

Project Proposal: Black Box  
Michail Turovskiy  
[mturovskiy@gmail.com](mailto:mturovskiy@gmail.com)  
Done for: CSCS299, Spring '08

#### Project Abstract:

The idea of this is to be a black box for use inside a car. It should record things such as time, current velocity of vehicle, acceleration, and GPS coordinate. It should all this info for future parsing and data mining. It should have a removable primary storage device. It should have a small profile; ideally it should be no bigger than a GPS navigational unit. It will not have any display devices initially. It will draw its power from batteries. Block-2 device should have a display for the velocity/location of the device and should be powered via batteries or the car charger. The use-case scenario for this device is that a person places it in his car, turns it on and drives. At the end of the day, the primary storage device is removed and, once the user is home, uploaded to their computer. This will give a verifiable record of where the car has been and when.

#### Hardware strategy:

I have identified the main components of this device.

The overall microcontroller shall be the Zilog Z8 provided in class and the development board for it. This is done for practical reasons; this is the only board that I know how to use! It has enough GPIO pins for all the sensors, timers to read them and standardized power supply of 3.3V.

A main memory storage device that will store all the data being logged by the sensors is also needed. This needs to have capacity in the Gigabytes, and be easily readable by a PC. For this purpose I have chosen to use an SD card format. I have originally wanted to write my own FAT16 interface but was afraid that I will not have enough time to finish the rest of the project in the time allotted. So I have decided to use a file-system on a chip; uAlFat. The manual for the device is available<sup>1</sup> and it is available online<sup>2</sup> (\$31.95). It will interface via the UART interface and will take an SD card as the removable memory device. What it does for me is abstracts the file system interface so I can write commands as-if I had a dos prompt, without implementing it myself. I have chosen this over Dos-in-a-box because according to online forums, it seems easier to use. Supposedly, this will plug right in and you can debug most things via hyperterminal. It takes 3.3V VCC and needs 16 pins.

I will be using a SiRF GPS<sup>3</sup> unit to track where the device is. This device was chosen because it was relatively cheap (\$47.99) and appeared to do everything I wanted: operates at 3.3V VCC, with a power-draw of 75mA. Other similar devices were a lot more

---

<sup>1</sup> <http://www.ghielectronics.com/download/uALFAT/uALFAT%20Manual.pdf>

<sup>2</sup> [http://www.saelig.com/miva/merchant.mvc?Screen=PROD&Product\\_Code=FF003&Category\\_Code=FF](http://www.saelig.com/miva/merchant.mvc?Screen=PROD&Product_Code=FF003&Category_Code=FF)

<sup>3</sup> 20 Channel EM-408 SiRF III Receiver with Antenna/MMCX, from  
<http://www.usglobalsat.com/p-47-em-408-sirf-iii.aspx>

expensive on SparkFun. Furthermore, this receiver has WAAS support, which gives a much tighter error estimate in optimal conditions. It includes built-in patch-antennae (it has an optional connection for an external MMCX antennae but I don't particularly want to use it) and connects with only 5-pins, of which only 2 are used for unidirectional serial communication. I am not sure exactly how that works or what kind of throughput that will give me so that is something that concerns me somewhat; I'd like to be able to get the data out real-time. The device outputs standard NMEA 0183 sentences for GPS coordinates, as well as SiRF proprietary binary format<sup>4</sup>. The data sheet is available as well<sup>5</sup>.

Reading acceleration will be accomplished with a triple-axis accelerometer<sup>6</sup>. I have looked at several products and considering I don't really need anything too fancy, any would do. It is likewise available from SparkFun<sup>7</sup>. It has input VCC rated at 2.2-3.6V, which is in range of the Zilog Z8 board and draws only 50uA in low-power mode. I have not been able to find the power draw for standard operations but I assume it can't be that much more. It comes in 16-pins packages, of which only 9 are connected. It needs only 3-pins for data output since it just writes the acceleration along the X,Y,Z axes. The only issue here is that the surface-mounted component comes in at \$11.80 while the breakout board is \$39.95; which to me is slightly ridiculous. I can probably solder the surface mount device myself if I had a PCB board to do it on. I doubt I'll be able to solder wires to the pinouts. I have not looked at how much it would cost to fab out a PCB that'll let me surface-mount the component, nor do I know how much time it will take to get it to me.

Reading the velocity will be done in software; assuming that when the device is turned on, there is no acceleration. The formula is:  $V_f = V_0 + A * T$ , where  $V_f$  = final velocity,  $V_0$  = initial velocity,  $A$ =acceleration, and  $T$ =time. One potential problem is how accurate this will be in a non-discreet system. I have no idea what kind of error margins to expect here. Also, timing will be an issue as well. The system must be able to keep up, and be able to solve this equation every small interval  $dT$ , for some value of  $dT$ .

