	2288	2078	2521	1768	2198	1287	1782	1250	25748
7	9245	2154	2679	2763	2022	1907	1938	2891	1816	1598	2743	2711	1256	26478
8	9246	2614	1986	1637	2300	1459	2006	2066	2677	2200	2826	1000	2970	25741
9	9250	2357	1571	1388	2281	2645	1791	2461	1582	3201	1476	2563	2109	25425
10	9237	2009	1411	3641	1299	1314	2244	2625	1894	2479	1908	2882	2547	26253
11	9263	1342	2097	2052	1415	2264	2774	1689	2330	1199	2281	2397	2846	24686
12	9264	2091	3290	1815	2313	1885	2650	1342	1761	2215	2575	1519	2924	26380
13	9265	3718	1320	1298	2854	1735	2283	2748	1621	1848	1198	2576	2791	25980
14	9295	2134	3021	1575	907	2647	2897	2282	1226	2052	2675	1999	1661	25076
15	9298	2183	2301	1122	1573	2108	2578	1733	3369	2345	2188	3210	1393	26103

Table F5: Research data of monthly service intervals for heavy vehicles

Sr No	Naha No	Jan	Feb	Mar	Apr	May	Jun	Jly	Ags	Sep	Oct	Nov	Dec	Total	Svc Int	condi oil
1	9224	02				12								24466	9193	Servic
2	9236	06					20					21		25377	9281	Servic
3	9240		18				14							25284	8902	Servic
4	9241		15		20							01		25996	9120	Oil ch near fu
5	9242		09		04				16					26103	9381	Oil ch near fu
6	9243			05		21								25748	9003	Servic
7	9245		15			08			20					26478	8968	Oil ch near fu
8	9246		01			02			10					25741	9372	Oil immed
9	9250			09							05			25325	9182	Unser
10	9237				18		25			07				26253	8452	Oil ch near fu
11	9263						27				23		28	24686	9172	Unser
12	9264			02			15					06		26380	9006	Oil immed
13	9265							11		27		26		25980	9456	Oil immed
14	9295				20		05				17			25076	9219	Oil immed
15	9298								03	19			06	24466	9193	Oil ch near fu

The engine oil (burned) checking procedure (spot test)

The checking of engine oil condition was done based on the carbon percentage containing in the oil. Visual inspection was carried out according to the guidance provided by the 'MTU' marine engine workshop manual.

The spot test procedure

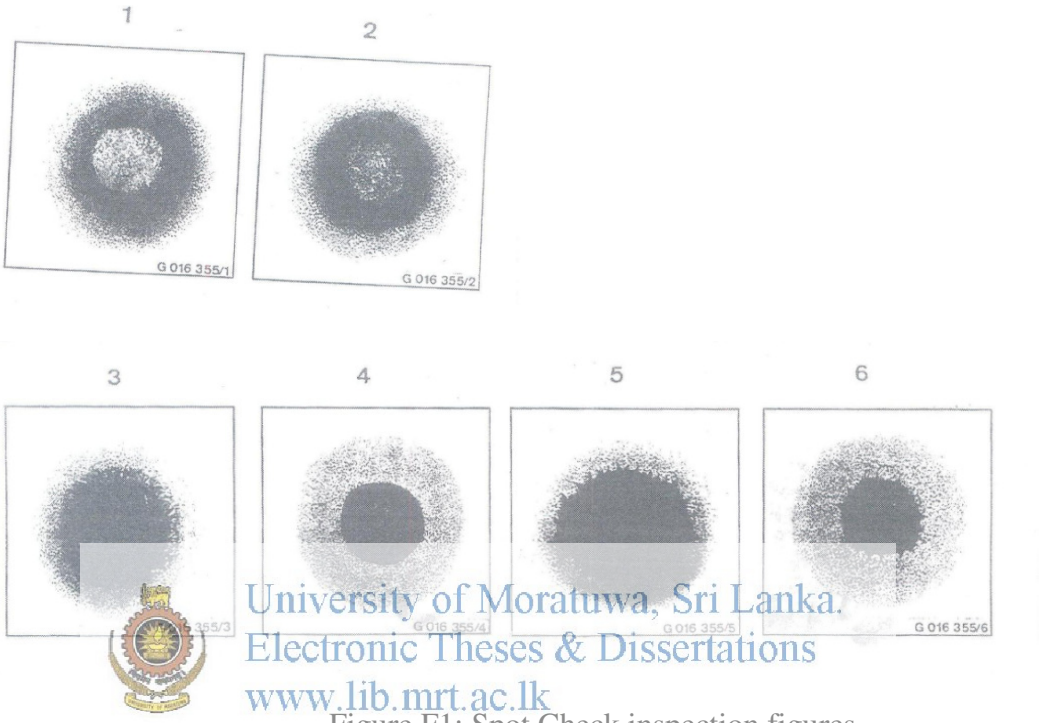


Figure F1: Spot Check inspection figures

Spot check figure 1 and 2 indicates that the oil is serviceable

Spot check figure 3 indicates that an oil change will be required in the near future

Spot check figure 4, 5 and 6 indicates that an oil change is required immediately

Annex G

Installation of the modified shock absorber to the vehicle

The shock absorber has been modified without disturbing its original structure. Most of the heavy vehicles fitted only front shock absorber and no one for rear side. Hence it needed to modify/ fabricate holding brackets as necessary.

Figure G1: Disassembling of shock absorber

Figures G2: Modification as distance measuring sensor

Figure G3: Fitted Infra red sensor

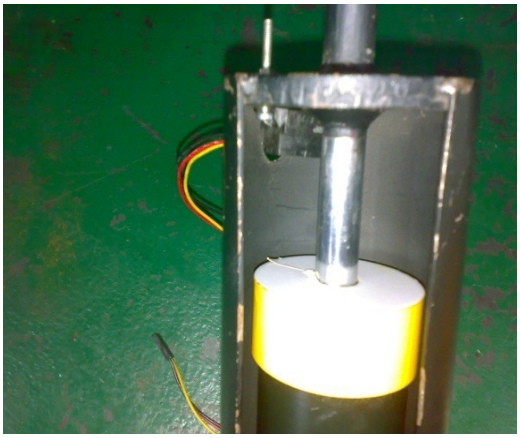


Figure G4: Preparing smoother surface flat form

Figure G5: Assembling to rear axle

Figure G6: Wiring the system

Figure G7: Modified rear low bracket

Figure G8: Modified rear upper bracket

The existing shock absorber was departed and welded with a mild steel circle plate of 6 mm thickness to the outer tube.

a. It was fitted to the distance sensor to the space as shown in figure G4 and the wire connection was taken from the small hole.

b. An aluminum cup (diameter 56mm) was prepared and the surface was smoothened before permanently fitting it to the top of the inner tube without disturbing its main structure and the hydraulic function.

c. A photo paper (white) was pasted on the smoothen surface of the aluminum cup. The mild steel plate was carefully fitted to the outer tube of the shock absorber by threads which can be easily dismantled if needed.

d. Other components such as rubber bushes were assembled to the shock absorber.

Figure G9: TATA 2516 LPT Truck

Calibrating process as follows:

a. The vehicle (TATA ten wheel trucks) was taken to a flat surface

b. Modified shock absorber (Load Sensing Unit) was mounted on each axel

c. The load sensing unit(LSU) was connected to the microcontroller, 18F 4520

d. With the known weight , observation were made for each configuration separately

e. All the values taken from micro controller was listed for further evaluation

f. The weight distribution for front axel 5700Kg, middle axel, rear axle 6650 Kg

g. Obtaining the sensor (LSU) reading by loading known weights

Figure G10: loading weight to configuration A

Figure G11: loading weight to configuration B

Figure G12, G13: Obtaining values from LSU by loading known weights

Figure G14, G15: Loading weight for uniform configuration

The details (digital value) taken by loading known weight to the vehicle

Configuration (A) - Loading weight near to driver cabin

Table G1: Digital values correspondent to load

Known weight (Ton)	Front LSU	Middle LSU	Rear LSU
1	159	128	103
2	163	128	103
3	167	131	103
4	171	131	103
5	176	131	103
6	184	143	103
7	188	143	103
8	192	147	103
9	196	149	103
10	201	153	103

Configuration (B) - Loading weight middle of rear deck

Table G2: Digital values correspondent to load

Known weight (Ton)	Front LSU	Middle LSU	Rear LSU
1	167	131	103
2	167	135	103
3	167	139	108
4	167	143	112
5	167	147	112
6	167	153	112
7	177	157	112
8	177	161	112
9	177	165	116
10	177	169	120

Configuration (C) - Loading weight to rear of the deck

Table G3: Digital values correspondent to load

Known weight (Ton)	**Front LSU**	**Middle LSU**	**Rear LSU**
1	156	129	103
2	145	131	106
3	135	133	113
4	131	135	120
5	124	137	127
6	112	139	134
7	112	141	141
8	108	144	145
9	108	147	149
10	102	149	153

Configuration (D) - Loading weight to the deck uniformly

Table G4: Digital values correspondent to load

Known weight (Ton)	Front LSU	Middle LSU	Rear LSU
1	159	120	112
2	193	124	113
3	167	124	115
4	171	126	117
5	175	128	119
6	179	130	121
7	179	132	125
8	180	134	131
9	180	136	137
10	181	138	143

