Interface Design

## Port Pins on Micro (PT6, PT1, PT2, PT3) (Included in Interface for Understanding)

The basic function of the port pins is to provide 3 states. We use 4 bits since our use constitutes an H-bridge which has four connections which need to be simultaneously operated and any combination using 2 bits proved to be insufficient.

The following states define the direction controlled mechanism which becomes evident in a next couple of pages.

|  |  |  |  |
| --- | --- | --- | --- |
| Port Pins | Move Forward | Move Right | Move Left |
| PT 6 | 1 | 0 | 1 |
| PT 1 | 0 | 1 | 0 |
| PT 2 | 1 | 1 | 0 |
| PT 3 | 0 | 0 | 1 |

## L298N H – Bridge

The L298N Dual H-Bridge motor driver has six inputs and two outputs.

The two outputs are connected to the two motors with a common ground.

Four inputs control the switches in the H-bridge and based on the configuration that is output by the active high port pins these switch states are decided which in turn decide the direction of the robot car.

There are two ENABLE pins which decide whether either of the motor is permanently enabled or disabled. These have to be coupled with the 5V logic pin of the H-bridge outputs of which are regulated by an RC connection on the bridge itself. ENABLE pins are in AND relationship with four switches, therefore only two PWM signals are required for two motors.

There is a separate Voltage source for the H-bridge which routes separately for the motors allowing us to rate the motors at a higher voltage source as compared to the logic pin as well as the Microcontroller specification.

There is a voltage stabilizer that convert higher voltage source (VCC for motor) to 5V for logic pins. Voltage stabilizer is ENABLE by a jumper-wire that allows user to supply motors with voltage as high as 12V without requiring external 5V source for logic pins.

*Note: The ground for the Vcc and the logic pin as well as microcontroller has to be common.*

## DC/PWM Controlled Motors

We use two separate motors to control two driving wheels separately. In this way, we save room for a complex mechanical design, and have a more precise control of the motion.

The direction control mechanism of the car robot mechanism is as follows:

1. To move right, left wheel moves forward and the right wheel backwards.
2. To move left it’s the opposite of step 1.
3. To keep moving forward the wheels need to be in phase and running at the same velocity.

The above is achieved using the L298N H-Bridge which is controlled by the Port Pins PT6, PT1, PT2, and PT 3.

## IR Receiver:

An IR Receiver with two PWM outputs is used to send instruction signals from the user-end to the interfacing device i.e. the microcontroller.

The transmitter consists of 6 pins two of which are required to power up using a 5V DC source.

The remaining four pins are pairs each sending a different signal corresponding to the signal sent by the RF Transmitter.

Based on the duty cycle of the waves generated by the pins on the IR Receiver, they are sampled to a certain range of inputs to provide basic two functions: one for changing the direction, the other for enabling any PTT pins in turn bringing the car to a stop or setting it in motion.

One thing needs to be noticed is that the last two pins require a diode added between of them to avoid signal distortion.

The signal distortion is caused by the build-in capacitor in the transmitter. Once the transmitter sends signal, the capacitor gives PWM varies fall time (not a square wave). PTT pin of our microcontroller is only able to give a 0 or 1 value, therefore a longer fall time gives significant more PTT high count. Adding a diode between 2 pins allow us to filter out most of the signal distortion.(Reference Schematic is attached on this document.

## RF- Transmitter:

An RF Transmitter controls the above mentioned IR Receiver. The amplitude of the signal sent by the controller communicated to give the corresponding duty cycle of the PWM wave form which we sample on two ATD pins and feed data into the micro to control a couple of if statements in the TIM sub interrupt routine to constantly read input and aid in setting port pins.

## 2x16 LCD with 16-pin single-row header

The SPI Module of the Microcontroller is used to interface an LCD to the PCB board to help aid the user in entering a 3-bit password sampled using ATD which is converted internal into its ASCII value. The password module is reset and the Enter Password screen keeps reappearing until the user has access to the right password upon which the timer sub interrupt routine is enabled in turn enabling the PTT pins.

