LART-CS08

Maintenance Manual

Senior Design - Spring 2008

Lafayette College

5/9/2008

MAINTENANCE MANUAL

Table of Contents:

Introduction

Purpose of this Manual

The purpose of this Maintenance Manual is to explain the unique technical principles and details of the system operation. In addition, it includes information on any advanced maintenance or calibration techniques that could be applied by an expert maintainer. This document includes schematics, pinouts of all the connectors, the signal assignments of all the cables, and the semantics of all the hardware and software interfaces.

Procedure

The manual includes tables, documents, and instructions on the methods needed to maintain and operate the whole system.

Fiscal Management

Fiscal responsibility is important because all the expenditures for the maintenance activities have to be properly documented to ensure the maintenance of the project is under the budget.

Principles of Operation (PoO)

The system is composed of the MasterPC, the octopus, the circuit boards and the cables that connect the circuit board to the rail system.

Circuit Board:

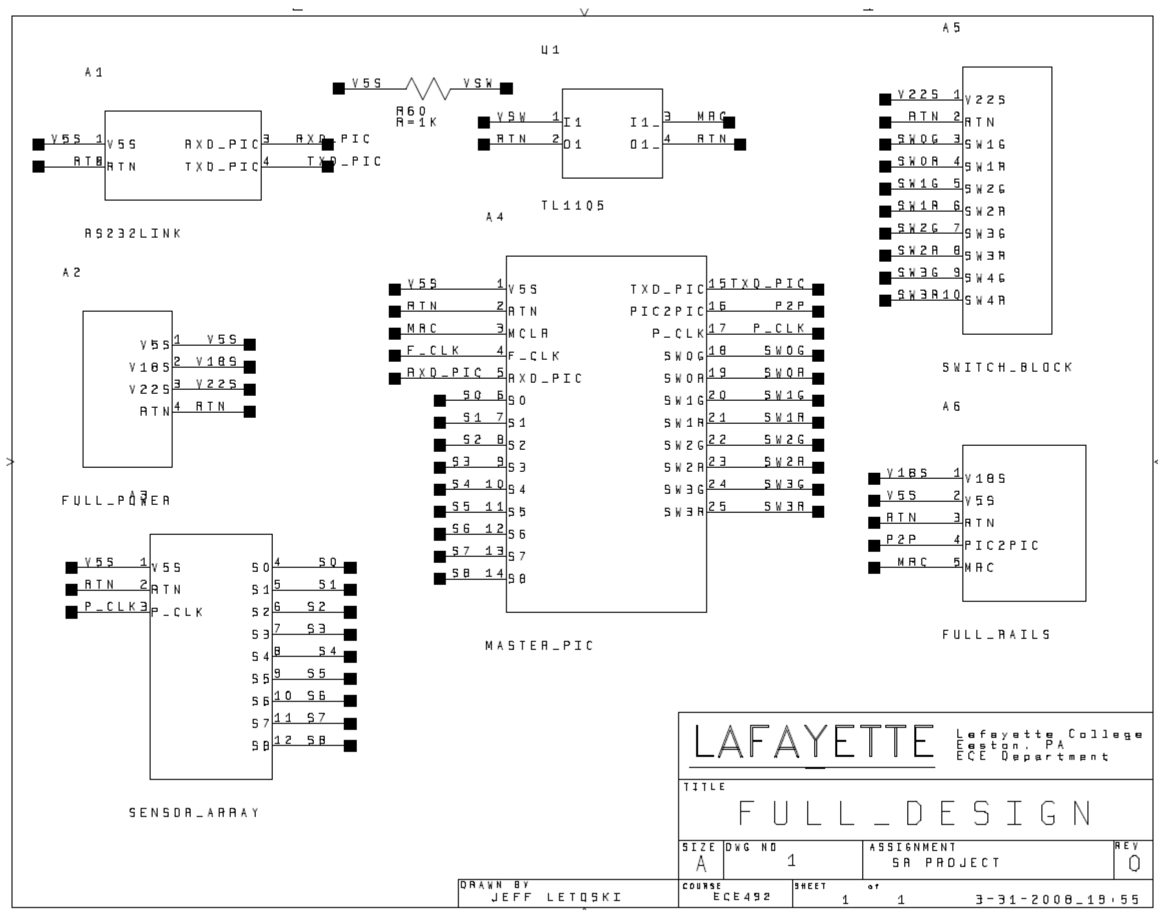
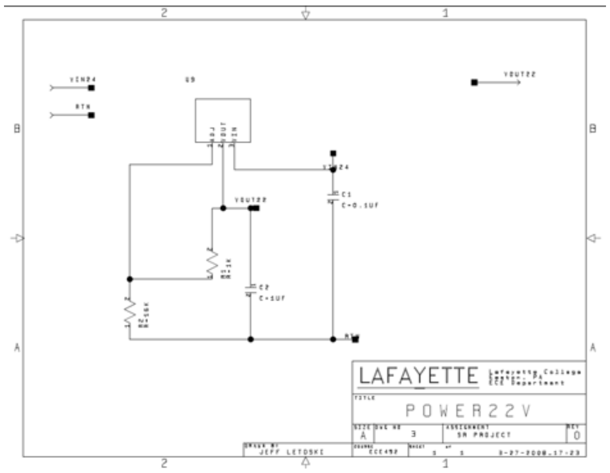
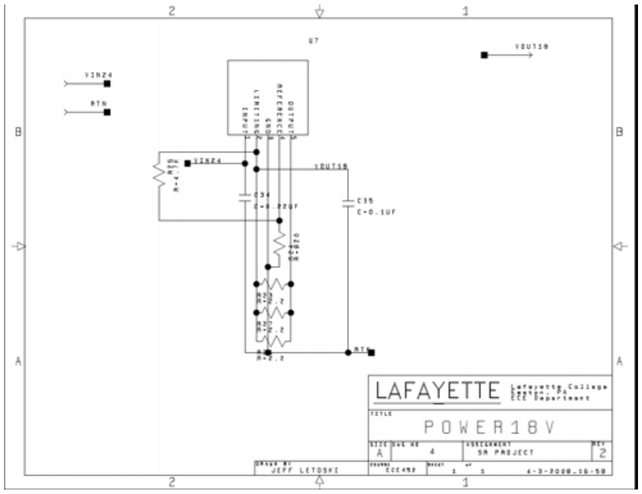


