Computer Engineering Senior Project Documentation

GPS Controlled RC Car with Collision Avoidance

GPSHummer-HC12

Brian Bradford
Huy Vo
Bryce McAllister
Seth Thorup

Table of Contents

Section 1. Introduction

To fulfill the senior project requirement for a Bachelors of Science in Computer Engineering at the University of Utah we have elected to design and build a GPS guided car with collision avoidance.

The GPS car is guided by an integrated GPS unit allowing the car to operate from start to destination with out intervention. The GPS car is also equipped with collision avoidance via an ultrasonic device. The ultrasonic device enables the GPS car to change course mid trip if an obstacle is in the destination path of the GPS car. The car is an R/C type model of a Hummer SUT and is the mount for all of the GPS car's peripherals as well as housing the vehicles motors. The interfaces between the motor, GPS unit and ultrasonic device are managed by a Motorola 68HC812 microcontroller. The microcontroller will handle all requests and messages from the GPS unit and ultrasonic device. The microcontroller facilitates all logical operations by the GPS car, making the car "smart" enough to control its peripherals and adjust to varying terrains.

Section 2. Overview

2.1 Design model

We have mounted the microcontroller, GPS unit and the Ultrasonic device to the body of an R/C model Hummer. All critical interfaces are through the HC12 microcontroller.

2.1.1 GPSHummer-HC12 Overview Diagram

2.2 Components

The custom built GPS hummer is composed of many different parts. All purchased through local retailers or via the internet. Each component was initially acquired with no modification specific to the GPSHummer-HC12

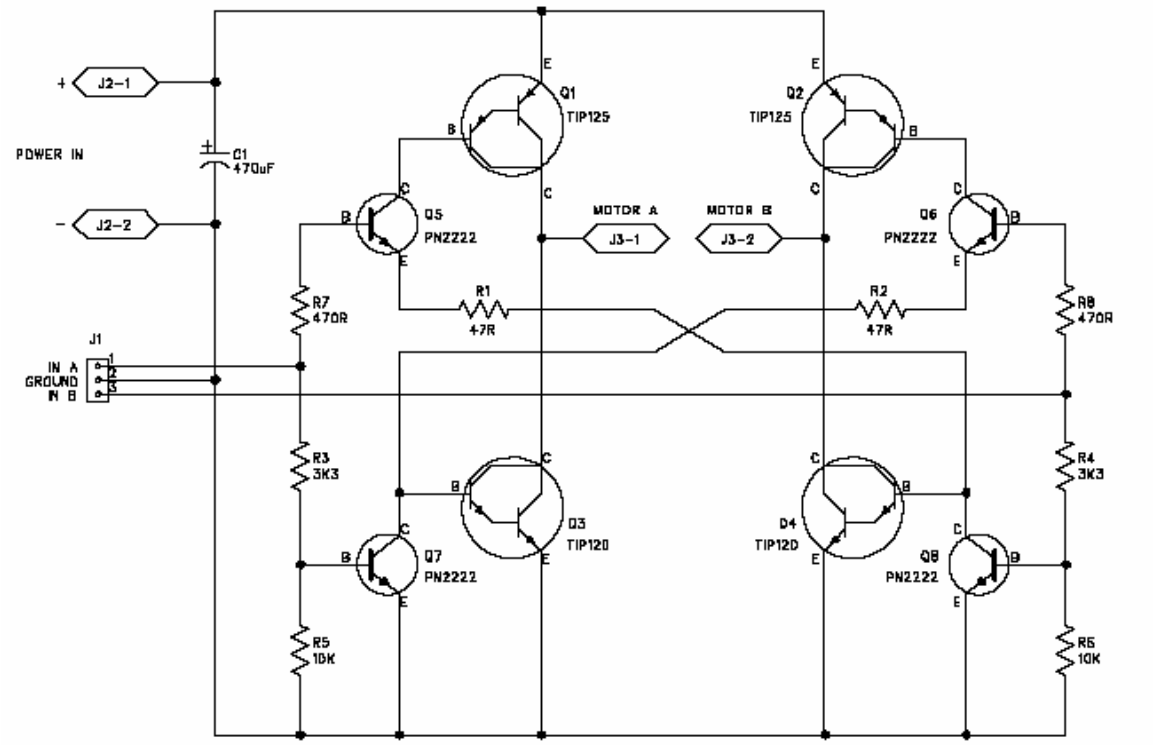
2.2.1 1:6 Radio-Controlled H2 Hummer SUT

- 28-inch, radio-control Hummer
- Steers left and right while moving in forward or reverse
- High and low gear speeds
- Uses one, 9.6-volt rechargeable battery
- Front and rear suspension
- Lights, horn and bumper

2.2.1.2 Problem with R/C Hummer's circuit board
The main circuit board which comes with the R/C Hummer uses TTL output from its main microcontroller chip. TTL doesn't work with a dual output which means if you try and tie an input to the circuit board in parallel with the Hummer microcontroller's output your get feedback into the Hummers microcontroller and burn up the Hummers chip.

