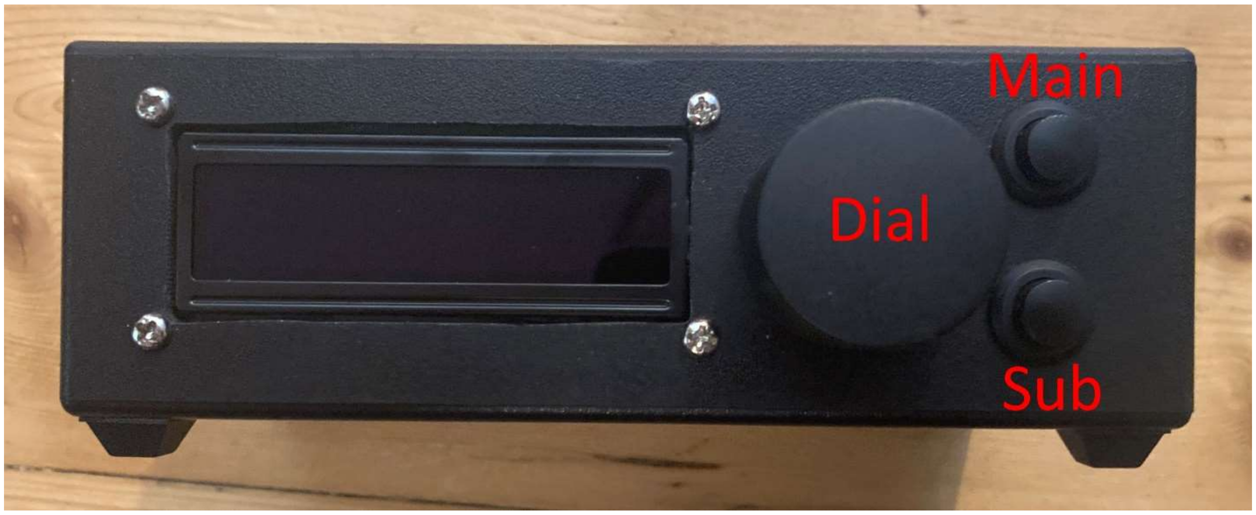
Fairy Light 3000



Introduction

Control of 2 sets of high voltage (~30V) fairy lights and a clock/alarm.

Features

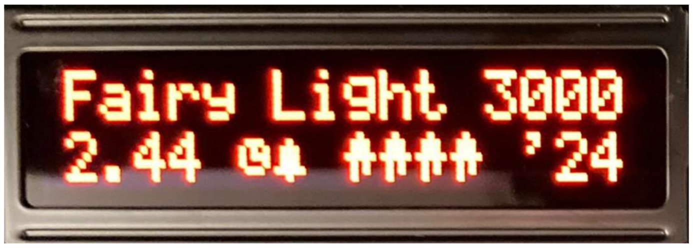
- Control of 2 sets of high voltage fairy lights
- Turbo mod – high voltage for extra bright lights
- Multiple patterns with tweakable controls
- Clock with multiple displays.
- Alarm with snooze, rising alarm volume, time limit.
- Lights can be triggered by the alarm.
- Light level sensor
- USB programming port / Time Debug port for timing debugging
- One press favorite light setting

Modes

There are 5 main screens which can be cycled through by pressing the **MAIN** button.

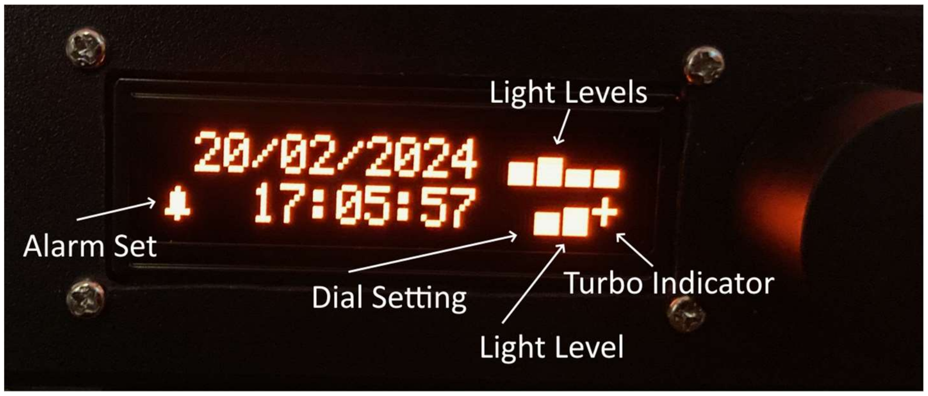
Start Up

A splash screen appears during loading.