Annex H

Pictures of modification of fan belt condition indication unit

Figure H1: Accessories for fan belt condition indicator unit

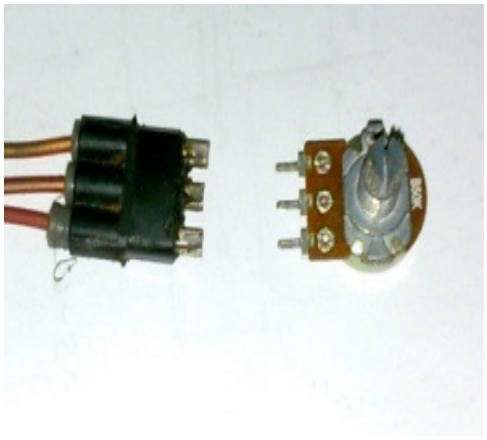


Figure H2: POT and connector

Figure H3: POT connected to modify flat form

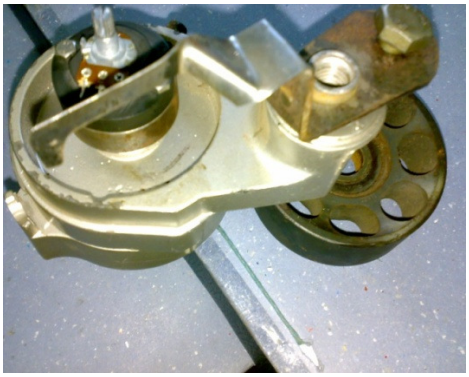


Figure H4: Modified indicator unit fitted to fan belt adjuster

Annex I

programming language for belt condition indicator design circuit

```
void set_ang_low()
{
    if(read_eeprom(set_low_flg)!=1){
    printf(LCD_PUTC, "\fSet LOW End:");
    }

    if(read_eeprom(set_low_flg)==1){
    printf(LCD_PUTC, "\fEdit LOW End:");
    }
    set_adc_channel(2);
    delay_us(10);
    temp1=read_adc();
    lcd_gotoxy(0,2);
    printf(LCD_PUTC, "U:%03u",temp1);
    if(read==0){if(read_eeprom(set_low_flg)==1)temp=read_eeprom(low_ang);read=1;}
    lcd_gotoxy(8,2);

    printf(LCD_PUTC, "Angle:%02u",temp);
    if(sub_state_1>10&&sub_state_2>10){
    if(input_state(inc)==0){delay_ms(20);lighton();temp+=1;if(temp>50)temp=0;}//increment angle
    if(input_state(dec)==0){delay_ms(20);lighton();temp-=1;if(temp==-1)temp=50;}//decrement angle
    if(input_state(cl)==0){delay_ms(20);lighton();temp=0;}//decrement angle

      if(input_state(set)==0)
{
        delay_ms(10);
        lighton();
        printf(LCD_PUTC, "\fSaving Data...");
        delay_ms(750);
        write_eeprom(low_ang,temp);
        write_eeprom(low_adc,temp1);
        write_eeprom(set_low_flg,1);
        printf(LCD_PUTC, "\nok...");
        delay_ms(500);
      }//save data
      exit_menu();
    }

}//ang low set ends


void set_ang_high(){
    if(read_eeprom(set_low_flg)!=1){
    printf(LCD_PUTC, "\fError!!");
```

```
    printf(LCD_PUTC, "\nSet Lo first!");
    delay_ms(750);
    lighton();
    sub_state_2=11;
}

    if(read_eeprom(set_low_flg)==1&&read_eeprom(set_high_flg)==1){
    printf(LCD_PUTC, "\fEdit Hi End:");
    }

    if(read_eeprom(set_low_flg)==1&&read_eeprom(set_high_flg)!=1){
    printf(LCD_PUTC, "\fSet Hi End:");
    }
    if(read_eeprom(set_low_flg)==1){
    set_adc_channel(2);
    delay_us(10);
    temp1=read_adc();
    lcd_gotoxy(0,2);
    printf(LCD_PUTC, "U:%03u",temp1);
    if(read==0){if(read_eeprom(set_high_flg)==1)temp=read_eeprom(high_ang);read=1;}
    lcd_gotoxy(8,2);
    printf(LCD_PUTC, "Angle:%02u",temp);
    if(temp<=read_eeprom(low_ang)){lcd_gotoxy(14,1);printf(LCD_PUTC,">H");}
    if(temp1<=read_eeprom(low_adc)){lcd_gotoxy(12,1);printf(LCD_PUTC,">U");}

if(temp>read_eeprom(low_ang)&&temp1>read_eeprom(low_adc)){lcd_gotoxy(13,1);printf(LCD_PUTC,"
Ok!");}

     if(sub_state_1>10&&sub_state_2>10){
    if(input_state(inc)==0){delay_ms(20);lighton();temp+=1;if(temp>50)temp=0;}//increment angle
    if(input_state(dec)==0){delay_ms(20);lighton();temp-=1;if(temp==-1)temp=50;}//decrement angle
    if(input_state(clr)==0){delay_ms(20);lighton();temp=0;}//decrement angle

    if(input_state(set)==0){
        delay_ms(10);
        lighton();
        printf(LCD_PUTC, "\fSaving Data...");
        delay_ms(750);
        write_eeprom(high_ang,temp);
        write_eeprom(high_adc,temp1);
        write_eeprom(set_high_flg,1);
        printf(LCD_PUTC, "\nok...");
        delay_ms(500);
      }//save data

      }//if sub1>10 and sub2>10
    }
    exit_menu();
```

```
}//ang high set ends

float Get_angle(float input){

float adc_unit,range;
float angle_const,angle;
float x,y,p,q;
x=read_eeprom(high_adc);
y=read_eeprom(low_adc);
p=read_eeprom(high_ang);
q=read_eeprom(low_ang);
adc_unit=x-y;
range=p-q;
angle_const=(range/adc_unit);
angle=((input-y)*angle_const)+q;
return angle;
}


void ang_alm_setup()
{
    if(read_eeprom(ang_alm_flg)!=1)printf(LCD_PUTC, "\fSet Angl: Limit");
   if(read_eeprom(ang_alm_flg)==1)printf(LCD_PUTC, "\fEdit Angl: Limit");

    if(read==0)if(read_eeprom(ang_alm_flg)==1)temp=read_eeprom(ang_alm);read=1;}
    printf(LCD_PUTC, "\nAngl: Limit =%02u",temp);

    if(sub_state_1>10){
    if(input_state(inc)==0){delay_ms(20);lighton();temp+=1;if(temp>50)temp=0;}//increment
temperature
    if(input_state(dec)==0){delay_ms(20);lighton();temp-=1;if(temp==-1)temp=50;}//decrement
temperature
    if(input_state(clr)==0){delay_ms(20);lighton();temp=0;}
      if(input_state(set)==0){
        delay_ms(10);
        lighton();
        printf(LCD_PUTC, "\fSaving Data...");
        delay_ms(750);
        write_eeprom(ang_alm,temp);
        write_eeprom(ang_alm_flg,1);
        printf(LCD_PUTC, "\nok...");
        delay_ms(500);
      }//save data

    }//if sub1>10 and sub2>10

}//end
```

Annex J

Programming language for engine cooling temperature indicator

```
//   adc initialize   inside PIC
  setup_adc(ADC_CLOCK_INTERNAL||VSS_VDD);
  setup_adc_ports(AN0_TO_AN2);
void dis_temp2(){
set_adc_channel(0);
delay_us(10);
ADC+=read_adc();
ADC/=2;
loop++;
if(loop>20){temperature=ADC*0.588;loop=0;} //
printf(LCD_PUTC,"\fTemp:=%4.2f",temperature);
if(temperature==90)printf(LCD_PUTC,"\fOver Temp: Err!");
// indicate over temperature
}
```

Annex K

Programming language for load sensing unit

RX data receive interrupt

```
#int_RDA
void  RDA_isr(void)
{
output_high(PIN_D3);
char c;
int rxptr=0;
disable_interrupts(INT_RDA);
/* Read the receive buffer till at least one or more character can be read */
  c = getc();
  switch (c)
  {  case '$':
       {
              RxDataReady=0;
              RxDataPtr = 0;
       break;
       }
 case '#': {
       RxDataReady = 1;
       break;
      }
}
    Rxdata[RxDataPtr++ & 0x7F] = c;
   if (RxDataReady)
 {
  for(rxptr=0;rxptr<11;rxptr++)
    {
       DataCopy[rxptr]=Rxdata[rxptr];
     }
      Datacopied=1;
 }  output_low(PIN_D3);
enable_interrupts(INT_RDA);
```


Load configuration setup

```
}void setup_LOAD_configuration()
{
setconfig=1;
printf(LCD_PUTC,"\f A | B | C | D ");
if(sub_state_1>10&&sub_state_2<10){if(input_state(inc)==0){
delay_ms(50);
```

```
sub_state_2++;
if(sub_state_2==5){
sub_state_2=1;}
lighton();}}
if (sub_state_2==1)
{printf(LCD_PUTC,"\n Front");
delay_ms(200) ;
printf(LCD_PUTC,"\f A*| B | C | D ");
delay_ms(200);
configuration=0;
printf(LCD_PUTC,"\n Front");
}
if (sub_state_2==2)
{
printf(LCD_PUTC,"\n Middle ");
delay_ms(200);
printf(LCD_PUTC,"\f A | B*| C | D ");
delay_ms(200);
configuration=1;
printf(LCD_PUTC,"\n Middle ");
}
if (sub_state_2==3)
{
printf(LCD_PUTC,"\n Rear ");
delay_ms(200);
printf(LCD_PUTC,"\f A | B | C*| D ");
delay_ms(200) ;
configuration =2;
printf(LCD_PUTC,"\n Rear ");
}
if (sub_state_2==4)
{
printf(LCD_PUTC,"\n uniform");
delay_ms(200);
printf(LCD_PUTC,"\f A | B | C | D*");
delay_ms(200) ;
configuration=3;
printf(LCD_PUTC,"\n uniform");
}
}
void load_distribution(){
float total_load=0;
int previousval1;
int previousval2;
int previousval3;
if(DataCopied==1 && RxDataReady==1)
```

```
{
if(DataCopy[1]=='S' && DataCopy[2]=='F' && DataCopy[3]=='R' && DataCopy[4]=='T' )
{
DataCopied=0;
val1=(DataCopy[5]-48)*100+(DataCopy[6]-48)*10+(DataCopy[7]-48);
previousval1=val1;
//printf(LCD_PUTC,"\n %u  | %u | %u ",val1,val2,val3
}
else val1=previousval1;
 if(DataCopy[1]=='S' && DataCopy[2]=='M' && DataCopy[3]=='I' && DataCopy[4]=='D' )
{
DataCopied=0;
val2=(DataCopy[5]-48)*100+(DataCopy[6]-48)*10+(DataCopy[7]-48);
//printf(LCD_PUTC,"\n%u  | %u | %u ",val1,val2,val3 );
previousval2=val2;
}
else val2=previousval2;
 if(DataCopy[1]=='S' && DataCopy[2]=='R' && DataCopy[3]=='E' && DataCopy[4]=='R' )
{
DataCopied=0;
val3=(DataCopy[5]-48)*100+(DataCopy[6]-48)*10+(DataCopy[7]-48);
previousval3=val3;
}
}
if (configuration==0) total_load=((0.206)*val1 - 31.68);
if (configuration==1) total_load=((0.232)*val2 - 29.30);
if (configuration==2) total_load=((0.122)*val3 - 27.13);
if (configuration==3) total_load=((0.5)*val2 - 59.0);
if(total_load<=0)total_load=loadtemp;
else loadtemp=total_load;
if (setconfig !=1)printf(LCD_PUTC,"\f configuration error");


if (configuration==0) printf(LCD_PUTC,"\f front config");
if (configuration==1) printf(LCD_PUTC,"\f middle config");
if (configuration==2) printf(LCD_PUTC,"\f rear config");
if (configuration==3) printf(LCD_PUTC,"\f uniform config");
printf(LCD_PUTC,"\fFront| Mid| Rear");
printf(LCD_PUTC,"\n%2.1gT |%2.1gT| %2.1gT",total_load*.3,total_load*.35,total_load*.35);
delay_ms(20);
}
```


Set next function as load distribution indicator

```
void load_distribution()
{
float total_load=0;
int previousval1;
int previousval2;
int previousval3;
if(DataCopied==1 && RxDataReady==1)
{
if(DataCopy[1]=='S' && DataCopy[2]=='F' && DataCopy[3]=='R' && DataCopy[4]=='T' )
{
DataCopied=0;
val1=(DataCopy[5]-48)*100+(DataCopy[6]-48)*10+(DataCopy[7]-48);
previousval1=val1;
//printf(LCD_PUTC,"\n %u  | %u | %u ",val1,val2,val3 );
}
else val1=previousval1;
 if(DataCopy[1]=='S' && DataCopy[2]=='M' && DataCopy[3]=='I' && DataCopy[4]=='D' )
{
DataCopied=0;
val2=(DataCopy[5]-48)*100+(DataCopy[6]-48)*10+(DataCopy[7]-48);
//printf(LCD_PUTC,"\n%u  | %u | %u ",val1,val2,val3 );
previousval2=val2;
}
else val2=previousval2;
 if(DataCopy[1]=='S' && DataCopy[2]=='R' && DataCopy[3]=='E' && DataCopy[4]=='R' )
{
DataCopied=0;
val3=(DataCopy[5]-48)*100+(DataCopy[6]-48)*10+(DataCopy[7]-48);
previousval3=val3;
}
}
if (configuration==0) total_load=((0.206)*val1 - 31.68);
if (configuration==1) total_load=((0.232)*val2 - 29.30);
if (configuration==2) total_load=((0.122)*val3 - 27.13);
if (configuration==3) total_load=((0.5)*val2 - 59.0);
if(total_load<=0)total_load=loadtemp;
else loadtemp=total_load;
if (setconfig !=1)printf(LCD_PUTC,"\f configuration error");
if (configuration==0) printf(LCD_PUTC,"\f front config");
if (configuration==1) printf(LCD_PUTC,"\f middle config");
if (configuration==2) printf(LCD_PUTC,"\f rear config");
if (configuration==3) printf(LCD_PUTC,"\f uniform config");
printf(LCD_PUTC,"\fFront| Mid| Rear");
printf(LCD_PUTC,"\n%2.1gT |%2.1gT| %2.1gT",total_load*.3,total_load*.35,total_load*.35);
delay_ms(20);
}
```