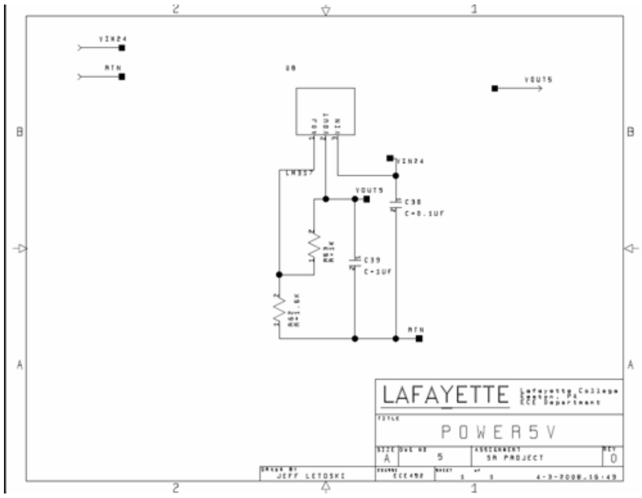
Figure 1: Block Diagram of the Circuit Board

Above is the block diagram for the whole circuit board, which is at every station. The board includes the circuits for the sensors, rail powering, switches, PICs, power system and RS232 linking. All the schematics for the subsequent circuits are attached.

Power circuit







Figures 2-a.) Power Circuit for 22V
b.) Power Circuit for 18V
c.) Power Circuit for 5V

The purpose of this circuit is to take the 120V AC, 60Hz source and transform it into an 18, 22 and two 5V sources, which are used by the Rail Switching Circuit, the Rail Power System, the PICs and the Cooling Fan. This can be seen in the figures above.
The circuit consists of 8 resistors, 6 capacitors, 2 LM 317s and 1 LM200.
The measured voltages and currents for the working systems are as followed:

Part	Total Current	Power Dissipated per part (W)
Master PIC	0.16	0.256
Slave PIC	0.09	0.009
H-Bridge	0.6	0.44
RC- Circuit	0.0176	0.0968
Flip-Flop	0.01	0.001
Sensor Circuit	0.0045	0.2025
Fan	0.1	2.4

			Voltage Drop	Power Dissipated per part (W)
L200-5v		0.2645	19	5.0255
LM317-22v		0.0176	2	0.0352
LM317-18v		0.6	6	3.6

Table 1 – Power and Current Values

There are two parts that dissipate more than one watt of power. In terms of heat, the L200 will produce more than one watt so we needed a fan and heat sink to keep within operable temperatures. In terms of EMI, we found a clause that exempts us to worry about power dissipation because this system is for public transportation.

RS232 circuit

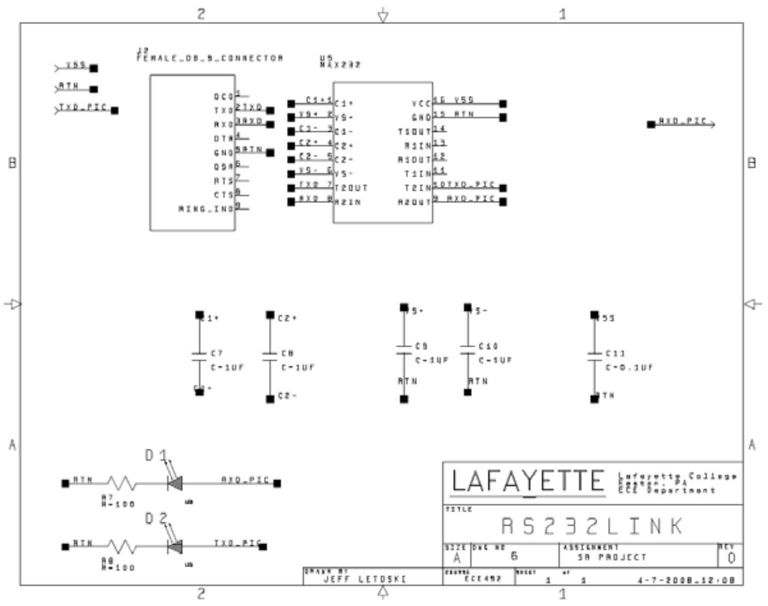
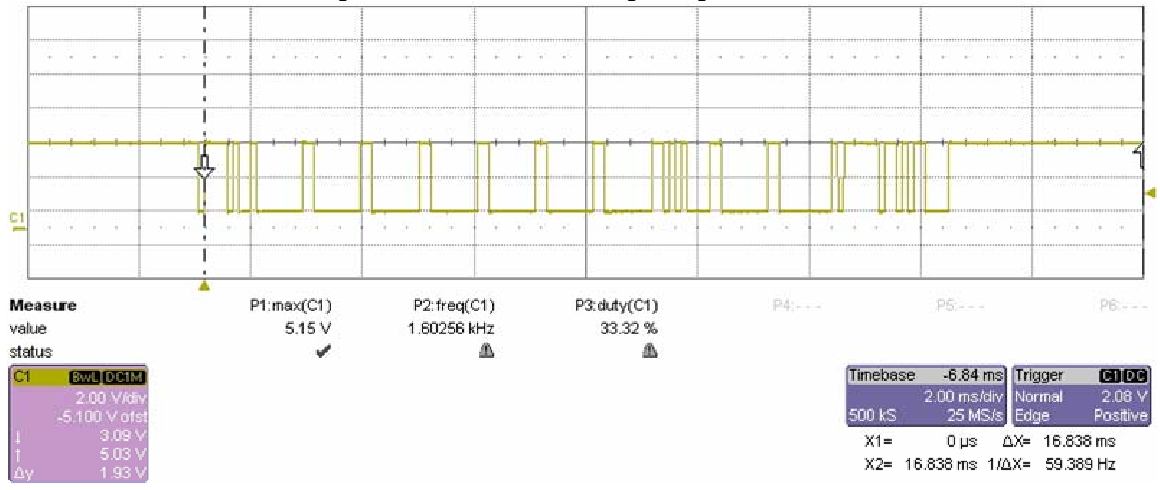


