Getting Started with OSK's

Pi-Sat

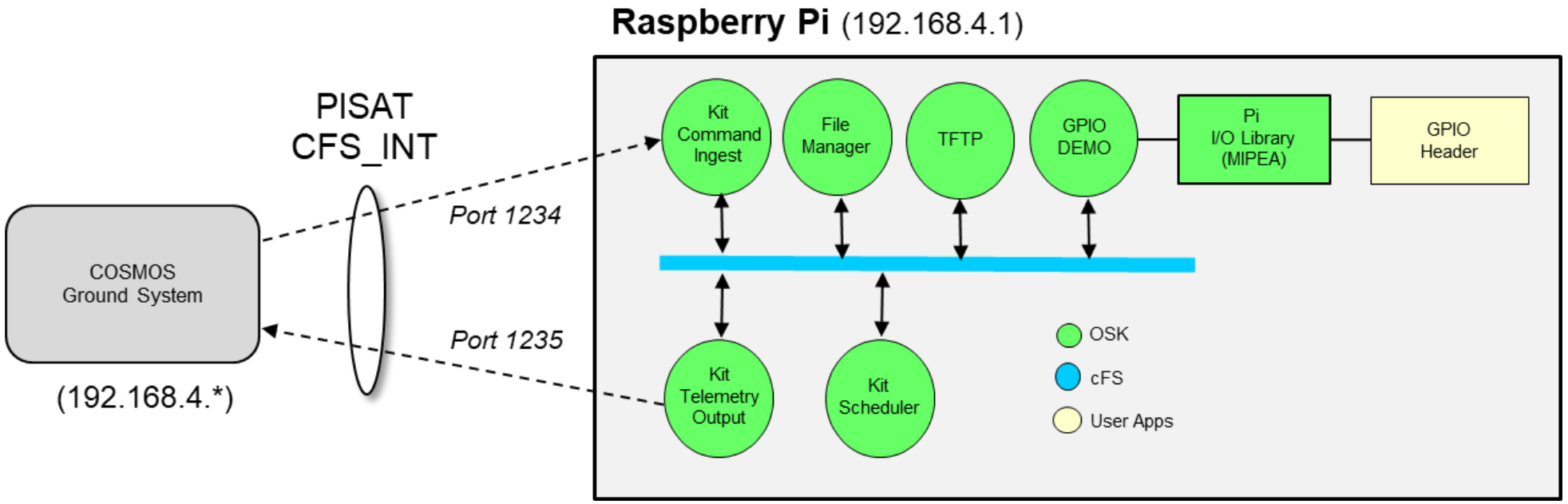
OSK v3.2

Pi-Sat v1.1

Introduction

- **The primary objectives of the OpenSatKit (OSK) Pi-Sat are to**
  1. Provide a platform that allows STEM educators and hobbyists to work with NASA's core Flight System (cFS) on the low-cost Raspberry Pi
  2. Support a configuration that allows the COSMOS ground system to remotely communicate with the Pi-Sat over WiFi
  3. Provide pathways for user expansion and customizations
- **Rationale**
  - NASA's cFS is a mature flight-proven open-source flight software (FSW) framework that has many educational benefits
  - Remotely controlling Pi-Sat instantiations allows a user to gain an appreciation of spacecraft FSW development and operations
- **OSK does not provide a turn-key project**, so the documentation provides workflows and sequences of activities, but the details may vary based on user platforms
  - Users can create a "Pi-Sat Kit" that provides a turn-key solution for a particular situation
- **Perquisites**
  - Installation and running demo requires basic Linux operating system knowledge
  - Developing apps requires cFS runtime environment app group knowledge and C programming proficiency

Pi-Sat cFS Target

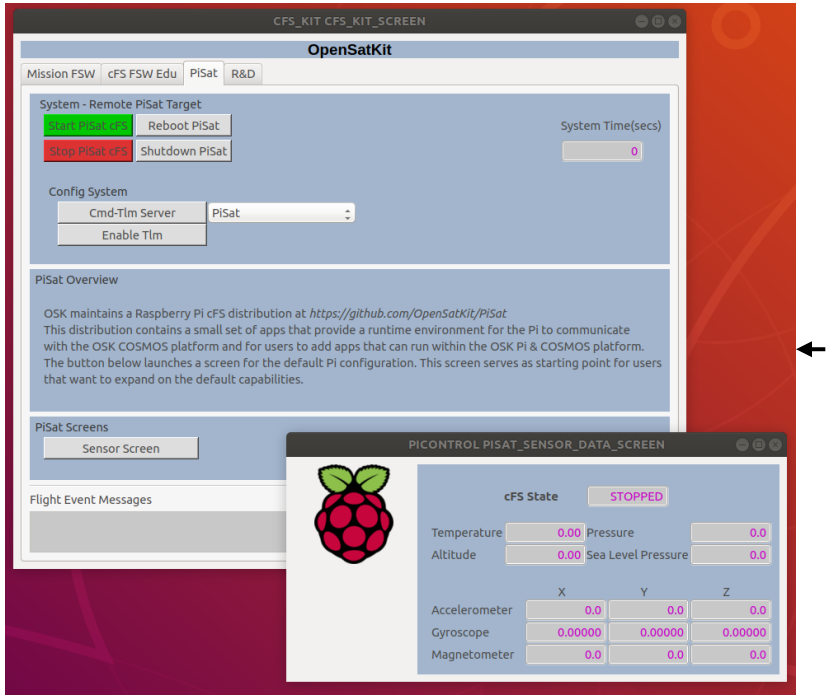


- **Minimal set of OSK apps that provide an app runtime environment and file management/transfer services**
- **Only use OSK apps that use JSON tables and not cFS binary tables**
- **GPIO demo app can be used to verify the cFS Pi-Sat target installation and it can serve as a starting point for users to create their own apps**
- **Separate repo at https://github.com/OpenSatKit/pi-sat**
  - Pi-Sat v1.1 uses cFE 6.8.* which has significant API changes from cFE 6.7.1 which is used by OSK therefore the apps in the Pi-Sat target are not identical to OSK's desktop apps
  - For Pi-Sat v1.1, File Manager and TFTP have not been fully transitioned to cFE 6.8.*. They are not required to run the GPIO demo