Annex L

Programming language for engine running hours counter

Used code for the interrupt is as follows.

```
#int_TIMER2
void  TIMER2_isr(void)
{
//output_toggle(pin_d1);
if(!input(pin_a3))
{
ticks++;
}
```

Ticks indicated that the total number of 10ms time. At main loop there are two functions, and those are,

1. calc_engine_hrs(); // calculating hours
2. dis_engine_hrs(); // display engine hours

1. Function calc_engine_hrs();

```
void calc_engine_hrs()
{
 if(ticks>18000)
 {
  ticks=0;
  toggle=~toggle;
  decamin++;
  if (decamin>9)
  {
  decamin=0;
  half_hour++;
  if((half_hour%2))
  {
    engine_hrs++;
     beeptwice();
  }
  }
  if(engine_hrs>999999)engine_hrs=0;
   Save_engine_hrs(engine_hrs);
 }
}
```


2. Function dis_engine_hrs();

```
void dis_engine_hrs()
{

 printf(LCD_PUTC,"\fEngine work hour:");
unsigned int m1,m2,m3,m4,m5,m6;
```

```
//m1 = read_eeprom(11);
//m2 = read_eeprom(12);
//m3 = read_eeprom(13);
//m4 = read_eeprom(14);
//m5 = read_eeprom(15);
//m6 = read_eeprom(16);
m1 = read_eeprom(ml100000);   // reading from EEPROM
m2 = read_eeprom(ml10000);
m3 = read_eeprom(ml1000);
m4 = read_eeprom(ml100);
m5 = read_eeprom(ml10);
m6 = read_eeprom(ml1);
//m1=0,m2=0,m3=0,m4=0,m5=0,m6=0;

lcd_gotoxy(1,2);                    //  write  to LCD
printf(LCD_PUTC,"%d",read_eeprom(11));
lcd_gotoxy(2,2);
printf(LCD_PUTC,"%d",read_eeprom(12));
lcd_gotoxy(3,2);
printf(LCD_PUTC,"%d",read_eeprom(13));
lcd_gotoxy(4,2);
printf(LCD_PUTC,"%d",read_eeprom(14));
lcd_gotoxy(5,2);
printf(LCD_PUTC,"%d",read_eeprom(15));
lcd_gotoxy(6,2);
printf(LCD_PUTC,"%d",read_eeprom(16));
lcd_gotoxy(7,2);
printf(LCD_PUTC," hrs");
}
```


At the beginning, the display (LCD) shows the current worked hours by loading data from EEPROM
(Electrical Erasable Programmable Read only Memory)
At the end, the PIC saves current working hours in EEPROM.

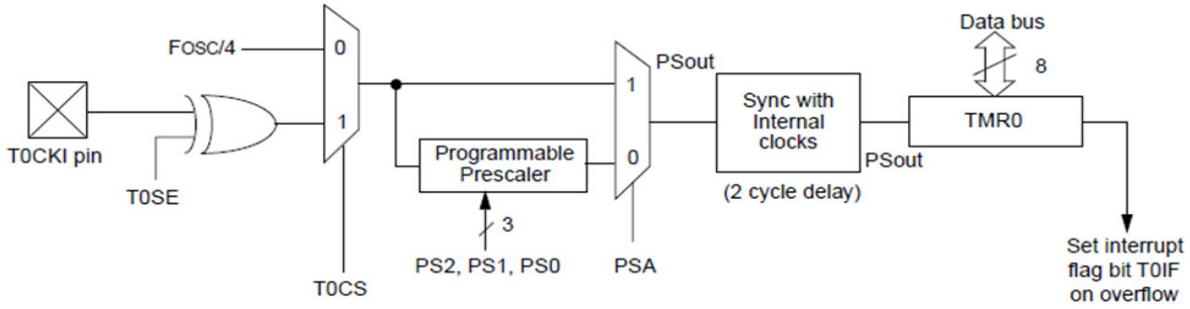


Figure L1: Timer 0 block diagram (Inside the PIC)

Note – T0CS, T0SE, PSA, PS2:PS0 (OPTION_REG<5:0>).
The prescaler is shared with watchdog timer

Annex M

Programming language for RS 485 Communication system

Front board data

```
#include <rs232.h>
int adc_value;
char DataCopy[10]={};
char RxData[15]={};
char Tx_data[13]={};
int1   RxDataReady=0;
int1   RxDataPtr = 0;
int1   Datacopied=1;
int get_ADC(void);
void set_data();
void transmit_data();
void main()
{
output_low(PIN_B5);
 // setup_comparator(NC_NC_NC_NC);// This device COMP currently not supported by the PICWizard
 // enable_interrupts(INT_TBE);
  //enable_interrupts(INT_RDA);
  //enable_interrupts(GLOBAL);
adc_value=get_ADC();
delay_ms(2000);
while(1)
{
adc_value=get_ADC();
delay_ms(100);
set_data();
delay_ms(100);
transmit_data();
delay_ms(4000);
}
}
int get_ADC(void)
{
 int value;
```

```
setup_adc_ports(sAN0|sAN1, VSS_VDD);
setup_adc(ADC_CLOCK_DIV_2);
set_adc_channel(0);
read_adc(ADC_START_ONLY);
int1 done = adc_done();
while(!done)
      {
      done = adc_done();
      }
        value = read_adc();
return value;
}
void transmit_data()
{
output_high(PIN_B2);
output_high(PIN_B5);
printf(Tx_data);
output_low(PIN_B5);
output_low(PIN_B2);
}
void set_data(){
int dec1;
int dec2;
int dec3;
dec1=adc_value/100;
dec2=(adc_value-100*dec1)/10;
dec3=(adc_value-100*dec1-10*dec2);
dec1=dec1+48;
dec2=dec2+48;
dec3=dec3+48;
Tx_data[0]='$';
Tx_data[1]='S';
Tx_data[2]='F';
Tx_data[3]='R';
Tx_data[4]='T';
Tx_data[5]= dec1;
Tx_data[6]= dec2;
Tx_data[7]= dec3;
Tx_data[10]= '#';
}
```

8.8.1 (B) Middle board data

```
#include <rs232.h>
int adc_value;
char DataCopy[10]={};
char RxData[15]={};
char Tx_data[13]={};
int1   RxDataReady=0;
int1   RxDataPtr = 0;
int1   Datacopied=1;
int get_ADC(void);
void set_data();
void transmit_data();
void main()
{
output_low(PIN_B5);
  // setup_comparator(NC_NC_NC_NC);// This device COMP currently not supported by the PICWizard
  // enable_interrupts(INT_TBE);
   //enable_interrupts(INT_RDA);
   //enable_interrupts(GLOBAL);
adc_value=get_ADC();
delay_ms(1000);
while(1)
{
adc_value=get_ADC();
delay_ms(100);
set_data();
delay_ms(100);
transmit_data();
delay_ms(4000);
}
}
int get_ADC(void)
{
 int value;
setup_adc_ports(sAN0|sAN1, VSS_VDD);
setup_adc(ADC_CLOCK_DIV_2);
set_adc_channel(0);
read_adc(ADC_START_ONLY);
int1 done = adc_done();
while(!done)
```

```
      {
   done = adc_done();
 }   value = read_adc();
return value;
} void transmit_data()
{
output_high(PIN_B2);
output_high(PIN_B5);
printf(Tx_data);

output_low(PIN_B5);
output_low(PIN_B2);
}
void set_data()
{
int dec1;
int dec2;
int dec3;
dec1=adc_value/100;
dec2=(adc_value-100*dec1)/10;
dec3=(adc_value-100*dec1-10*dec2);
dec1=dec1+48;
dec2=dec2+48;
dec3=dec3+48;
// set data packet
Tx_data[0]='$';
Tx_data[1]='S';
Tx_data[2]='M';
Tx_data[3]='I';
Tx_data[4]='D';
Tx_data[5]= dec1;
Tx_data[6]= dec2;
Tx_data[7]= dec3;
Tx_data[10]= '#';
}
```


8.8.1 (C) Rear board data

```
#include <rs232.h>
int adc_value;
char DataCopy[10]={};
char RxData[15]={};
char Tx_data[13]={};
int1   RxDataReady=0;
```

```
int1   RxDataPtr = 0;
int1   Datacopied=1;
int get_ADC(void);
void set_data();
void transmit_data();
void main()
{
output_low(PIN_B5);
  // setup_comparator(NC_NC_NC_NC);// This device COMP currently not supported by the PICWizard
  // enable_interrupts(INT_TBE);
   //enable_interrupts(INT_RDA);
   //enable_interrupts(GLOBAL);
adc_value=get_ADC();
while(1)
{
adc_value=get_ADC();
delay_ms(100);
set_data();
delay_ms(100);
transmit_data();
delay_ms(4000);
}
}
int get_ADC(void)
{
 int value;
setup_adc_ports(sAN0|sAN1, VSS_VDD);
setup_adc(ADC_CLOCK_DIV_2);
set_adc_channel(0);
read_adc(ADC_START_ONLY);
int1 done = adc_done();
while(!done)
       {
       done = adc_done();
       }
value = read_adc();
return value;
}
void transmit_data()
{output_high(PIN_B2);
output_high(PIN_B5);
printf(Tx_data);
output_low(PIN_B5);
```

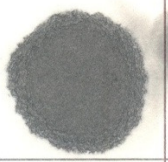
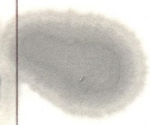
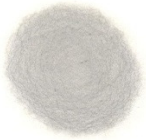
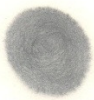
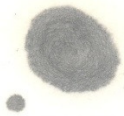
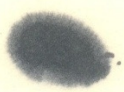
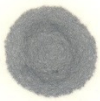
```
output_low(PIN_B2);
}void set_data()
{ int dec1;
int dec2;
int dec3;
dec1=adc_value/100;
dec2=(adc_value-100*dec1)/10;
dec3=(adc_value-100*dec1-10*dec2);
dec1=dec1+48;
dec2=dec2+48;
dec3=dec3+48;
// data packet set
Tx_data[0]='$';
Tx_data[1]='S';
Tx_data[2]='R';
Tx_data[3]='E';
Tx_data[4]='R';
Tx_data[5]= dec1;
Tx_data[6]= dec2;
Tx_data[7]= dec3;
Tx_data[10]= '#';
}
```

Appendix A

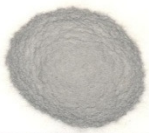
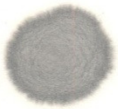
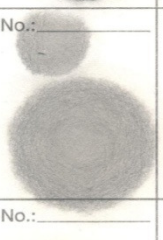
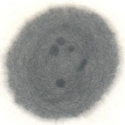
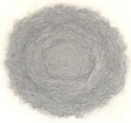
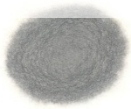
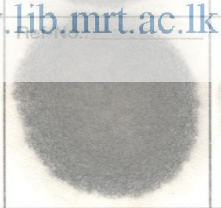
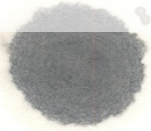
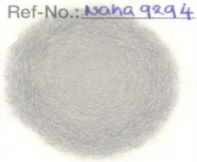
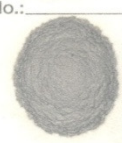
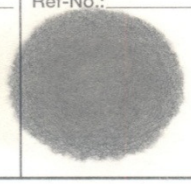
The Spot Check Data

SPOT CHECK

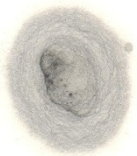
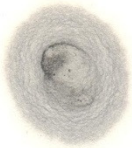
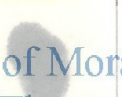
Vehicle Driven Inside the Colombo City.