Figure 3: RS-232 Linking Diagram


```
Measure      P1:max(C1)   P2:freq(C1)   P3:duty(C1)   P4:- - -   P5:- - -   P6:- - -
value        5.15 V       1.60256 kHz   33.32 %
status       ✓            ⚠             ⚠

C1   BwL DC1M                     Timebase   -6.84 ms   Trigger   C1 DC
         2.00 V/div                          2.00 ms/div Normal   2.08 V
     -5.100 V ofst                500 kS     25 MS/s    Edge     Positive
↓        3.09 V                     X1=         0 µs    ΔX=   16.838 ms
↑        5.03 V                     X2=  16.838 ms  1/ΔX=   59.389 Hz
Δy       1.93 V
```

Figure 4: RS-232 Signal

The purpose of this circuit is to connect the RS232 cables from the track system onto the board and to limit current in the circuits. The circuit consists of an MAX232 chip, 5 capacitors, 2 diodes and 2 resistors. The circuit converts the RS232 voltage levels to TTL voltage levels. As seen in Fig. 4, the signal is a serial binary signal that is at a rate of 9600 baud/s. The condition that produced this signal was on Station 5, where a rail power was changed to speed 5. The scope is connected to the Rx pin of the PIC.

PIC circuit

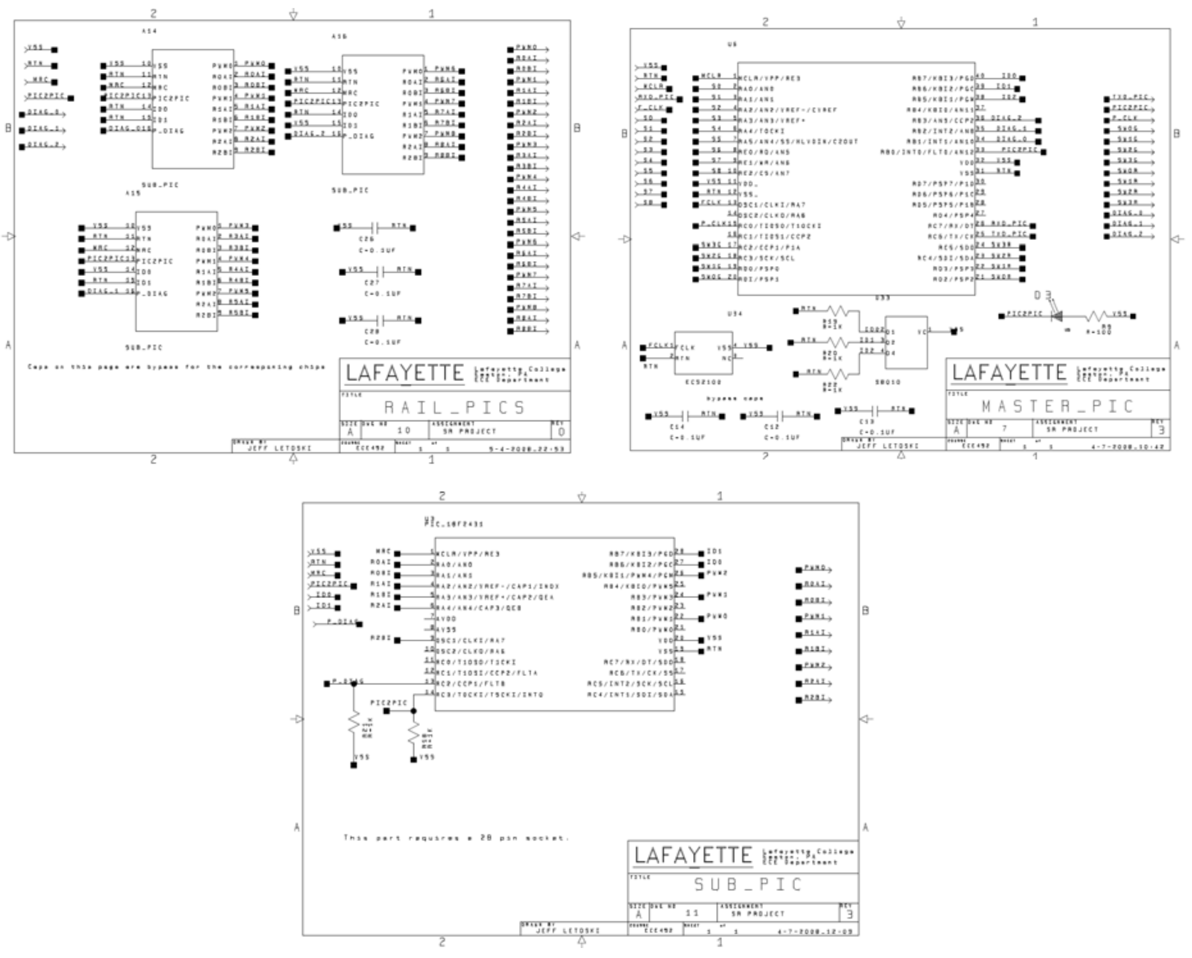


Figure 5: All PIC Diagrams

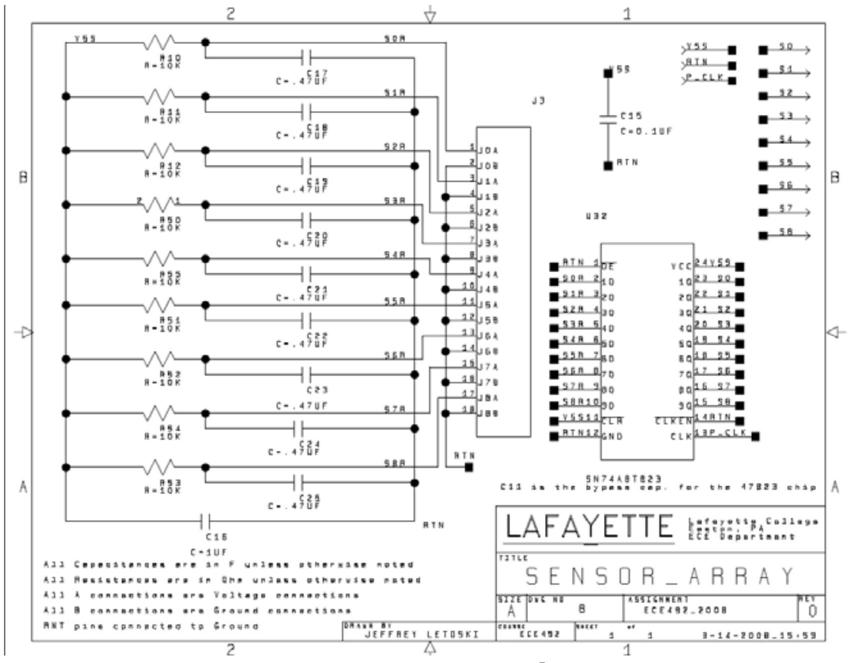
The purpose of this circuit is to make sure that the PICS are all working. This circuit includes a S8010 chip (rotary switch), a ECS2100 chip (clock), the 4 PICs (3 Slave PICs and 1 Master PIC), 3 capacitors, 4 resistors, and 1 diode. The Master PIC controls the switches according to the input sent by the computer and sends back sensor information when sensors are triggered. The switches are control by outputting 5.6ms pulse. The switch tasks are stored in a queue and executed on a FIFO basis. The Master PIC also sends the rail information to the slaves. The slave PICs output PWM signals based on this information. The duty cycle of the PWM signals are changed incrementally/decrementally from the current PWM to the desired PWM signal. The following is the last date modified, size and name of the PIC code used in the project:

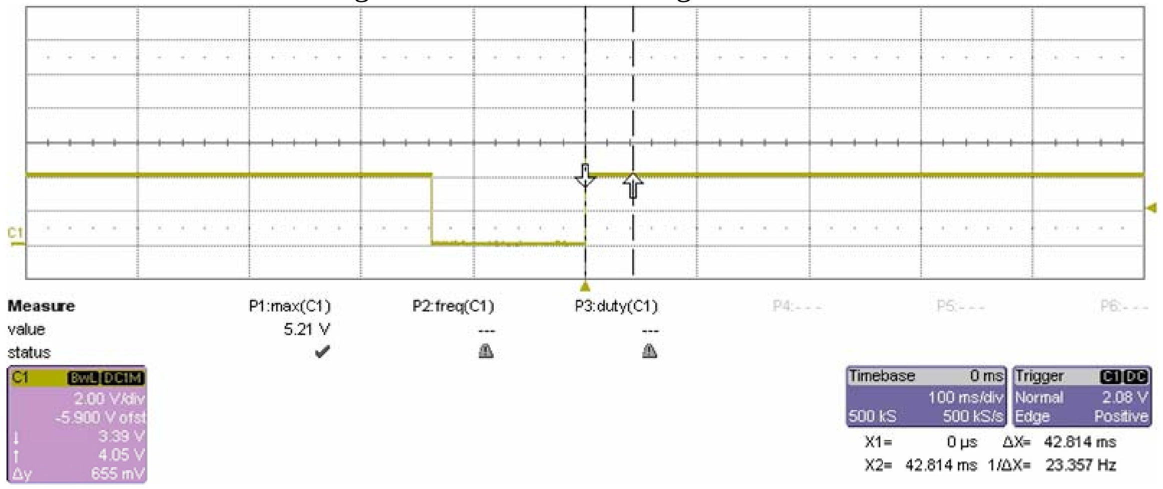
i) MasterPic.c 4/29/2008 10:55PM (16,216 Bytes, checksum: 918DA73FC26BE5D1197DE761FB86DA31)
ii) MasterPic.h 4/29/2008 10:56PM (2,177 Bytes, checksum: 44B7DC039EF02417DEFEB2FD14423CF2)
iii) SlavePic.c 4/29/2008 10:55PM (6,634 Bytes, checksum: 26AFB28675E9E1AB8C9F1EF1E19A5045)