Quick Start Steps

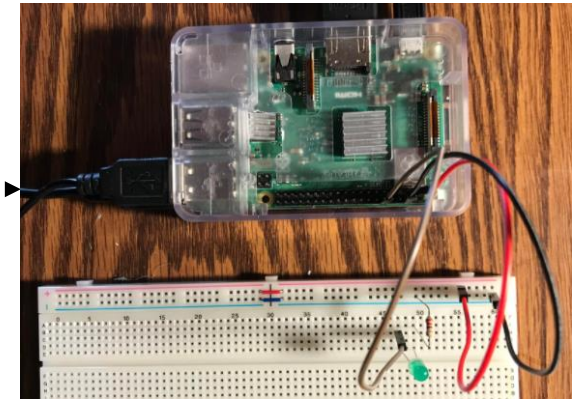
1. **Configure your Pi hardware for GPIO demo**
   - Recommended so you can run the demo to verify later steps
2. **Install cFS Pi-Sat target on your Pi**
   - Run demo locally
3. **Configure your Pi as a WiFi access point**
4. **Install Pi-Sat remote control program**
5. **Configure COSMOS for Pi-Sat Communications**
6. **Connect COSMOS to PiSat**
   - Run demo remotely

COSMOS



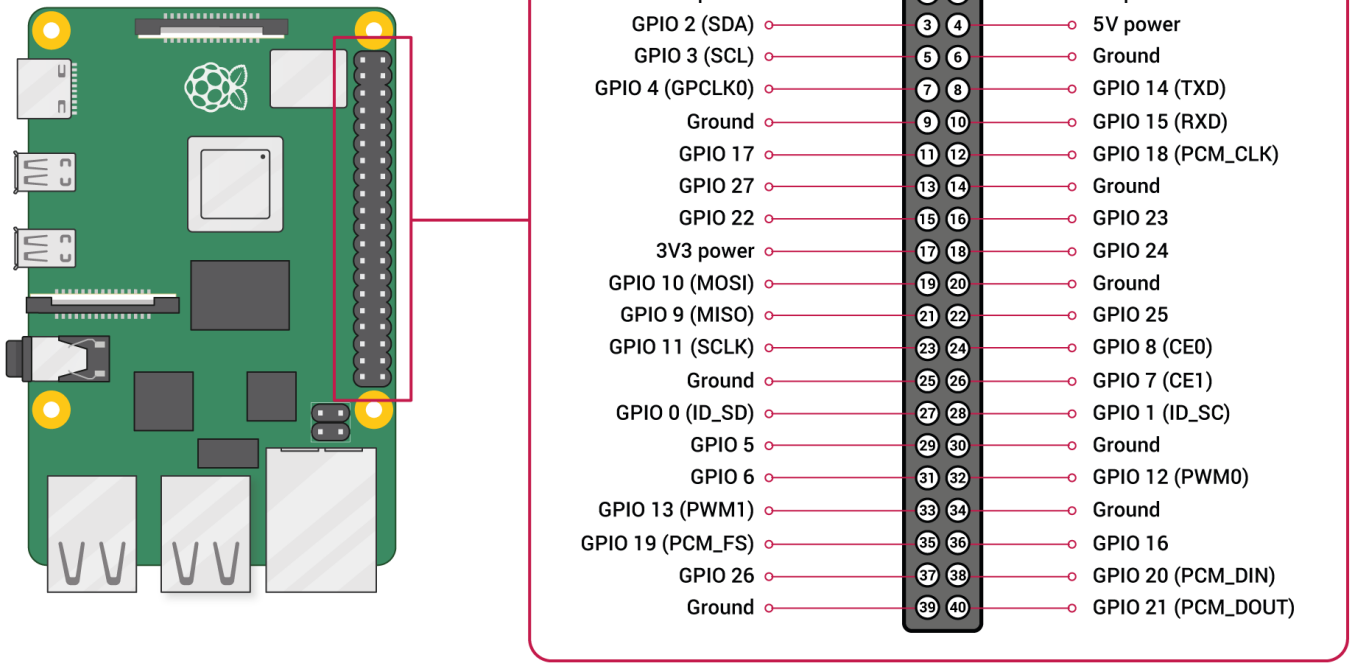
WiFi

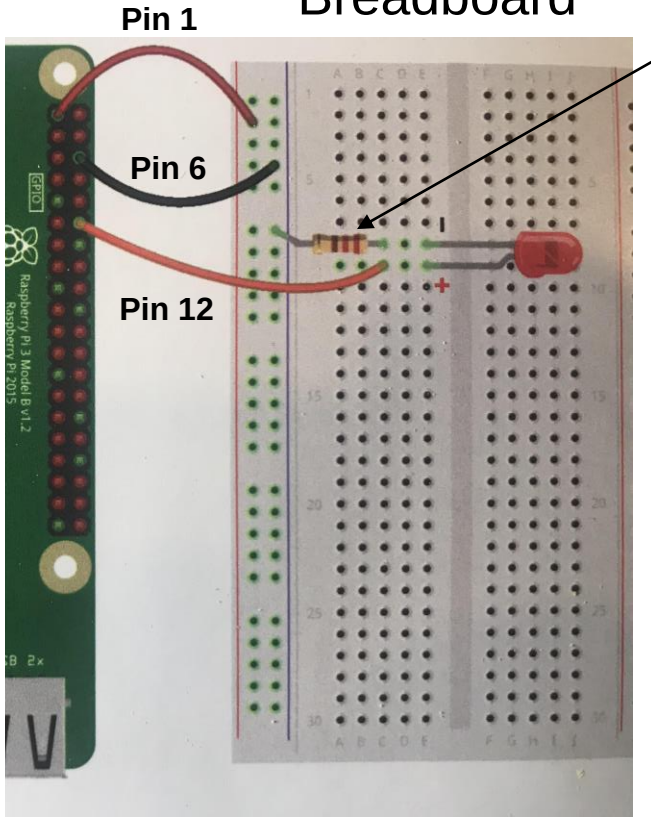
Raspberry PI GPIO Demo



1 – Configure Your Pi Hardware for GPIO Demo

- **The cFS Pi-Sat target comes with an app called GPIO_DEMO that blinks an LED connected to the Pi's General-Purpose Input/Output (GPIO)**
  - Refer to the cFS Pi-Sat target section later in this package for software details
- **Configure and connect a breadboard to the Pi as shown below**
  - Note physical pin 12 is GPIO logical pin 18





