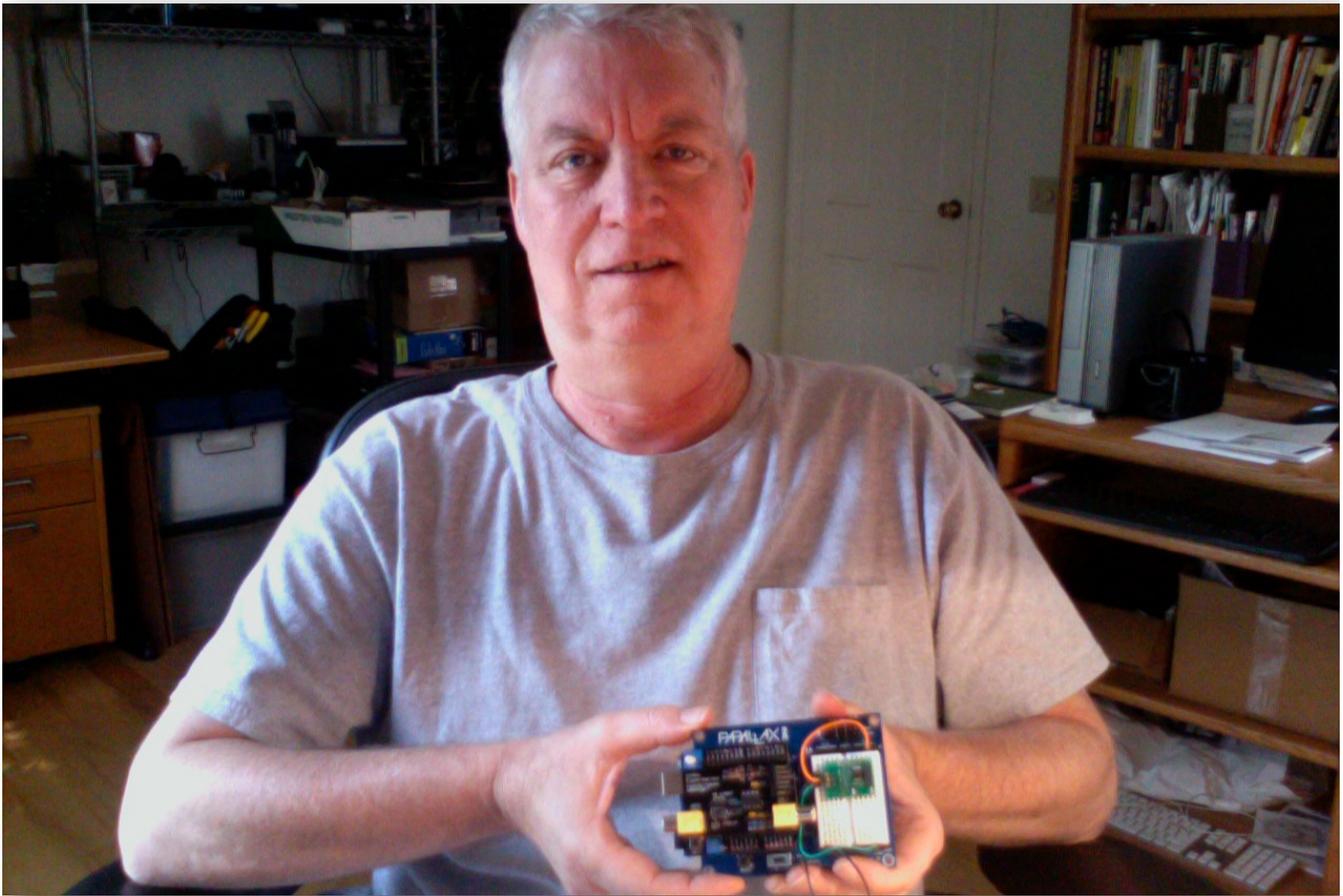


Parallax microMedic Contest Entry

Real-time TV Overlay of Vital Sign Sensor Data
Project ID: **micro13LK298**

Project Report



Lawrence Keyes
PO Box 194
Colchester VT 05446
lkeyes@mxdesign.net

microMedic Heart-Rate Monitor, TV Display

Introduction and Objectives

The genesis of my microMedic entry was a request from a research partner to overlay vital-sign readings (oxygen saturation and heart rate) on an existing video image of a patient being viewed in real-time over an internet video connection using our dedicated video endpoint that we call the DocBox. (Figure 1.) These units have been used for tele-exercise programs for participants who are managing chronic conditions such as chronic obstructive pulmonary disease (COPD), and diabetes, or those who have a fallen or have a fear of falling. In the figure 1 image, we would like to see each participant's heart rate and blood-oxygen saturation appear in a corner of their screen image.