2.2.2 H-Bridge
We decided that instead of trying to use the main board, which comes stock with the R/C Hummer, we would build an H-Bridge to control the front and rear servo-motors.

2.2.2.1 H-Bridge schematic



http://www.bobblick.com/techref/projects/hbridge/hb01sch.pdf

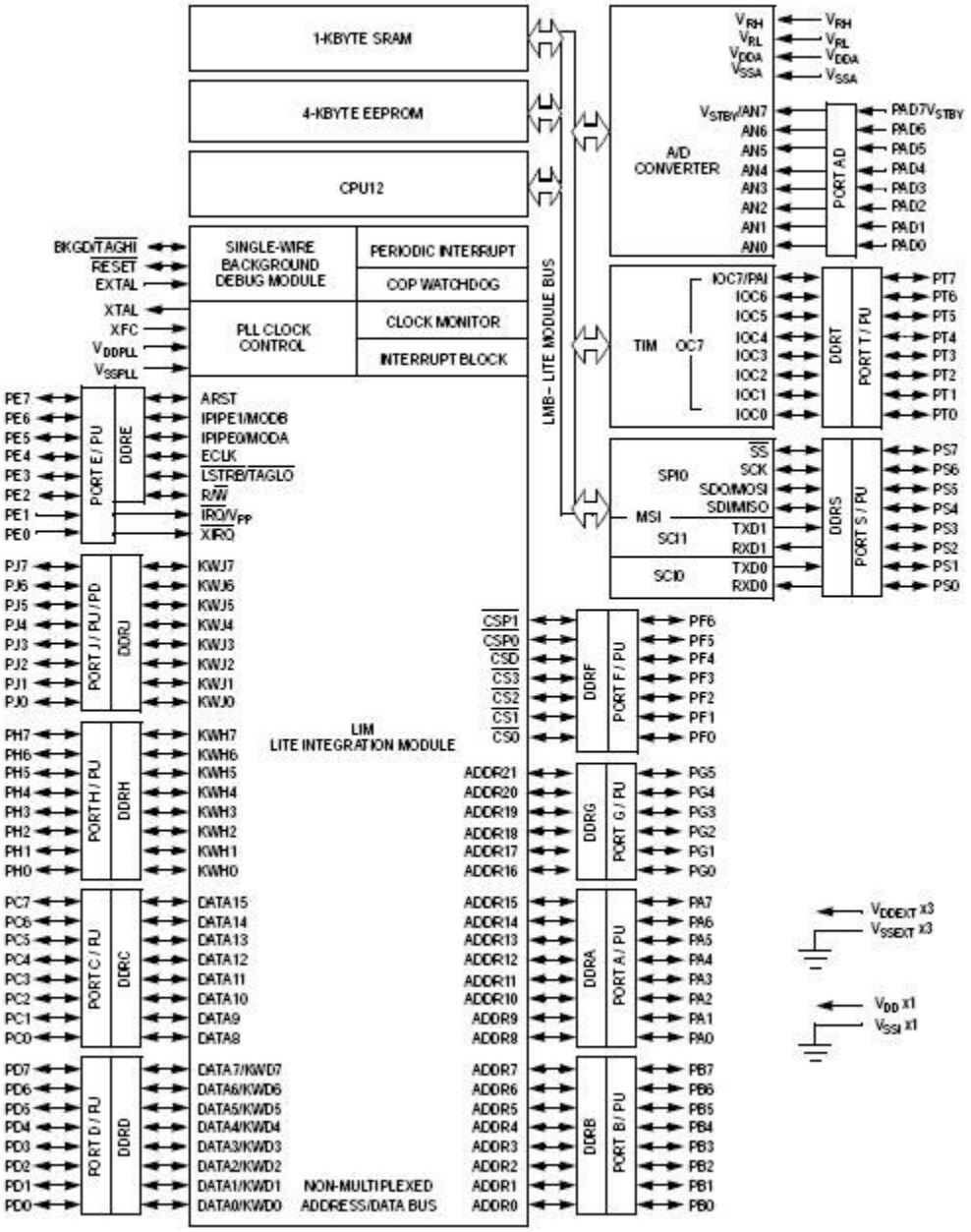
2.2.2.2 H-Bridge design
Each H-Bridge will power two motors, we required 2 such H-Bridges, one for forward and reverse control and one for left and right control. Power supply voltage for the H-Bridge is 5 to 40 volts. Output current up to 5 amps is allowed if the power supply voltage is 18 volts or less. Peak current must be kept below 8 amps at all times.
(http://www.bobblick.com/techref/projects/hbridge/hbridge.html)

2.2.3 Motorola M68HC12 Microcontroller
The M68HC12 is a low-power, high speed CPU with power-saving stop and wait modes. There is 1024-byte Ram and 4096-byte EEPROM with on-chip memory mapping which allows expansion to more than 5-Mbyte address space.