Ref-No.: Wana 9235 	Ref-No.: 	Ref-No.: 	Ref-No.:
Ref-No.: Wana 9238 	Ref-No.: 	Ref-No.: 	Ref-No.:
Ref-No.: Wana 9239  	Ref-No.:	Ref-No.:	Ref-No.:
Ref-No.: Wana 9249 	Ref-No.:	Ref-No.:	Ref-No.:
Ref-No.: Wana 9254   	Ref-No.:	Ref-No.:	Ref-No.:
Ref-No.: Wana 9277   	Ref-No.:	Ref-No.:	Ref-No.:

SPOT CHECK

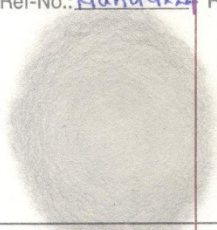
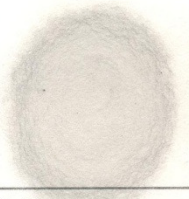
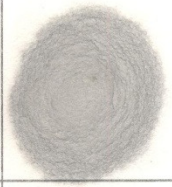
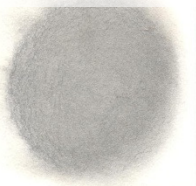
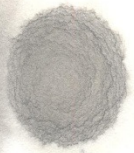
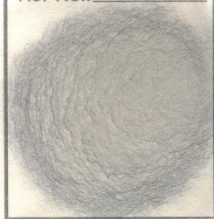
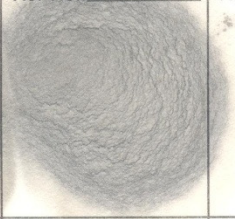
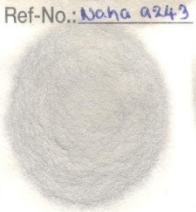
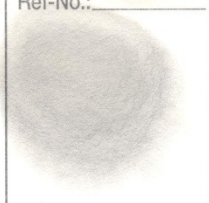
Ref-No.: Nana 9283 	Ref-No.: 	Ref-No.: 	Ref-No.:
Ref-No.: Nana 9289 	Ref-No.: 	Ref-No.:	Ref-No.:
Ref-No.: Nana 9291 	Ref-No.: 	Ref-No.:	Ref-No.:
Ref-No.: Nana 9292 	Ref-No.:	Ref-No.:	Ref-No.:
Ref-No.: Nana 9293  	Ref-No.:	Ref-No.: 	Ref-No.:
Ref-No.: Nana 9294 	Ref-No.: 	Ref-No.: 	Ref-No.:

SPOT CHECK

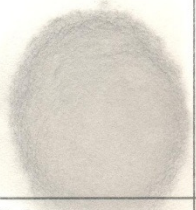
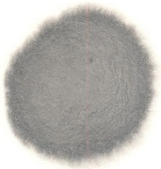
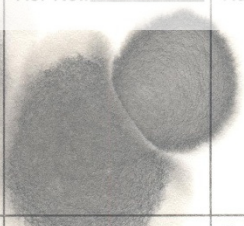
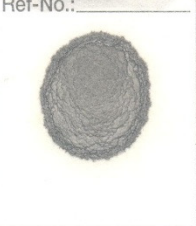
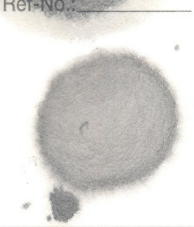
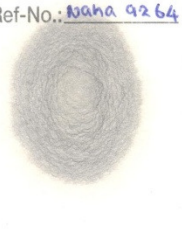
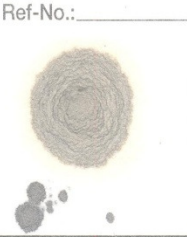
Ref-No.: Naha 9296 	Ref-No.:_____ 	Ref-No.:_____ 	Ref-No.:_____
Ref-No.: Naha 9255 	Ref-No.:_____ 	Ref-No.:_____ 	Ref-No.:_____
Ref-No.: Naha 9251 	Ref-No.:_____ 	Ref-No.:_____ 	Ref-No.:_____

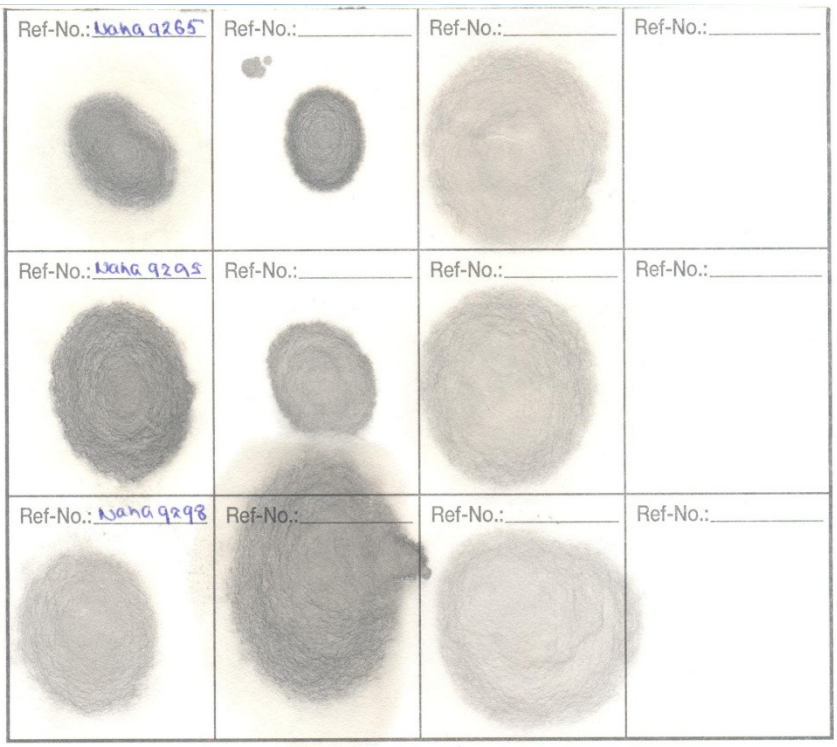
SPOT CHECK

Vehicles Driven out side of Colombo

Ref-No.: Nana9224 	Ref-No.: 	Ref-No.:	Ref-No.:
Ref-No.: Nana 9236 	Ref-No.: 	Ref-No.: 	Ref-No.:
Ref-No.: Nana 9240 	Ref-No.: 	Ref-No.:	Ref-No.:
Ref-No.: Nana 9241 	Ref-No.: 	Ref-No.: 	Ref-No.:
Ref-No.: Nana 9242 	Ref-No.: 	Ref-No.: 	Ref-No.:
Ref-No.: Nana 9243 	Ref-No.: 	Ref-No.:	Ref-No.:

SPOT CHECK

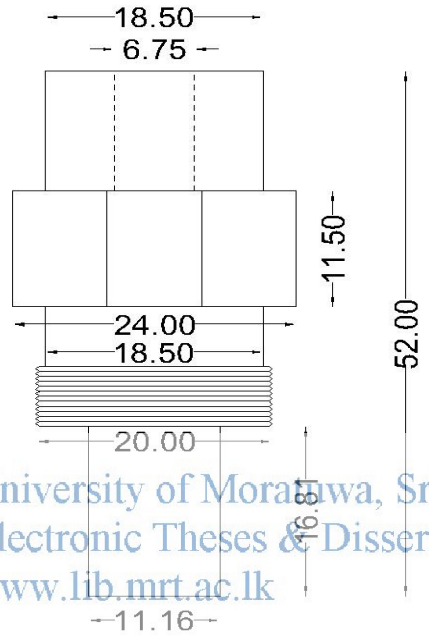
Ref-No.: Naha 9245 	Ref-No.: 	Ref-No.: 	Ref-No.:
Ref-No.: Naha 9246 	Ref-No.: 	Ref-No.: 	Ref-No.:
Ref-No.: Naha 9250 	Ref-No.: 	Ref-No.:	Ref-No.:
Ref-No.: Naha 9237 	Ref-No.: 	Ref-No.: 	Ref-No.:
Ref-No.: Naha 9263 	Ref-No.: 	Ref-No.: 	Ref-No.:
Ref-No.: Naha 9264 	Ref-No.: 	Ref-No.: 	Ref-No.:

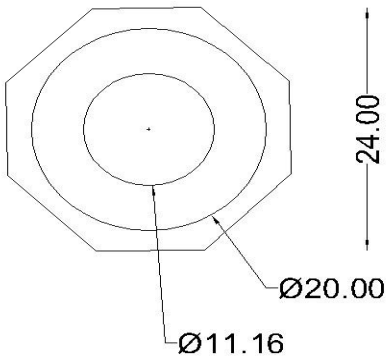


Appendix B

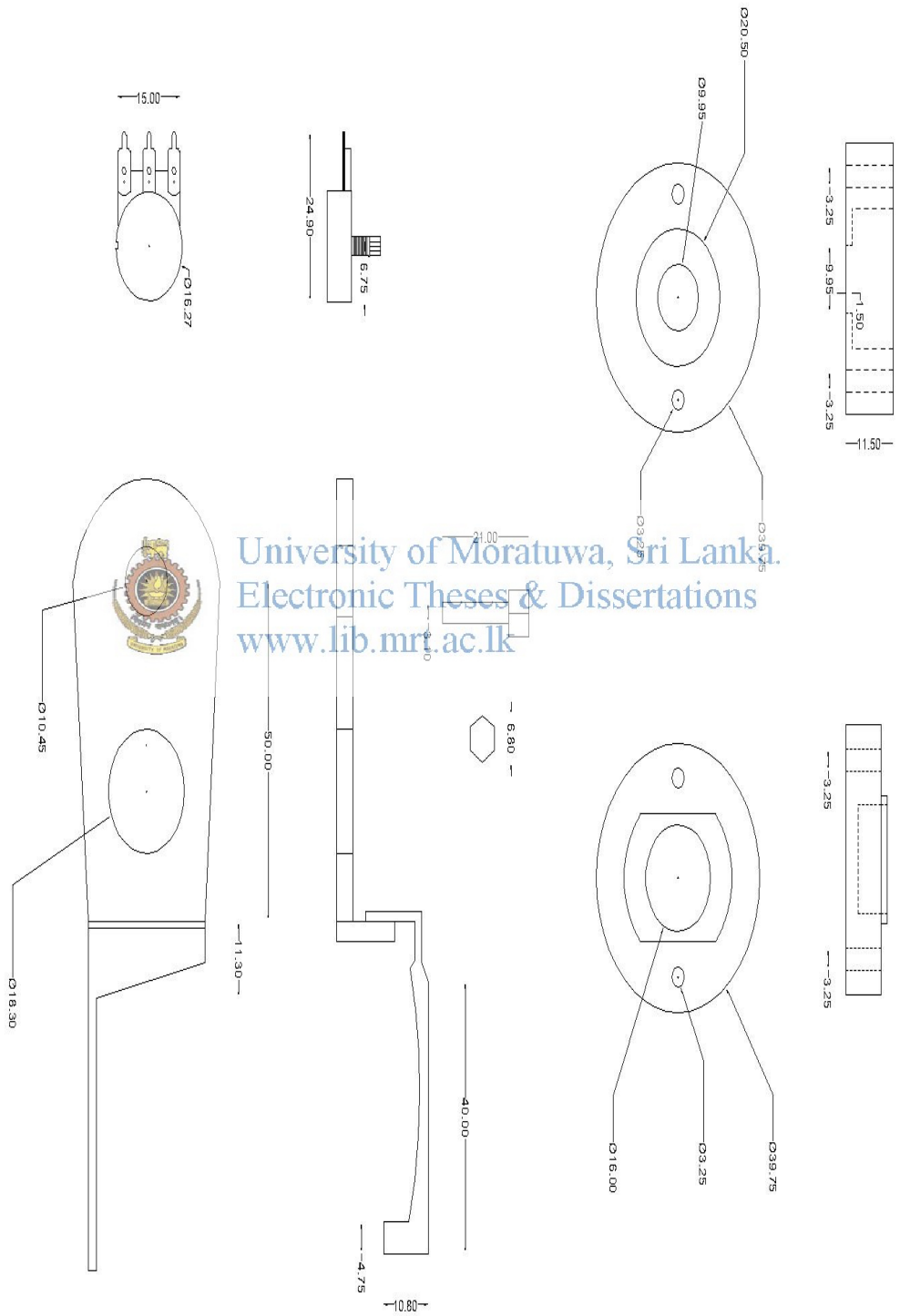
DRAWING OF TEMPERATURE SENSOR HOUSING

(All diamentions are mm)