After which there is an option for sampling the ATD using a push-button and potential meter to play any of the 3-tracks.

## ATF 22V10C 15PU

An ATF chip is used as a shift register to interface the LCD to the Microcontroller for displaying the corresponding bits onto the LCD.

This chip has an advantage over the GAL22V10 of a better constant supply of bits.

A Left Shift register is routed to pins 14-21 of the ATF chip which are connected Most Significant bitwise to pins of the LCD.

Radioshack Pushbutton

Pushbutton is served as ENTER for user to confirm their action. It is connected to AN6 pin which is sampled every RTI routine.

Potential Meter

It is a Linear Potential Meter which allows user to change their input value smoothly. Potential Meter is used as 0-9 number input from user. It is sampled at AN0 pin. We design a function -- convert\_atd() in order to convert value stored in ATDDR0H to a ACII value between 0-9. This function is called constantly for user to see updates of their input.

Internal structure design

Because all the wires have to be fit into the limited space in the car, we have to design 5 PCB for distinct functions. The PCB on the back of the car is mounted with Microcontroller. PCB in the middle section of the car serves as a Power Panel which collects all the Vcc and Grounds. PCB on the top of the car is soldered with push-buttons and potential meter which communicates with user. PCB on the front of the car is designed for the complex wiring between LCD and ATF 22V10C chip. PCB on the bottom of the car is used for powering the tail-lights. Carefully designed PCBs significantly reduced the number of floating-wires and bad connections. It also reduce the noise produced by the wires.

Microcontroller Resource Utilization

The modules that we significantly utilized of the MS12C9SC32 are the ATD, TIM, PWM, and SPI.

## Timer Module

The timer module on the Microcontroller is used by creating a timer sub-routine which provides an interrupt at every 0.01 mille-seconds by setting the TIE\_C7 value to the corresponding value.

Every time the timer subroutine is called which is at the interval specified above the AN4 and AN5 pins are sampled respectively. AN4 and An5 pins are enabled as digital input instead of analog input to give us a one bit value. The AN4 and AN5 pins sample the outputs from the IR Receiver in the following way.

Algorithm for the sampling within the Timer Module

Sample\_atd\_convert () We only use function to determine directions.

{

* Read PWM Signal from the AN4 pin
* Use a counter to see for how long the PWM cycle is high in turn measuring the PWM duty cycle.
* Based on the counter assign particular speed to the output pin by enabling the corresponding PT pins
* Read PWM Signal from the AN5 pin
* Use a similar counter measuring the time of counts the PWM signal is high.
* Assign particular direction to the car by enabling corresponding PT pins.

}

## Analog to Digital Module

This module has been explained in the Timer module above. The sampling on ATD is done at every Timer interrupt, the algorithm for which has been specified. In this case, the PORT AD is set to give only digital input (similar to AN6 for push bottom).

The only instance where the ATD is used to sample in exclusion of the timer module is when TC7 is disabled i.e. at the beginning of the program where the password is asked. Here the potentiometer on AN0 is sampled to convert it into ASCII using the user-defined function convert\_atd()(potential meter is only sampled each time we call convert\_atd() function), which samples up to 3 bits locked with the PTAD6 push-button until the combination specified is right in order to enable TC7 to start the timer routine which is disabled otherwise Every time user turns the potential meter, ATDDR0H will give a different value between 0 to 255. Therefore, we are able to choose 10 threshold ranges in order to assign number between 0-9.

## Pulse Width Modulation Module

Music – The major function that our PWM aids is to provide music which is output on PT0 using a certain amount of initializations in the code which allow the frequency to be heard by the human ear.

An array of sounds, Do, Re , Mi, Fa , So , La , Si ,D0 (Sa Re Ga Ma Pa) are the basis of every music hence we sampled the sounds at different PWM’s to generate these 7 segments which in turn form the basis for three arrays of music that we created using different combinations of these above sounds.

