ECE 3992 Final Project Proposal

4/16/08

Ben Meakin

benlm54@gmail.com

Eric Hsu

erichsu@hotmail.com

Dan Rolfe

dan.rolfe@gmail.com

Calvin Yan

pc611652003@hotmail.com

Abstract

Advances in technology have lead to the development of the "smart car" concept. High-end vehicles offer motorists with gadgets such as back-up cameras, mp3 players, and GPS. This paper is a proposal for a project that would take the current automotive technology trends in a new direction by utilizing sensor nodes and wireless networking strategies to provide motorists with real-time information. We believe that a low-cost system can be developed to further the "smart car" movement and provide a platform for many useful new applications.

Introduction

City traffic can be a significant limitation to an individual's productivity. Our project will utilize motes (described below) and GPS technology to provide motorists with real-time driving directions that avoid traffic congestion. Taking traffic into consideration, optimized travel routes will be calculated and displayed to the driver through an on dash LCD display. This document outlines the key hardware and software components that will be needed to implement this system. Several high level implementation strategies are discussed. It also provides a list of design tasks, a schedule for their completion, a bill of materials, and an assessment of risks involved with this project.

Hardware

Motes

A mote is a small, low power device that includes most of the hardware needed to create a wireless sensor network. It has a microcontroller/CPU, memory (RAM and ROM), a wireless radio for networking, and several buses for the connection of various sensors. The key hardware design component of this project will be to interface a mote with a GPS chipset and an LCD display for output. These motes will also be networked with motes in other vehicles and at traffic intersections.

One of the most important hardware design decisions that must be made is the selection of the mote device to use. The range of the wireless device is very important. Without at least 30-40 meters the device will likely not work under

most circumstances. The processing power is also important. The capabilities of the device could be severely limited without ample CPU power. Memory and bandwidth are also vital to building a good system. The device must provide enough bandwidth to allow large amounts of traffic and city data to be received and transmitted to other devices. Performing operations on this data could require a lot of memory as well. Useful I/O busses are also very important. We needed a device that could interface with a GPS chipset and an LCD module. Based on these criteria, we have determined that the Crossbow Imote2 would be a good choice as it has an Intel XScale CPU that can run at up to 416 MHz. By soldering on an optional antenna to the board the device has a range of about 35 meters. The Imote2 also has a bandwidth of up to 250 kbps and 32 MB of Flash memory, making it a perfect choice for this application. It also includes RS-232 and USB serial busses for interfacing with external I/O [1].

GPS

The GPS chipset to use was also an important device to select. We needed a device that was low cost, was fairly accurate with respect to providing position and velocity, and could interface with a serial bus. We chose the Garmin GPS 15H sensor board. It can determine position to within 3 meters and velocity to within 0.1 knot RMS steady state. The manufacturer also claims that has very good EMI/RFI performance [2]. This is important because we plan to use the device alongside an 802.15.4 wireless radio. If the GPS device were to cause a lot of electromagnetic interference it could result in a lot of corrupted data from the WIFI radio.

Output

The user output system that we original thought of would involve the use of an LCD screen to display a lot of graphical representations of driving directions to the user. We ran into complications with this approach and came up with a slightly different solution. We found a device that converts from an RS-232 input to a VGA output. The device takes characters transmitted via an RS-232 protocol which uses a 2400 baud rate, 1 start bit, 8 data bits, no parity, and 1 stop bit [3]. It can then display these

characters onto an LCD screen via a VGA output. Using this device we will be able to display text based driving directions to the user rather than a visual representation.

Input

For user input we have chosen to use a Digilent rotary encoder and push button module. Using this device it will be possible for the user to quickly scroll through characters and push the button to enter them one by one. It will also allow the user to scroll through menus and will provide other functionality that is crucial to having an intuitive user interface. The module is easy to interface with. It will only require the use of 3 general purpose I/O pins on the mote [5].

Power Supply

One of the more important hardware design tasks that must be considered is that of a power supply. The car battery will provide the main power source. However, the devices that we have chosen to use have various voltage and current requirements. Therefore, it will be necessary to build a custom power supply and voltage regulator. Table 1 shows the operating voltage and current levels of the various devices in our system. Figure 1 shows a diagram representing the entire hardware system.

