Biomedical Signal Analyser

Jinu Jayachandran

December 2017

Contents

List of Figures

1 Introduction

The display of signals generated by various sources in an efficient manner is of great importance in the field of research. There are instruments which can display the signals at different levels of accuracy. The more accurate the instrument is the more costlier it becomes and the relationship is exponential. Also the range of frequency of signals (or in other words the bandwidth of the signals) is also an important factor. But is it possible to make an in-house display for relatively small frequency signals using minimum components and less cost?. This possibility is being explored in the project. The scope of the project is mainly confined to displaying biomedical signals which will range only in hundreds of frequency. Also the project aims to display two signals at the same time.

2 The System

For the display of analog signals the three important components needed are the sampler, the processing unit and the display. The system can also be designed in such a way that sample-processing unit will itself form a single entity or even better, all three are in the single system. In this project all the three are three different units and are handled individually.

2.1 Analog to Digital Converter (ADC)

An ADC will perform the function of a sampler. Depending on the range of signals to be displayed an appropriate ADC can be chosen. By 'appropriate' it means that a decision should be made on what should be the sampling time (or speed), resolution, voltage range, interface type etc of the ADC. Currently in the first phase of the project ADC is not used. Instead the waveform to be displayed is generated in the processing unit and sent to the display.

2.2 Processing Unit

The three main tasks of the processing unit are - Receive data, Process data and Send data to display. The data that is being received from the ADC can be through an interface. The interface can be a serial interface like UART, SPI, I2C etc or a parallel one. To process data received there should be a unit which can analyze the data and modify it so that it can be displayed suitably. Thus a microcontroller is one of the choices of the processing unit. In this

project we are using Raspberry Pi Rev2.0Model B [3] as the processing unit. Fig(1) shows the Raspberry Pi Rev 2.0 Model B which is used in the current project.

Figure 1: Raspberry Pi

The Raspberry Pi can be considered as a minicomputer with lot of interfaces. Even a monitor and keyboard can be attached to pi and can be used as a normal computer. But the resources to the pi are limited. The only memory it can have is via an SD Card. The pi can only work with a custom OS. The most commonly used OS is raspbian [4]

2.3 Display

The display that is used is uLCD-28PTU from 4D systems [6] (shown in fig(2)) The display is controlled by PICASO processor which can be configured. The product also comes with a software from 4D systems. Although the display can be configured by different means the current project is concentrated on controlling the display via serial commands. The detailed documentation of the product along with the list of serial commands available are listed in the documentation for PICASO Serial Environment Command Set [7]. To play with the display unit you can make use of the 4D systems software(running in desktop/laptop) connected to the display directly without using raspberry pi. The display can be connected to a system either via a USB to serial converter. Some sample examples are given at the begining of the document in [7]. The pins of the `H2` group (as shown in fig(2))in the display shall be connected to the USB to Serial converter. The connection diagram between the display and the USB to Serial converter is as shown in the fig (3).

Figure 2: Display:ULCD-28PTU

The software can used with the USB to Serial converter only for learning

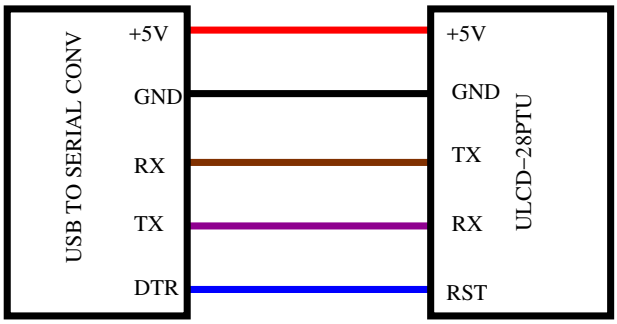

Figure 3: Connection diagram for USB to Serial Converter and Display

purposes. To make a stand alone device the display needs to be interfaced with other stand alone systems like Raspberry Pi.

3 Interfacing Raspberry Pi

Before making the complete product it is necessarry to have a setup in which the system can be programmed, debugged and analysed. This setup mainly concerns with how the Pi (which is connected to the display) can be interfaced to a computer.