By controlling PWMDTY and PWMPRCLK, we are able to play 7 music tone with extremely low error. For example, Mi and Fa has 6.1% difference in frequency and we can tune that into 5.95% which only have a 0.15% error.

Serial Peripheral Interface Module (SPI)

This module is used to interface our LCD display using the ATF/GAL 22V10 shift register to display different characters corresponding to their ASCII values which are provided.

Two functions i.e. pmsglcd and print\_c are the functions which are used to display characters on the LCD module which utilize SPI initializations to interface the LCD Display.

## RTI Sub Interrupt

The RTI is used to sample one push-button on PTAD6 that we have attached externally (not on the docking module) for locking the password bits every-time they are entered. This sampling occurs at every 2.048 ms.

Software Narrative

Software can be concluded into three catalogs: Input sampling, output generation and UI.

**Input sampling:**

**Direction signal from transmitter** is a PWM waveform and is sampled from AN4. We set AN4 pin to digital input and we compare port value with logic high every 0.01ms. In order to precisely sample a 20ms period input signal, we collected 2000 samples and save the number of high value into a counter. By comparing the counter with certain threshold we can determine the direction of the car.

**Speed signal from transmitter** is also a PWM waveform and is sampled the same way as direction signal. However, speed signal has a period of 5ms. Therefore, we only collect 500 samples before we use the counter.

**Signal from push-button** is sent to AN6. We use RTI to check if push-button is set. AN6 is also enabled as a digital input.

**Signal from potential-meter** is used to collect user input and it is sent to AN0. This time, we use Port AD to complete ATD conversion. We collect the 8bit result from ATDDR0H. We create a function to convert the 8bit value from ATDDROH to an ACII number. Therefore, every time we call the function, it will sample and update the new user input to display on LCD.

**Output generation:**

**Direction outputs** are generated by setting PT1, PT2, PT3 and PT6 to different combinations of logic high and low. This process happens every 20ms when direction signal is sampled.

**Speed output** is generated in a very simple way. When the signal from transmitter ask the car to stop, we set every PT pin to logic low. If the signal from transmitter commands car to run, we assign direction output to PT pins.

**Display output** is shift to ATF device by SPI module. By writing to SPI data register, it is abled to shift out strings to display on the LCD.

**Music output is** generated by creating PWM waveform with different frequencies. We created two arrays to construct music. The first array is functioned to set PWM wave form to a certain frequency by setting PWMDTY0. By doing so we can create seven basic tones that can form any song. The second array determined the bit of the music. It sets time delays between two tones. By cycling two arrays through a for-loop, we are able to send PWM signal with the decided frequency and holding time through PT0 to speaker.

**User Interface:**

**Password** hasto be entered and checked in order to enable TIM module to receive signal from transmitter. After user enters 3-number password, we save user inputs into an array. We compare the array to the correct password. If the user input is incorrect, we ask user to input password again. If the user input password is correct, we process to next step and enable the timer module.

**Press to play** step happens right after timer module is enabled. It’s a while-loop that only breaks if users press push-button (rghtpb=1). User then is asked to **pick a song** by using potential-meter to input a number and pressing push-button to confirm the number. The number is then sampled and used to pick the 3 sets of arrays for 3 songs. Notice that PWM is only enabled if one of the PT pins is sat to logic high. This enables music to be played only when car is moving.

Summary and Conclusions

The project has given us a wide array of application frontiers of knowledge acquired through theory.

# Learnt Objectives

* The most important part of the project is the brainstorming phase where so many different ideas are generated for using the modules provided to create something physically meaningful.
* The application of using a Microcontroller compared to an entire circuit of diodes and resistors shows that the compressed technology is a better means for data transfer.
* Practical Application deviates from theory to a large extent. The error values can only be experimentally determined.
  + - * For example, during the project we thought of incorporating an auto-run function for our robot car, this function requires using the delay function for time which could be based on any interrupt.
      * The accuracy of interrupts cannot be determined to an exact precise value and hence this had to be tested experimentally.