Keeping the time will be done via a real-time clock integrated circuit from Intersil<sup>8</sup>. It has a battery-backed up SRAM that keeps the time when the main power is off. It is available from DigiKey for \$3.03<sup>9</sup> and has an available datasheet<sup>10</sup>. It comes in a 16-pin package of which only 9 pins are connected. It uses 1 bidirectional pin to communicate (via SDA). There were many similar real-time clocks but a lot of them could only be ordered in bulk and I've chosen this one because I could order it in single-digit quantities. One problem is that it requires an external 32.768 kHz quartz crystal to work. I have only found that out by reading the data sheets for this part; it was not listed anywhere else, which I thought

---

<sup>4</sup> [http://www.usglobalsat.com/downloads/SiRF\\_Binary\\_Protocol.pdf](http://www.usglobalsat.com/downloads/SiRF_Binary_Protocol.pdf)

<sup>5</sup> [http://www.usglobalsat.com/download/47/em408\\_v1.1.pdf](http://www.usglobalsat.com/download/47/em408_v1.1.pdf)

<sup>6</sup> Triple Axis Accelerometer MMA7260Q

<sup>7</sup> [http://www.sparkfun.com/commerce/product\\_info.php?products\\_id=308](http://www.sparkfun.com/commerce/product_info.php?products_id=308)

or

[http://www.sparkfun.com/commerce/product\\_info.php?products\\_id=252](http://www.sparkfun.com/commerce/product_info.php?products_id=252)

<sup>8</sup> ISL1220IUZ integrated real-time clock

<sup>9</sup> [http://www.newark.com/50M3935/supplier-direct-ship/product.us0?sku=INTERSIL-ISL1220IUZ&\\_requestid=33532](http://www.newark.com/50M3935/supplier-direct-ship/product.us0?sku=INTERSIL-ISL1220IUZ&_requestid=33532) ; SKU#: 50M3935

<sup>10</sup> <http://www.intersil.com/data/fn/fn6315.pdf>

was cheap! So I have found a suitable crystal from DigiKey<sup>11</sup>. I have chosen the through-hole package because I don't want to deal with anymore SMT components. The datasheet is available<sup>12</sup>. It is \$0.63. I am not sure how to connect the crystal with the clock yet; that is something I need to figure out.

Software strategy:

First and foremost, I must figure out how to interface with the UalFat device and get FAT32 on the chip. The reason I want FAT32 was for the longer file names. The plan was that on startup, the device would create a new directory with the name: YY\_DD\_HH\_MM, where Y=year, D=day, H=hour and M=minute. All the data until the device has been turned off will be recorded inside this directory. Furthermore each file inside the directory would correspond to some small time interval  $\Delta T$ . Then they would be named with DD\_HH\_MM\_SS\_GPSCOORDINATES\_ACCELERATION\_VELOCITY name scheme. So that nothing would need to be stored inside the file and all the information would be in the filename; sort of like \proc. This would make some pretty long file names and FAT16 would not be sufficient.

Additionally, I would need to implement a routine to figure out the current velocity of the black box, based on the time elapsed, and the history of acceleration. The formulae have been given above and it should be trivial to implement. Except that I would like to avoid using floating point arithmetic as much as possible for speed. The hard part here will be getting the timing correctly and synchronizing everything.

Finally, I will need a NMEA parser to read the output of the GPS unit and put it in the right mode. This should not be too difficult as it spits out binary data.

Development strategy:

My development strategy is to integrate every component separately.

- \* First I would like to get the FAT32 system up and running and be able to read/write to SD disk and have it be recognized by a PC. Testing for long filenames is essential.
- \* Then, I will need to integrate the GPS unit and read its output and add it to the filenames being generated.
- \* Then, adding the acceleration module and getting some hard numbers for the velocity calculation to figure out how often I can poll the accelerometer.
- \* Then adding the clock module and getting it to output time correctly
- \* Then, integrating all 4 modules and generating properly named files.

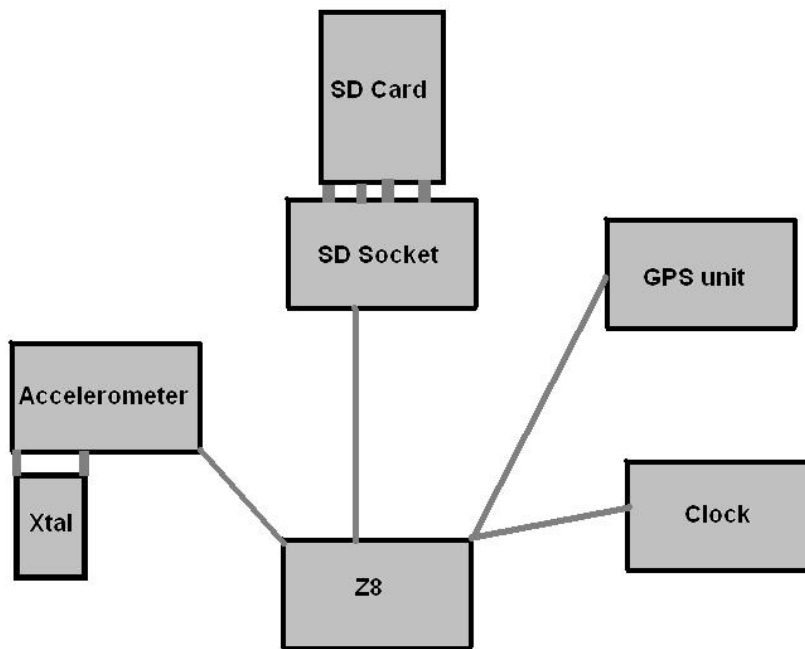
The hard part in integrating everything will likely be getting the timing correctly; I am not sure Z8 will be able to keep up to the GPS receiver and read it in real-time, and still read the accelerometer and the clock and write out the file on the SD disk. So chances are I will need to figure out some kind of timing loop wherein I read every other NMEA sentence. That is an unknown for now.