2 – Install cFS Pi-Sat Target on your Pi

- **Prepare the development environment**
  1. `sudo apt install cmake`
- **Clone OSK pi-sat repo and build the cFS target****
  1. `git clone https://github.com/OpenSatKit/pi-sat`
  2. `Edit cfs/apps/mipea/config.h and define the Broadcom chip for your Pi`
  3. `cd pi-sat/cfs`
  4. `make SIMULATION=native prep`
  5. `make install`
- **Run the cFS target (must be run from executable location)**
  1. `cd build/exe/cpu1`
  2. `sudo /core-cpu1`
- **GPIO Demo runs by default**
  - `It sends an information event message each time it turns on/off the LED`
  - `If the LED is not blinking, then double check your LED wiring and the mipea/config.h setting`

**Default configuration is for 32-bit Raspbian. See /cfs/rpi_defs/toolchain-cpu1.cmake

Install Software Packages

1. **Ensure Pi OS is up to date (this can take a while depending on your situation)**
   a. `sudo apt update`
   b. `sudo apt full-upgrade`
2. **Install host access point package**
   a. `sudo apt install hostapd`
3. **Enable wirelss access point service**
   a. `systemctl unmask hostapd`
   b. `systemctl enable hostapd`
4. **Provide DNS & DHCP network services to wireless clients using dnsmasq**
   a. `sudo apt install dnsmasq`
5. **Install utilities that help save and restore firewall settings when the Pi reboots**
   a. `sudo DEBIAN_FRONTEND=noninteractive apt install -y netfilter-persistent iptables-persistent`

** Summarized from: https://www.raspberrypi.org/documentation/configuration/wireless/access-point-routed.md

3 –Configure your Pi as a WiFi Access Point (2 of 3)

Configure Static IP Address

- **Pi-Sat runs a DHCP server for the wireless network which requires static IP configuration for the wlan0 wireless interface. The Pi also acts as the router on the wireless network it will be given the first IP address in the network 192.168.4.1**

1. **Edit DHCP configuration file and at the end of the file add the contents in step a**

   a. `sudo nano /etc/dhcpcd.conf`

   ```
   interface wlan0
       static ip_address=192.168.4.1/24
       nohook wpa_supplicant
   ```

2. **Save original dnsmasq configuration. Edit a new dnsmasq configuration file and at the end of the file add the contents in step b**

   a. `sudo mv /etc/dnsmasq.conf /etc/dnsmasq.conf.orig`

   b. `sudo nano /etc/dnsmasq.conf`

   ```
   interface=wlan0
      dhcp-range=192.168.4.2,192.168.0.20,255.255.255.0,8h
      domain=wlan
      address=/gw.wlan/192.168.4.1
   ```

Configure the access point

1. Create a hostapd configuration file

a. `sudo nano /etc/hostapd/hostapd.conf`

```
country_code=US
interface=wlan0
driver=nl80211
ssid=OSK-Pi-Sat
hw_mode=g
channel=7
wmm_enabled=0
macaddr_acl=0
auth_algs=1
ignore_broadcast_ssid=0
wpa=2
wpa_passphrase=pisatcfs
wpa_key_mgmt=WPA-PSK
wpa_pairwise=TKIP
rsn_pairwise=CCMP
```

2. Run the new wireless access point**

a. `sudo systemctl reboot`

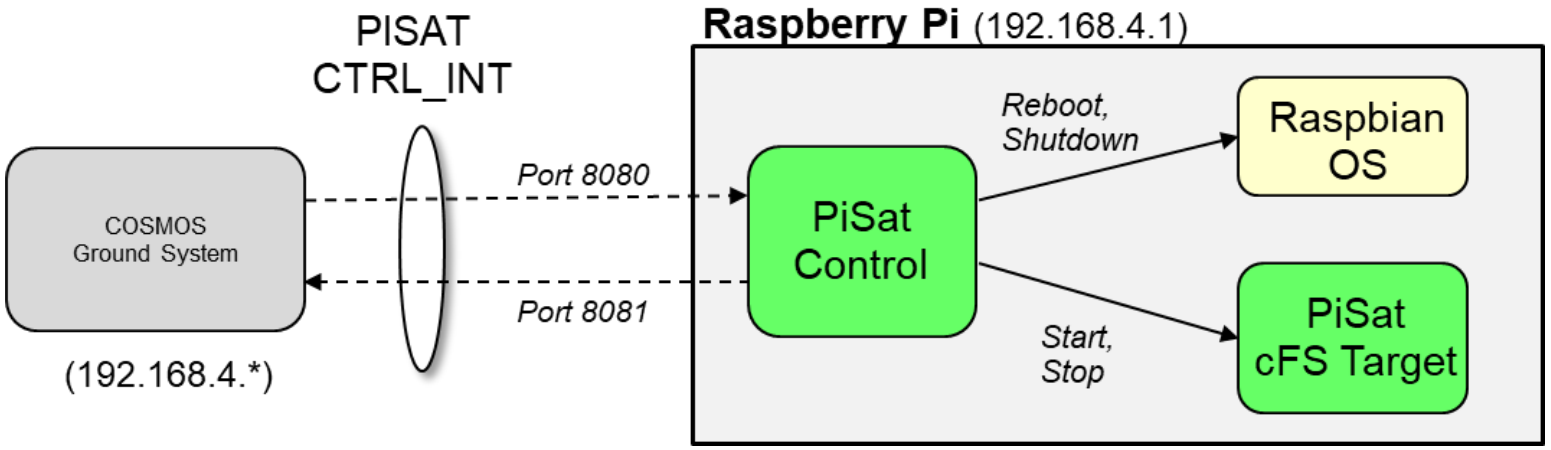
b. On COSMOS host search for "OSK-Pi-Sat"

c. If needed access via ssh: pi@192.168.4.1 or ssh pi@gw.wlan

**See Appendix B for instructions to enable/disable the WiFi access point

4 – Install Pi-Sat Remote Control Process (1 of 2)

picrtl.py is a python program that receives commands to manage the cFS target from COSMOS over a UDP socket and sends status telemetry



