DigITal Scoreboard

Final Documentation

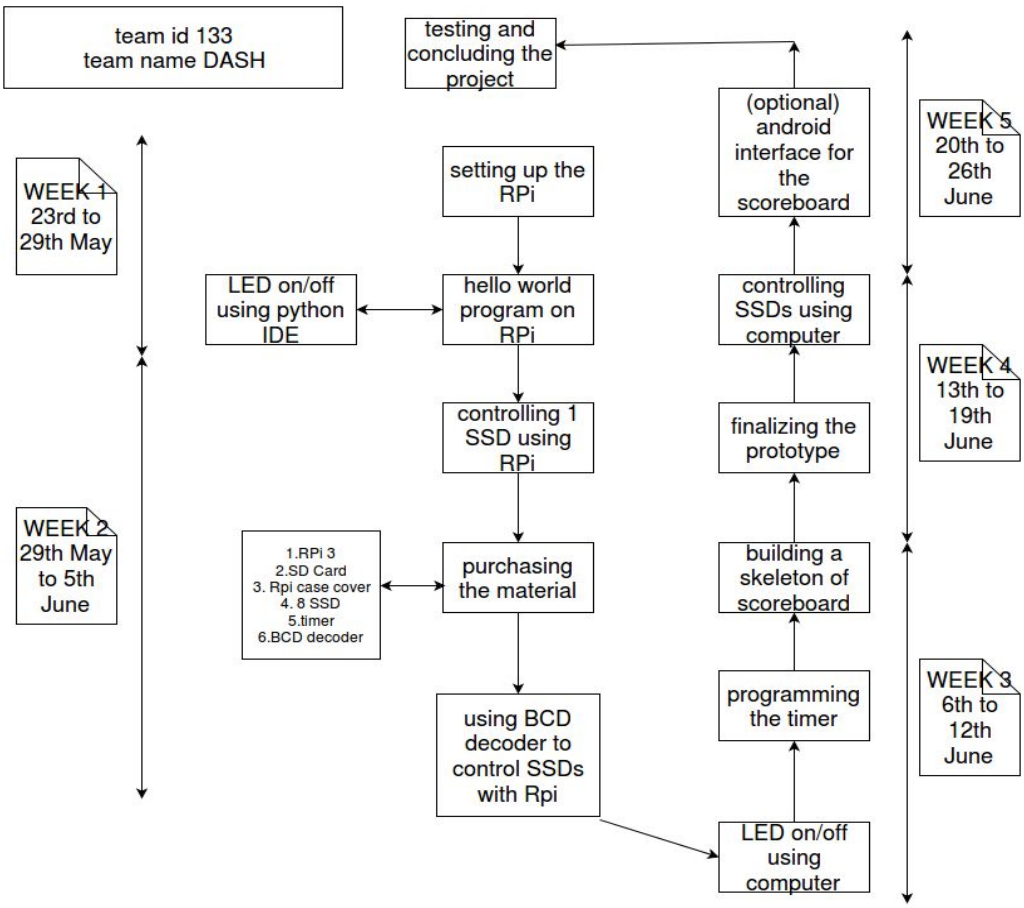
Team name : D.A.S.H
Project name : Digital scoreboard

Team members :

- Deep Tavker
- Hrushikesh Loya
- Ayush Khandelwal
- Suyash Vyas

Abstract : We have made a versatile digital scoreboard which can be used for volleyball, basketball, water polo and football . It can be remotely controlled by an android app. The operational backbone of the score-board is Raspberry pi.

Action Plan :



Hardware used by us

- Raspberry Pi 3B (comes with inbuilt Wi-Fi and bluetooth) - 3200/-
- Micro SD card (8GB grade 10) - 350/-
- HDMI cable, Mouse, keyboard and a monitor (for initial setup of Rpi)
- USB - Micro USB cable (for powering up RPi)
- Laptop
- Basic electronic components like breadboard, jumpers and solder gun.
- BCD IC, Demux IC, TM1637 module, SSDs(of various sizes) - 1500/-
- Acrylic sheets (for the score-board skeleton)

Raspberry Pi Basics

We started off with a Raspberry Pi - 3B model. Except the peripherals part (which is mouse, keyboard and digital output) RPi is essentially just like a traditional computer. Useful documentation for RPi can be found on its official website (follow this link for ‘booting up’ your RPi).