It is equipped with single-wire background Debug mode, seven programmable chip selects with clock stretching and 8-channel enhanced 16-bit timer with programmable prescaller. Meaning all channels are configurable as input capture or output compare with flexible choice of clock source.

The microcontroller is a 16-bit accumulator, a real-time interrupt circuit, computer operating properly (COP) watchdog and a clock monitor. Also built into the 68HC12 is phase-locked loop (PLL), Two enhanced asynchronous non-return-to-zero (NRZ) serial communication interfaces (SCI), 8-channel, 8-bit analog-to-digital converter (ATD) and up to 24 key wakeup lines with interrupt capability. (The following diagram and microcontroller explanation was taken from Motorola's Data Sheet for the M68HC12.)

2.2.3.1 M68HC12 Block Diagram



2.2.4 Adapt812DXLT

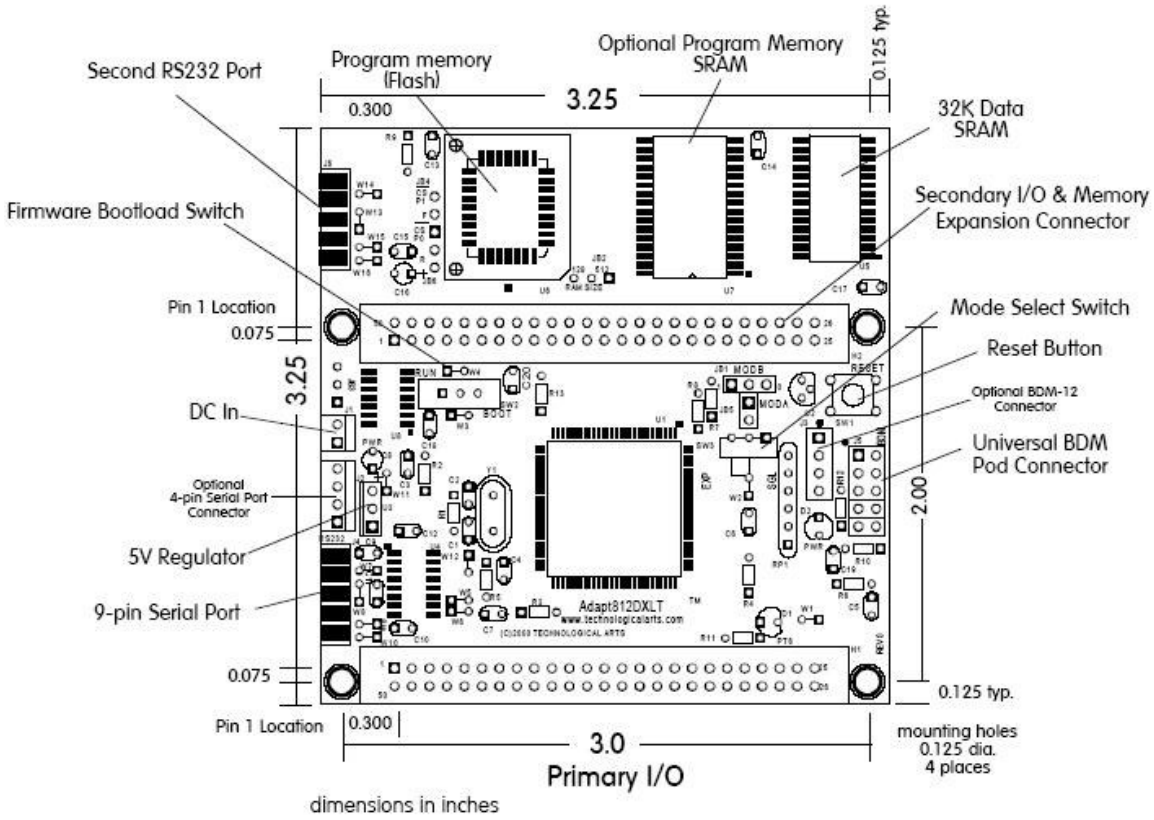
The M68HC12 mircocontroller is embedded in a Adapt812DXLT development board purchased from technicalarts.com. The Adapt812DXLT is form factor (3.25" x 3.25") and is a low-cost narrow expanded mode implementation of the 68HC812A4 MCU (16 MHz crystal) which is part of the M68HC12 microcontroller family described above.