4 – Install Pi-Sat Remote Control Process (2 of 2)

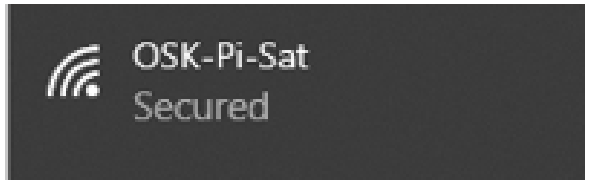
- **These instructions assume OSK's pi-sat repo is cloned in /home/pi**
- **Perform the following steps in a terminal window to cause picrtl.py to run during your Pi boot process**

1. **cd /etc**
2. **sudo nano rc.local**
   - Nano is simple editor included with Raspbian
   - Elevated privilege is required to modify rc.local
3. **Before the "exit 0" line enter the following:**
   - # Start OSK Pi-Sat control
   - python3 /home/pi/pi-sat/tools/pictrl.py &
4. **Note**
   - You will need to edit /home/pi/pi-sat/tools/pictrl.ini in a later step to configure the COSMOS IP Address
   - Pictrl creates a log file named /tmp/pictrl.log that can be helpful for troubleshooting

6 – Connect COSMOS to Pi-Sat (1 of 4)

- **Here's the current communication situation**
  - In step #3, Pi-Sat was configured as a WiFi access point with a static IP of 192.168.4.1
  - In step #5, COSMOS was configured to communicate with the Pi-Sat's static IP address of 192.168.4.1
  - In step #4, PiCtrl was installed but it doesn't know how to communicate to COSMOS
  - This step needs to configure PiCtrl's ini file and reboot the Pi to establish COSMOS-to-PiSat PiCtrl communications

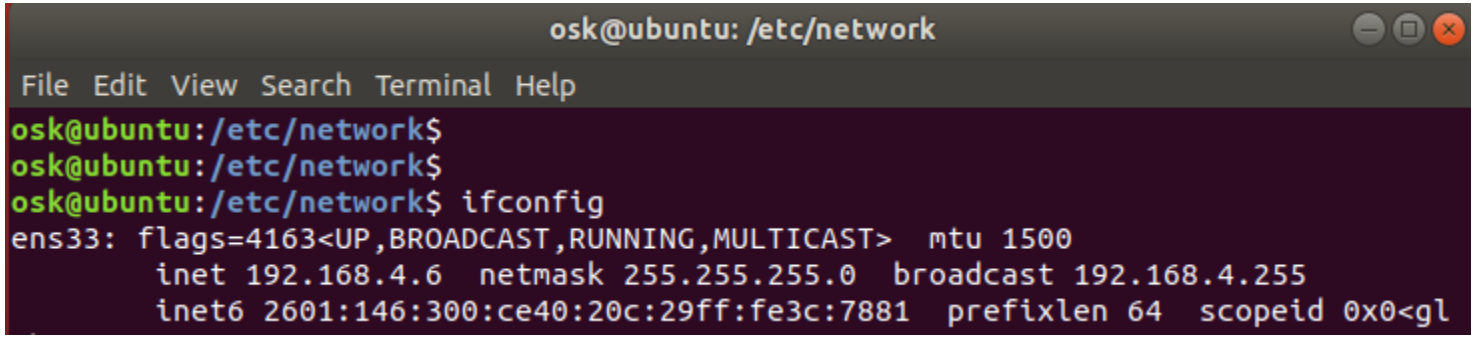
1. **On your COSMOS host machine connect to the Pi-Sat WiFi access point.**
   - **You should see the SSID name specified in the `hostapd.conf` file**



6 – Connect COSMOS to Pi-Sat (2 of 4)

2. **If you can configure your COSMOS platform to have a static IP address** then you only need to do the following steps once, otherwise they will need to be repeated each time you configure COSMOS-to-Pi-Sat communications**

   a. **On your COSMOS platform run ifconfig to get the host's IP address**



   b. **Edit Pi-Sat's** home/pi/pi-sat/tools/pictrl.ini and set the network section's "cmd-tlm--ip-addr" key to the COSMOS host's IP address

   c. **Reboot the Pi by entering** sudo systemctl reboot

** See Appendix A for some example static IP setup procedures

6 – Connect COSMOS to Pi-Sat (3 of 4)

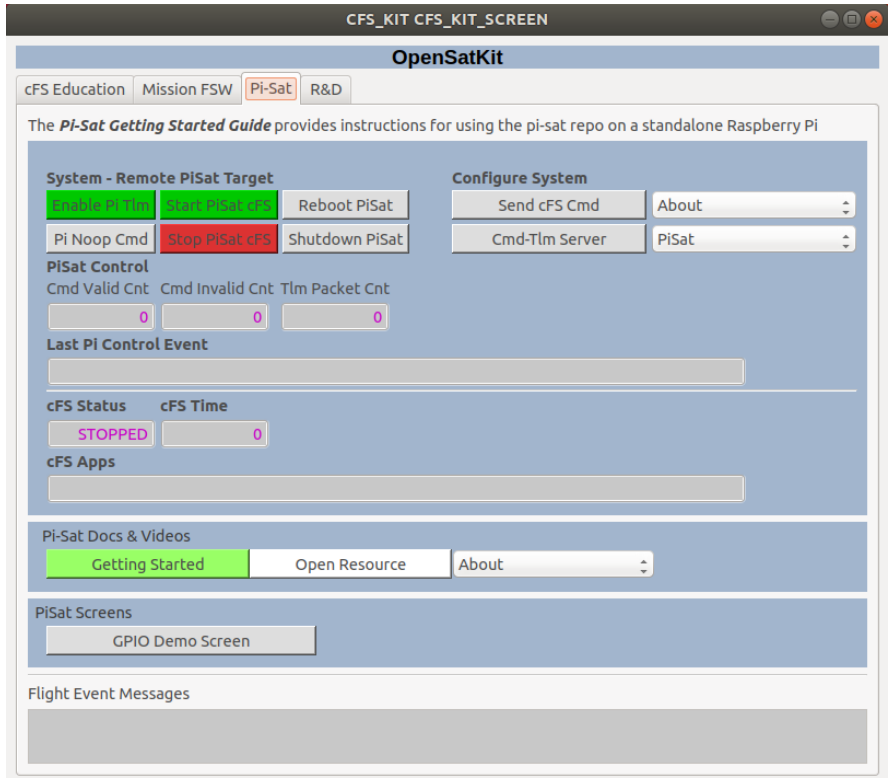
3. Configure COSMOS's command and telemetry server to connect to Pi-Sat and not the local IP host