# Time Constraint

* A week more for this project would have instilled in us inspiration to aim for more functions and features since making the robot-car run using the 9S12C32 in itself was a very time consuming task.
* By the end of it, building more functions onto it is a conceivable task given more time.
  + - * We had planned to install a line follower system. This could be done by integrating a sensor connected to the microcontroller which would be sampled using PWM, the only extra requirement would be interfacing the sensor using the SCI module on the microcontroller chip.
      * The second most important function we thought was using the Bluetooth module from any transmitter to move the car in directions based on any input interrupt which would also require SCI interfacing.
* Packaging of the Product
* As explained above our 3D printed model comes close to accurate but we could have spent more time on the design phase which would have led to a perfectly sealed robo-car.
* Also the specifics of the car body could have been made better if the time constraint had allowed us to spend more time on building the model.

# Conclusion

It was one of the best learning experiences of our career which will instill in us capability to build more and better designs in future classes as well as industry.

**Appendix A:**

**Individual Contributions**

**and**

**Activity Logs**

**Activity Log for:** <Siddhant Ekale, 7745-E> **Role:** <Software Interfacing & Delegation>

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***Activity*** | ***Date*** | ***Start Time*** | ***End Time*** | ***Time Spent*** |
| ***Assembling and analyzing available hardware*** | 04/22/2015 | 6:00p.m. | 11:00 p.m | 5 hours |
| Reconsidering hardware specifications and parts order for backup | 04/23/2015 | 6:00p.m | 8:00 p.m | 2 hours |
| Generating Left Shift Register on GAL to output data based on analysed PWM waveforms to shift bits onto the breadboard | 04/24/2015 | 6:00 p.m | 1:00 a.m. | 7 hours |
| Considering problems with IR Remote Chip since bits shifting invariably and reset problems due the GAL chip not sourcing enough current the pins | 04/25/2015 | 8:00 p.m. | 3:00 a.m. | 7 hours |
| Rewriting Software to rout outputs directly without using a shift register instead setting PTT pins as output pins | 04/26/2015 | 6:00 pm | 5 :00 am | 11 hours |
| Timer Subinterrupt used for continuous sampling of signal generation and total circuit simulated using a password lock function  Travel to Indianapolis for replacing burnt H-bridge | 04/27/2015 | 3:00 pm | 10:00 pm | 7 hours |
| Interfacing H-bridge with password locking module enabled | 04/28/2015 | 1:00 am | 4:00 am | 3 hours |
| H-bridge connectivity working on simulated circuit but showing problems when connected as a standalone machine | 04/28/2015 | 6:00 pm | 12:00am | 6 hours |
| PWM signals figured out by Chen Kai, interfacing them to the microcontroller and robo car hardware building assistance | 04/28/2015 | 1:00 pm | 3 : 00 am | 12-14 hours |
| Building Timer Generates only when motion of car is set and disabling all PTT otherwise | 04/29/2015 | 2:00am | 5:00 am | 2 hours |
| Packaging Soldering and Regenration of the entire circuit using manufactured PCB’s to isolate the ground and live wires loose connections causing the micro to reset  Auto-Run Function Testing/ Demo to Prof.Meyer | 04/29/2015 | 7:00 am | 3:00 pm | Around 8 hours |
| Report for Hardware Module/Summary/Conclusions | 04/30/2015 | 12:00 am | 4:00 am | 4 hours |
|  |  |  |  |  |
|  |  |  |  |  |

**Written Summary of Technical Contributions:** <Siddhant Ekale>

*My technical role revolved around 3 aspects of the software design phase: 1) Analyzing the PWM waveform to create a counter 2) Sending the right bits to the PTT pins based on the sampling rate and the frequency 3) Password Module 4) Modulation of every function based on the physical movement of the robot car 5) Assistance in soldering and circuit realization*

