

Circuit-Box

!!World’s First Virtual Lab!!

Raju Chaluva, Ganesh, Shruthi

08/03/2019

# 

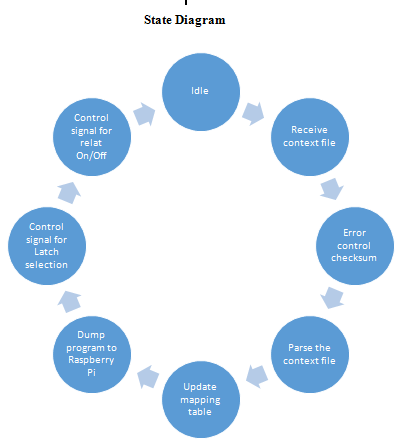
# 

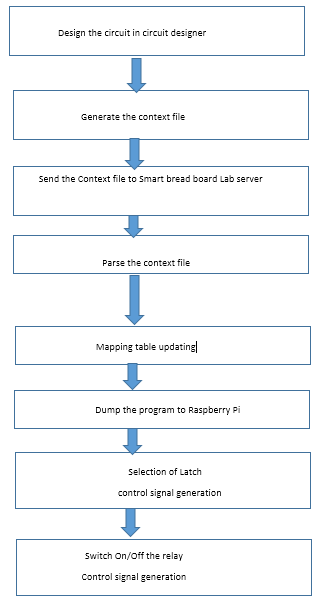
## Introduction

Virtual labs provide a remotely operated lab environment for students to perform experiments. This can be useful in testing the equipment’s and devices remotely. The lab has various equipment that can be remotely started or stopped. The virtual lab can be defined as virtual studying and learning environment that stimulates the real lab. It provides the students with tools, materials and lab sets on computer in order to perform experiments subjectively or within a group at anywhere and anytime.

The circuit simulation tool functions at the client end. A client can login to the V-lab website from the remote place and access the tool to design the test circuit. At the server end, the circuit is rigged up in a smart way without any human intervention and output is made visible to the client with the help of video conferencing. This feature of testing and analyzing the circuits virtually, from the remote place is said to be virtual electronics lab.

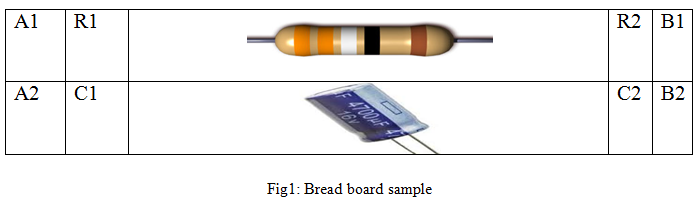
## Software State Machine



****

**3. RELAY CONTROL LOGIC**

With reference to the hardware architecture of smart bread board, Every D-Latch connected to Analog switch, which consists of 4 relays. Each relay connects a node and a pin of a component. The Analog switch is activated by the output of the D-Latch as per the assignment made in the program. For example consider the following sample bread board with a resistor and a capacitor with notations shown below.



The hardware consists of demux 1:16 with 4 select lines namely S0,S1,S2,S3. The each of the 15 outputs of Demux is given to the LE pins of each of the D-Latch is which is enabled by the select line data given in the program. Thus one latch is enabled at a time. The output of each of the 8 GPIO's from Raspberry-pi is fed to each of 8 inputs of D-Latch that is enabled, The control data to connect pin and node by activating relay is sent through it .Each Latch controls two analog switches(AS) summing up to 8 relays. It means that one output of latch drives one relay. Thus the hardware is designed as follows,

· Latch, L1: switches 1-8;( AS 1 & AS 2)

· Latch, L2 : Switches 9-16; ( AS 3 & AS 4)

.

.

.