iv) SlavePic.h 4/29/2008 10:56PM (1,148 Bytes, checksum: E82619DBB480269F94C215DCAE9F0321)
v) MasterPIC.hex 4/29/2008 10:56PM (9,408 Bytes, checksum: 541CD91D5507F6C30E2CA46891157C1C)
vi) SlavePIC.hex 4/29/2008 10:56PM (7,024 Bytes, checksum: 6C674FDF161AC865C9278B0D2391FF60)

The steps required to burn the above code is the following:

1) Compile code with the PIC IDE
2) Obtain a .hex file
3) Open up the Xeltek 280 program
4) Choose correct PIC device from the device list
5) Load your .hex file
6) Put PIC in socket
7) Fasten with mechanical arm
8) Click on erase
9) Click on program
    a) If auto is set then click on auto

Sensor circuit


LAFAYETTE
Lafayette College Easton, PA ECE Department
SENSOR_ARRAY
SN74ABT823
JEFFREY LETOSKI
ECE492_2008
3-14-2008_15:59
All Capacitances are in F unless otherwise noted
All Resistances are in Ohm unless otherwise noted
All A connections are Voltage connections
All B connections are Ground connections
ANT pins connected to Ground
Figure 6: Sensor circuit diagram


```
Measure   P1:max(C1)   P2:freq(C1)   P3:duty(C1)   P4:---   P5:---   P6:---
value     5.21 V       ---           ---
status    ✓            ⚠             ⚠

C1  BwL DC1M                 Timebase        0 ms   Trigger     C1 DC
        2.00 V/div                   100 ms/div     Normal     2.08 V
      -5.900 V ofst          500 kS    500 kS/s     Edge     Positive
↓          3.39 V
↑          4.05 V            X1=        0 µs    ΔX=    42.814 ms
Δy         655 mV            X2= 42.814 ms    1/ΔX=   23.357 Hz
```
Figure 7: Sensor Signal

The purpose of this circuit will interface with the reed switch imbedded in the track as well as the PIC to let the user know where the train is. The circuit consists of 1 SN74ABT823 (flip-flop), 10 capacitors, and 9 resistors. Using a 5V source this circuit has been designed to output TTL voltage levels to the PIC indicating either the presence or absence of the train. As you can see in Fig. 7, when a train passes over a reed switch the switch acts as a short, causing little or no voltage drop across it. When the train is not present the reed switch appears as an open circuit and thus all the voltage drop occurs across it. This voltage drop is the output to the PIC. The condition that produced this signal was on Station 5, where a tram went over sensor 2. The scope was connected to the third pin of the PIC where the sensor 2 output is connected.