1. *Analysis of the PWM waveform*

* *With assistance of understanding the duty cycles from my team mate, I was able to develop a solid algorithm for sampling the PWM signal to create a counter that could sample the signals sent in by the Infrared controlled remote.*
* *The difficulty was maintaining the thresholds since they varied significantly. These could only be debugged by setting breakpoints and then sending signals while the breadboard was connected to the oscilloscope and looking at the counter values after which we narrowed down to three values the PWM waveform could not go beyond for that particular output.*
* *Final analysis could only narrow down the direction counter and not the speed so that still needed some working which my team mate Chen Kai figured out.*

1. *PTT pin activation*

* *Major issue that we were to face was sampling the PTT pins for output to the H-bridge. I first used a left shift register to modify different values by twiddling only 4 bits between pins 13-17 of the 22V10 chip which were connected to the “arms of the H-bridge”, so testing phase included only LED’s.*
* *This worked perfectly after the experimental values based on the signals were studied.*
* *Second issue was thresholds couldn’t always be right because there was some bit shifting hence on suggestion of TA’s the 22v10 chip was discarded for this use and the PTT pins were directly used as outputs.*
* *My contribution involved setting the PTT pins and figuring out that a 2 bit logic will not work even if there were only 3 states since the H-bridge did not permit bridged connection.(drawback of the L298N H-bridge)*

1. *This has been by the far the major contribution I have given since this was my individual effort completely. A password module had to be set to activate the robot car. Since we were already sampling in the Timer module the password module was rendered useless in the beginning until I thought of disabling PT7 and running the module before the infinite main for loop and then enabling TC7. The password module involved sampling the ATD values, shifting those signals on the LCD by converting to its appropriate ASCII values and then storing them into a loop check where I set flags which would be set/reset based on the entries. This worked much quicker than expected.*
2. *Since music had been worked upon it was to be only played when the car moved and not when it stopped. I did this by disabling and enabling the PWME value in the Timer sub interrupt.*
3. *Assistance to teammates in rebuilding circuit and individual simulation on breadboard.*

**Activity Log for:** <Chen-Kai. Ling 0692-L> **Role:** <Software and Hardware>

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***Activity*** | ***Date*** | ***Start Time*** | ***End Time*** | ***Time Spent*** |
| Learn and understand the available hardware. Focus on transmitter, remote controller and H-Bridge. | 4/22/2015 | 6:00 pm | 11:00pm | 5 hours |
| Order necessary parts, discuss and make plans | 4/23/2015 | 6:00pm | 8:00pm | 2 hours |
| Be able to sample signal for direction control. Trying to find a way to sample signal for speed control. Build shift register in GAL chip for direction control. Build H-bridge but find out it can’t operate properly. | 04/24/2015 | 6:00 pm | 1:00 am | 7 hours |
| Realize GAL chip can’t source enough current for H-bridge. Car’s direction can not be fully controlled by Remote Controller. | 04/25/2015 | 8:00 pm | 3:00 am | 7 hours |
| Use 4 PTT port for four switches on H-bridge. Sample signal with PTT instead of ATD to avoid timing issue. Microchip pin arrangement to free up pin for other function. | 04/26/2015 | 6:00 pm | 5 :00 am | 11 hours |
| Apply password function to car. The car can now run with remote control and LCD display. Use 9-volt battery to make car runs faster. Design and develop software for music. | 04/27/2015 | 3:00 pm | 10:00 pm | 7 hours |
| Add multiple UI and improve the performance of H-bridge. | 04/28/2015 | 1:00 am | 4:00 am | 3 hours |
| Soldering 2 PCB for LCD and Microcontroller. LCD won’t initialize without using reset pin. Set up reset push-bottom. Car sometimes resets it-self. | 04/28/2015 | 6:00 pm | 12:00am | 6 hours |
| Successfully understand the speed signal from RC transmitter and is able to sample by building a diode bridge between 2 signal pin. | 04/28/2015 | 1:00 pm | 3 : 00 am | 12 hours |
| Install speaker onto the car. The car is now able to play music while it is moving and have 3 song tracks. We are able to create our own music array. Trying multiple ways to solve program-reset problem. | 04/29/2015 | 2:00am | 5:00 am | 3 hours |
| Rewiring the whole circuit in order to 1) have a better looking. 2) minimize floating wires and try to solve program self-reset problem. 3) be able to fit all hardware into limited space. Finally realize the cause of self-reset is noise from low quality long wire. Redesign wiring and software to solve reset problem.  Car is fully functional and ready for the demo. | 04/29/2015 | 7:00 am | 3:00 pm | 8 hours |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**Written Summary of Technical Contributions:** <chen-kai.ling>