The development board has 128K shadow RAM and supports software breakpoints. There also is 32K SRAM, two RS232 interfaces and an on-board 5V 500mA regulator. The Adapt812DXLT is a fully functional, standalone implementation of the MC68HC812A4 microcontroller designed to run in Normal Expanded Narrow Mode. You can configure the board to run in Normal single chip mode by jumping JB1 which sets MODA on the microcontroller to zero. We are running in Narrow Mode. (http://www.technologicalarts.com/myfiles/812dxlt.html)

Prior to implementation we plugged the Adapt812DXLT into a solderless breadboard for testing. The Adapt812DXLT has allowed us to easily integrate the GPS and Ultrasonic units with the 68HC12 chip. This evaluation board came with the microcontroller per-loaded with software for testing ports and other features on the board. Although we took from pieces of this code like the reg.asm file, we have done almost all the assembly code independent of any other source.

The RS232 interfaces on the Adapt812DXLT has allowed for easy loading of .s19 files to program the board. Also we have been able to connect through the second RS232 serial port directly to the GPS unit's development board.

2.2.4.1 Adapt812DXLT diagram



(See Adapt812DXLT Data Sheet http://www.technologicalarts.com/myfiles/data/812dxltpins0.pdf)

2.2.5 M12+ GPS Receiver

Built around Motorola's Instant GPS MG4100, the M12+ GPS receiver modules are lightweight and compact making them ideal for attaching to an R/C car. Each channel of the GPS independently tracks both code and carrier. The M12+ was specifically designed for embedded applications and allows for a lot of engineering freedom. (See M12+ Users Guide Version 6.0)

The GPS unit comprises of three main components, M12+ Positioning Receiver, M12+ Timing Receiver and a Hawk antenna.

Some of the main functions of the M12+ Receivers are: (See M12+ Users Guide Version 6.0)

- 12-channel parallel receiver design
- Code plus carrier tracking (carrier-aided tracking)
- Position filtering
- Antenna current sense circuitry
- Operation from +2.85 to +3.15 Vdc regulated power
- 3V CMOS/TTL serial interface to host equipment
- 3-dimensional positioning within 25 meters, SEP (with Selective Availability [SA]

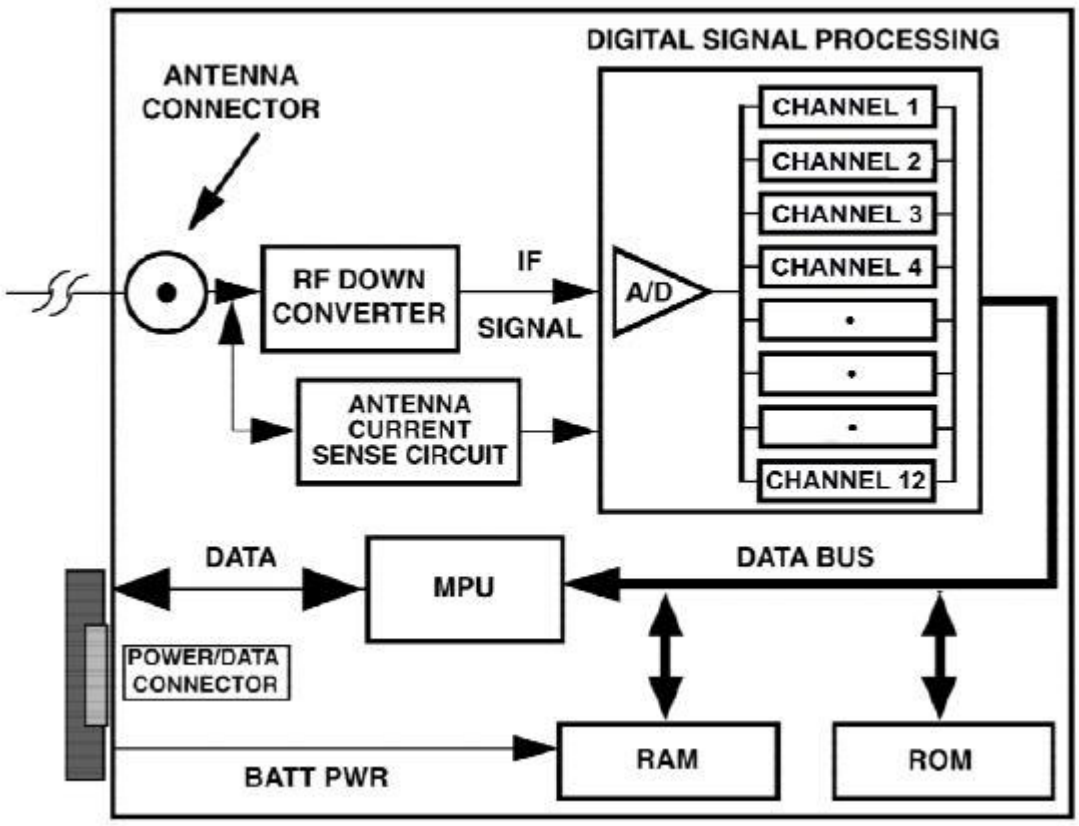
- Latitude, longitude, height, velocity, heading, time, and satellite status information
- transmitted at user determined rates (continuously or polled)
- Straight 10-pin power/data header for low-profile flat mounting against host circuit board.
- An optional right angle header is available for vertical PWA mounting.
- Optional on-board Lithium battery

2.2.5.1 M12+ Positioning Receiver
The positioning unit supports inverse differential GPS operation, RTCM differential GPS on the second serial port and a user controlled velocity filter.