a. Delete ~/cosmos/outputs/tmp/*.bin

b. Edit ~/cosmos/lib/hw_target.rb and follow instructions in the file to set ID = "PISAT"

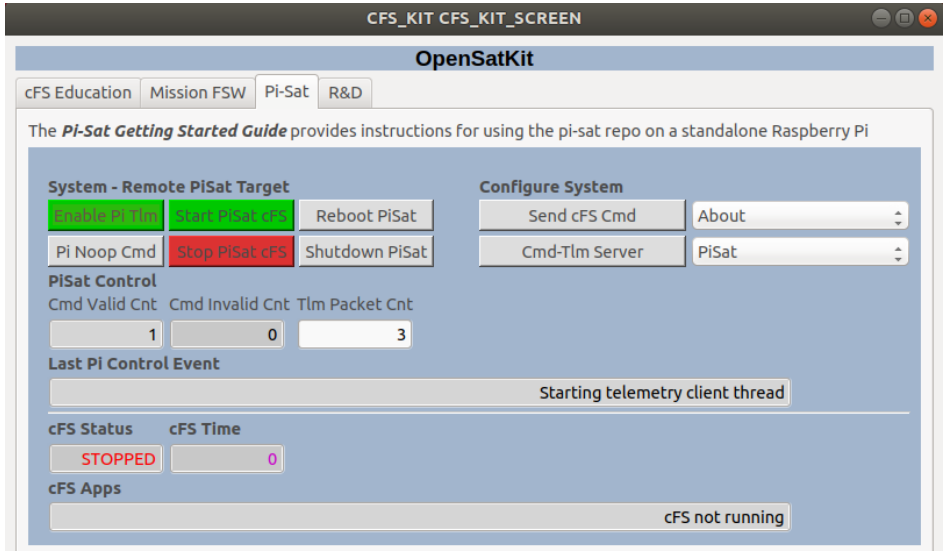
c. Start COSMOS/OSK as usual : In a terminal window cd ~/cosmos, ruby Launcher

- Your Pi-Sat main should look as follow:

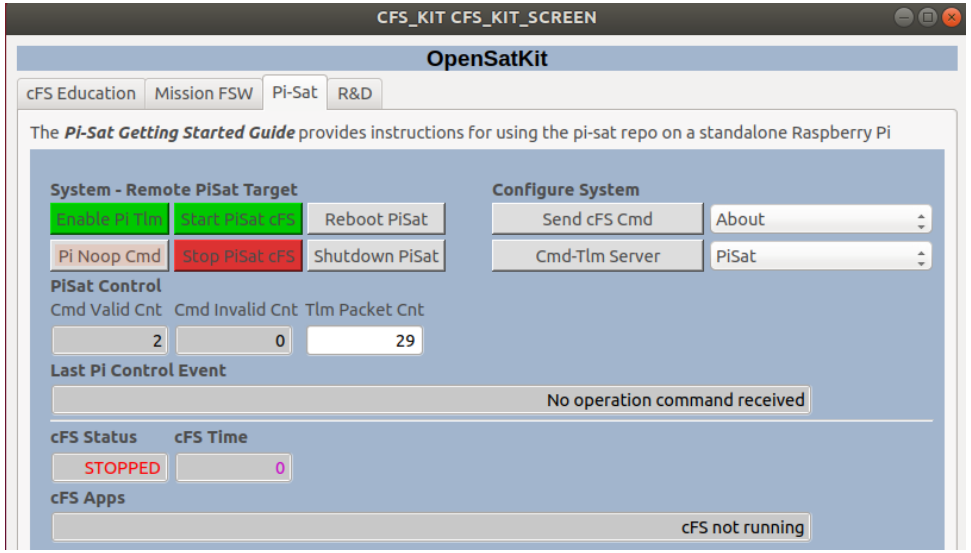


6 – Connect COSMOS to Pi-Sat (4 of 4)

4. **Select "Enable Pi-Sat" button to command PiCtrl to start sending telemetry to COSMOS**