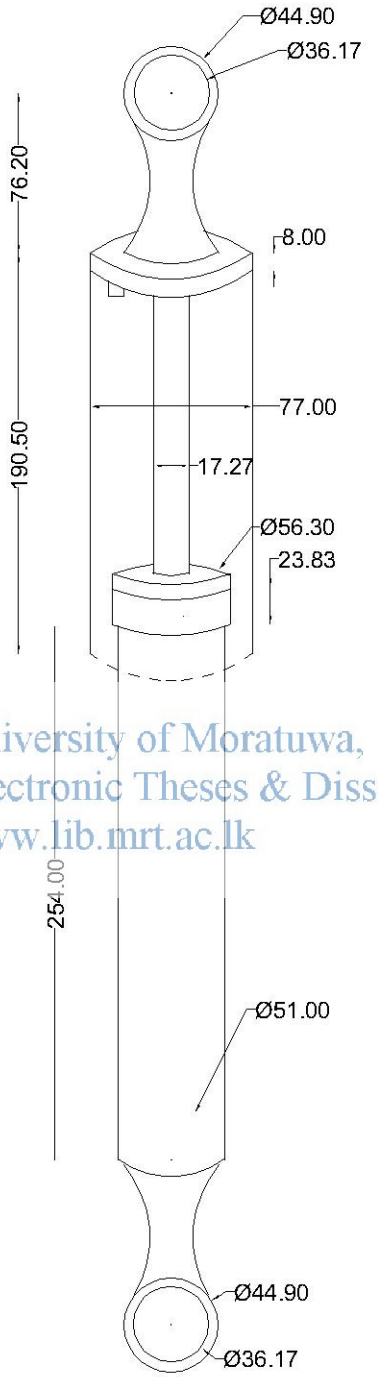




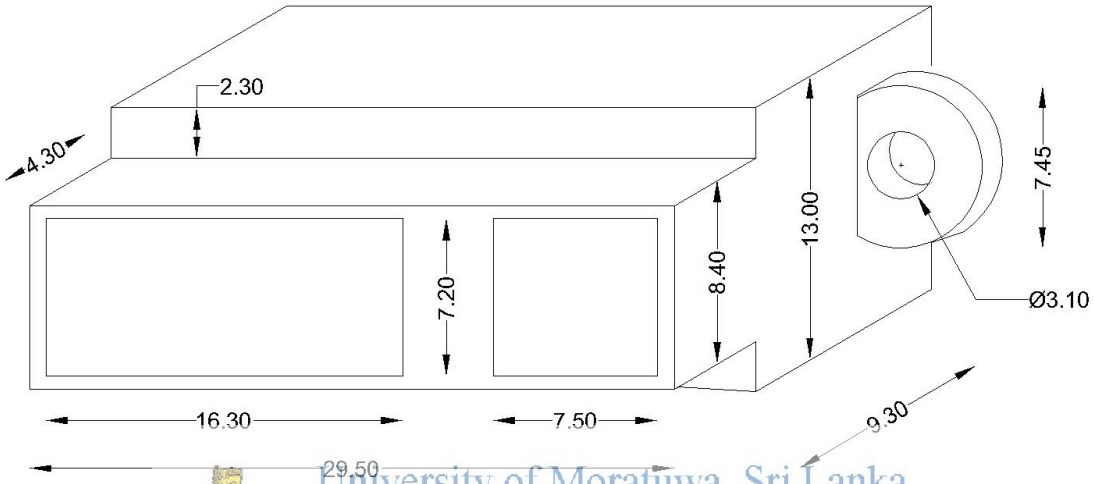
DRAWING OF BOLT ADJUSTER MOUNTING BRACKET



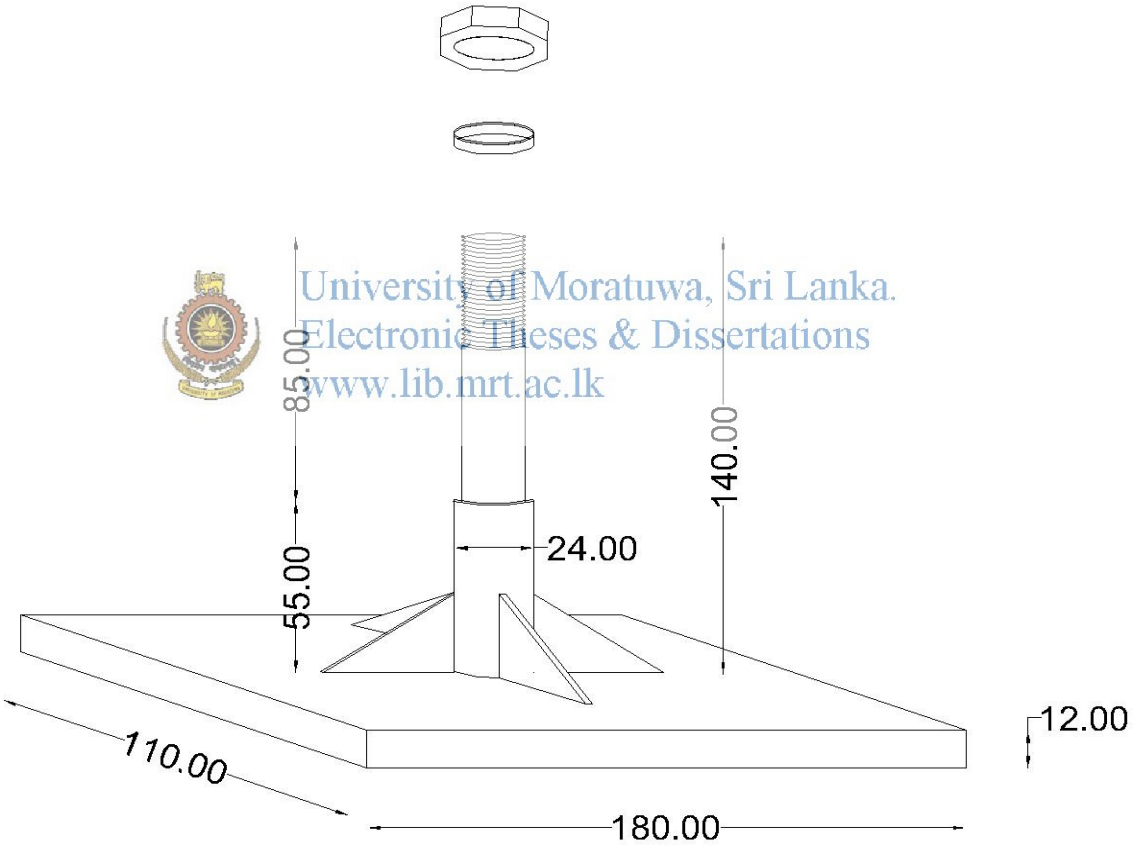
DRAWING OF LOAD INDICATING SENSOR HOUSING



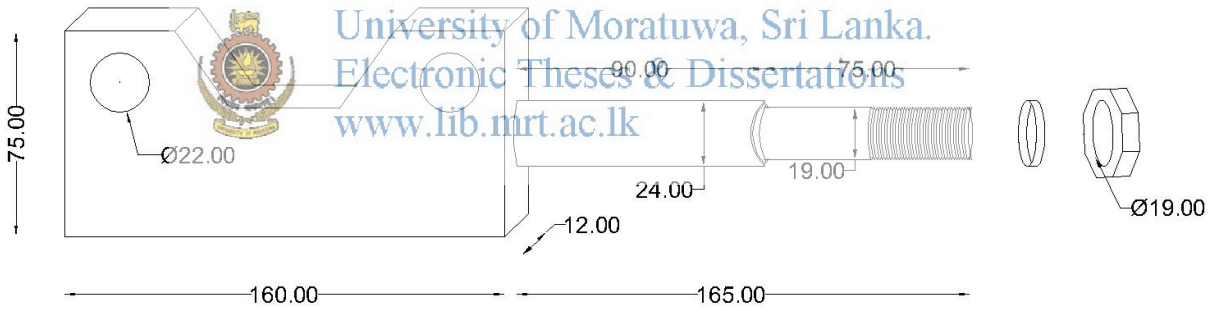
DRAWING OF LOAD SENSOR



DRAWING OF LOAD INDICATING SENSOR HOUSING BRACKET



DRAWING OF LOAD INDICATING SENSOR HOUSING BRACKET



Appendix C

HEAVY VHEHICLE MAINTANANCE SCHEDULLING WORK SHEET

Date -	Detail of cargo -
Vehicle No -	Total mileage for previous day -
Drivers Name -	Distance travelled – From - To -
Type of job -	Total engine running hours for pervious day -
Authorized person -	Total engine running hours per today -

Morning inspection chart

1. Engine

   Oil Level | Good | Weak |

   Engine cooling fan belt | Good | Normal | Weak | Error |

   Engine visual inspection | Good | Bad | Remarks _ _ _ _ _ _ _ _ _ _ _ _

   Radiator coolant level | Good | Bad | Remarks _ _ _ _ _ _ _ _ _ _ _ _

2. Electrical system

   Cleanness of battery | Good | Bad |

   Any bulb not working _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _

   Horn working | Good | Bad |

   Battery charging | Yes | No |

3. Suspension /Steering system

   Visual inspection of leaf springs | Good | Bad |

   Defective ball joints or cross joints | Yes | No | Remarks _ _ _ _ _ _ _ _ _ _

4. Brake system

   Remove water in air tanks | Yes | No |

   | Yes | No |

Start engine and check air pressure gauges are working

5. Body inspection

Body condition | Normal | Any abnormalities |

Condition of mirrors, side shutters, windscreen | Normal | Any abnormalities |

6. Tyre pressure and tyre condition

Tyre pressure guage reading | Recommended | High | Low |

Tyre ware | Normal | Tyre ware more than 75% | FL | FR | RLI | RLO | RRI | RRO | SPW |

Note-

FR- Front left

FL – Front Right

RLI – Rear Left Inner

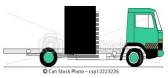
RLO- Rear Left Outer

RRI- Rear Right Inner

RRO- Rear Right Outer

Loading details – (Note. The driver should fill this part when the vehicle is loading)

The total weight can be load to the vehicle: _____ (19 Ton for TATA 2516)

Loading configuration (A)  Loading configuration (B) 

Recommended capacity		Current loading capacity		Recommended capacity		Current loading capacity	
Front Axel		Front Axel		Front Axel		Front Axel	
Rear Axle 1		Rear Axle 1		Rear Axle 1		Rear Axle 1	
Rear Axle 2 (Optional)		Rear Axle 2 (Optional)		Rear Axle 2 (Optional)		Rear Axle 2 (Optional)	
Total load for all axels		Total load for all axels		Total load for all axels		Total load for all axels	

Loading configuration (C)  Loading configuration (D) 

Recommended capacity		Current loading capacity		Recommended capacity		Current loading capacity	
Front Axle		Front Axle		Front Axle		Front Axle	

Rear Axle 1	Rear Axle 1	Rear Axle 1	Rear Axle 1
Rear Axle 2 (Optional)	Rear Axle 2 (Optional)	Rear Axle 2 (Optional)	Rear Axle 2 (Optional)
Total load for all axles	Total load for all axles	Total load for all axles	Total load for all axles

Instructions and guidance for users.

1. This works sheet to be filled by the maintenance crew and the driver daily and all the works sheet to be forwarded to the supervisor of the vehicle fleet once a week.

2. If the task is not finished within a particular day, you should continue the same work sheet. But important details are to be noted down clearly.

3. According to the nature of the task, the maintenance crew can miss daily inspection. For such occasions the driver ensures to fill this form by fist available time when crew is available.

4. The users ensure to get correct weight limitations for each and every configuration for respective vehicles in advance.