3.1 Installing Raspbian

As mentioned before an OS, Raspbian, shall be installed in the SD card of the Pi to perform the required functions. Although 'bare metal' programming of Pi is possible it is a complex proceedure in addition of need of extra hardwares for programming. A forum dedicated to the bare metal programming of Pi is available here. But here we will focus on programming via the OS.

Any image burning software can be used to burn the ISO image downloaded from the Raspbian site. A recommended software is Etcher. A detailed explanation on how it can be used to burn the image can be found in here [1].

Once the image is burned into the SD card, make an empty file named *SSH* in the SD Card in the topmost directory. Raspberry Pi version 2.0 has SSH disabled by default. We are making this change to enable it. The SD card can then be inserted to the PI and then can be powered up. If a monitor with a HDMI port and a usb keyboard are available they can be directly plugged into the pi for interactions. But as these extra items were unavailable at the time of the project the Pi is configured to program remotely by SSH.

3.2 Connection Raspbian Pi via SSH

Placing the empty *SSH* file is of utmost importance for connecting the Pi, without keyboard or monitor, to a computer. The IP address to Raspberry pi can be assigned in two ways

3.2.1 Static IP address allocation

1. Insert the SD card to the computer (via a card reader) running on linux. It shows all the folders in the original OS.

2. Go the file `/etc/network/interfaces`. Replace the line `iface eth0 inet dhcp` with `iface eth0 inet static`

3. Add the following lines after the previous step

   ```
   address=<ip address>
   netmask=<net mask>
   network=<network address>
   broadcast=<broadcast address>
   gateway=<gateway address>
   ```

   where the $'<>'$ indicates the values to be given. Make sure the computer to which the Pi is connected is also in the same subnet mask.

4. Go the file `/etc/dhcpcd.conf` and add the following lines at the end of the file

   ```
   interface <interface name eg:  eth0>
   static ip_address=<ip address>
   ```

5. Insert the SD card to the Pi and boot. Once booted up, connect Pi to computer via ethernet cable. For SSH login to the Pi, use software like *putty* in windows. In linux use the ssh command (for eg: `ssh pi@<ip address>`). The default username is `pi` and the password is `raspberry`.

The disadvantage of this method is that Pi will not be connected to the internet if you want to download and install any packages. So a better solution is using dynamic IP allocation.

3.2.2 Dynamic IP allocation

Here WiFi connection will be shared with the ethernet port for dynamic IP allocation and making available internet to Pi.

1. After burning the image and putting an empty *SSH* file, connect the SD card to Pi and connect Pi to the computer, with a WiFi connection, via ethernet cable. Here it is assumed that the file `/etc/network/interfaces` has `iface eth0 inet manual` as default setup. If there is failure of connection, then the file needed to be manualy edited to `iface eth0 inet manual` with all the IP address commented throught the steps mentioned in previous section.

2. Make sure the Wifi is connected to some network. Share the WiFi adapter to share its internet connection with the ethernet. Make sure the IP address are kept as *Obtain IP address dynamically* for both the WiFi and ethernet. The system will assign an ip address to the ethernet port and the device connected to it automatically. That is raspberry pi will be assigned an IP address.

3. To find out what is the ip address assigned to Pi, first get the IP address of the ethernet port of the computer (this can be obtained by giving `ipconfig` command in windows). Install `nmap` from here. If the ip address obtained for ethernet is `a.b.c.d` then given the command `nmap -sn a.b.c.0/24` in the command window. It will lists all the devices within this range connected to the ethernet port along with the name of the device.

4. Identify the ip address of Pi and connect it via putty.

The current configuration of Pi with static ip address of 192.168.137.2
A python program will be running in the Pi which controls the display.

4 Interfacing Raspberry Pi with ULCD-28PTU

The Pi is interfaced to the display via UART. The schematic of the PI can be found in the website here [5]. Only the relevant portion of the schematic is shown in fig(4)

The connectors shown in cyan colored box in the fig(4) are connected to the connectors in yellow colored box in fig(2). The fig(3) will be the same except that the USB to serial converter block will be replaced by Pi and the `DTR` pin of the converter will be replaced by `GPIO_GEN1` pin of the Pi for resetting purposes.