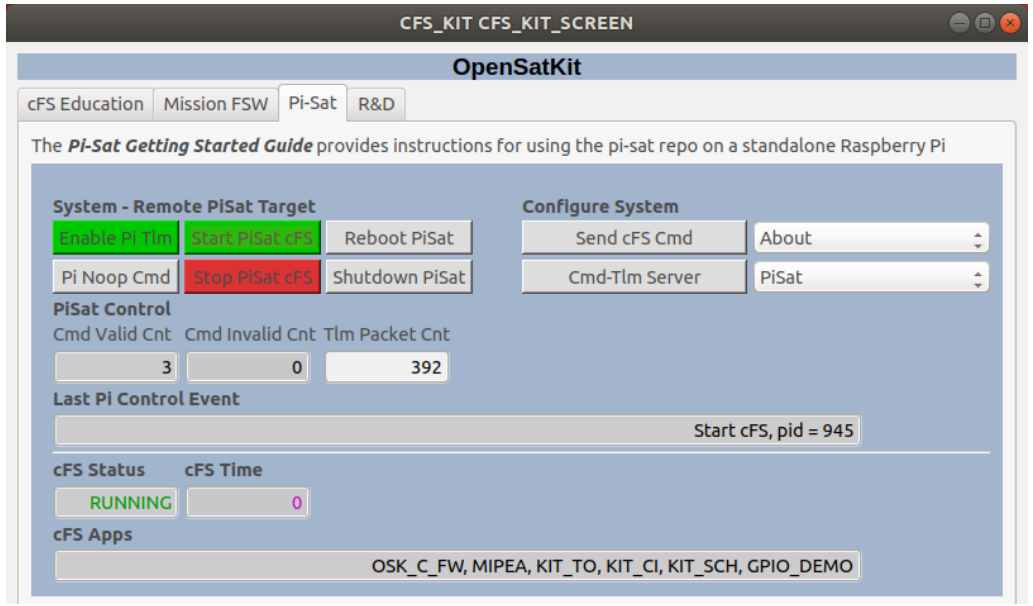


5. **Test the connection to Pi-Sat Control by sending a Noop command**

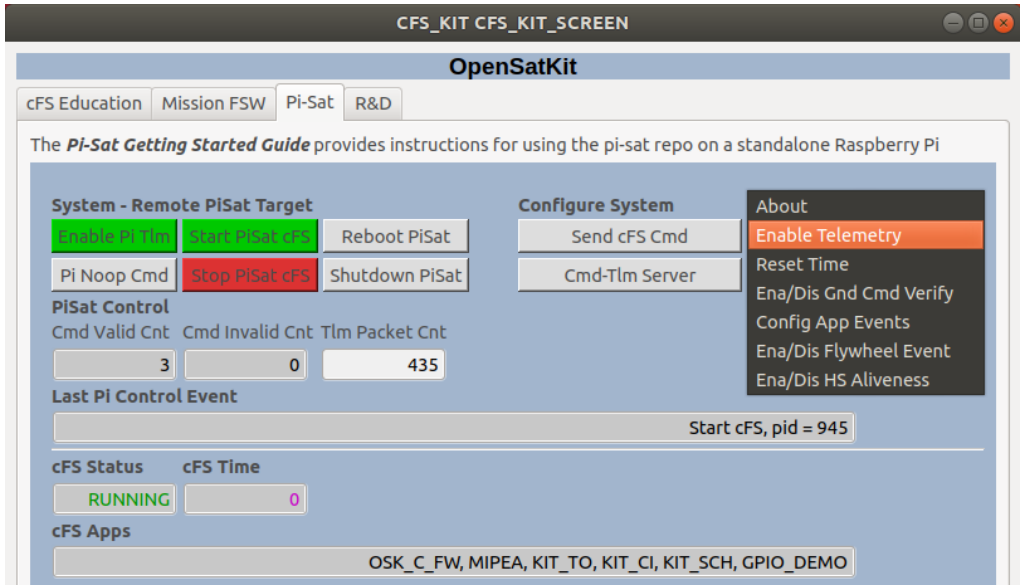


6 – Run GPIO Demo (1 of 2)

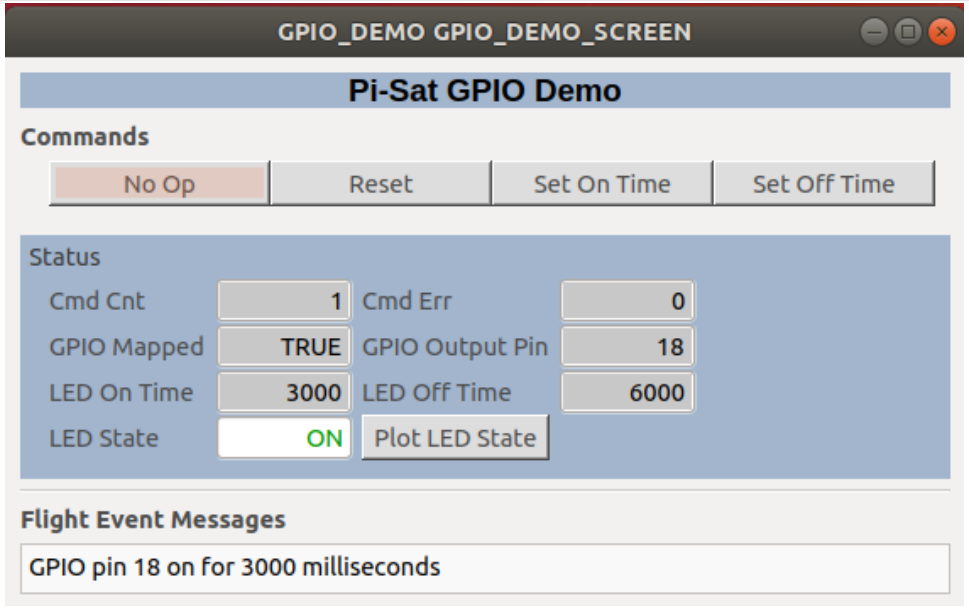
1. **Select the <Start Pi-Sat cFS> button**
   - This starts the cFS. Loads GIO_DEMO app, and the LED should start blinking