Switch circuit

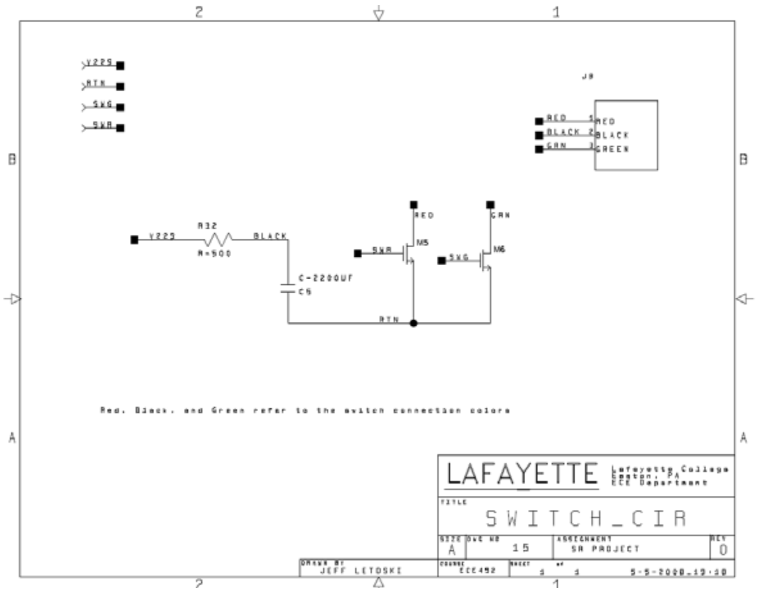
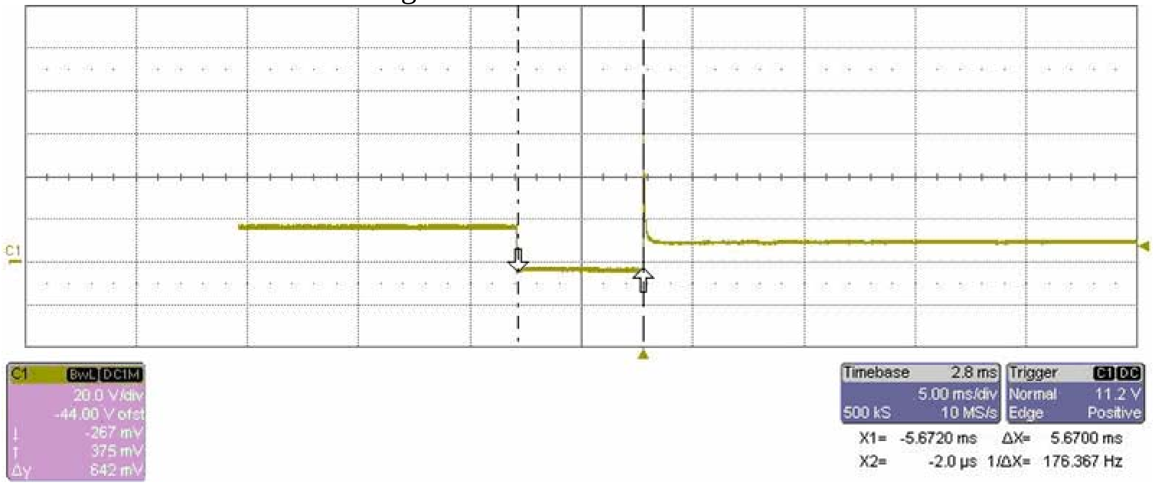


Figure 8: Switch Circuit



```
C1  BwL DC1M
    20.0 V/div
    -44.00 V ofst
↓   -267 mV
↑   375 mV
ΔY  642 mV

Timebase   2.8 ms   Trigger   C1 DC
        5.00 ms/div  Normal   11.2 V
500 kS   10 MS/s     Edge     Positive

X1=  -5.6720 ms    ΔX=   5.6700 ms
X2=     -2.0 µs  1/ΔX=  176.367 Hz
```

Figure 9: Switch Signal

The purpose of this circuit is to control the train's path on the system. The circuit consists of 2 transistors, 1 capacitor and 1 resistor. The 22V source will charge the capacitor in a period of 4 seconds to the desired voltage of 22V. When the switch needs to be changed, a 5V pulse will be sent to the gate of the appropriate transistor and will discharge the energy of the capacitor across the switch. Figure 9 shows that the capacitor is discharging onto the switch, which takes 5.6 ms. This discharge time is because of the pulse from the PIC. After the discharge, the capacitor charges up; this should take about 4s. We can't really see the discharge as it would be seen on the inductors because we have to plug in the scope in serial with the output of the board and the coils. This is because for the discharge to happen there needs to be the load of the coils on the line. Thus the picture doesn't represent the signal exactly as it would be on the coil but rather provide a framework to extract the very essence of the switch discharging signal. The reason the voltage scale was so stingy, was because one could see the gradual charging of the capacitor. The condition that produced this signal was on Station 5, where a switch 0 was turned. The ground of the scope was connected to the ground of the board and the other end of the scope was connected to the left coil of switch 0.