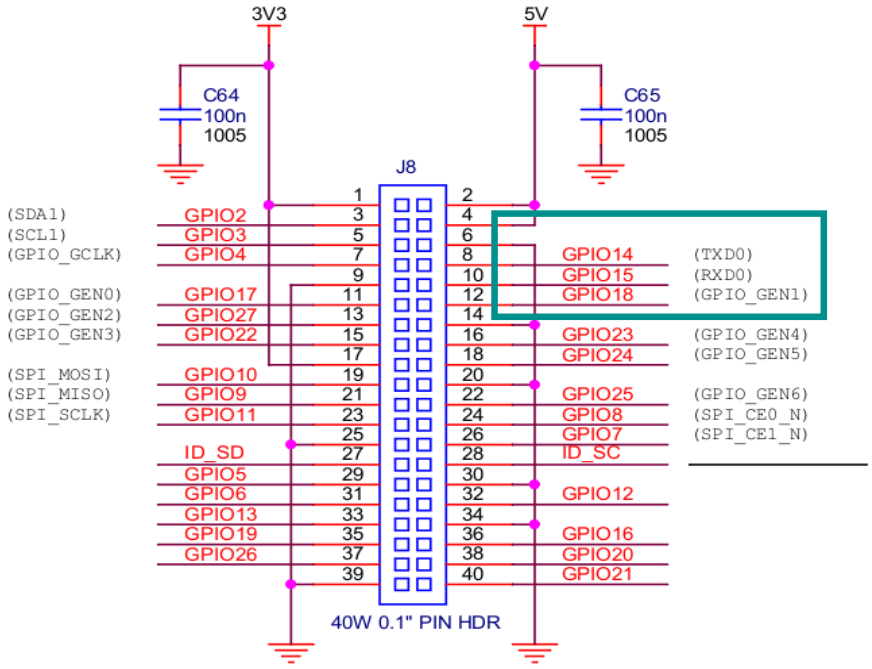

Figure 4: Schematic of Raspberry Pi for connecting with the display

4.1 The Code

Once these connections are made each and every pixel of the ULCD display can be controlled by serial commands send from the Pi. A sample program has been done in the github. The link is here [2]. The `dso.py` file have the latest code of displaying two sine waves simultaneously in the display. The comments of the code gives information of the algorithm used. After proper connections are made run the code for the display.

5 What's Next?

The system can be improved as follows

1. Improving the current algorithm. The current algorithm has some amount of delay in showing one waveform after the other. The main reason is because the screen is cleared after displaying one complete wave. The algorithm can be improved so that we will observe a standing wave. The logic of trigger functionality of the practical DSO's can be made use for the same.

2. Interfacing of external ADC to Pi via SPI or I2C.

3. Accumulate, Process and Display the values obtained from ADC. The current code can be modified in such a way that instead of generating sine values, the values can shall be obtained from ADC, convert it to voltage and scale the values to display on display.

4. Finding the parameters of the wave such as frequency, peak voltage, average voltage etc and displaying them are not yet explored.

5. The ULCD-28PTU display is also a touch screen. There are serial commands which allows to get the values of pixel when touched. This is can be used to make it more user friendly. For example by scrolling the screen increases resolution of the signal displayed.

References

[1] Etcher. Installing raspbian using etcher. `://www.raspberrypi.org/magpi/pi-sd-etcher/`.

[2] Jinu Jayachandran. Simultaneous display of two sine waves. `https://github.com/jinujayachandran/dso`.

[3] Raspberry Pi Main Page. Raspberry pi foundation. `https://www.raspberrypi.org/`.

[4] RASPBIAN. Raspberry pi foundation. `https://www.raspberrypi.org/downloads/raspbian/`.

[5] Schematic. Raspberry pi schematic. `https://www.raspberrypi.org/documentation/hardware/raspberrypi/schematics/Raspberry-Pi-2B-V1.2-Schematics.pdf`.

[6] uLcd 28PTU. 4d systems. `http://www.4dsystems.com.au/product/uLCD_28PTU/`.

[7] PICASSO uLcd 28PTU. ulcd-28ptu - serial command documentation. `http://www.4dsystems.com.au/productpages/PICASO/downloads/PICASO_serialcmdmanual_R_2_0.pdf`.