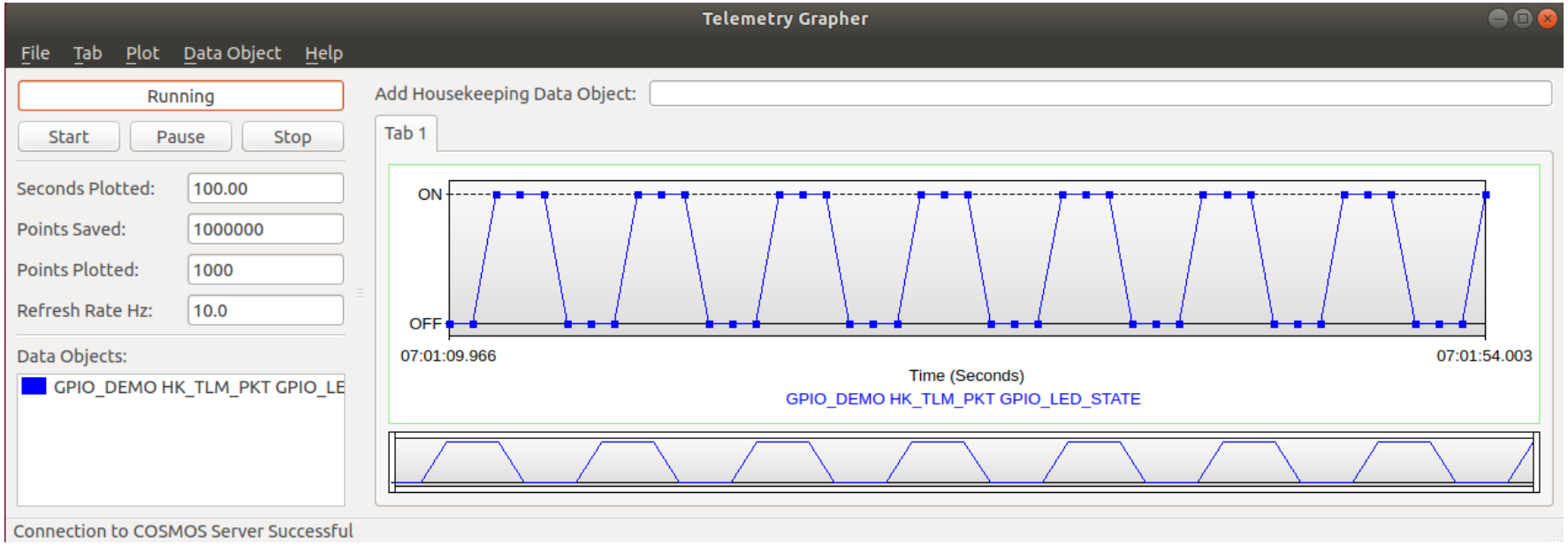


2. **Enabled cFS telemetry by using the "Send cFS Cmd" under the Configure System heading**
   - Enter your COSMOS host's IP address



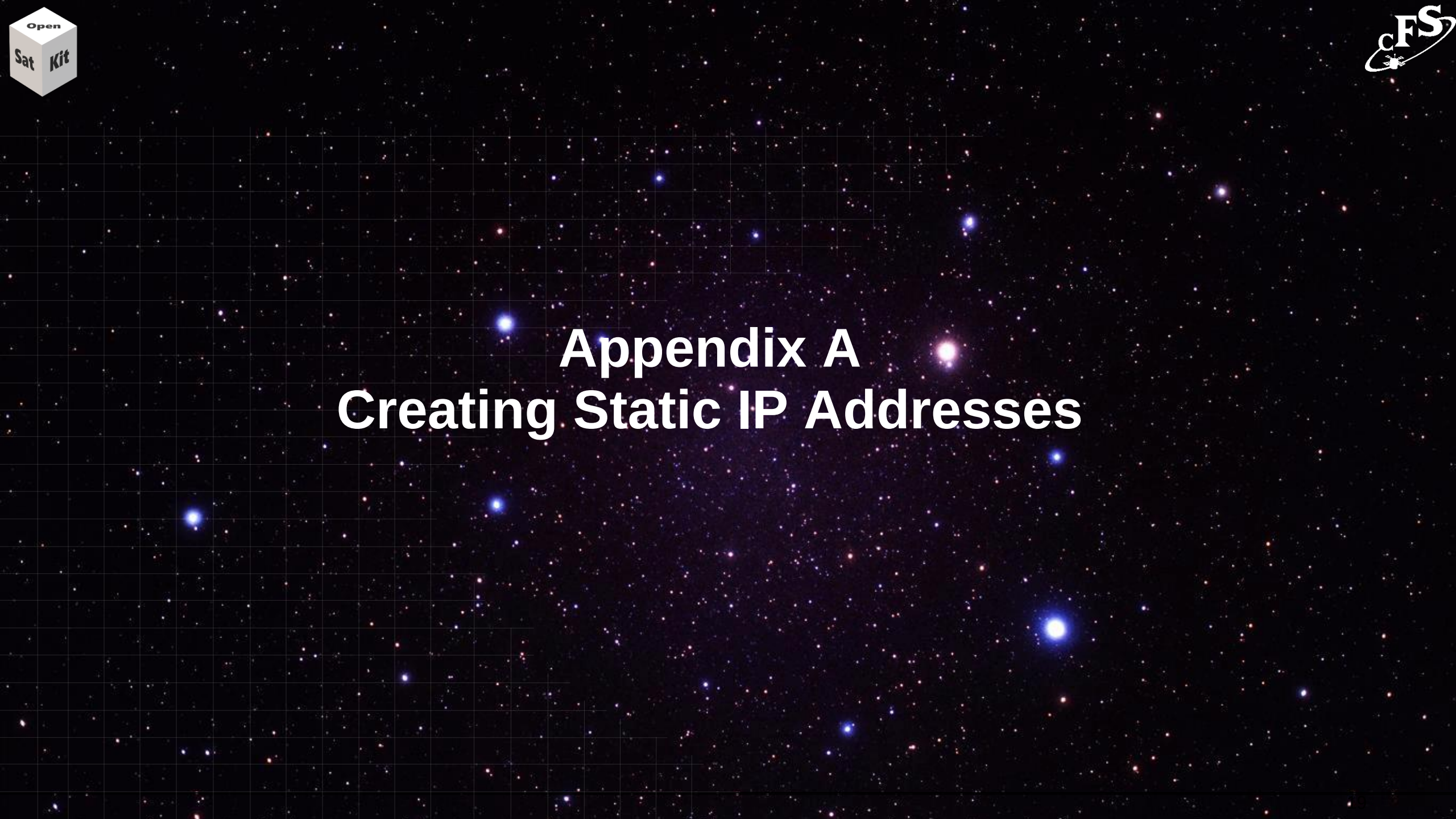
3. **Launch the GPIO_DEMO_SCREEN**
   - GIO_DEMO app is loaded and should you see the LED blink
4. **Select "Set Off Time" and set the time to 3000 milliseconds**
5. **Select "Plot LED Status" and you should eventually see a plot similar to the one below**





