Digital Aspect of the Design of An Embedded System for Heart Rate Monitoring

ieee.cse.ucsc.edu/~bear/125paper.pdf

Charles Shiflett - CE*
bear@soe.ucsc.edu

Brian Miller - EE
bmiller@ucsc.edu

Michael Rubin - EE
paperboy@ucsc.edu

June 16, 2003

1 Abstract

This paper is about my role in the design of a embedded system for wireless heart rate monitoring. My role consisted of all non-analog components of the heart rate monitor. The wireless heart rate monitor project was intended as a prototype system to show the feasibility of being able to hook up a small inexpensive system to a athlete to remotely monitor heart conditions.

This paper is intended to serve as both a way of documenting how we implemented our project as well as to provide a starting point for people wishing to program or create embedded systems using similar hardware.

*Primarily responsible for this paper

2 Introduction

The project I worked on was to create an advanced stage prototype of a heart rate monitor, capable of analyzing the three major components of a heart contraction. By showing that we could find the peaks and measure the amplitude of each peak, we would have demonstrated that our project could be used to analyze heart conditions.

I was charged with designing and implementing all digital aspects of this board, this included:

- Analog to digital conversion
- On-chip processing of data to
  - Provide feedback for the automatic gain control
  - Compress the data enough to send it out over a low-bandwidth link
  - Generate a serial bit stream
- Providing a low noise, high precision conversion
- Providing power regulation (+5V, +3.3V, -3.3V, -5V).
- Creating a programmer to program the chip
- Filtering the data on PC as needed to get a clean signal
- Doing peak detection on PC of the three major heart contractions
- Displaying the final result to the user

This paper is organized starting with a history of how I ended up on CFP's heart rate project, then going into the selection of hardware and design environment. Next the design of the hardware will be discussed as well as an analysis of its effectiveness. Finally, the PC side of the software will be discussed, and then, a summary will be presented summarizing both the class, the effectiveness of our project, and what I personally learned by taking this class.

3 Pre CFP's Wireless Heart Rate Project

We started our two quarter sequence for this class learning about teamwork, coming up with ideas, and forming into groups composed of people from different majors within the school of engineering sharing similar interests. When I heard about the idea of a heart rate monitor, I thought they were crazy. Did they realize just how small of a current and voltage was produced by a heart? Didn't they know a body has 3 Mega Ohms of resistance? Were they even aware of the need for wearing extremely painful electrodes to start to have a chance at reading electrical signals from a heart beat? Didn't they know that this wasn't a biology class?

With that in mind, I formed into a group with Erick Castillo[1] to work on Patrick E. Mantey's water control project[2]. There we divided the labor such that I was primarily responsible for hardware and Erick Castillo was primarily responsible for software. As we designed our project we worked together very closely to ensure that the hardware would be suitable for both of our needs, and that the software architecture would handle the needs of the hardware. We defined abstractions, came up with a design, and implemented a initial prototype system which consisted of a hardware design (with printed circuit board), and an initial cut at the software needed by the system.

In that group, I took the role of project manager, although with only two people working on the project, we both took leadership roles as needed. We implemented a time line which consisted of multiple things to be finished at different times as the project progressed, and had deadlines set by what needed to finished by a specific date to ensure that we could both be productive. I learned a great deal about working with a second person, delegating, and coming up with goals.

In spite of this, however, my interests were primarily in the project, and not in the final product. I viewed this class as a learning experience, and felt that part of that learning experience is coming up with new designs, and implementing things primarily in software (I am CE after all). In particular, P. E. Mantey felt that my design for a primarily digital modulator was too ambitious and that I should be implementing a modulator/demodulator in hardware as opposed to software.

I couldn't handle the second guessing of my work, and the things being said behind my back, so I left the project with a nearly complete software modem, a prototype hardware design, a build environment, a programmer, and all the research and design I put into the project. I joined CFP's heart rate monitor project, being impressed with the work they were putting into the project, and the results they were getting. I felt that with my experience doing

[1] ecast@ucsc.edu

[2] See reports by either Erick Castillo or Alexandra Carey of Team Telemetry II

digital signal processing, and with hardware design I had, I would be ideal to do the CE aspects of CFP's heart rate monitor.