---

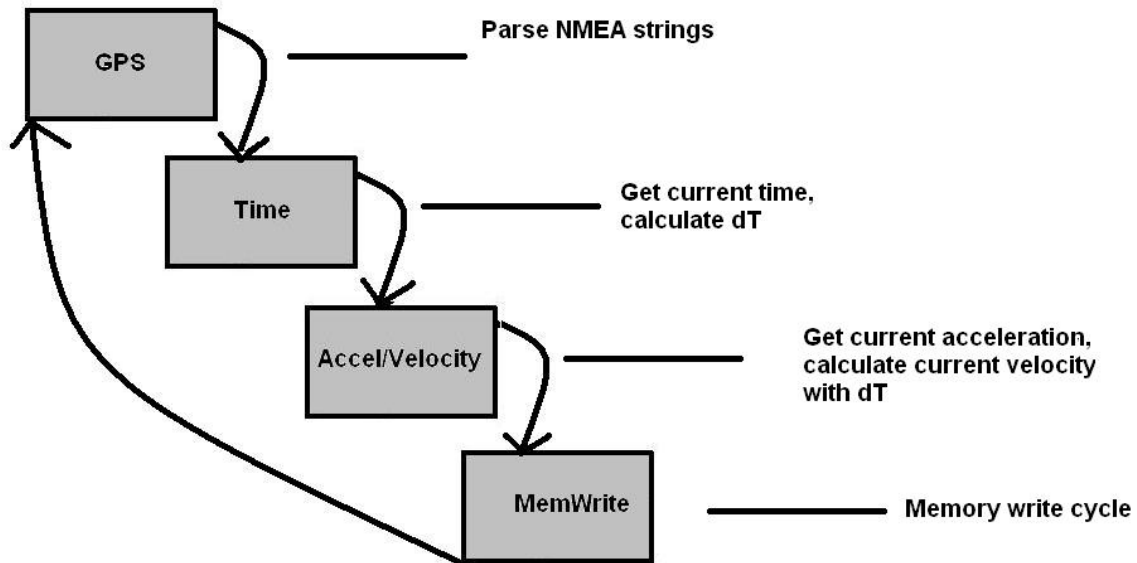
<sup>11</sup> digikey part #: X1125-ND

<sup>12</sup> <http://www.ecsxtal.com/store/pdf/ecs-31x.pdf>

Architectural Design:



*Hardware architecture*



### *Software architecture*

#### Resources:

- Zilog Z8, as provided in class
- 32 pins on Z8 will be used, most for GPIO; some will be duplicated VCC/GND
- SD socket + SD card
- SiRF EW-406 GPS unit
- Accelerometer
- Intersil real-time clock IC
- 32.768 kHz quartz crystal

I will also need assorted resistors/isolating capacitors but have not had time to figure out exactly how many/what yet.

#### Timeline:

I am ordering the parts this week and would like to start integrating ASAP, because it seems like a lot of work.

#### Power supplies:

I do not know how I will drive this setup. I believe the current provided by the development board will be enough. If I have time, I would very much like to make it portable and run it off a battery. I have not investigated it at all. The draw rate of the GPS + accelerometer is approximately 75mA.

Unknowns:

I have no idea how long this project can be expected to put together. I usually have very optimistic estimates for software projects so I imagine my estimating abilities when it comes to hardware are not any better. To reduce this risk I will have to start as early as possible; ordering the supplies this week so I have enough time.

Also, I do not know if Z8 will be able to keep up with every device and be near-real-time. A potential problem is that after I finish or get close, I find out that Z8 isn't fast enough. I refuse to think about this possibility; that would be a nightmare!

I am pretty worried about the memory system. I have read the manuals over how to connect the device but can not definitively say that it should work for me; they provide a FAT32 library for use with their chips<sup>13</sup> but it will need to be ported to Z8 and I don't even understand why I'd need it because it should be pre-loaded when I get it.

---

<sup>13</sup> [http://www.ghielectronics.com/download/uALFAT/uALFAT\\_lib.zip](http://www.ghielectronics.com/download/uALFAT/uALFAT_lib.zip)