Rail powering circuit

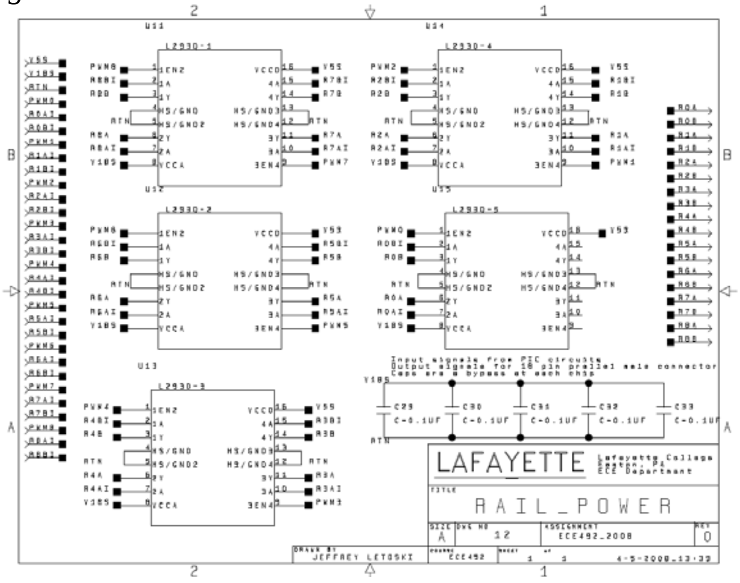
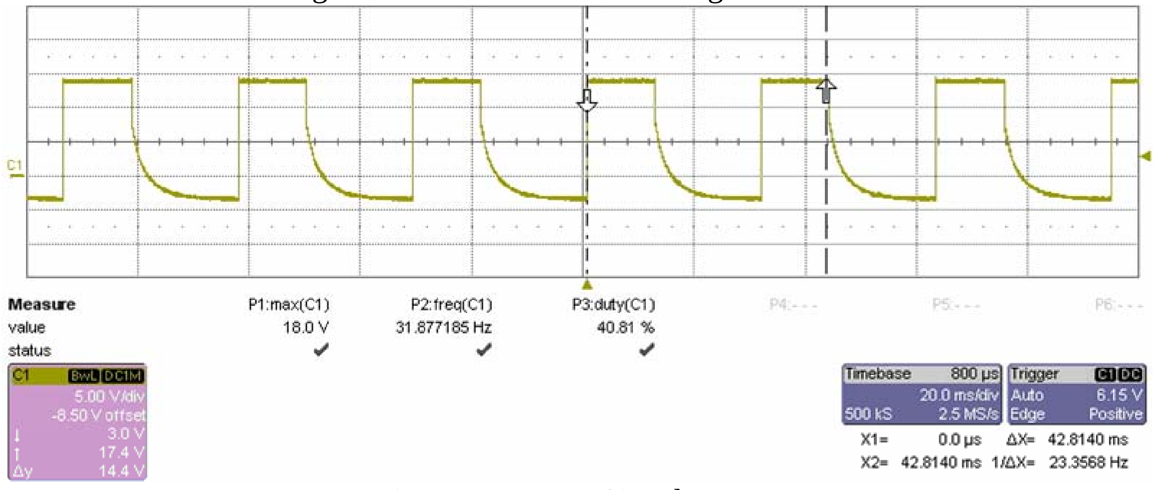


Figure 10: Rail Power Block Diagram



```
Measure    P1:max(C1)    P2:freq(C1)     P3:duty(C1)
value      18.0 V        31.877185 Hz    40.81 %
status     ✓             ✓               ✓

C1  BwL DC1M
5.00 V/div
-8.50 V offset
↓ 3.0 V
↑ 17.4 V
Δy 14.4 V

Timebase 800 µs    Trigger C1 DC
20.0 ms/div        Auto  6.15 V
500 kS  2.5 MS/s   Edge  Positive
X1=  0.0 µs         ΔX=  42.8140 ms
X2=  42.8140 ms   1/ΔX=  23.3568 Hz
```

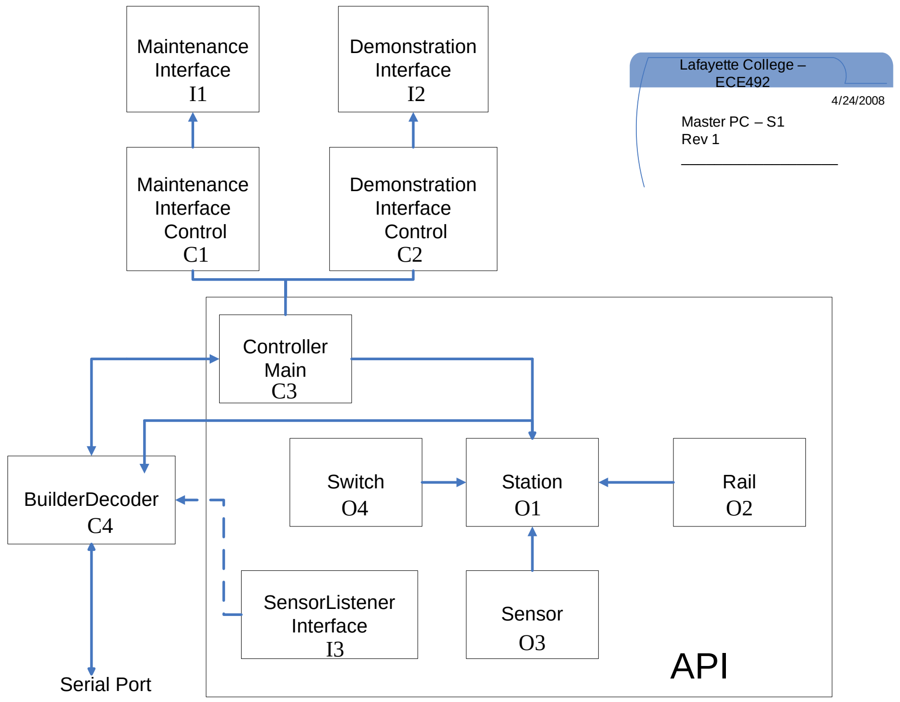
Figure 11: PWM Signal

The purpose of this circuit is to amplify a PWM signal from the PICs and then use that to set the rail speeds and direction. The circuit consists of 5 L293D's (h-bridges), 8 capacitors, 11 resistors, and 11 diodes. The PWM signal is set to the enable input pins of the h-bridges. The polarity is controlled by the 5V source from the PIC to the input pins of the h-bridges. The result is a modified PWM signal which is high when the rail should be powered and when it's low the output is in a high impendence state. The condition that produced this signal was on Station 5, where a tram went over sensor 2. The scope was connected to the third pin of the PIC where sensor 2 output is connected. The condition that produced this signal was on Station 5, where the speed was set to 5. The scope was connected between ground of the board and the top rail.