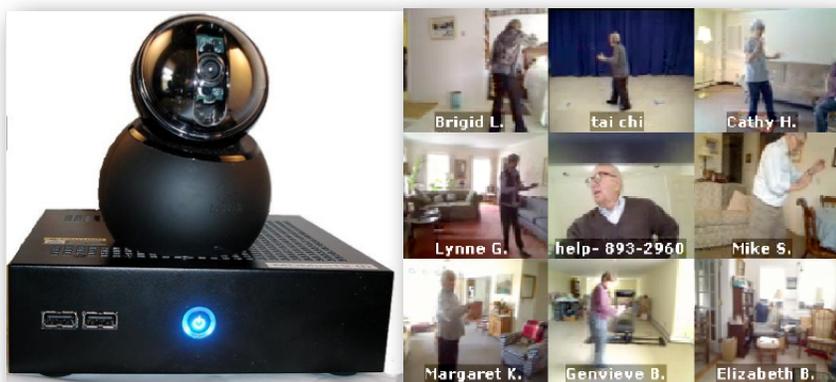


Figure 1. DocBox set-top videoconferencing and view of 8 exercise class participants

Having had some prior exposure to the Arduino, I applied for an Arduino Board of Education (BOE) kit. Given the relatively short time frame, (I started work on the contest in mid-June), it seemed prudent to limit the objectives of the contest entry to what was probably the easiest sensor (the heart rate monitor), and to investigate ways of overlaying the received heart rate data on to a video image.

Materials and Methods

I chose four components from the microMedic Arduino BOE kit:

- Arduino BOE
- Polar WearLink Heart Rate Transmitter
- Heart Rate Receiver
- 4-Digit 7-segment LED display

During the evolution of the project, the components were augmented with the [Nootropic Video Experimenter](#) board which ultimately provided the data overlay on the TV.

Heart Rate (HR) monitor

The Polar heart rate monitor consists of a chest strap transmitter and the heart rate receiver. The receiver is a pcb, 1" x 5/8" with three connections; 5v, GND and signal. The signal lead is connected to a digital input pin on the Arduino.

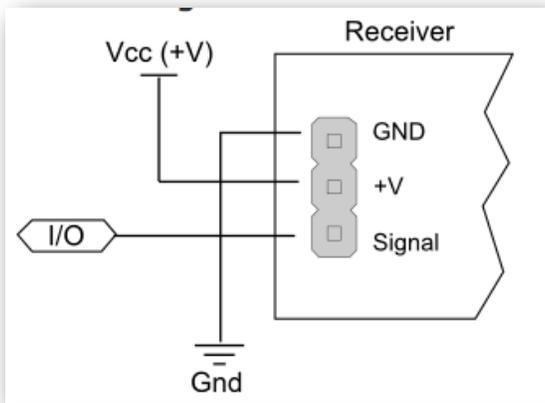


Figure 2: Connections for the Polar heart rate monitor

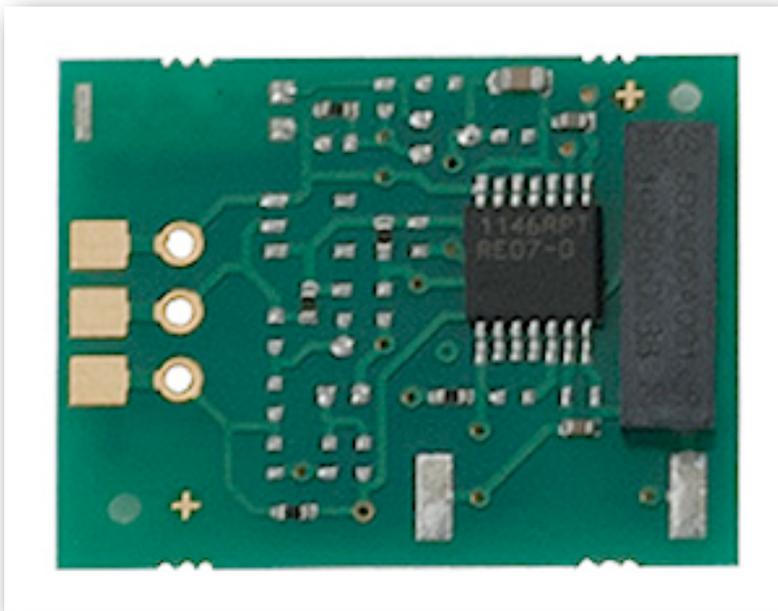


Figure 3: Parallax Polar Heart Rate Receiver

A three-pin header was soldered to the three available holes at the left of the receiver board and the board was inserted into the prototyping breadboard of the Parallax BOE shield.