After receiving a signal there is a minimal computational delay of 1 second to compute outputs. This delay limits the frequency at which an update of position can be taken.

2.2.5.2 M12+ Timing Receiver
The timing receiver has precise 1PPS output (+/- 25ns accuracy) without sawtooth correction and is also capable of selectable 100PPS output. Also included is a built in time RAIM (Time-Receiver Autonomous Integrity Monitoring) algorithm for checking timing solution integrity. Automatic site survey is also a component of the timing receiver.

2.2.5.3 M12+ Oncore Receiver Functional Block Diagram



(See M12+ Users Guide Version 6.0)

2.2.6 Evaluation Kit of the FS Oncore GPS Module

We purchased the M12+ GPS receiver as part of an evaluation kit from http://www.synergy-gps.com/minimodule.html. This kit allows for faster development of an embedded GPS device because of the easy integration of the GPS and microcontroller units. The evaluation kit came with all the extra accessories needed to allow for full GPS integration for the rest of the GPS car.

2.2.6.1 FS Oncore Carrier PCB

A main piece of the evaluation kit it is the FS Oncore Carrier PCB, which is the board the GPS receiver is mounted to. The carrier PCB provides TTL to serial conversion, allowing easy connection to our microcontroller. It also provides a regulated 3 volts to the GPS receiver. The Carrier PCB requires 9 volts for operation. It also provides LEDs for easy troubleshooting. It has a passive patch antenna, voltage regulators and zero ohm links to select antenna.

Even though it was possible to use the GPS chip without the Carrier PCB, the Carrier PCB allowed for easier troubleshooting.

2.2.6.2 Hawk antenna

The Hawk antenna receives satellite signals used by the M12+ GPS receiver to determain position. The antenna doesn't work unless it has a good view of the sky.

2.2.6.3 Low Cost Evaluation Kit Contents for the FS Oncore GPS Module

(See Motorola's Semiconductor Technical Data, Evaluation Tools – Preliminary for the Low Cost Evaluation Kit for the FS Oncore GPS Module)

2.2.6.4 Ionosphere distortion with GPS signal

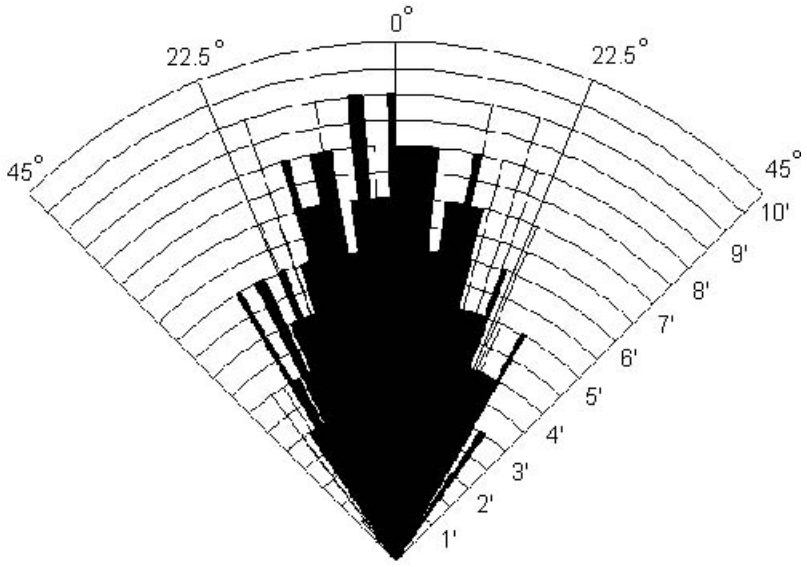
When we first started testing accuracy we found that a point would drift up to 100 feet. We found that our accuracy and reliability of a point was greatly improved when we masked out satellites resting too low in the horizon. Although the M12+ has ionospheric correction, the correction algorithms do not work well for satellites resting near the horizon. Signals from satellites positioned near the horizon in reference to the antenna are distorted by the ionosphere causing the signal to bend and attenuate. You can mask out satellites by setting the mask angle with the @@al command.