In case it doesn’t boot up, first check if the HDMI cable if working. If yes, the problem is probably with the SD card.

RPi terminal is same as your linux terminal just on a different OS (Raspbian) which means you can use all the linux commands here.
Here’s a quick guide for bash commands. Configure your RPi using this link.

GPIO pins

GPIO pins stand for general purpose I/O pins. These are what we mainly used for our project. Basically these are easy-to-use pins for making a circuit with fairly involved logic.

Read about GPIO pins and led blinking here. Also, you’ll need basic python knowledge for coding this and running your code through the terminal.

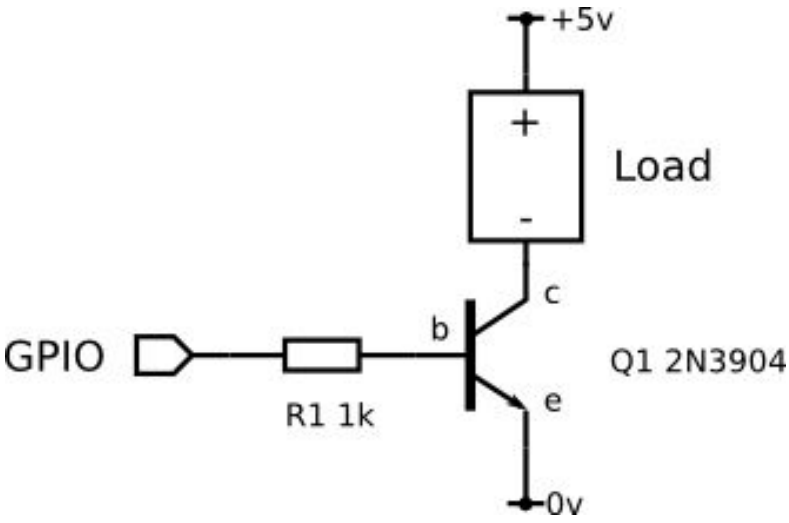
Controlling Rpi using your laptop

The setup might get a bit messy with keyboard and monitors not available at your rooms. For convenience we used VNC for controlling RPi through our laptop. Here’s a bit about that.

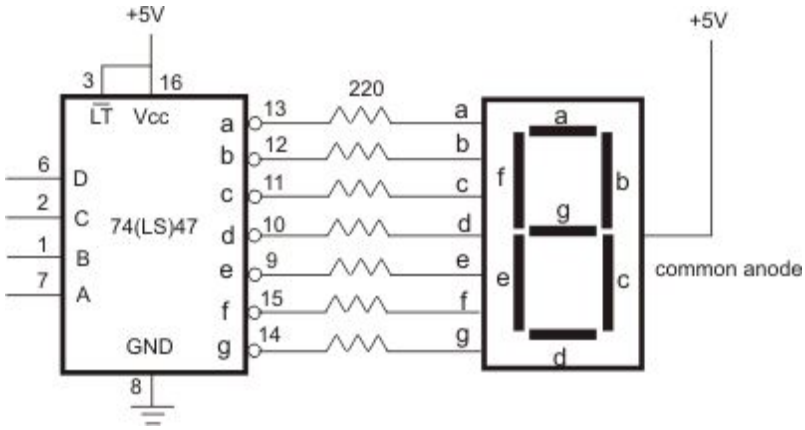
Also you can SSH into your RPi given that both RPI and your laptop is connected to the same network.

Seven segment display (SSD) and BCD IC

Seven leds are independently controlled by the 10 pins provided at the back of SSDs. In order to make things easier, we can use a BCD IC so that by controlling 4 pins, we can control all seven leds of the SSDs for depicting 0-9 digits on our SSD.
Since the SSDs we used were not getting enough power supply from RPi, we used the concept of transistor as a switch and used external power supply for that purpose.