· Latch, L15: Switches 113-120; ( AS 29 & AS 30);

| Sl.no | Pin, node | Switch | Select lines | | | | RPi O/P control data | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| S0 | S1 | S2 | S3 | GPIO's | | | | | | | |
| 8 | 4 | 2 | 1 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| 1 | A1,N1 | SW1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | A1,N2 | SW2 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 3 | A1,N3 | SW3 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 4 | A1,N4 | SW4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 5 | A1,N5 | SW5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 6 | A1,N6 | SW6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 7 | B1,N1 | SW7 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 8 | B2,N2 | SW8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 9 | B2,N3 | SW9 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 10 | B2,N4 | SW10 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 11 | B2,N5 | SW11 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 12 | B2,N6 | SW12 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 13 | A2,N1 | SW13 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 14 | A2,N2 | SW14 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 15 | A2,N3 | SW15 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 16 | A2,N4 | SW16 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 17 | A2,N5 | SW17 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 18 | A2,N6 | SW18 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 19 |  | SW | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 20 |  | SW | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 21 |  | SW | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 22 |  | SW | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 23 |  | SW | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 24 |  | SW | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
|  |  | SW | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**4. CONTEXT FILE FORMAT TABLE**

| ***Node*** | ***File name*** | ***Component*** | ***Pin*** |
| --- | --- | --- | --- |
| ***1*** | ***Clipper*** | ***V2*** | ***1*** |
| ***1*** | ***Clipper*** | ***R1*** | ***1*** |
| ***2*** | ***Clipper*** | ***D1*** | ***A*** |
| ***2*** | ***Clipper*** | ***R1*** | ***2*** |
| ***2*** | ***Clipper*** | ***XSC1*** | ***1*** |
| ***3*** | ***Clipper*** | ***XSC2*** | ***4*** |
| ***3*** | ***Clipper*** | ***V2*** | ***2*** |
| ***3*** | ***Clipper*** | ***D1*** | ***K*** |
| ***Gnd*** | ***Clipper*** | ***GND*** | ***GND*** |

**5. GPIO Toggling for LED**

import RPi.GPIO as GPIO

import time

GPIO.setmode(GPIO.BCM)

GPIO.setwarnings(False)

GPIO.setup(18,GPIO.OUT)

print "LED on"

GPIO.output(18,GPIO.HIGH)

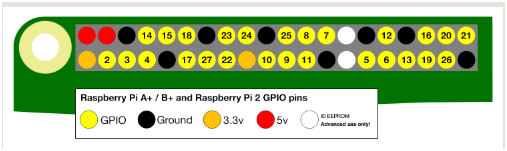
time.sleep(1)

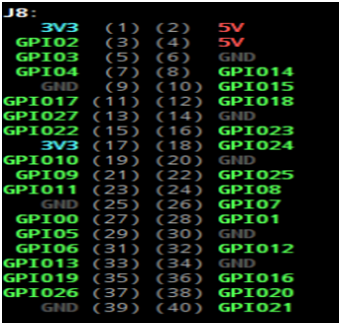
print "LED off"

GPIO.output(18,GPIO.LOW)

| import RPi.GPIO as GPIO | The first line tells the Python interpreter (the thing that runs the Python code) that it will be using a ‘library’ that will tell it how to work with the Raspberry Pi’s GPIO pins. A ‘library’ gives a programming language extra commands that can be used to do something different that it previously did not know how to do. This is like adding a new channel to your TV so you can watch something different. |
| --- | --- |
| import time | Imports the Time library so that we can pause the script later on. |
| GPIO.setmode(GPIO.BCM) | Each pin on the Raspberry Pi has several different names, so you need to tell the program which naming convention is to be used. |
| GPIO.setwarnings(False) | This tells Python not to print GPIO warning messages to the screen. |
| GPIO.setup(18,GPIO.OUT) | This line tells the Python interpreter that pin 18 is going to be used for outputting information, which means you are going to be able to turn the pin ‘on’ and ‘off’. |
| print "LED on" | This line prints some information to the terminal. |
| GPIO.output(18,GPIO.HIGH) | This turns the GPIO pin ‘on’. What this actually means is that the pin is made to provide power of 3.3volts. This is enough to turn the LED in our circuit on. |
| time.sleep(1) | Pauses the Python program for 1 second |
| print "LED off" | This line prints some information to the terminal. |

**6. GPIO pins**





**6. Receive Context file flow diagram**

6.1 USB working

6.2 Ethernet data TX

6.3 Cloud

6.4 Parsing the context file

6.5 Mapping of Components

6.6 Circuit diagram

6.7 BOM(Bill of Materials)