4 Hardware and Design Environment

A significant component of the project consisted of the selection of a microcontroller, and the creation of a build environment. I wanted a surface mountable microcontroller as they have better noise properties, and they can be soldered directly onto a PCB (as opposed to, say, a Rabbit). I also wanted a Microcontroller that had as many bells and whistles as possible. Our search led us to the AVR ATmega series of Microcontrollers which have built in A/D, RAM, FLASH, EEPROM, UART, and so on, as well as enough processing power to be able to do integer-based digital filtering in a reasonable amount of time.

This processor turned out to be an ideal selection for both projects, however, it did require that a programmer be built, and that a build environment be set up to support the microcontroller. And, while documentation did exist, very little documentation existed for our particular microcontroller, and the quirks had to be found by experimentation.

What made the whole interface possible is the fact that the generation of object code for the processor and the communication between the processor and computer is seperate. When I began to construct the interface, the first thing to do was to attempt to communicate with the processor. It turns out that the inputs to the buffer must be pulled up/down, and that how buffering is done is very important. It also turns out that the miso/mosi pins are not consistent between chips, and actually change locations depending on if you are programming the chip, or using the chip's three-wire interface.

The biggest problem with the initial programmer was the fragility of the chips and the cable. When you hack apart a cable and then force the split wires into a breadboard, and then hack together a connection from the chip to the microcontroller, it leaves many points of failure. This, combined by the fact that the chips I was using had a tendency of blowing up, caused serious problems in the reliability of the programmer. This led me to eventually choose carefully the chips I was using, pulling up of inputs, and the robustness of the programmer. You can see these points in the redesigned programmer used in our final prototype of the heart rate monitor. This redesigned programmer never failed and saved countless hours.

5 Hardware Design

The primary considerations for the hardware design was to keep everything as low noise as possible while keeping accuracy high. To do this I approached the problem from both a hardware and a software perspective. We also carefully shielded all cables. The result was a very accurate representation of the output produced from analog section of the heart rate monitor.

In designing the PCB, the primary concern was in keeping a solid ground plane, and secondarily, keeping all switching traces as short as possible. The results of this design can be seen by the PCB design in my engineering notes.

On the software side, the analog data was sampled at a rate of 2.4Khz[3] and then averaged 20 times, which resulted in a sample rate of 120 samples/second. This required a bandwidth of 1Khz. This was about the ideal speed as it was well below the speed of the wireless link, but, fast enough to provide enough resolution for sampling the heart beat. Additionally, the low bandwidth effectively filtered 60hz noise.

In addition to the averaging, the chip entered a idle mode after every sample, such that as little of the chip was running at any given time. So while the final layout did indeed have flaws, they were easy to work around, and the result was a very low noise, very high accuracy analog to digital conversion.

6 Software design

My idea for the software was to present the data in a format similar to an oscilloscope, thus we would be able to determine the quality of the data we are sampling, and view the peak results in comparison to the sampled data.

To implement this, I needed a polling loop to read the data, and then some sort of interface. My teammates wanted to see a interface that could be ported to environments other than X. This limited the selection to a few toolkits, such as FLTK, and WxWindows. The documentation for WxWindows was really bad, and FLTK was quirky, so I selected FLTK.

As FLTK was event driven, and object oriented, it took some time to get it to work with the coding I wanted, which was realtime and polling based. However, it wasn't too bad to hack together an interface, and the results were exactly what I was looking for.

[3] 1Mhz / (13 == ADC conversion time * 32 == prescaler)