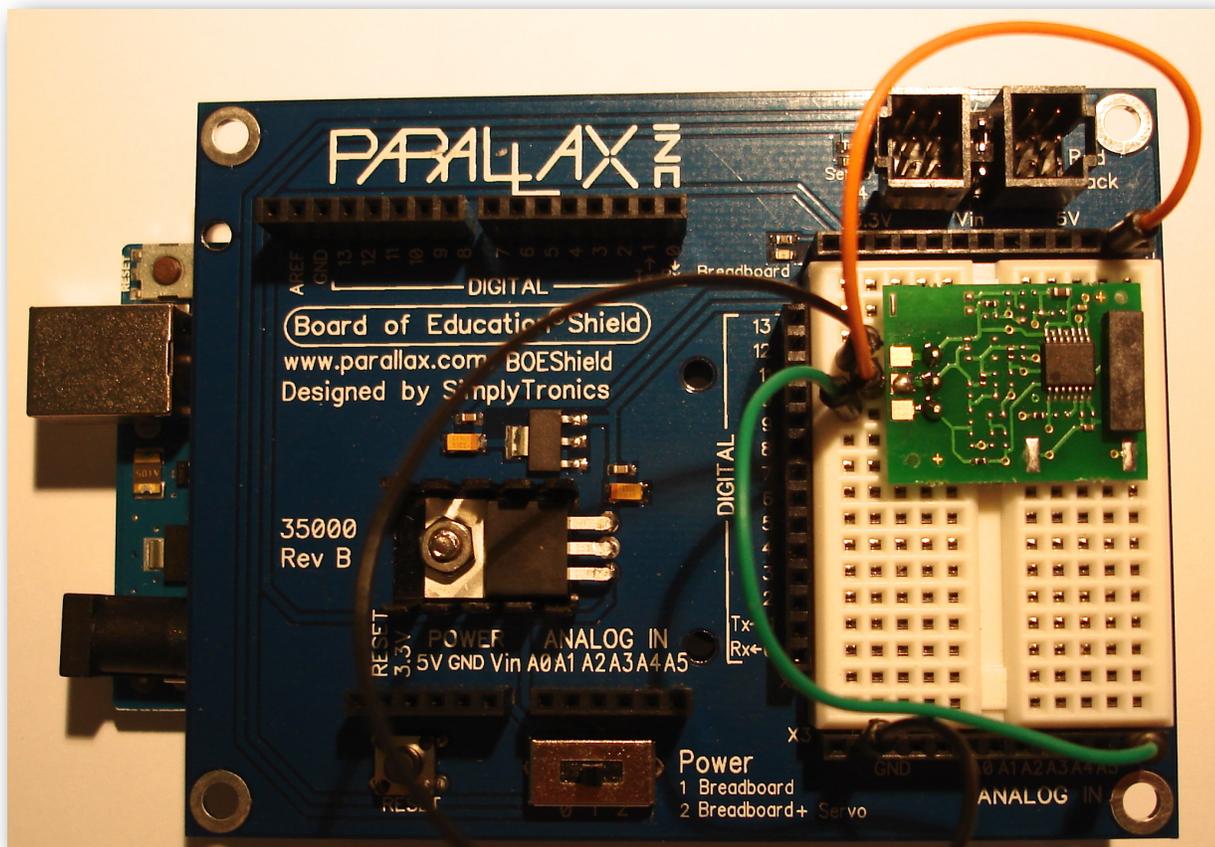


Figure 4: Heart Rate Receiver mounted on Board of Education Shield

Figure 4 shows the BOE mounted on top of the Arduino Uno. (You can see the USB and power connectors of the Uno to the left and underneath of the BOE.