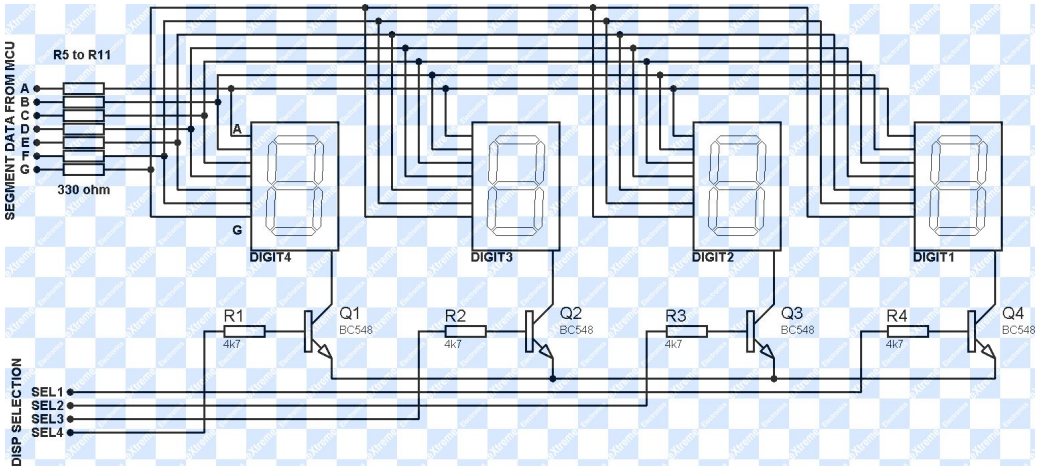
Below is the circuit for bcd-ssd connection and here are some links (1, 2) for reference.



Demux IC

Since the number of GPIO pins were not enough for us, we had to somehow use the existing pins to control more than it could independently do. For this we resorted to the concept of *demultiplexing*. Refer to this link for more info. Here's the datasheet for demux.

Below is the circuit diagram for Demux.



Timer : TM1637

Our score-board also depicts time frames for gameplay. TM1637 is an accurate clock with an inbuilt crystal system. We have directly used an inbuilt IC for the timer.

Kicking in some IOT

In order to remotely control our scoreboard, we used flask to fabricate a web api. Some links for more info are 1, 2.

On the coding front, we had to use several libraries and a concept called *multithreading* to ensure smooth and simultaneous working of all the components of the score-board.

For the libraries and the code, you can refer to our Github repo here.

We ran the server on Raspberry Pi and using MIT app inventor, created an app that could access it given that the client and the server are on the same *local network*. The .aia file for the MIT app_inventor app is on github.

#Tip : When you save the .py files, don't use names like random.py or system.py since they are pre-existing and might result into your modules not working as expected.

Electro-Mechanical work

Adjusting the bcd and demux circuits on a breadboard gets a bit cumbersome. To make stuff more convenient, we printed the required circuits. The designing of the PCB was done using *Eagle Cad.* Here's an excellent tutorial on Eagle Cad. For further details regarding printing, check out the itsp2k16 facebook page.

The schematic and .brd file of our circuits are on github too.

#Tip : When designing circuit board, make sure there's enough spacing between the routes so as to let you solder the components easily. Doing otherwise got us into quite some trouble.

#Tip : When you are about to print the PCB, make sure you have a picture of how it's gonna be printed because ICs (in our case BCD IC) are orientation specific.

Links:
Github link: https://github.com/hrushikeshloya/itsp2k16
Youtube link:

Reference Links:

- https://www.raspberrypi.org/downloads/noobs/ : downloading NOOBS
- https://www.raspberrypi.org/help/noobs-setup/ : procedure for booting Rpi with NOOBS
- https://www.raspberrypi.org/learning/physical-computing-with-python/worksheet/ :led on/off using Rpi
- https://www.raspberrypi.org/documentation/remote-access/vnc/ :VNC (Virtual Network Computing). Accessing Rpi using Android phone or laptop.
- https://www.raspberrypi.org/learning/python-web-server-with-flask/ : hosting web server using flask in raspberry.
- https://translate.google.com/translate?sl=auto&tl=en&js=y&prev=_t&hl=en&ie=UTF-8&u=http%3A%2F%2Fwww.forum-raspberrypi.de%2FThread-led-4-segment-i2c-display&edit-text=&act=url : The sample timer code for TM1637 module for raspberrry pi. The library that finally worked.