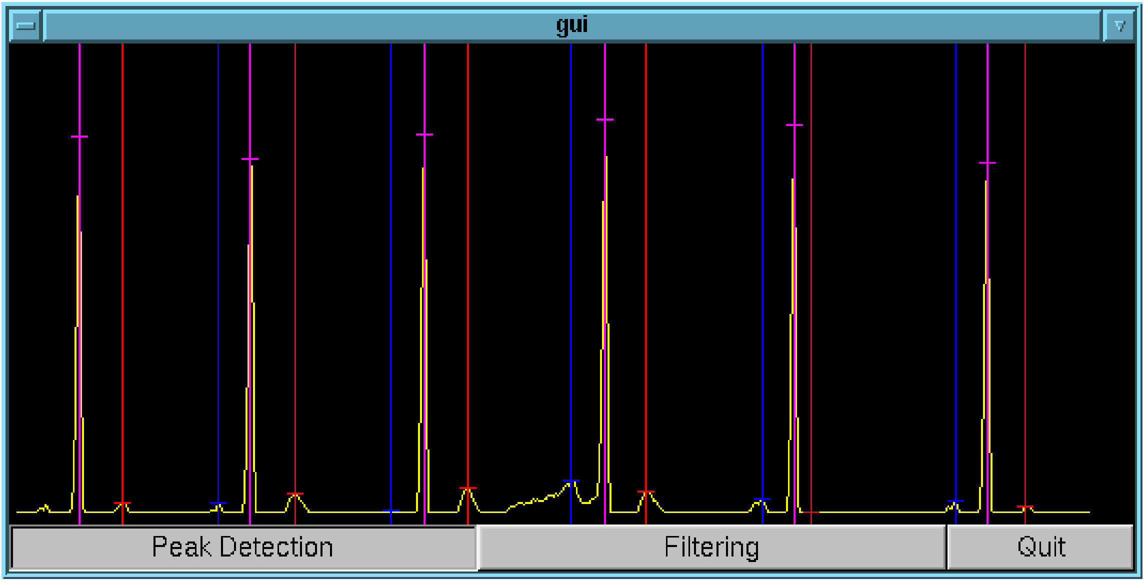

Figure 1: The results of peak detection

Even with the GUI complete, I still needed to implement peak detection. I implemented this simply by sorting the data, and then setting thresholds. Working backwards from my thresholds, I then found peaks, and sub peaks. This turned out to be a very quick and effective way of finding the data I needed.

The final result can be seen in Figure 1. The GUI color codes the three different peaks: magenta for the primary peak, and blue and red for the secondary peaks. Additionally the cross hairs represent the amplitude of each peak.

The biggest flaw is that the peak detection assumes a peak exists, which is a bad assumption, especially if there is not enough data, or if the data is bad. In addition, biasing was never implemented, so the negative peaks (or troughs), are nonexistent, which loses additional data. Other than that, the peak detection code worked very well.

No filtering was needed, as the signal coming in from the heart rate monitor is very clean after the hardware filtering and processing. It turned out that all the work I put into creating a software modem was a waste of time. *C'est La Vie.*

7 Summary

Our project was a success, and it was a lot of fun to work on. Figure 1 is proof of our success, and is an example of monitoring Brian Miller's heart. We did not add all the frills we had hoped for, but we did successfully show that we could monitor a heart and find the location and amplitude of the three main components of a heart contraction. Using this information we could then check a patient to ensure that there heart is functioning within normal parameters. Additionally, with our final prototype, the EKG stayed stable regardless of the actions taken by the wearer of the electrodes.

We had no idea what we were getting into when we started the project, but we learned what we needed when we needed to. We found what we needed to do by experimentation, research, and an extension of theory. We worked through our problems (and there were a lot), and, we have a prototype to show for it. It is still a little temperamental, but, the biggest problem seems to be with electrode placement, and ensuring that all of the wires are properly connected (and grounded).

In the two groups I worked with, I found that each had its own way of going about assigning responsibilities and working through problems. I thought that my original method from the first quarter while working with team telemetry was a very good method of assigning responsibilities, but when I switched over to CFP's heart rate monitor project, they had a much more laid back, and less formalized method of assigning responsibilities, but it worked equally effectively due to the leadership of Michael Rubin.

I found that it was a lot of fun working with EE's, and I was very pleased that they shielded me from having to learn too much about how a heart beat is actually converted to a electrical impulse, and so on. This allowed me to concentrate on the digital side of the project, and deliver a working prototype.

I learned a lot about myself, about working in a team, about doing independent research, about meeting deadlines, and about seeing a project through from start to completion. I am a much better engineer as a result of this class.

8 ABET Questions

Economic Both the development costs and the realization costs of this project were inexpensive given the nature of the project. The cost to me to develop two boards[4] was 85$ for the

[4]This included two each of the programmer and the microcontroller

PCB, 20$ for the microprocessor and parts, and 30$ for the connectors. In total, two boards cost 135 dollars, fully assembled. This is very reasonable and would come down considerably if produced in bulk.

Environmental This project has little environmental impact

Manufactuarability This project is very manufacturable. It uses standard components along with a simple design.

Ethical This project has nothing to do with ethics.

Health and Safety This project furthers the health and safety of it's user

Social and Political This project will allow people with health conditions to safely play sports. This helps to make sports safer.

Did you know everything you needed to complete your project? Yes, and no. I didn't know everything, but I knew enough to be able to research it.

How did your group develop a plan for you? Our team leader drew a picture on the board. On it, it said, "Magic happens here". This was where I took over. It worked great. We stayed in close communication, and when it came down to connecting everything up, it worked.