2.2.7 Devantech SRF04 Ultrasonic Range Finder

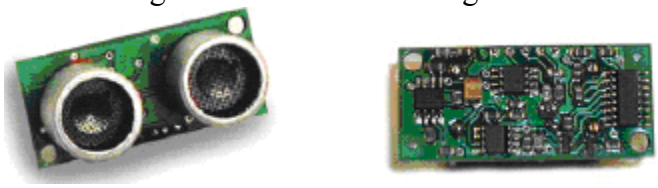
The ultrasonic ranger has a range from 3cm to 3m. There is a logic line used to trigger a pulse with the echo returned on a second line. It has a self contained design making it ideal for our GPS car. The digital I/O lines from the microcontroller are the only interface with the range finder.

Between each pulse there is a minimum 3ms delay to allow for the echo return to time out.

2.2.7.1 Beam pattern of the Ultrasonic Range Finder



2.2.7.2 Image of the Ultrasonic Range Finder



(See http://www.acroname.com/robotics/parts/R93-SRF04.html)

2.2.8 Power

Power for the GPSHummer-HC12 comes from two major sources; first the motors, lights, horn and other Hummer accessories are powered by a 9.6V rechargeable battery that comes with the stock R/C Hummer. The remaining power will come from a 9V rechargeable battery purchased as a spare for the R/C Hummer.

With some trial and error trying to power the GPSHummer-HC12 and after burning up a few circuit boards and blowing out the fuse in each of our batteries, we finally were able to power the H-Bridge/servo-motors on one battery and the HC12 chip and the GPS chip on the other battery.

Although we get about 10V from each battery by powering both the GPS and the Microcontroller from the same battery we are able to keep in a safe power range. The Ultrasonic Range finder is powered from the HC12 chip.

2.2.8.1 Required Power by module for the GPSHummer-HC12

Module	Power Min	Max
Motors	9.6V	11V+
Adapt812DXLT	5V	9V
M12+ GPS Receiver	3V	9V
Ultrasonic Range finder	5V	5V

2.3 Interfaces

Some of the more critical configurations that we have done on the GPS car have to do with its interfaces. The 68HC12 chip is the hub for all of our interfaces, no one device communicates with any other module unless through the microcontroller.

2.3.1 Microcontroller - GPS

The 68HC12 microcontroller communicates to the GPS unit via a null modem serial connection integrated into each devices development board. The GPS chipset is configured to give a position update once every second. Data is taken from the GPS bit by bit.

2.3.2 Microcontroller – Ultrasonic Range Finder

The ultrasonic range finder is controlled by the microcontroller through PORTT, PT2 for the “Echo Output” and PT3 for “Pulse Trigger Input.”

2.3.3 Microcontroller – Motor circuit

The motor circuit only requires 1V to operate; the microcontroller sends directional signals through PORTJ. PT7 sends the forward signal, PT6 sends a reverse signal, PT5 and PT4 send the turn right and left signal with PT3 needing to be set to zero when a left or right turn signal is given.

2.4 Tools

The main tools and accessories that we have used to design, develop and implement our GPSHummer-HC12 are:

- Internet – Research, purchasing and documentation.
- DC Power supply – Power to microcontroller development board, car motors, GPS evaluation board, ultrasonic rangefinder etc. during design and development.
- MinIDE – An assembler program that also has an editor and a serial communication terminal. You can download a copy of this software from http://www.mgtek.com/miniide/.
- Glue Gun – Mounting of modules to cars frame.
- Oscilloscope – Test signals from Rangefinder and microcontroller.
- WinOncore – The is an PC program which comes with the GPS development kit. Use to experiment with configuration and commands for the GPS chip such as masking of satellites.

Section 3. Drivers

3.1 Motor Driver

The processor is interfaced to the H-bridges through 4 ports. Two of the ports Control forward/reverse and the other two control left/right. The motor driver provides basic control of the motors. When MotorForward is called, the processor lowers the reverse port and raises the forward port. Just the opposite occurs when MotorBack is called. When MotorBrake is called the processor raises both the forward and reverse ports. This has the effect of "braking" the motor, not allowing it to spin. The operation of MotorLeft and MotorRight is similar.

3.2 Ultrasonic Sensor Driver

The sensor has only one logical input and one logical output. When the input it raised high, the sensor sends out an ultrasonic pulse and then raises its output. When the sensor receives the echo to that pulse it lowers the output. If an echo is not received then the sensor will timeout after about 30ms and lower its output.

The sensor driver sends the initiating high input, waits for the output to go high and then begins timing how long until the output is dropped again. From the time it measures it calculates the approximate distance in feet of the object. If for some reason the sensor malfunctions or an object is not detecting then the driver will timeout and return an error code.

Section 4. Main Program

The main program brings all of the peripheral drivers together. It also serves as the logic unit and makes all the movement decisions. GPS coordinates are stored to the RAM as specified by the main program.