The connections from the HR monitor, from top to bottom are as follows:

- Top (black lead) -> Ground
- Middle (orange) -> Signal, connected to Analog Pin 5
- Bottom (green) -> 5 volts

This configuration allows confirmation that the HR receiver is working using the sample Arduino code from Gordon Macomb provided by Parallax. This code is shown in Listing 1 and forms one of the software building blocks of the project. The code outputs the heart beat to the Arduino's serial port, where it can be monitored in the serial port window (Figure 5).

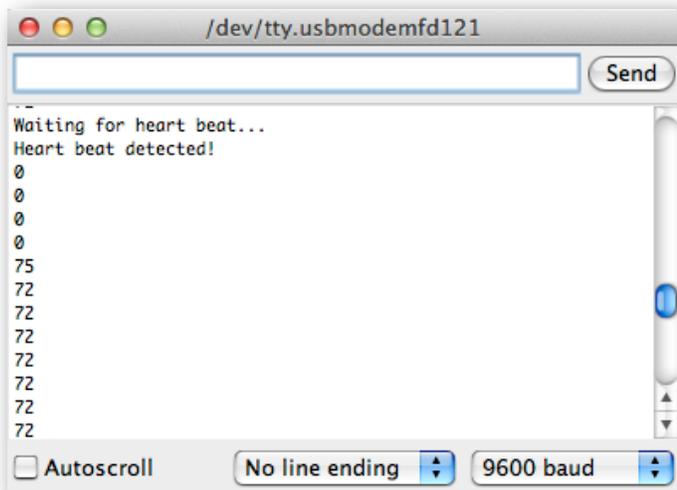


Figure 5. Serial monitor readout showing heart rate

As an Arduino novice, I found it helpful to be able to build the project in increments, working from a known state. The first modification was to add an LED to show the pulse. This resulted in Listing 2. This pulses an LED on pin 13, either an external one mounted in the digital pin header, or the internal one that is already on the Arduino Uno board.

The next modification was to add a buffer using five variables to provide a reading which is the average of five individual readings. The point of this is to smooth out the variations in readings. This code for this is shown in Listing 3.

As a side trip, I thought it would be helpful to connect the heart-rate monitor to the 7 segment display. I wired this up on a breadboard, (Figure 4) and modified the code to include an output to the display by incorporating kickstart code, again provided by Gordon Macomb of Parallax. The result is shown in Listing 4. This code builds on the previous code by incorporating three additional lines:

- An `#include <Multiplex7SEg.h>` to incorporate the library for the seven segment display
- A line in the setup routine to initialize the display, and assign the Arduino pins
- A line which displays the integer value of the hear rate

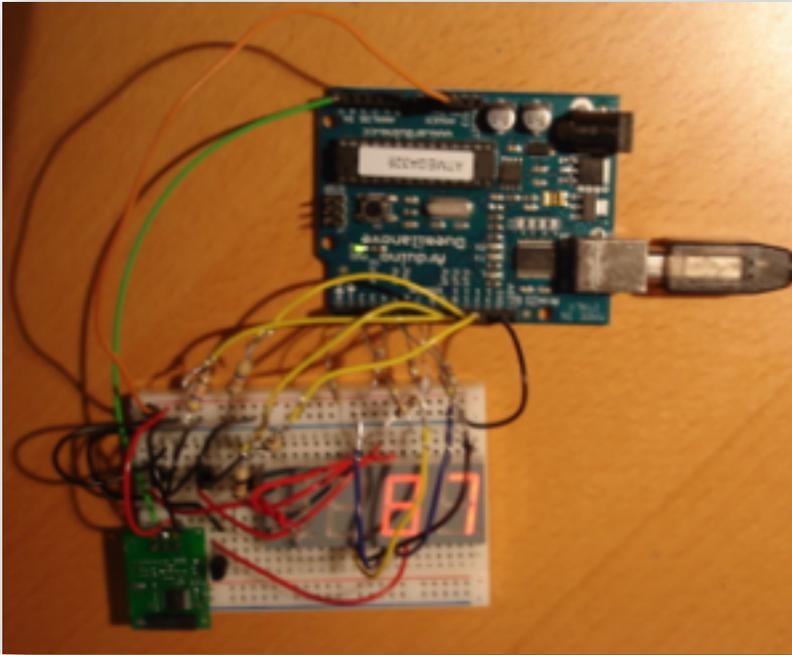


Figure 6: HR receiver (lower left) and seven segment display on breadboard, connected to Arduino

In all cases, I compared the heart rate shown in the serial port monitor or on the seven-segment display to my Polar wrist monitor. In general, I found that the output was within a beat or two of what was displayed on the wrist monitor.