5. All drivers are to attend any minor repairs, important points highlighted by morning inspection prior to attend for the job.

Appendix D

C+ codes for the total circuit are as follows

```
#include <final_main.h>

#include <math.h>

//#define EXP_OUT_ENABLE  PIN_D0
//#define EXP_OUT_CLOCK   PIN_D1
//#define EXP_OUT_DO      PIN_D2
//#define NUMBER_OF_74595 1
//#include <74595.c>

#define set_low_flg    1
#define set_high_flg   2
#define low_adc        3
#define high_adc       4
#define low_ang        5
#define high_ang       6
#define temp_alm       7
#define temp_flg       8
#define ang_alm_flg    9
#define ang_alm        10
#define ml100000       11
#define ml10000        12
#define ml1000         13
#define ml100          14
#define ml10           15
#define ml1            16

# pragma config WDT=OFF
# pragma config LVP=OFF
# pragma config OSC = HS
#define Engine_comp pin_a3

#define frant_initial 20
#define middle_initial 20                    //  initial adc value of each sensor
#define rear_initial  20

#define frant_max_load 13
#define middle_max_load 10
#define rear_max_load 1                       // max loads


char state=0;//main state controle variable
char sub_state_1 = 1;
char sub_state_2 = 1;
char backlight_timer = 255;
char adc_unit,anlge;
float angle_const,angle;
char k,l,m,n;
```

```
char alm_off=0;
unsigned int  temp=0,temp1=0,temp2=0,temp3=0,temp4=0,read=0;
unsigned int rotation_adc,temp_adc;
unsigned char rotadc[2]={};
char display;
int alt_adc;
unsigned long engine_hrs;
unsigned char hundredthousands,tenthousands,thousands,hundreds,tens,ones;
float ADC;
int loop;
float temperature;
char ticks_en=0;
int frant=0;
int middle=0;
int rear=0;
int datasize ;     // difined size of data
int configuration ; // configuration A=0/B=1/C=2/D=3
int frant_index;
int rear_index;
int middle_index;


int frant_load=0;
int middle_load=0;
int rear_load=0;
// uart communication
int  RxDataReady=0;
int  RxDataPtr = 0;
int  DataCopied=0;

char DataCopy[10]={};
char Rxdata[10]={};

int setconfig=0;

int8 val1;
int8 val2;
int8 val3;
int8 decamin=0;
float loadtemp=0;
int belt=0;

int  table[3][25][2]={};
void beep();
void beeptwice();
void lighton();
void exit_menu();
void Settings_menu();
void Angular_meter_setup();
void set_ang_low();
void set_ang_high();
void clear_miles();
float Get_angle(float input);
void Master_reset();
void ang_alm_setup();
unsigned int get_adc();
void dis_adjuster_ang();
void dis_units(int16 adcval);
void dis_temp();
void dis_();
void dis_engine_hrs();
int check_engine();
```

```
void dis_alt_voltage();
void dis_temp2();
void get_engine_hrs();
void Save_engine_hrs(int16 input);
void read_LOAD_fr();
void read_LOAD_mid();
void read_LOAD_rear();
void load_distribution();
void calc_engine_hrs();
void setup_LOAD_configuration();
void display_LOAD();

//items

void engine_item();




//char indicator=255;
int16 ticks;
int16 half_hour=0;

//button definitions
#define  mode   PIN_B2
#define  set    PIN_B3
#define  scroll  PIN_B4
#define  exit   PIN_B5
#define  inc    PIN_D5
#define  dec    PIN_D6
#define  clr    PIN_D7

//lcd backlight and buzzer definitions
#define  buzzer  PIN_D4
#define  backlit PIN_A4


#define LCD_DB4   PIN_C0
#define LCD_DB5   PIN_C1
#define LCD_DB6   PIN_C2
#define LCD_DB7   PIN_C3

#define LCD_RS    PIN_E0
#define LCD_RW    PIN_E1
#define LCD_E     PIN_E2
#include <flex_lcd.c>

//write_expanded_outputs(eo);

#int_TIMER0
void  TIMER0_isr(void)
{

//output_high(backlit);

}

#int_TIMER1
void  TIMER1_isr(void)
{
```

```
//output_toggle(backlit);

}

#int_TIMER2
void  TIMER2_isr(void)
{
//output_toggle(pin_d1);
if(!input(pin_a3))
{

ticks++;


 }




}

#int_EXT
void  EXT_isr(void)
{//mileage counter functions goes here

}

#int_EXT1
void  EXT1_isr(void)
{

}

#int_RDA
void  RDA_isr(void)
{

output_high(PIN_D3);


char c;
int rxptr=0;

disable_interrupts(INT_RDA);
/* Read the receive buffer till atleast one or more character can be read */
  c = getc();
  switch (c)
  {
    case '$':
      {
              RxDataReady=0;
              RxDataPtr = 0;
      break;
      }


    case '#':
    {
      RxDataReady = 1;
      break;
     }
}
```

```
   Rxdata[RxDataPtr++ & 0x7F] = c;
  if (RxDataReady)
 {
  for(rxptr=0;rxptr<11;rxptr++)
    {
        DataCopy[rxptr]=Rxdata[rxptr];
      }
       Datacopied=1;
 }

 output_low(PIN_D3);
enable_interrupts(INT_RDA);

}

#int_TBE
void  TBE_isr(void)
{//load calculation system data transmitting goes here


}




void main()
{

 output_low(PIN_C4);
 output_low(PIN_C5);

 if(read_eeprom(11)==-1)write_eeprom(ml100000,0);
 if(read_eeprom(12)==-1)write_eeprom(ml10000,0);
 if(read_eeprom(13)==-1)write_eeprom(ml1000,0);
 if(read_eeprom(14)==-1)write_eeprom(ml100,0);
 if(read_eeprom(15)==-1)write_eeprom(ml10,0);
 if(read_eeprom(16)==-1)write_eeprom(ml1,0);

  //adc settings
  setup_adc(ADC_CLOCK_INTERNAL||VSS_VDD);
  setup_adc_ports(AN0_TO_AN2);


  //timer settings
  //setup_timer_0(RTCC_INTERNAL|RTCC_DIV_256);     //3.4 s overflow
  //setup_timer_1(T1_INTERNAL|T1_DIV_BY_8);     //104 ms overflow
  setup_timer_2(T2_DIV_BY_16,241,13);     //774 us overflow, 10.0 ms interrupt
  //setup_timer_3(T3_DISABLED|T3_DIV_BY_1);

  //timer interrupt settings
  //enable_interrupts(INT_TIMER0);
  //enable_interrupts(INT_TIMER1);
  enable_interrupts(INT_TIMER2);

  //external interrupt settings
 // enable_interrupts(INT_EXT);
  //enable_interrupts(INT_EXT1);

  //rs232 interrupt settings
  //enable_interrupts(INT_RDA);
//enable_interrupts(INT_TBE);
```

```
enable_interrupts(INT_RDA);

output_low(pin_c4);

  enable_interrupts(GLOBAL);




    //output_high(backlit);
lcd_init();
sub_state_1=1;
display=0;
output_high(backlit);
state=0;
loop=0;
ADC=0;

  get_engine_hrs();
  dis_engine_hrs();
  delay_ms(2500);


  while(1)
   {
  calc_engine_hrs();

    //mode button events
     if(input_state(mode)==0){//state controle loop
     delay_ms(5);
      state++;
      if(state==2)state=0;
       lighton();
    }//mode button event ends

    //monitoring system
      if(state==0)  {

    //engine running
    if(!input(pin_a3)){
  //  dis_alt_voltage();
     dis_temp();
    }

    //engine stop
    if(input(pin_a3)){
      if(display==0)dis_adjuster_ang();
      if(display==1)dis_temp2();
      if(display==2) display_LOAD();
      if(display==3)load_distribution();
      if(display==4)dis_engine_hrs();



    if(input_state(scroll)==0){beep();display++; delay_ms(200);}
    if(display>4)display=0;
    }
```

```
    }//state 0 ends


    if(state==1){
    Settings_menu();
    if(sub_state_1<10){if(input_state(dec)==0){lighton();delay_ms(20);sub_state_1+=10;}}//enter sub menu in sys menu
    if(sub_state_1<10){if(input_state(inc)==0){lighton();delay_ms(20);sub_state_1++;if(sub_state_1==5)sub_state_1=1;}}//change
items in sys menu

    }//state 1 options

     exit_menu();

     delay_ms(100);


   }//while loop ends


}//main loop ends here

void calc_engine_hrs()
{
 if(ticks>18000)
 {
  ticks=0;
  decamin++;
  if (decamin>9)
  {
  decamin=0;
  half_hour++;
  if((half_hour%2)==0)
  {
   engine_hrs++;
    beeptwice();
  }
  }
  if(engine_hrs>999999)engine_hrs=0;
   Save_engine_hrs(engine_hrs);
 }
}

void beep(){
output_high(buzzer);
delay_ms(25);
output_low(buzzer);
}

void beeptwice(){
output_high(buzzer);
delay_ms(25);
output_low(buzzer);
delay_ms(10);
output_high(buzzer);
delay_ms(25);
output_low(buzzer);
}

void lighton(){
output_high(backlit);
```

```
backlight_timer = 255;
beep();
}

void exit_menu(){
if(input_state(exit)==0){//state controle loop
  delay_ms(5);
  state=0;
  sub_state_1=1;
  sub_state_2=1;
  lighton();
  beeptwice();
  read=0;
  }
}

void Settings_menu(){
printf(LCD_PUTC, "\fSystem Settings");
//settings menu options
if(sub_state_1==1){printf(LCD_PUTC, "\nBelt setup");}
//if(sub_state_1==2){printf(LCD_PUTC, "\nMileage count");}
if(sub_state_1==2){printf(LCD_PUTC, "\nBelt alm setup");}
if(sub_state_1==3){printf(LCD_PUTC, "\n Load config");}
if(sub_state_1==4){printf(LCD_PUTC, "\nMaster Reset!");}
if(sub_state_1==11)Angular_meter_setup();
//if(sub_state_1==12)clear_miles();
if(sub_state_1==12)ang_alm_setup();
if(sub_state_1==13) setup_LOAD_configuration();
if(sub_state_1==14)Master_reset();
if(sub_state_1>10){if(input_state(scroll)==0){delay_ms(200);sub_state_1-=10;lighton();}}
}//settings menu ends

void Angular_meter_setup(){
printf(LCD_PUTC, "\fBelt ang setup");
if(sub_state_2==1){printf(LCD_PUTC, "\nSet low angle:");}
if(sub_state_2==2){printf(LCD_PUTC, "\nSet high angle:");}
if(sub_state_2==11)set_ang_low();
if(sub_state_2==12)set_ang_high();


if(sub_state_1>10&&sub_state_2<10){if(input_state(inc)==0){
delay_ms(5);
sub_state_2++;
if(sub_state_2==3){
sub_state_2=1;}
lighton();}}

  if(sub_state_1>10&&sub_state_2<10){
    if(input_state(dec)==0){
    delay_ms(5);
    sub_state_2+=10;
    temp=0;temp1=0;
    if(sub_state_2>13)sub_state_2=11;