Device	Voltage	Current
IMote2	5 V	< 66 mA
GPS 15H	12 V	< 40 mA
Rs-Big-Print	Bus Powered	NA
LCD	12 V	
Rotary Encoder	Bus Powered	NA

Table 1

Software

The software design component of this project will be significant. The first task will be to become familiar with the .NET Micro Framework which runs on the Imote2 device [1]. It is upon this platform that we will build our software. A high level model of what the software will need to do can be described as follows: the mote will need to constantly be searching for other motes in range, connecting with those motes one by one, sending and receiving information so that both motes have up to date traffic information, and then performing a shortest-path routine on a weighted

graph data structure that represents city streets and compute the time required to traverse them. We will also need to write a fair amount of software for displaying a decent text based interface to the user on an LCD screen and for outputting optimized driving directions.

Networking

Communication between motes is also an important issue. Car based motes will have four different modes of communication (2 types of transmission and 2 types of reception): broadcast, update, request, and fulfill request. In "Broadcast" mode, the car mote is constantly transmitting 3 data frames each containing the ID of the mote, as well as the motes GPS coordinate, direction, and velocity respectively. This information will be used by the street motes to identify areas of traffic congestion. In "Update" mode, the car mote will constantly be downloading new map data. If no new map data is available and/or needed then nothing is downloaded. In "Request" mode, the car mote can send one of several requests for different kinds of data. Data frames for requests will include the car motes ID as well as an OPCODE for the given request. Requests can range from asking for traffic information to asking for locations of local gas stations, restaurants, or hotels. The "Fulfill Request" mode is where the car mote receives the resulting data of this request.

Data Structures and Maps

One important software design decision that we have had to make already is how we will represent city streets. We determined that we could not use pre-designed map files on the mote device due to the encryption and large size of these types of files. Therefore, we came up with a solution that will require close integration with the street based motes. Each street based mote will be installed with a graph based representation of the streets around it. Each node in the graph will represent an intersection and will contain the names of the streets at the intersection and a GPS coordinate. When a car based mote come within range of a street based mote the map for that area will be downloaded and saved into flash memory for future use. Flash memory management will be managed with timestamps where map areas used least recently will be deleted to make room

for new data. If the driver is seeking optimized directions for the entire trip, a street address will be sent in a request to the city based motes for a larger map area that includes the endpoint. We realize that this could result in a huge amount of data that needs to be downloaded. To optimize this we will probably need to coordinate with the street based motes and write a pruning algorithm that removes irrelevant paths to the endpoint. However, if I driver drives a route regularly, it is likely that very little new map data will need to be downloaded on a day to day basis because of the flash memory.

