The PyBinaryClock

Technical description

The circuit board is a type of 'hat' to a raspberry pi zero with 20 WS2812B programmable LEDs for the clock-display, a 2x20 header for the pi connection and a real-time clock circuit. The board was manufactured by https://aisler.net that has a great quality service for small series of prototypes.

The kit is mounted between two part of plexi-glass panels to make a clock that can be mounted on the wall or put on a table. See pictures below.

The hardware:

Raspberry Pi zero W

- Install rasbian light and configure it for your localization etc.
- Enable the I2C interface: https://www.raspberrypi-spy.co.uk/2014/11/enabling-the-i2c-interface-on-the-raspberry-pi/
- Install the RTC clock module software and set it up the properly. Follow this guide: https://www.raspberrypi-spy.co.uk/2015/05/adding-a-ds3231-real-time-clock-to-the-raspberry-pi/ but be aware of that hardware is already present on the circuit-board in a correct way.
- Get the python lib for WS281x programmable leds: https://github.com/rpi-ws281x/rpi-ws281x-python
- Get the software for the binary clock from from github: https://github.com/teddycool/BinaryClock
- Fix autostart at boot, add these lines before 'exit' in /etc/rc.local
  - cd /home/pi/PyBinaryClock
  - sudo python Main.py

Binary display

The circuit board design is presented in the 'Eagle' directory and version 3 of the board can be directly ordered from here: https://aisler.net/p/VFLSRFLZ

A bill of material is not compiled yet but all components are defined in the circuit diagram, also presented in the 'Eagle'-directory.

The PyBinaryClock software:

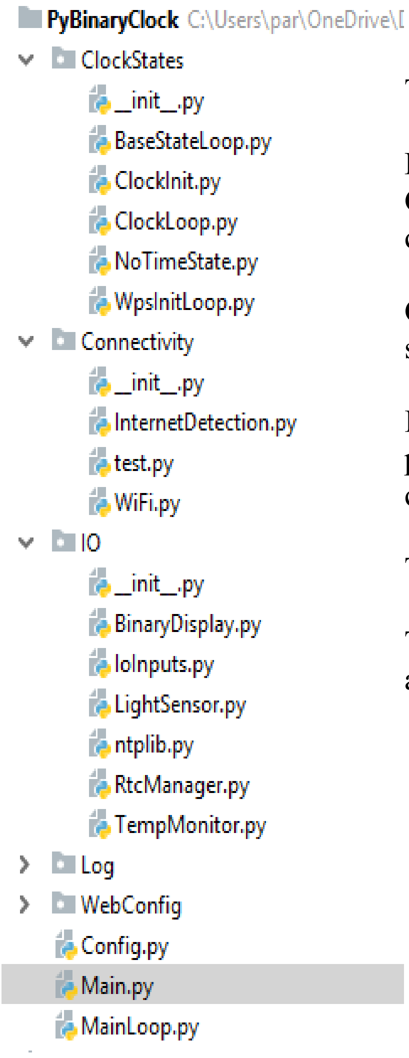
The code depends on several libraries and the Raspberry Pi has to be set up for i2c, a real-time clock and ws281x among other things. Follow the hardware chapter above or the refs in the source-code to get going.

The work-horse of the software is a very simple 'game-loop'. Everything starts with running Main.py which in turn starts the MainLoop.py that initialize everything and then the update-method in this module is run over and over again.

The second important part is the state-machine. This sets a state depending on some prerequisites and the the current state is run by MainLoop.update.

The ClockLoop state updates the time and display every 1/5th of a second. Taking current time and split to each figure (column) and then calculate the bit-pattern and set the LEDs accordingly.

Code structure



```
PyBinaryClock C:\Users\par\OneDrive\[
  ClockStates
      __init__.py
      BaseStateLoop.py
      ClockInit.py
      ClockLoop.py
      NoTimeState.py
      WpsInitLoop.py
  Connectivity
      __init__.py
      InternetDetection.py
      test.py
      WiFi.py
  IO
      __init__.py
      BinaryDisplay.py
      IoInputs.py
      LightSensor.py
      ntplib.py
      RtcManager.py
      TempMonitor.py
  Log
  WebConfig
  Config.py
  Main.py
  MainLoop.py
```

The software is split into several modules.

From top:
ClockStates is the state-machine with one base-class and inherited classes for each defined state.

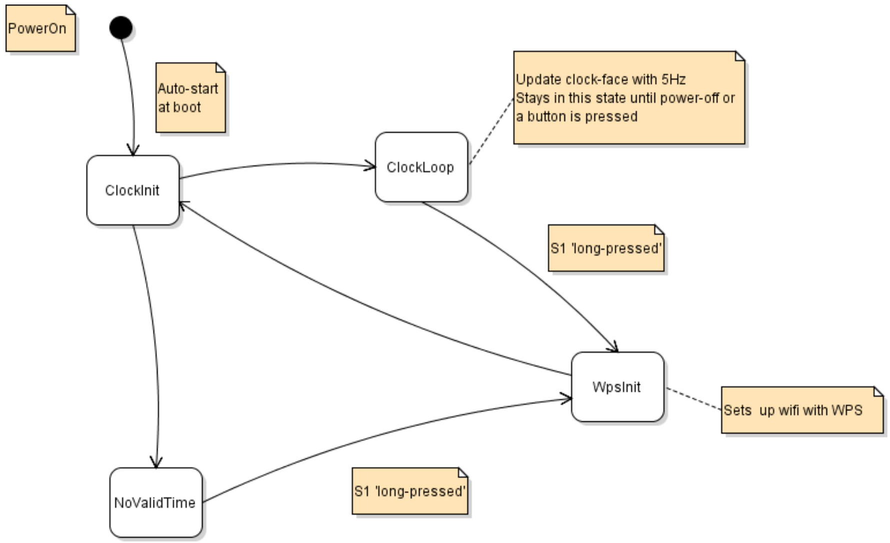
Connectivity: These classes handles internet detection and WiFi setup etc.

IO: These classes handles the hardware of the clock-display. Each part of the hardware has its own class. The one with most code is the clock display itself (BinaryDisplay.py)

The Log and the WebConfig is not yet implemented.

The files in the 'root' is the startup file Main which calls MainLoop as described above.

The State-machine



The clock starts in the 'init' state where network connection is checked and if there is a valid time present, either from the network or from the real-time clock.
If a valid time is detected the clock jumps to ClockLoop state and stays there until power off or if S1 is 'long-pressed'.
A valid time doesn't necessarily mean the correct time....

If no valid time is detected (no network connection and also no valid time from the real-time clock) the clock jumps to NoValidTime state. This state is kept until either a network is detected or Wps is set up via a long-press on S1.