Displaying the Heart Rate on the TV

Once I was confident that I had a reasonable ability to receive the heart rate, it was time to look for a way to project it on to a television screen. This was made possible using the TVOut library written by Miles Metzler described at <http://code.google.com/p/arduino-tvout/>

The TVOut code allows projection of characters and graphics using NTSC composite or PAL video. It uses two resistors and an RCA plug to connect the Arduino to the composite video input on a TV. (Figure 7)

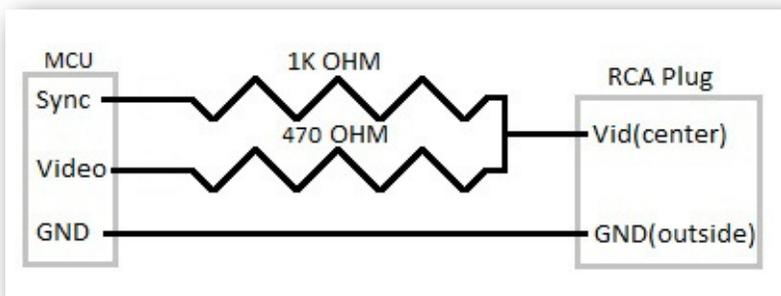


Figure 7: Arduino on the left, and RCA plug on the right
Sync is connected to digital pin 9

Video is connected to digital pin 7
GND connected to the GND on the Arduino board.

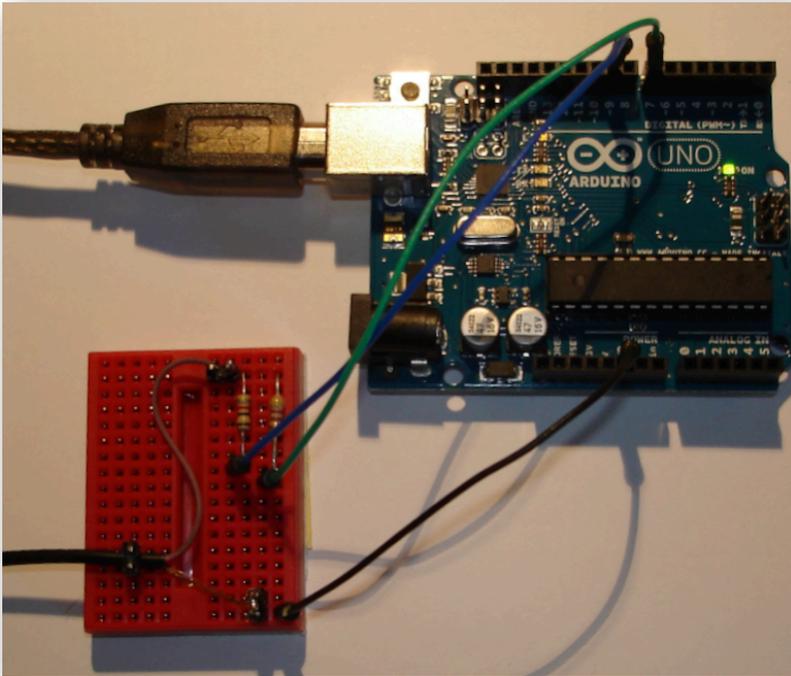


Figure 8. TVOut Connection to Arduino. Larger black wire at lower left connects to the composite RCA connection on the TV. Jumpers connect to digital pins 9 and 7 and to GND on the Arduino.



Figure 9. Simulated output on the TV, "HR 75" Low quality, but legible.

Listing 5 shows the modified TVOut code to display an integer on the TV screen.

Overlaying HR on to an existing video image

After researching options for video overlay, I found the [nootropic video overlay shield](#) (Figure 10) which uses an enhanced version of the TVOut library to overlay character-generated information over an existing composite video signal. The basis for the hardware design is the National Semiconductor [LM1881 video sync separator](#). Their assembled units were out of stock, but kit version was available, so I ordered and built it.



Figure 10. Nootropic Video Experimenter Shield

The Nootropic sample TVOut code generates a dramatic real-time display which includes a crawl on the bottom of the screen, an X,Y graph with points plotted in real time, and real time display of milliseconds in the upper left-hand corner. (Figure 11.)