4.1 Logic
Source code for the main program is available upon request. The following files are what make up the GPSHummer-HC12 Microcontroller logic.

The basic operation of the car is very simplistic:

```
loop {
        Is anything in our way?
        Our we there yet?
        Update our heading..
    }
```

4.1.1 MAIN.asm
This file contains the main logic loop, along with the GPS driver. Dirfifo, sensor, direction, test and Guide functions are called and implemented in main.asm. Also included in main.asm are EEPROM and initialization definitions, delay, Input points subroutines and all extended GPS subroutines.

4.1.2 DIRFIFO.asm
Subroutine put point and get point fifo routines. This function allows for multiple way point entries.

4.1.3 SENSOR.asm
Ultrasonic range finder driver and all related subroutines.

4.1.4 DIRECTION.asm
Allows user to input directions to destination in feet along the cardinal directions.

4.1.5 TEST.asm
Test routine which bypasses the GPS to allow for calibrating car's amount of turn with out having to be outside.

4.1.6 GUIDE.asm
Decision function for car turn right/left by 45, 90, 135 or 190 degrees. Update course, normalize and round turn subroutines.

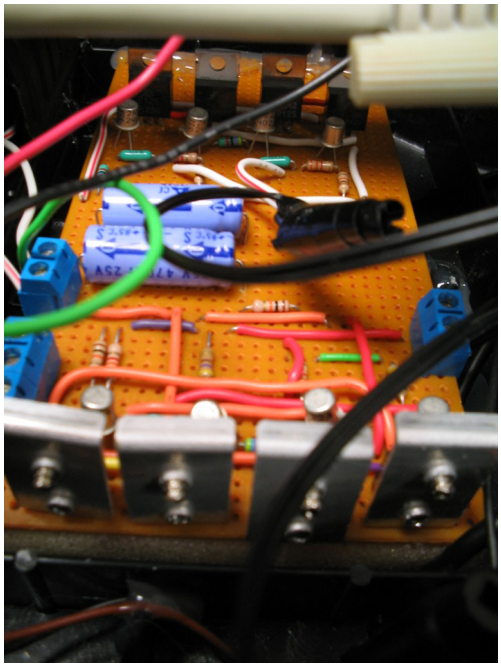
Section 5. Final Product

5.1 Images of Complete GPSHummer

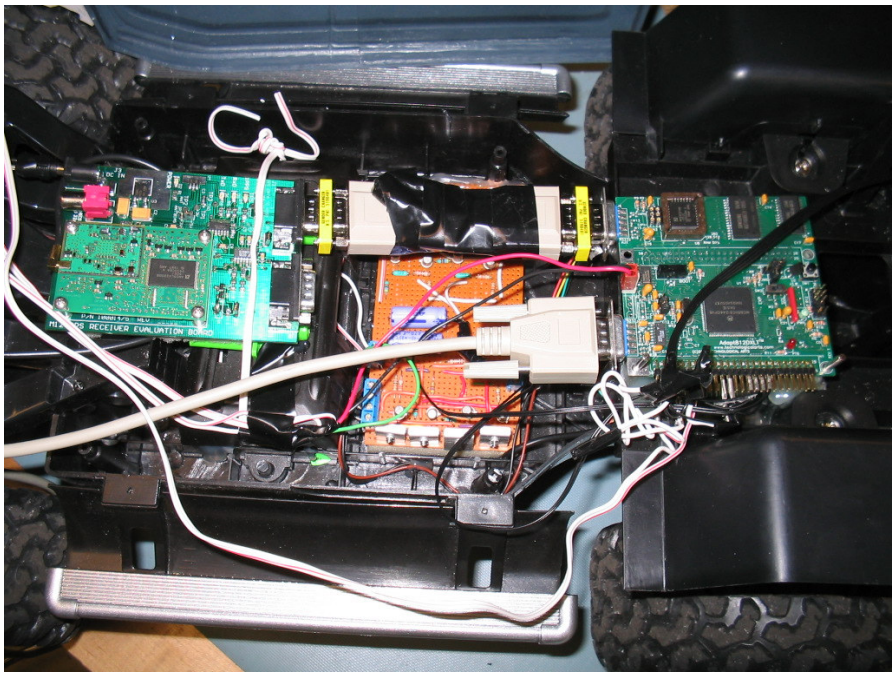
The completed GPSHummer-HC12 is shown below. Things of note are the cut out in the hood to allow access to the run/program mode switch on the HC12 and the serial cable extended from within the car to allow for programming the HC12 chip.