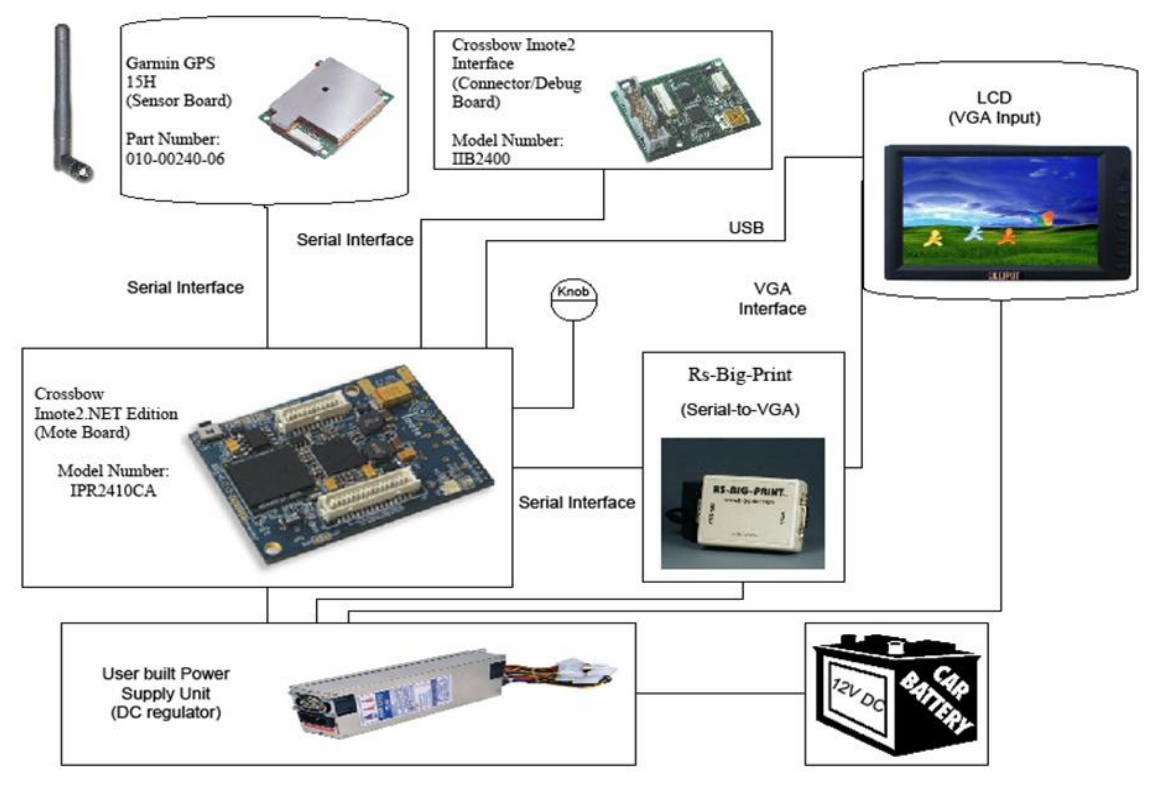

Fig. 1

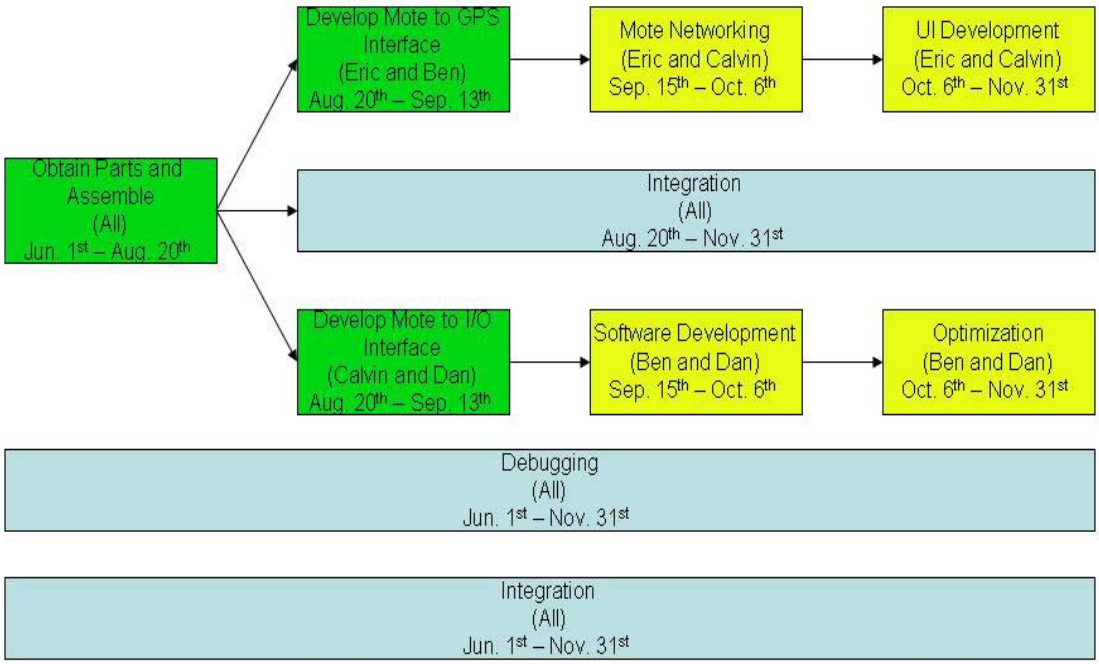

Fig. 2

Tasks

Hardware

Mote to GPS Interface

- Hsu and Meakin

Mote to LCD Interface

- Rolfe and Yan

Power Supply

- Hsu and Rolfe

Assembly

- Meakin and Yan

Debugging - All

Documentation - All

Integration - All

Software

Networking

- Hsu and Yan

Algorithms

- Meakin and Rolfe

User Interface

- Hsu and Yan

Optimization

- Meakin and Rolfe

Debugging - All

Documentation - All

Integration - All

Bill of Materials

Crossbow Imote2.NET Edition
- Model Number: IPR2410CA
- Price: $299
- Source: www.xbow.com online store
- Contact: sales@xbow.com

Crossbow Imote2 Interface – Connector/Debug Board
- Model Number: IIB2400
- Price: $150
- Source: www.xbow.com online store
- Contact: sales@xbow.com

Garmin GPS 15H – Sensor Board
- Part Number: 010-00240-06
- Price: $55.50
- Source: www.garmin.com online store
- Product Support: 1-800-800-1020

Cylab Inc. RS-232 to VGA Converter
- Part: rs-big-print
- Price: $62.50
- Source: www.rs-big-print.com
- Contact: sales@rs-big-print.com

Lilliput 7" Touch screen LCD VGA
- Part: EBY701
- Price: $163
- Source: www.ebay.com

Digilent Rotary Encoder Module
- Part: PmodENC
- Price: $10
- Source: www.digilentinc.com
- Contact: sales@digilentinc.com

Risks

Many of the problems that we originally thought we would encounter with this project have been eliminated as a result of design decisions that we have made up to this point. We shouldn't have any trouble with interfacing the mote with GPS. The GPS sensor module that we have chosen has a simple serial interface which should pair well with the motes busses. The issues surrounding the use of a graphical user interface will be avoided by the use of a text based interface. We will also not have to deal with using large, proprietary map files by implementing our system using the methods described in the software section.

However, these design decisions have not eliminated all risks. It is common in engineering that the solution of one problem may lead to several new