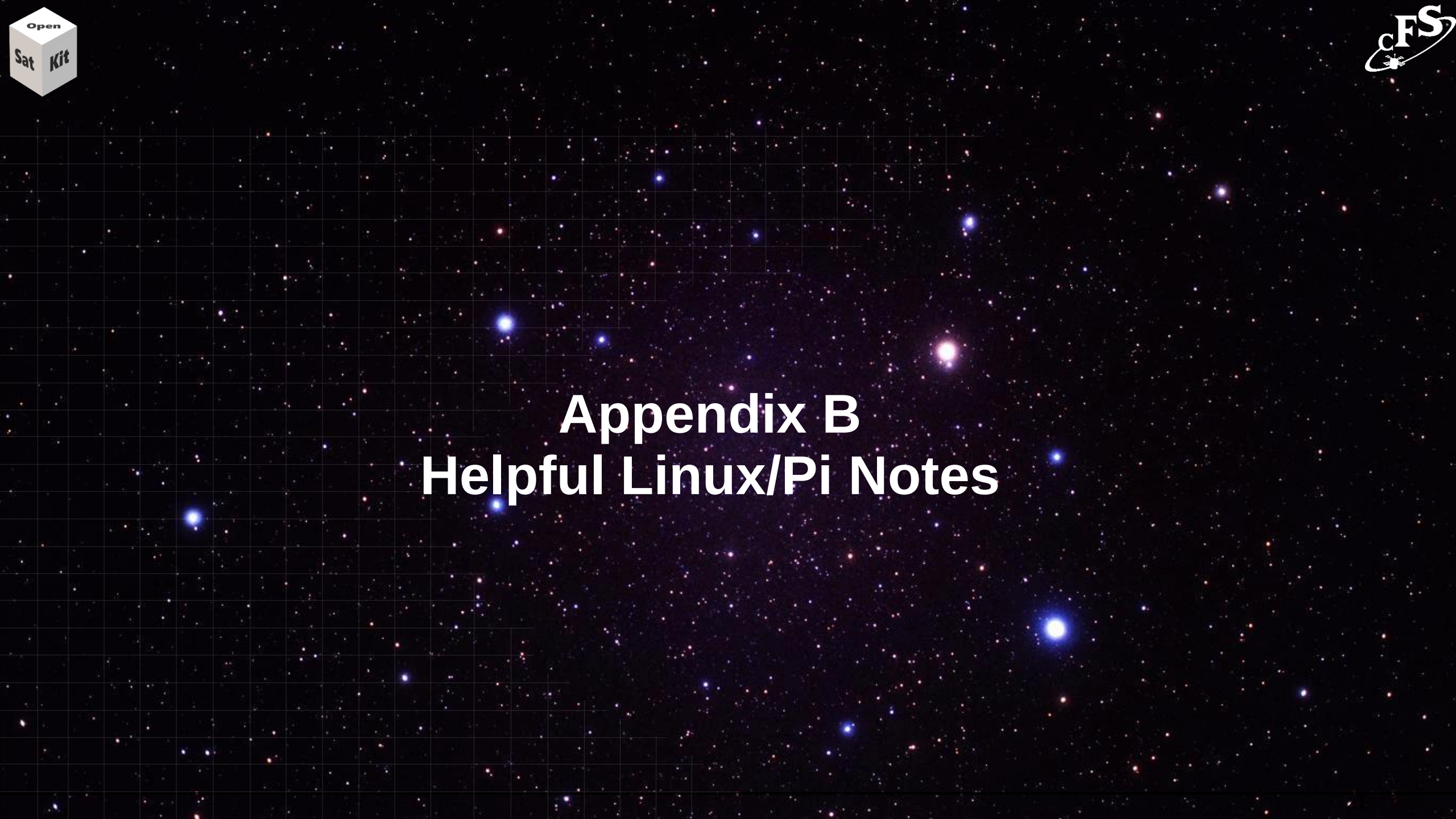
Congratulations!

You successfully completed the OSK Pi-Sat installation and demo



Appendix A
Creating Static IP Addresses

Ubuntu 18.04 hosted by VMware



Appendix B
Helpful Linux/Pi Notes

Enable-Disable PiSat WiFi Access Point

- **Disable WiFi Access Point**
  1. sudo systemctl disable hostapd dnsmasq
  2. comment the static ip config in /etc/dhcpcd.conf
  3. sudo reboot

- **Enable WiFi Access Point**
  1. sudo systemctl enable hostapd dnsmasq
  2. uncomment the static IP config in /etc/dhcpcd.conf
  3. sudo reboot

Handy OS Commands

Linux/Rasbian Command	Notes
ifconfig	Displays current network status
hostname -i	Displays host IP address
netcat	Command-line utility that reads and writes data across network connections using TCP or UDP Useful for trouble shooting pictrl program. If configured for local host enter the following to send the string XYZ to the pictrl program echo XYZ | nc –u 127.0.0.1 8080
pgrep xyz	process grep command returns process IDs for all process names containing the string xyz pgrep core # Display cFS core images running pgrep python3 # Display all python3 programs

Recover/Reset Pi Password

1. Power down and pull the SD card out from your Pi and put it into your computer.
2. Open the file 'cmdline.txt' and add 'init=/bin/sh' to the end. This will cause the machine to boot to single user mode.
3. Put the SD card back in the Pi and boot.
4. When the prompt comes up, type 'su' to log in as root (no password needed).
5. Type "passwd pi" and then follow the prompts to enter a new password.
6. Shut the machine down, then pull the card again and put the cmdline.txt file back the way it was by removing the 'init=/bin/sh' bit.

Appendix C
GPIO_DEMO App Design

GPIO_DEMO App Overview

- **Describe GPIO app design so user's can use it as a starting point for their own designs**