*Provide a concise but sufficiently detailed description of your technical contributions to the project.*

*My role includes design and developing hardware and software.*

1. *Apply RC & RC transmitter.*

* *Exam input and output signal from transmitter*
* *Design signal sampling method using PTT/TIM and digital input of ATD. By analyzing the signal we are able to determine the frequency and pattern of the signal sent from transmitter. Therefore, we are able to create our sampling method with the highest accuracy.*

1. *Internal structure design.*

* *Design 5 PCBs: PCB for user input, PCB for power, PCB for LCD and ATF device, PCB for microcontroller and PCB for LEDs.*
* *Design wire connection to minimize floating wire and noise from high current wire. The first structure design we made has many floating wire which cause significant connection and signal noise issue. These issues make debugging extremely difficult. The wire with constantly changing current or high current causing our logic output unstable and sometimes causing microcontroller reset issue.*
* *Fit everything into the limited space of the car. Be able to understand the*

1. *Debugging and solving the hardware and software issue*

* *Solving the problem that speed signal can’t be sampled by building the diode bridge between input pins.*
* *Solving the problem that LCD can’t initialize by rewiring the Vcc and Ground PCB and give LCD more time delay to initialize in the code. By changing the distance between Vcc and ground position has an unexpected impact on hardware initialization. It is very important to power up other hardware parts first, so it can have enough time to receive initialization code from microcontroller.*
* *Solving the issue that music can’t be played properly by giving proper time delay.*

1. *Motor and H-bridge*

* *Build the connection between H-bridge, microcontroller and motors*
* *Take part in designing the software that is then applied to control the speed and direction of the car*
* *Apply sampled input into H-bridge. This is achieved by understand the internal structure of H-bridge—the way voltage stabilizer is connected and the way switches is sat to control the directions.*

1. *Microcontroller resource organizing*

*Since our project requires many pins for inputs and outputs, it is very important to correctly use port pins with their proper function. In order to fit all user inputs and microcontroller outputs into one chip, we have to carefully design and organize the port pin usage.*

*Length should be about one page.***Activity Log for:** Tianyu Deng **Role:** debug software and hardware,idea on funtion

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ***Activity*** | ***Date*** | ***Start Time*** | ***End Time*** | ***Time Spent*** |  |
| ***Learn and understand the circuit connection*** | 04/22/2015 | 6:00pm | 11:00pm | 5 hours |  |
| Testing the hardware and preparing the backup parts | 04/23/2015 | 6:00pm | 8:00pm | 2 hours |  |
| Working on the direction control and speed control, testing the signal generated by transmitter. | 04/24/2015 | 6:00 pm | 1:00 am | 7 hours |  |
| Building the shift register to control direction and fixing the H-bridge to make it worked stably and constantly. | 04/25/2015 | 8:00 pm | 3:00 am | 7 hours |  |
| Changing our design circuit , sampling the signal with PTT , thinking the function of PWM could be applied to car. | 04/26/2015 | 6:00 pm | 5:00 am | 11 hours |  |
| Debugging the password function code for our car, checking the connection of LCD on other breadboard | 04/27/2015 | 3:00 pm | 10:00 pm | 7 hours |  |
| Thinking and writing the code for the music function of our car | 04/28/2015 | 10:00 pm | 1:00 am | 3 hours |  |
| Soldering the circuit , debugging the hardware problem that causing the LCD displaying mistakenly | 04/28/2015 | 6:00 pm | 12:00 am | 6 hours |  |
| Solving the problem that RC transmitter did not provide speed control signal stably. | 04/29/2015 | 1:00 pm | 3:00 am | 12 hours |  |
| Installing the speaker to the car and creating the code for user to choose among three music | 04/29/2015 | 2:00 am | 5:00 am | 3 hours |  |
| Soldering some wire again and packaging the car , making it looks more neatly . | 04/29/2015 | 7:00 am | 3:00 pm | 8 hours |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**Written Summary of Technical Contributions:** Tianyu Deng