    lighton();
    }
  }
if(sub_state_1>10&&sub_state_2>10){if(input_state(scroll)==0){delay_ms(200);
sub_state_2-=10;lighton();read=0;
temp=temp1=temp2=0;}}
}//angular meter setup ends
```

```
void set_ang_low(){
    if(read_eeprom(set_low_flg)!=1){
    printf(LCD_PUTC, "\fSet LOW End:");
    }

    if(read_eeprom(set_low_flg)==1){
    printf(LCD_PUTC, "\fEdit LOW End:");
    }
    set_adc_channel(2);
    delay_us(10);
    temp1=read_adc();
    lcd_gotoxy(0,2);
    printf(LCD_PUTC, "U:%03u",temp1);
    if(read==0){if(read_eeprom(set_low_flg)==1)temp=read_eeprom(low_ang);read=1;}
    lcd_gotoxy(8,2);
    printf(LCD_PUTC, "Angle:%02u",temp);
    if(sub_state_1>10&&sub_state_2>10){
    if(input_state(inc)==0){delay_ms(20);lighton();temp+=1;if(temp>50)temp=0;}//increment angle
    if(input_state(dec)==0){delay_ms(20);lighton();temp-=1;if(temp==-1)temp=50;}//decrement angle
    if(input_state(clr)==0){delay_ms(20);lighton();temp=0;}//decrement angle

      if(input_state(set)==0){
        delay_ms(10);
        lighton();
        printf(LCD_PUTC, "\fSaving Data...");
        delay_ms(750);
        write_eeprom(low_ang,temp);
        write_eeprom(low_adc,temp1);
        write_eeprom(set_low_flg,1);
        printf(LCD_PUTC, "\nok...");
        delay_ms(500);
      }//save data
      exit_menu();
    }
}//ang low set ends

void set_ang_high(){
    if(read_eeprom(set_low_flg)!=1){
    printf(LCD_PUTC, "\fError!!");
    printf(LCD_PUTC, "\nSet Lo first!");
    delay_ms(750);
    lighton();
    sub_state_2=11;
    }

    if(read_eeprom(set_low_flg)==1&&read_eeprom(set_high_flg)==1){
    printf(LCD_PUTC, "\fEdit Hi End:");
    }

    if(read_eeprom(set_low_flg)==1&&read_eeprom(set_high_flg)!=1){
    printf(LCD_PUTC, "\fSet Hi End:");
    }
    if(read_eeprom(set_low_flg)==1){
    set_adc_channel(2);
    delay_us(10);
    temp1=read_adc();
    lcd_gotoxy(0,2);
    printf(LCD_PUTC, "U:%03u",temp1);
    if(read==0){if(read_eeprom(set_high_flg)==1)temp=read_eeprom(high_ang);read=1;}
    lcd_gotoxy(8,2);
    printf(LCD_PUTC, "Angle:%02u",temp);
```

```
    if(temp<=read_eeprom(low_ang)){lcd_gotoxy(14,1);printf(LCD_PUTC,">H");}
    if(temp1<=read_eeprom(low_adc)){lcd_gotoxy(12,1);printf(LCD_PUTC,">U");}
    if(temp>read_eeprom(low_ang)&&temp1>read_eeprom(low_adc)){lcd_gotoxy(13,1);printf(LCD_PUTC,"Ok!");}

      if(sub_state_1>10&&sub_state_2>10){
    if(input_state(inc)==0){delay_ms(20);lighton();temp+=1;if(temp>50)temp=0;}//increment angle
    if(input_state(dec)==0){delay_ms(20);lighton();temp-=1;if(temp==-1)temp=50;}//decrement angle
    if(input_state(clr)==0){delay_ms(20);lighton();temp=0;}//decrement angle

    if(input_state(set)==0){
        delay_ms(10);
        lighton();
        printf(LCD_PUTC, "\fSaving Data...");
        delay_ms(750);
        write_eeprom(high_ang,temp);
        write_eeprom(high_adc,temp1);
        write_eeprom(set_high_flg,1);
        printf(LCD_PUTC, "\nok...");
        delay_ms(500);
      }//save data

      }//if sub1>10 and sub2>10
    }
    exit_menu();


}//ang high set ends

float Get_angle(float input){

float adc_unit,range;
float angle_const,angle;
float x,y,p,q;
x=read_eeprom(high_adc);
y=read_eeprom(low_adc);
p=read_eeprom(high_ang);
q=read_eeprom(low_ang);
adc_unit=x-y;
range=p-q;
angle_const=(range/adc_unit);
angle=((input-y)*angle_const)+q;
return angle;
}

void clear_miles(){
    printf(LCD_PUTC, "\fClear Counter:");

    if(sub_state_1>10){
    if(input_state(inc)==0){delay_ms(20);lighton();temp+=1;if(temp>100)temp=0;}//increment temperature
    if(input_state(dec)==0){delay_ms(20);lighton();temp-=1;if(temp==-1)temp=100;}//decrement temperature
    if(input_state(clr)==0){delay_ms(20);lighton();temp=0;}
      if(input_state(set)==0){
        delay_ms(10);
        lighton();
        printf(LCD_PUTC, "\fSaving Data...");
        delay_ms(750);
        write_eeprom(temp_alm,temp);
        write_eeprom(temp_flg,1);
        printf(LCD_PUTC, "\nok...");
        delay_ms(500);
      }//save data
```

```
    }//if sub1>10 and sub2>10

}

void Master_reset(){
     printf(LCD_PUTC, "\fReset System!!!");
    printf(LCD_PUTC, "\nPress CLR");

if(read_eeprom(temp_flg)==0&&read_eeprom(set_low_flg)==0&&read_eeprom(set_high_flg)==0&&read_eeprom(ang_alm_flg)==
0){
    printf(LCD_PUTC, "\fSystem Cleared!");
    printf(LCD_PUTC, "\nCal: Again!");
    delay_ms(1000);
    printf(LCD_PUTC, "\fRedirecting to");
    printf(LCD_PUTC, "\nSettings menu!");
    delay_ms(500);
    sub_state_1=11;
     }
    if(sub_state_1>10){
        if(input_state(clr)==0){
        delay_ms(10);
        lighton();
        printf(LCD_PUTC, "\fErasing Data...");
        delay_ms(750);
        write_eeprom(temp_alm,0);
        write_eeprom(temp_flg,0);
        write_eeprom(low_adc,0);
        write_eeprom(high_adc,0);
        write_eeprom(low_ang,0);
        write_eeprom(high_ang,0);
        write_eeprom(set_low_flg,0);
        write_eeprom(set_high_flg,0);
        write_eeprom(ang_alm,0);
        write_eeprom(ang_alm_flg,0);
        alm_off=0;
        printf(LCD_PUTC, "\nok...");
        delay_ms(500);
      }


}
}

void ang_alm_setup(){
    if(read_eeprom(ang_alm_flg)!=1)printf(LCD_PUTC, "\fSet Angl: Limit");
   if(read_eeprom(ang_alm_flg)==1)printf(LCD_PUTC, "\fEdit Angl: Limit");

    if(read==0){if(read_eeprom(ang_alm_flg)==1)temp=read_eeprom(ang_alm);read=1;}
    printf(LCD_PUTC, "\nAngl: Limit =%02u",temp);

    if(sub_state_1>10){
    if(input_state(inc)==0){delay_ms(20);lighton();temp+=1;if(temp>50)temp=0;}//increment temperature
    if(input_state(dec)==0){delay_ms(20);lighton();temp-=1;if(temp==-1)temp=50;}//decrement temperature
    if(input_state(clr)==0){delay_ms(20);lighton();temp=0;}
      if(input_state(set)==0){
        delay_ms(10);
        lighton();
        printf(LCD_PUTC, "\fSaving Data...");
        delay_ms(750);
        write_eeprom(ang_alm,temp);
        write_eeprom(ang_alm_flg,1);
        printf(LCD_PUTC, "\nok...");
```

```
        delay_ms(500);
      }//save data

    }//if sub1>10 and sub2>10



}//end


void dis_adjuster_ang()
{
int value;
float adc_unit,range;
float angle_const,angle;
float x,y,p,q;


    k=read_eeprom(set_low_flg);
    l=read_eeprom(set_high_flg);
    //m=read_eeprom(temp_flg);
    n=read_eeprom(ang_alm_flg);
    set_adc_channel(2);
    delay_us(10);
    rotation_adc=read_adc();

x=read_eeprom(high_adc);
y=read_eeprom(low_adc);
p=read_eeprom(high_ang);
q=read_eeprom(low_ang);
adc_unit=x-y;
range=p-q;
angle_const=(range/adc_unit);


    //rotation_adc=get_adc();
    //if(l==1&&k==1&&n!=1){printf(LCD_PUTC, "\fSet:Angl: Limit!");delay_ms(10);}
    if(l!=1&&k!=1)printf(LCD_PUTC, "\fCal:Belt meter!");

    if(k==1&&l==1){

    value=((rotation_adc-y)*angle_const)+q;



   if(25.00<value)
   {printf(LCD_PUTC,"\f Belt:Error "); delay_ms(20);}

  else if (22.00<=value)
  {
  printf(LCD_PUTC,"\f Belt:Weak ");delay_ms(20);
  }

  else if (17.10<=value)
  {
  printf(LCD_PUTC,"\f Belt:Good");delay_ms(20);
  }
```

```
   else
   {
   printf(LCD_PUTC,"\f Belt:New");delay_ms(20);
   }

     if(value>read_eeprom(ang_alm ) )
      if(alm_off==0){output_high(buzzer);delay_ms(500);output_low(buzzer);delay_ms(500+);if(input_state(set)==0)alm_off=1;}


    }
 delay_ms(100);
}

void dis_temp2(){
set_adc_channel(0);
delay_us(10);
ADC+=read_adc();
ADC/=2;
loop++;
if(loop>20){temperature=ADC*0.588;loop=0;}
printf(LCD_PUTC,"\fTemp:=%4.2f",temperature);
if(temperature==90)printf(LCD_PUTC,"\fOver Temp: Err!");
}

void dis_temp(){
set_adc_channel(0);
delay_us(10);
ADC+=read_adc();
ADC/=2;
loop++;
if(loop>50){temperature=ADC*0.588;loop=0;}
printf(LCD_PUTC,"\nTemp:=%4.2f",temperature);
if(temperature==90)printf(LCD_PUTC,"\nOver Temp: Err!");
}

void dis_alt_voltage(){
    set_adc_channel(1);
    delay_us(10);
    alt_adc=read_adc();
    float voltage;
    voltage = 0.1176470588*alt_adc;
printf(LCD_PUTC,"\fALT Voltage=%2.1g",voltage);
if(voltage>28)printf(LCD_PUTC,"\fOver Volt: Err!");
}



void dis_load(){
    set_adc_channel(3);
    delay_us(10);
    alt_adc=read_adc();
    //float voltage;
printf(LCD_PUTC,"\fload=%d",alt_adc);
}

void dis_engine_hrs(){


 printf(LCD_PUTC,"\fEngine work hour:");
unsigned int m1,m2,m3,m4,m5,m6;
//m1 = read_eeprom(11);
//m2 = read_eeprom(12);
```

```
//m3 = read_eeprom(13);
//m4 = read_eeprom(14);
//m5 = read_eeprom(15);
//m6 = read_eeprom(16);
m1 = read_eeprom(ml100000);
m2 = read_eeprom(ml10000);
m3 = read_eeprom(ml1000);
m4 = read_eeprom(ml100);
m5 = read_eeprom(ml10);
m6 = read_eeprom(ml1);
//m1=0,m2=0,m3=0,m4=0,m5=0,m6=0;

lcd_gotoxy(1,2);
printf(LCD_PUTC,"%d",read_eeprom(11));
lcd_gotoxy(2,2);
printf(LCD_PUTC,"%d",read_eeprom(12));
lcd_gotoxy(3,2);
printf(LCD_PUTC,"%d",read_eeprom(13));
lcd_gotoxy(4,2);
printf(LCD_PUTC,"%d",read_eeprom(14));
lcd_gotoxy(5,2);
printf(LCD_PUTC,"%d",read_eeprom(15));
lcd_gotoxy(6,2);
printf(LCD_PUTC,"%d",read_eeprom(16));