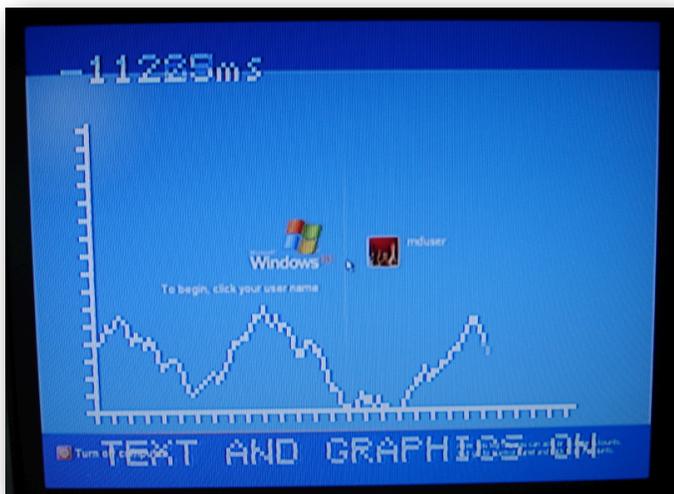


Figure 11. Data overlay on to a Windows log-in screen displayed on a TV set.

Putting it all together

The final hardware “stack” (literally), consists of the Arduino Uno, the Parallax Arduino Board of Education shield, and and Nootropic Video Experimenter shield. (Figure 12.)

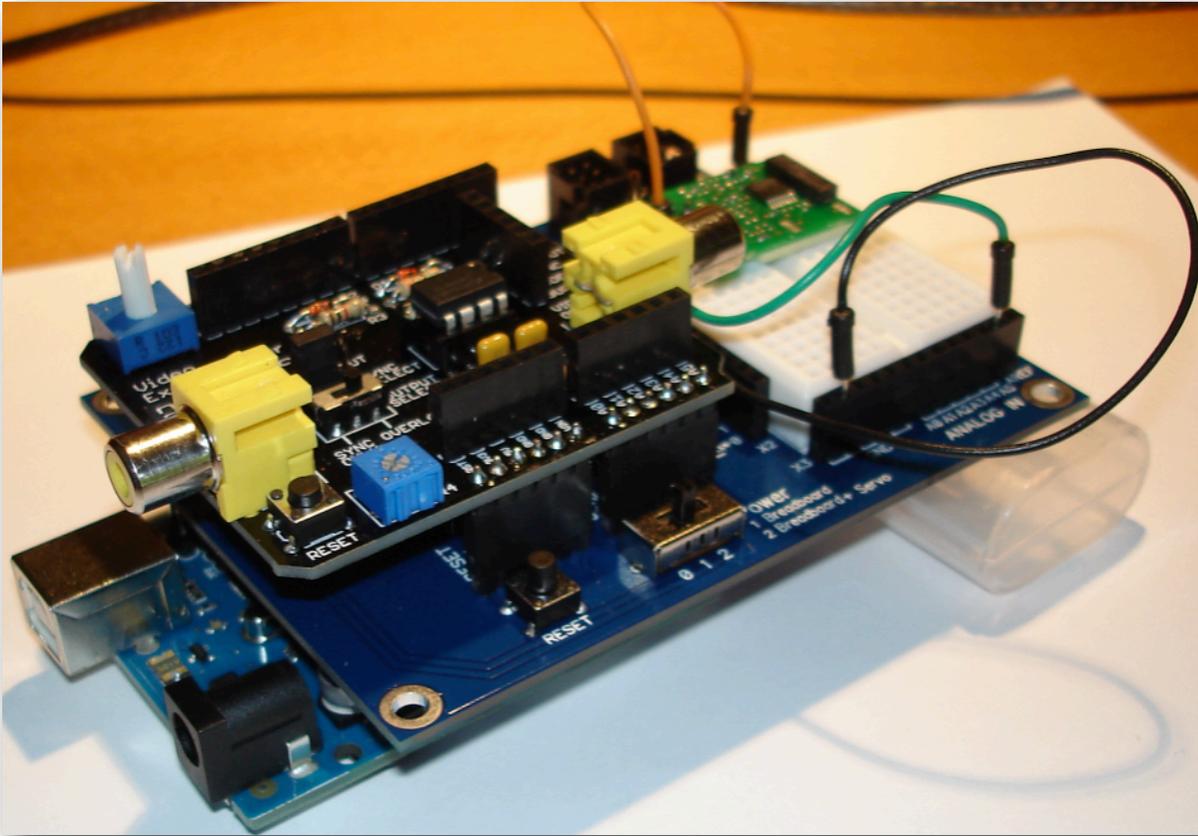


Figure 12. Hardware Stack. Arduino at the bottom, BOE in middle, Video board at the top.