Octopus:

The Octopus 550 is a Peripheral Component Interconnect (PCI) to Serial-RS232 port expansion card. It will be placed in the Master PC and supports up to 8-serial ports. This card will be used to connect to each of the stations in parallel via serial cables. There are no drivers for this device. To install the Octopus on LINUX, please go to the following web-site for details: http://support.lavalink.com/index.php?id=471

PC:


Maintenance Interface I1
Demonstration Interface I2
Lafayette College – ECE492
4/24/2008
Master PC – S1
Rev 1
Maintenance Interface Control C1
Demonstration Interface Control C2
Controller Main C3
BuilderDecoder C4
Switch O4
Station O1
Rail O2
SensorListener Interface I3
Sensor O3
API
Serial Port

To log in onto the LINUX machine, one should use the following login (kissingt) and password (ece492). The last modified date for all the source code is 4/29/08 and the total size of the code is 146.01 KB. The OS version is linux lisa 2.6.22.9a. The following is the last date modified, size and name of the JAVA code used in the project:

i) BuilderDecoder.java 4/29/2008 11:05PM (14 KB, checksum: 419DEECF6F7ED0B578EC733969E60117)

ii)BuilderDecoder.class 4/29/2008 11:05PM (13 KB, checksum: 33685EC6FD8300452950A1570C08D39D)

iii) ControllerMain.java 4/29/2008 11:05PM (7 KB, checksum: `C8052907EF65F35A5E13E97BE86B7CDE`)
iv) ControllerMain.class 4/29/2008 11:05PM (8 KB, checksum: `DBCC28BD05D30B25BE0F031FBE2245C4`)
v) ControllerMainTest.java 4/29/2008 11:05PM (3 KB, checksum: `64ABADC7F0D012997D3E4393E2A5BAA6`)
vi) ControllerMainTest.class 4/29/2008 11:05PM (3 KB, checksum: `0CD3A7527F8B8A1AC26EF362EC28864A`)
v) DemoTest.java 4/29/2008 11:05PM (10 KB, checksum: `97BB8916A01D959203BFEBF4E2B9E476`)
vi) DemoTest.class 4/29/2008 11:05PM (5 KB, checksum: `EF9AAEBC7C78CEE0B0B41F4B3132685B`)
vii) MaintenanceInterface.java 4/29/2008 11:05PM (60 KB, checksum: `E535F08ECC4FA9B1370A187D99540A0A`)
viii) MaintenanceInterface.class 4/29/2008 11:05PM (36 KB, checksum: `963E767564CAB4DE0F545533E8E4FED9`)
ix) Rail.java 4/29/2008 11:05PM (4 KB, checksum: `F2A7782D6DD61A2C2876BD7635672811`)
x) Rail.class 4/29/2008 11:05PM (4 KB, checksum: `7C81B28C36728CD500774F1FD78EFCC1`)
xi) RailTest.java 4/29/2008 11:05PM (2 KB, checksum: `95374C1F748E1CFB477B7C7F5411788E`)
xii) RailTest.class 4/29/2008 11:05PM (3 KB, checksum: `A227265274CEA6B15A2C705A21C01898`)
xiii) Sensor.java 4/29/2008 11:05PM (2 KB, checksum: `647DBDF3CC1934A74FCD94E7A1DC081F`)
xiv) Sensor.class 4/29/2008 11:05PM (2 KB, checksum: `526F500E4A488CA356F4093A2E3653E5`)
xv) SensorTest.java 4/29/2008 11:05PM (1 KB, checksum: `B125B6D3BDCAE90C95EB19849E5C4F84`)
xvi) SensorTest.class 4/29/2008 11:05PM (2 KB, checksum: `FC3442386C872FF1CAC593BA7DCB27DC`)
xvii) SensorListener.java 4/29/2008 11:05PM (1 KB, checksum: `214DCC47291DB6D7CE390D4BDD6737E4`)
xviii) SensorListener.class 4/29/2008 11:05PM (1 KB, checksum: `96602429764A92DE67A23EA1EA3E4C8F`)
xix) Station.java 4/29/2008 11:05PM (11 KB, checksum: `EAC523022690A5230202A84BD85ED3C9`)
xx) Station.class 4/29/2008 11:05PM (8 KB, checksum: `84017D47D0F11D924FBB860537F321C0`)
xxi) StationTest.java 4/29/2008 11:05PM (2 KB, checksum: `85DF8BC4DC2679911DB7A8F3E826C509`)
xxi) StationTest.class 4/29/2008 11:05PM (2 KB, checksum: `9788B4FFAFA46D03B0F7E2CC60862709`)
xxiii) Switch.java 4/29/2008 11:05PM (5 KB, checksum: `0FA1D1D5B7A06102ACC2FA091CAE3033`)

xxiv) Switch.class 4/29/2008 11:05PM (3 KB, checksum: `424EF66C2FA8C36BDDAC8C68EB919AAC`)
xxv) SwitchTest.java 4/29/2008 11:05PM (3 KB, checksum: `C8E816A48830B6F3C2B6D19F4ACC9E35`)
xxvi) SwitchTest.class 4/29/2008 11:05PM (4 KB, checksum: `EDDAE02612465719DB9AC7029F3BE800`)
xxvii) demo.GIF 4/29/2008 11:05PM (6 KB, checksum: `1B5144EBD5826256BF478AB709F6B2D5`)
xxviii) left_down.GIF 4/29/2008 11:05PM (1 KB, checksum: `2230345BA2FA07E3D20087C099EEE37F`)
xxix) left_straight.GIF 4/29/2008 11:05PM (1 KB, checksum: `E55B5CA781AC01CC87CFF86F62E924D1`)
xxx) left_up.GIF 4/29/2008 11:05PM (1 KB, checksum: `4F4F57A91F9E920C44AD3C9AEF4457EC`)
xxxi) right_down.GIF 4/29/2008 11:05PM (1 KB, checksum: `5F4B48B4DE3CF6EAF9E10072B38AE752`)
xxxii) right_straight.GIF 4/29/2008 11:05PM (1 KB, checksum: `63942CCCEAD2315E7AF20FBE97A2DF39`)
xxxiii) right_up.GIF 4/29/2008 11:05PM (1 KB, checksum: `E0AF4C2792B2B6798DD872EE847518BE`)
xxxiv) tracks.GIF 4/29/2008 11:05PM (4 KB, checksum: `FF31EBF547447B67F066D1F1AF1CAB5F`)
xxxv) trackConfig.xml 4/29/2008 11:05PM (15 KB, checksum: `26AFA84D8177AEF6AC09AC64395A3885`)

User Interface:

The maintenance GUI is for a train operator who would use it to monitor the trains and to control their movements and speed. It will also include an emergency stop button that the operator could press if there is potential for disaster. The maintenance GUI will include combo boxes to control the speed of the train on each rail segment. In addition it will include buttons for each switch and radio buttons for detecting sensor action. The demonstration GUI includes a log to inform the user of the positions of the trains. In addition there are buttons to start the demonstration and to stop the trains.