*Provide a concise but sufficiently detailed description of your technical contributions to the project.*

*Length should be about one page.*

On our team, my main job was helping to solve hardware errors and testing the components that I bought to make sure they are suited for our microcontroller. When we first assembled our car, the transmitter provided a very unstable signal when we tried to hold the accelerator. I recorded the difference in the current situation of our car and found out that if the battery inside the RF remote was lower, the signal tended to be more unstable. To address the issue, we changed the battery and discovered that the remote power was not the main issue causing noise to our signal. Then, I thought the transmitter might not have been working efficiently when we removed it from the original toy. To test this theory, we bought other infrared remote control toys and kept testing for a stable square wave signal. All these unsuccessful tests contributed to our final decision to provide a diode between the direction control wire and the motor power-up wire. As we wired the circuit, I also consulted with our TA and realized that we did not need a GAL chip to store the data for direction control and acceleration since we already had enough pins for it and the GAL chip needs to be used for LCD. I learned that the GAL chip was another main issue that could cause the motor and H-bridge to be unstable. At last, I collaborated with my teammates and soldered most of the wires clearly.

On the software side of things, I determined that the DDRT was not enabled and that we could not sample the signal provided by the transmitter, which caused the motor not to run ideally. I came up with the idea that we could make our car run after entering the correct password by adjusting the potential meter and that we could use our PWM peripheral to output different music. I searched online and, knowing that different frequencies can produce various sounds, I looked up the piano’s structure and decided I needed to make 14 melodies in order to produce any song we like. I searched for the frequency of those melodies and tested the sound. Afterwards, I found the music score of some simple songs and made an array that contained the elements in the correct sequence from the melody array. In the main loop, I wrote a loop that kept changing the PWM period, which provided a different frequency and the sequence matched the music score array for each song. Between each melody, I made a delay function that determined the tempo by giving it blank for loop. After everything was done, I connected the speaker to the PWM output and tested for appropriate tempo.

  At the last step of the project, I helped my teammates package the car by taping it with a 3D printed shell and making a video, which we uploaded online.

**Activity Log for:** <name-4> **Role:** <role on team>

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***Activity*** | ***Date*** | ***Start Time*** | ***End Time*** | ***Time Spent*** |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**Written Summary of Technical Contributions:** <name-4>

*Provide a concise but sufficiently detailed description of your technical contributions to the project.*

*Length should be about one page.*

**Appendix B:**

**Interface Schematic**

*Paste a copy of your Eagle or OrCAD interface schematic here.*

*Be sure to clearly identify the team member(s) responsible for producing this documentation.*

**Appendix C:**

**Software Flowcharts**

*Include software flow diagrams and/or pseudo code here.*

*Be sure to clearly identify the team member(s) responsible for producing this documentation.*

*NOTE: Software source listing file must be submitted on-line and should NOT be included here.*

**Appendix D:**

**Packaging Design**

*Paste illustrations/pictures of your project packaging here.*

*Be sure to clearly identify the team member(s) responsible for producing this documentation.*