5.11 Completed Car

5.1.2 Left - Front view of car/View of Range finder. Right – H-Bridge



5.1.3 GPS Unit connected to Microcontroller



5.1.4 Top View/GPS antenna

Section 6. Problems/Improvements

6.1 Controlling the motors
Initially the plan was to cannibalize the motor-control circuit that came with the Hummer. We thought this would be pretty simple and would allow us to spend more time on other design issues. This decision cost us a lot of time in the long run. After frying the board that came with the Hummer and reparing it then frying it again, and after shorting out both of our batteries, we decided that the best thing to do would be to quit messing with that board and build our own H-bridges to control the motors.

After researching several designs on the internet we found the one we wanted at http://www.bobblick.com/techref/projects/hbridge/hb01sch.pdf. After building the h-bridges ourselves we never had another problem with the motors.

6.2 Interrupts
We decided that running the sensor off a timer interrupt would give us the optimal collision-detection system. It would ensure that every 30 ms or so the sensor was being used to make sure nothing was in our way. We ran into problems that we could not get around.

The first problem we discovered was that the timer interrupt from which we were running the sensor had a higher priority than the serial port interrupt from which the GPS chip was running. Because the sensor interrupt was occurring about every 30 ms the GPS chip was getting starved. We discovered that we could assign an interrupt to the highest priority and thought this would solve our problem. We gave the GPS the highest priority but when we tested it, the GPS was working but only about once every ten seconds or so. This was due to the fact that only one character of the message is sent per SCI interrupt. Because the sensor would interrupt before a complete message could be received, invalidating the message.

Due to lack of time we decided the easiest thing to do was to take the sensor off the interrupt and run it in gadfly style within the main program. Because of this, every once in a while, the Hummer doesn't detect an object as quickly as it would had it run off interrupts.

6.3 Improvements
If given more time and resources we would have added more sensors, refined the collision avoidance logic and improve the navigation software. In addition we would have optimized the car for higher speeds.

Section 7. GPSHummer-HC12 User Manual

7.1 Introduction

The GPS Hummer is simple to operate. It only has to be given points to drive to and then it can be set free to go and find them. You can only enter a maximum of 10 points. There are two different methods for giving it points. These will be discussed in the next two sections.

7.2 Inputting points manually

In the default mode, when the Hummer is turned on or reset, there is a three second delay after which the green LED turns on. The green light indicates that it is waiting for points. Walk the Hummer to the first point, push the black button located on the rear of the Hummer. The first point has now been entered. Continue this until you arrive at the last point you want to enter. On the last point push the black button but this time hold it down until the green LED turns off (about three seconds). When the light goes off the Hummer is ready to find the first point. If the motor switch (located underneath the Hummer) is already turned on it will immediately start driving. Otherwise, as soon as you flip the switch it will begin to drive.

7.3 Inputting points through a terminal

Note: This feature can only be turned on by changing a flag in the actual program, reassembling it, and uploading it to the HC12 processor. We did not have time to add an external switch to control the flag.

In this mode, when the Hummer is turned on or reset it is ready to be given points through the serial port. Hook the serial cable up to a computer and connect to the port with a program such as Hyperterminal. There are two different methods for giving the Hummer points. The first is to give it absolute GPS coordinates. This is done by sending commands in the following format.

#XXYY

- ASCII '#' (Beginning of command)
XX - 2 bytes that define the X coordinate
YY - 2 bytes that define the Y coordinate

When you are done entering points send the Hummer this command:

! - ASCII '!' (No more points)

You can also give directional commands such as "Go 100 ft north, then 200 ft northeast, then 100 feet south..." by sending commands of this format:

nDf

n - 1 byte representing feet (255 ft maximum)
D - One of the following letters representing a direction:
- -N North
- -E East
- -S South
- -W West
- -n northeast
- -e southeast
- -s southwest
- -w northwest

f - ASCII 'f' (feet)

When you are done entering points send the Hummer this command:

! - ASCII '!' (No more points)

Disconnect the serial cable, place the Hummer at the starting position and turn the motors on. The Hummer is now waiting for the black button to be pushed. When it is pushed if it was given directional points, it will calculate the absolute points and start driving to the first point.

7.4 Accuracy

The Hummer most likely won't arrive at the exact point. It will usually end up between 5 and 10 feet away from the point. The accuracy of the GPS Hummer is affected by two things. First, it is only as accurate as the GPS points it receives from the satelites. These points can be off by a few meters or more. Second, because of this it is necessary to define large points. Each point the GPS is given is actually a rectangle of about 10 ft by 13 ft.

7.5 Collisions

The GPS Hummer is equipped with one ultrasonic sensor on the front grill. Because there is only one sensor the Hummer works best in large open areas with occasional objects in the way. If there are two many objects it will spend most of it's time trying to get away from them and very little time trying to find the points. When the sensor detects on object within 3.5 - 6.5 feet, it will first try to swerve to avoid it. If anything is detected less than 3.5 feet then it will stop, backup, and try a different direction. Although the collision avoidance system works fairly well, occasionally it will have a slight crash