problems. A new risk that arises from our choice of map representation described in the software section is that bandwidth may be too significant of a limitation for this solution to work. We will seek to develop graph pruning algorithms to minimize the amount of data that needs to be transmitted. However, if these algorithms fail to deliver enough of an improvement then we will still run into bandwidth problems. In the end this risk will probably not be an issue as this project is just a small scale proof of concept. It will not actually be implemented in a city wide, large scale application. Therefore, bandwidth will not be an issue for our purposes. We will however, do our best to build a system that could be used in a large scale application.

Another risk that we face is that none of the engineers working on this project have much experience with wireless networking. Because of this, the learning curve may impose a significant limitation on our productivity. It will be important to allow time to study and learn wireless networking practices so that the project can be completed on time.

A significant risk also arises from the fact that to demonstrate our project, we depend on another team to successfully complete their project (see "Demonstration Layout" section below). The stationary mote team must be able to provide a system to communicate with our car based mote. We have not thought of a backup plan yet. However, it is likely that through close integration between the two groups this will not be an issue.

Demonstration Layout

The system that we are designing must be built in such a way that it can be demonstrated on a small scale. For many reasons, it is not feasible to demonstrate this system in a city wide application. We plan to have one car based mote system that interacts with one city based mote system. By having the city based mote feed imaginary data to the car mote we can model how the system would work in a real application. This is acceptable because the main purpose of this project is to develop a "proof of concept." We want to prove that given modern technology it is possible to take

the “smart car” concept to a whole new level by giving motorists a wide variety of real-time information.

Bibliography

1. Imote2.Net Datasheet: http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/Imote2.NET_ED_Datasheet.pdf

2. Garmin GPS 15H Tech Specs: http://www8.garmin.com/manuals/237_TechnicalSpecifications.pdf

3. Cylab, Inc. Rs-Big-Print Operating Manual: http://www.panix.com/~ht/vggdoc

4. Lilliput Website: http://www.lilliputweb.net/home.html

5. Digilent Rotary Encoder Reference Manual: http://www.digilentinc.com/Data/Products/PMOD-ENC/PmodENC_rm_RevA.pdf