I combined the code sections from the Nootropic TVOut library with the sections for receiving the heart rate, and the buffer code. This results in the final Listing 6.

Figure 13 shows the output of the HR monitor compared with the Polar wrist monitor.

Figure 14 shows an example of a “patient” image during a video conference which displays the heart rate as part of the image.



Figure 13. HR monitor output compared with Polar wrist monitor.



Figure 14. "Patient" image overlaid with HR reading

Discussion

While working through the development process several areas of further development and improvement were identified:

Packaging

Because of the limited range of the Polar heart-rate transmitter, it seems probable that the receiver module should be located on the user's body, that the receiver be battery powered, and that the receiver then transmit wirelessly via Bluetooth or X-Bee to the DocBox for integration into the video stream. This might best be served with a custom circuit board, and a wearable case which incorporates batteries.

Measurement Accuracy

There is considerable discussion on the Parallax discussion board and elsewhere about the best way to verify (and guarantee) accuracy of the heart rate. These include:

- Increase the buffer to ten positions thereby doubling the number of readings used for the moving average calculation of the heart rate.
- Include a calculation that throws away readings outside a normal range of 40-220.
- Use one of the two hardware interrupts available on the Arduino to guarantee priority for the HR monitor over other processes.
- Take EKG readings at the same time as the heart beat readings and compare the EKG information with the HR information.
- Dedicate one processor to reading the HR and a second processor to deal with the video display.
- Use an Arduino Mega, to provide increased processing speed.

Video Overlay

When the Nootropic Video Experimenter board overlays data on to a composite image, it inserts the data in series with the composite video signal before the signal is sent to the TV for display. Essentially, it performs the function of a video mixer. In the next iteration, we would like to be able to perform the same function with an image from a webcam sent connected via USB to a computer. In this case, however we want the mixing to take place before the image is processed by the videoconferencing software, so that the data appears as part of any image exchanged between video endpoints.

The Nootropic Video Experimenter board is limited to composite video only. Ultimately, it would be helpful to display the data at higher resolution on a more typical VGA, DVI, or HDMI video screen.

Additional sensors

The original request was for a pulse-oximetry display to be shown on the patient's video image. This would be the patient's pulse, and the blood oxygen saturation. A pulse-ox sensor is included in the microMedic package, and is the next candidate for development.

Evaluation

Comments on the Judging Criteria:

- *USE OF MICROCONTROLLER: Does the project use a microcontroller to function? Winning applications will use a microcontroller as the brain of the application.*

The project uses the Arduino Uno. As mentioned in the discussion section, it is an open question whether another platform may be more suitable. The Arduino Mega has a higher processing speed, the Raspberry Pi includes HDMI already configured on its PCB, and the Propeller has the ability to run independent processes that would allow separation of the sensor monitoring code and the television display code. (The latter will become more critical as additional sensors are added).

- *INNOVATION: Is the prototype concept creative and forward thinking? Does the prototype show a cheap and easy way to do something that was previously difficult and expensive?*

While no explicit design costs were projected, an informal target to add this capability to our existing DocBox, has been US\$100.00 or less per unit. The component cost (Qty 1) of the current prototype was about \$125.00. By way of a price comparison, a Nonin Blue-Tooth pulse-oximeter which is clipped to the patient's finger has a list price of US\$395 for the sensor alone, and does not include the receiver. The Nonin unit is relatively bulky, and is not suitable for use when the patient is actively performing an exercise program.

- *EXECUTION: Is the prototype well-built or impressive in its operation? How well is the project documented? Submittals should be typed, organized and spell checked. Pictures and video must be in focus. The project report should be clear and well written.*

The prototype has been built from modular components which appear to work reliably. Results displayed by the unit need further verification but were accurate when evaluated against a Polar wrist strap.

- *UTILITY: Is the prototype potentially useful? Does the vision for the prototype advance military medicine?*

The project was an initial feasibility and learning experience which revealed tantalizing opportunities for refinement.

Opportunities for order-of-magnitude cost reductions for vital-sign sensors are on the horizon using commercial off-the-shelf components packaged with modular open-source software.