lcd_gotoxy(7,2);
printf(LCD_PUTC," hrs");

}


void Save_engine_hrs(int16 input){
if(engine_hrs>999999)engine_hrs=0;
   hundredthousands=input/100000;
   tenthousands =(input%100000)/10000;
   thousands =((input%100000)%10000)/1000;
   hundreds  =(((input%100000)%10000)%1000)/100;
   tens     =((((input%100000)%10000)%1000)%100)/10;
   ones     =((((input%100000)%10000)%1000)%100)%10;

 write_eeprom(ml100000,hundredthousands);
 write_eeprom(ml10000,tenthousands);
 write_eeprom(ml1000,thousands);
 write_eeprom(ml100,hundreds);
 write_eeprom(ml10,tens);
 write_eeprom(ml1,ones);

}

void get_engine_hrs(){
int m1,m2,m3,m4,m5,m6;
m1 = read_eeprom(ml100000)*100000;
m2 = read_eeprom(ml10000)*10000;
m3 = read_eeprom(ml1000)*1000;
m4 = read_eeprom(ml100)*100;
m5 = read_eeprom(ml10)*10;
m6 = read_eeprom(ml1)*1;
engine_hrs = m1+m2+m3+m4+m5+m6;
if(engine_hrs>999999)engine_hrs=0;
}

int check_engine(){
```

```
    set_adc_channel(1);
    delay_us(10);
    alt_adc=read_adc();
    float voltage;
    voltage = 0.117*alt_adc;

    if(voltage>6){
    return 1;
     ticks_en=1;
    }

    if(voltage<6){
    return 0;
     ticks_en=0;
    }
}

void setup_LOAD_configuration()
{
setconfig=1;

printf(LCD_PUTC,"\f A | B | C | D ");

if(sub_state_1>10&&sub_state_2<10){if(input_state(inc)==0){
delay_ms(50);
sub_state_2++;
if(sub_state_2==5){
sub_state_2=1;}
lighton();}}
if (sub_state_2==1)
{printf(LCD_PUTC,"\n Front");
delay_ms(200) ;
printf(LCD_PUTC,"\f A*| B | C | D ");
delay_ms(200);
configuration=0;
printf(LCD_PUTC,"\n Front");
}
if (sub_state_2==2)
{
printf(LCD_PUTC,"\n Middle ");
delay_ms(200);
printf(LCD_PUTC,"\f A | B*| C | D ");
delay_ms(200);
configuration=1;
printf(LCD_PUTC,"\n Middle ");
}
if (sub_state_2==3)
{
printf(LCD_PUTC,"\n Rear ");
delay_ms(200);
printf(LCD_PUTC,"\f A | B | C*| D ");
delay_ms(200) ;
configuration=2;
printf(LCD_PUTC,"\n Rear ");
}
if (sub_state_2==4)
{
printf(LCD_PUTC,"\n uniform");
delay_ms(200);
printf(LCD_PUTC,"\f A | B | C | D*");
delay_ms(200) ;
```

```
configuration=3;
printf(LCD_PUTC,"\n uniform");
}

}



void read_LOAD_fr()
{
output_high(PIN_C4);
output_high(PIN_C5);

printf("$MAFRT\n");

printf("$MAFRT\n");

printf("$MAFRT\n");

output_low(PIN_C4);
output_low(PIN_C5);


int1 ack=0;
while(!ack)
{
if( RxDataReady && Datacopied)
  {
   if(DataCopy[0]=="$" && DataCopy[1]=="S" &&DataCopy[2]=="L" &&DataCopy[3]=="F" && DataCopy[4]=="R"
&&DataCopy[5]=="T" )
   {
   frant=(DataCopy[6]-48)*100+(DataCopy[7]-48)*10+DataCopy[8]-48;
   ack=1;
   output_high(PIN_C4);

output_high(PIN_C5);
   printf("$MAFAK"); printf("$MAFAK"); printf("$MAFAK"); printf("$MAFAK"); printf("$MAFAK");
   delay_ms(100);
   output_low(PIN_C4);

output_low(PIN_C5);
   }

  }
}
}

void read_LOAD_mid()
{
int1 ack=0;
while(!ack)
{
if( RxDataReady && Datacopied)
  {
   if(DataCopy[0]=="$" && DataCopy[1]=="S" &&DataCopy[2]=="L" &&DataCopy[3]=="M" && DataCopy[4]=="I"
&&DataCopy[5]=="D" )
   {
   middle=(DataCopy[6]-48)*100+(DataCopy[7]-48)*10+DataCopy[8]-48;

   ack=1;
   printf("$MAMAK");
   }
```

```
  }
}

}


void read_LOAD_rear()
{
int1 ack=0;
while(!ack)
{
if( RxDataReady && Datacopied)
  {
    if(DataCopy[0]=="$" && DataCopy[1]=="S" &&DataCopy[2]=="L" &&DataCopy[3]=="R" && DataCopy[4]=="E"
&&DataCopy[5]=="R" )
    {
    rear=(DataCopy[6]-48)*100+(DataCopy[7]-48)*10+DataCopy[8]-48;

    ack=1;
    printf("$MARAK");
    }

  }


}
}

void serch_LOAD()
{
int frant_margin;
int middle_margin;
int rear_margin;
```



```
int prescaler=1;
int min_frant_margin=0;
int min_middle_margin=0;
int min_rear_margin=0;

int fr_data_index=0;
int mi_data_index=0;
int re_data_index=0;



min_frant_margin= (table[configuration][0][fr_data_index])-frant;
min_middle_margin=(table[configuration][1][mi_data_index])-middle;
min_rear_margin  =  (table[configuration][2][re_data_index])-rear;
// searching frant load
for (fr_data_index=1;fr_data_index<=datasize; fr_data_index++)
  {
  frant_margin=(table[configuration][0][fr_data_index])-frant;
  if (frant_margin<0)frant_margin=-frant_margin;
  if (min_frant_margin>frant_margin)
  {
    frant_margin=min_frant_margin;
    frant_index=fr_data_index;


  }
```

```
  }
  // serching for middle load
for (mi_data_index=1;mi_data_index<=datasize; mi_data_index++)
  {
  middle_margin=(table[configuration][1][mi_data_index])-middle;
  if (middle_margin<0)middle_margin=-middle_margin;
  if (min_middle_margin>middle_margin)
  {
    middle_margin=min_middle_margin;
    middle_index=mi_data_index;


  }

  }

  // serching for rear load

  for (re_data_index=1;re_data_index<=datasize; re_data_index++)
  {
  rear_margin=(table[configuration][2][re_data_index])-rear;
  if (rear_margin<0)rear_margin=-rear_margin;
  if (min_rear_margin>rear_margin)
  {
    rear_margin=min_rear_margin;
    rear_index=re_data_index;


  }

  }

frant_load=frant_index*prescaler;
middle_load=middle_index*prescaler;
rear_load=rear_index*prescaler;


}

void display_LOAD()

{
float total_load=0.0;
int previousval1=0.0;
int previousval2=0.0;
int previousval3=0.0;


if(DataCopied==1 && RxDataReady==1)
{
if(DataCopy[1]=='S' && DataCopy[2]=='F' && DataCopy[3]=='R' && DataCopy[4]=='T' )
{
DataCopied=0;


val1=(DataCopy[5]-48)*100+(DataCopy[6]-48)*10+(DataCopy[7]-48);
previousval1=val1;
//printf(LCD_PUTC,"\n %u  | %u | %u ",val1,val2,val3 );

}
```

```
else val1=previousval1;
 if(DataCopy[1]=='S' && DataCopy[2]=='M' && DataCopy[3]=='I' && DataCopy[4]=='D' )
{
DataCopied=0;
val2=(DataCopy[5]-48)*100+(DataCopy[6]-48)*10+(DataCopy[7]-48);
//printf(LCD_PUTC,"\n%u  | %u | %u ",val1,val2,val3 );
previousval2=val2;
}

else val2=previousval2;


 if(DataCopy[1]=='S' && DataCopy[2]=='R' && DataCopy[3]=='E' && DataCopy[4]=='R' )
{
DataCopied=0;
val3=(DataCopy[5]-48)*100+(DataCopy[6]-48)*10+(DataCopy[7]-48);
previousval3=val3;
}

}


if (configuration==0) total_load=((0.206)*val1 - 31.68);
if (configuration==1) total_load=((0.232)*val2 - 29.30);
if (configuration==2) total_load=((0.122)*val3 - 27.13);
if (configuration==3) total_load=((0.5)*val2 - 59.0);

if(total_load<=0)total_load=loadtemp;
else loadtemp=total_load;

if (setconfig !=1)printf(LCD_PUTC,"\f configuration error");

printf(LCD_PUTC,"\f Total Load=%2.1gT ",total_load);
if (configuration==0) printf(LCD_PUTC,"\n front config");
if (configuration==1) printf(LCD_PUTC,"\n middle config");
if (configuration==2) printf(LCD_PUTC,"\n rear config");
if (configuration==3) printf(LCD_PUTC,"\n uniform config");

delay_ms(20);

}


void load_distribution()

{
float total_load=0;
int previousval1;
int previousval2;
int previousval3;


if(DataCopied==1 && RxDataReady==1)
{
if(DataCopy[1]=='S' && DataCopy[2]=='F' && DataCopy[3]=='R' && DataCopy[4]=='T' )
{
DataCopied=0;


val1=(DataCopy[5]-48)*100+(DataCopy[6]-48)*10+(DataCopy[7]-48);
previousval1=val1;
//printf(LCD_PUTC,"\n %u  | %u | %u ",val1,val2,val3 );
```

```
}

else val1=previousval1;
 if(DataCopy[1]=='S' && DataCopy[2]=='M' && DataCopy[3]=='I' && DataCopy[4]=='D' )
{
DataCopied=0;
val2=(DataCopy[5]-48)*100+(DataCopy[6]-48)*10+(DataCopy[7]-48);
//printf(LCD_PUTC,"\n%u  | %u | %u ",val1,val2,val3 );
previousval2=val2;
}

else val2=previousval2;


 if(DataCopy[1]=='S' && DataCopy[2]=='R' && DataCopy[3]=='E' && DataCopy[4]=='R' )
{
DataCopied=0;
val3=(DataCopy[5]-48)*100+(DataCopy[6]-48)*10+(DataCopy[7]-48);
previousval3=val3;
}

}



if (configuration==0) total_load=((0.206)*val1 - 31.68);
if (configuration==1) total_load=((0.232)*val2 - 29.30);
if (configuration==2) total_load=((0.122)*val3 - 27.13);
if (configuration==3) total_load=((0.5)*val2 - 59.0);

if(total_load<=0)total_load=loadtemp;
else loadtemp=total_load;

if (setconfig !=1)printf(LCD_PUTC,"\f configuration error");


if (configuration==0) printf(LCD_PUTC,"\f front config");
if (configuration==1) printf(LCD_PUTC,"\f middle config");
if (configuration==2) printf(LCD_PUTC,"\f rear config");
if (configuration==3) printf(LCD_PUTC,"\f uniform config");

printf(LCD_PUTC,"\fFront| Mid| Rear");

printf(LCD_PUTC,"\n%2.1gT |%2.1gT| %2.1gT",total_load*.3,total_load*.35,total_load*.35);
delay_ms(20);




}

// items

void engine_item()
{
printf(LCD_PUTC,"\fE");
delay_ms(50);
printf(LCD_PUTC,"\fEn:");
delay_ms(50);
printf(LCD_PUTC,"\fEng");
```

```
delay_ms(50);
printf(LCD_PUTC,"\fEngi");

delay_ms(50);
printf(LCD_PUTC,"\fEngin");
delay_ms(50);
printf(LCD_PUTC,"\fEngine");
delay_ms(50);
printf(LCD_PUTC,"\fEngine w");
delay_ms(50);
printf(LCD_PUTC,"\fEngine wo");
delay_ms(50);
printf(LCD_PUTC,"\fEngine wor");

delay_ms(50);
printf(LCD_PUTC,"\fEngine work");

delay_ms(50);
printf(LCD_PUTC,"\fEngine work h");
delay_ms(50);
printf(LCD_PUTC,"\fEngine work ho");
delay_ms(50);
printf(LCD_PUTC,"\fEngine work hou");
delay_ms(50);
printf(LCD_PUTC,"\fEngine work hour");
delay_ms(50);
printf(LCD_PUTC,"\fEngine work hour:");


}
```

Appendix E

Pictures of QMU and other events of the research