Demonstration Mode:

The demonstration control logic will choose which stations the trains will be running, what speed they will be running at, and the direction they will be moving. The logic will also have a failsafe, which would detect an error if for example, trains are going too fast, and then the whole system will pause. The operator can also at his discretion stop the whole system. This also sends the sensor signals to the GUI. For the demonstration mode, the following has to be done: all stations have to be reset and the PC has to be turned on for the train to move.

Maintenance Mode:

The maintenance control logic will send method call backs to control the track layout, including switches, sensors, and speed control at each track segment.

Builder Decoder:

The BuilderDecoder class organizes information from the ControllerMain class about the switches and rails and transmits it through the serial port. In addition it receives sensor information from the SensorListener Interface and sends it to the ControllerMain class.

API:

The ControllerMain class will parse the XML document to get characteristics for each station. This will create a map for each station. It sends the values for each station to the control classes. It receives data from the BuilderDecoder class which sends sensor values to the control classes. In addition it sends information from the XML document to the Station class.

The Station class gets information from the Switch, Sensor, and Rail class to build each station.

Expandability:

If another station, rail, sensor, or switch is added to the track layout, we will have to modify the XML documents so as to incorporate those changes.
If a new station is added, then the following is modified:
<NAME>**Station 1 - 3rd Street Terminal**</NAME>
<NUMRAILS>**7**</NUMRAILS>
<NUMSENSORS>**6**</NUMSENSORS>
<NUMSWITCHES>**2**</NUMSWITCHES>

If a new rail is added, then the following is modified:
<STATIONNAME>**Station 1**</STATIONNAME>
<NUM>**1.0**</NUM>
<LEFTCON>**-1.0**</LEFTCON>
<RIGHTCON>**1.1**</RIGHTCON>

<ROW>**1**</ROW>
<LENGTH>**300**</LENGTH>

The parameters LEFTCON and RIGHTCON indicate which rail number are connected to the left or right of this rail. The (-1) means that there is no rail connected to it.

If a new sensor is added, then the following is modified:

<STATIONNAME>**Station 1**</STATIONNAME>
<NUM>**0**</NUM>
<RAILNUM>**1.0**</RAILNUM>
<RAILPOS>**RAIL_MIDDLE**</RAILPOS>

The RAILNUM parameter indicates which rail the sensor is connected to and the RAILPOS parameter indicates on what part of the rail is the sensor.

If a new switch is added, then the following is modified:

<STATIONNAME>**Station 1**</STATIONNAME>
<NUM>**0**</NUM>
<LEFTRAIL>**1.1**</LEFTRAIL>
<RIGHTRAIL>**1.6**</RIGHTRAIL>
<LEFTPOS>**RAIL_LEFT**</LEFTPOS>
<RIGHTPOS>**RAIL_MIDDLE**</RIGHTPOS>
<DIRECTION>**RIGHT**</DIRECTION>
<POLARITY>**NORMAL**</POLARITY>

The LEFTRAIL and RIGHTRAIL parameters indicate which rail numbers are connected to the left or right of this rail. The LEFTPOS and RIGHTPOS parameters indicate which the position the switches are on the rail. The DIRECTION parameter indicates the direction in which the tram moves and the POLARITY parameter indicates whether the switch is turned or not.

Errors

The software checks for the following errors:

1) If there is a small syntax error in the XML, then it is a JDOM exception and an XML syntax error gets print out.
2) If the code is trying to read the XML file and it isn’t there, an IOException error gets print out.
3) If the trams are in the wrong place, the following error message gets print out, “The tram is in the wrong place, please put it in its appropriate position.”

Troubleshooting

Follow the Acceptance Test Plan for maintenance procedures and if all test pass then the system is functional.

If the ATP fails then check all critical voltages to see what circuit is malfunctioning. Replace circuit on the board as necessary. If multiple circuits have failed then use a new circuit board.

If the ATP fails and all the critical voltages are correct then review cables and software.

Board Maintenance:

1. Check if the PIC chips are in the sockets, but make sure they are not fully fitted; otherwise they won’t function properly.
2. Check if all necessary RS232, power, and ribbon cables are connected.
3. Check the board fan functionality.
4. Clean any excess dust inside box.

Computer and Cable Maintenance:

1. Check for any damaged cables.

Track Maintenance

1. Check tracks, sensors, or switches for any failing components.

Safety Warnings

1. Keep your fingers away from the fan.
2. Don’t touch the voltage regulators if the fan is not turned on.
3. Food or drink is not allowed.
4. In case of emergency, call 911.
5. Use care when handling equipment power plugs.
6. Never solder a live circuit or touch one with your bare hand.

Manufacturability

To make a new board, use a 4-layered board with minimum spacing between the traces equal to .010 inches. The document that is needed as a guide to put components on the board is the Board_Layout.pdf file that is attached. The details of each circuits are also attached in the Final Schematics.pdf. The minimum parts that are needed are included in the parts list.

Parts List

Parts list will contain the components of the system and the quantity needed to make a circuit board.

Test Procedures

The tests are located in the Acceptance Test Plan (ATP).

Tables for critical voltages and signals

The values are located in the Acceptance Test Report (ATR) and the Quality Assurance (QA) report.

Interface Control Document

The detailed connection scheme is shown in the Interface Control Document (ICD).