Clock

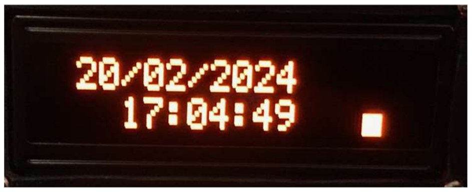
This is the main clock view.



It has 3 sub pages which are accessed by pressing the **Sub** button.

If an alarm is set a bell icon is visible on the bottom left of the screen.

Date

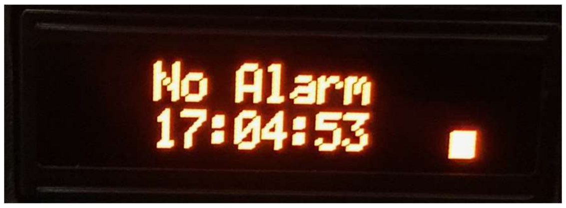


This screen displays the time and the above it the date. The current light values and analog inputs are visible on the right-hand side of the screen

Alarm

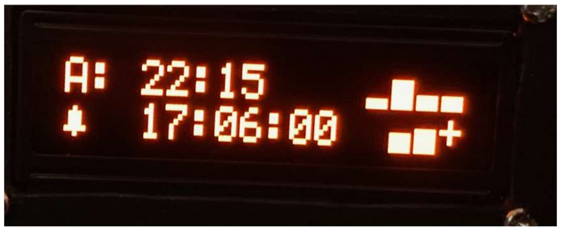
This screen displays the time and alarm. The current light values and analog inputs are visible on the right-hand side of the screen

If no alarm is set, it will simply display 'No Alarm'



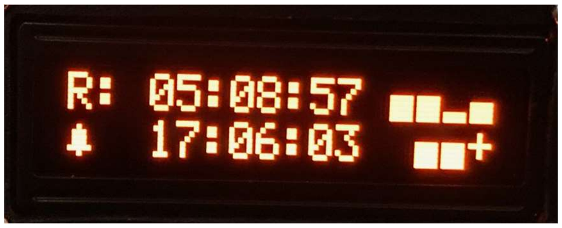
If a alarm is set, then turning the dial will show 3 different views:

Alarm Time



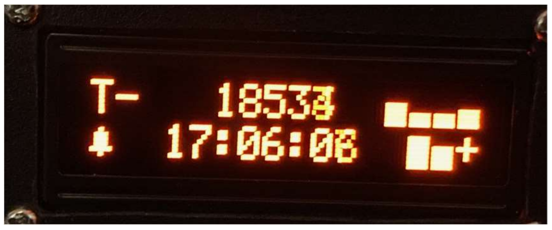
the time alarm will sound

Remaining Time



time in HH:MM:SS until the alarm will sound. Once the alarm is sounding, then this will display the amount of time the alarm has been sounding for.

T-Minus

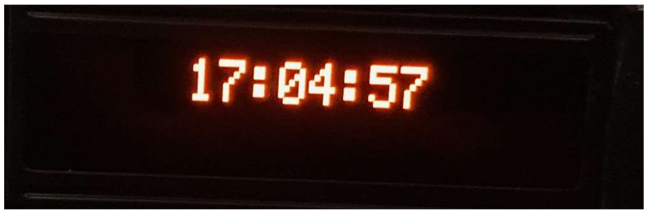


Number of seconds until the alarm sounds. Inspired by the countdown in the film Alien. Once the alarm is sounding, then this will display the amount of time the alarm has been sounding for.

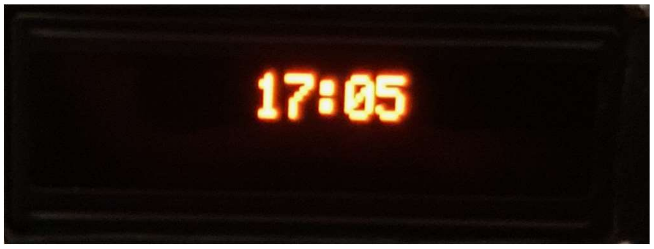
Minimal

This display shows a minimal time display for minimal brightness whilst sleeping. Turning the dial will show 5 different views:

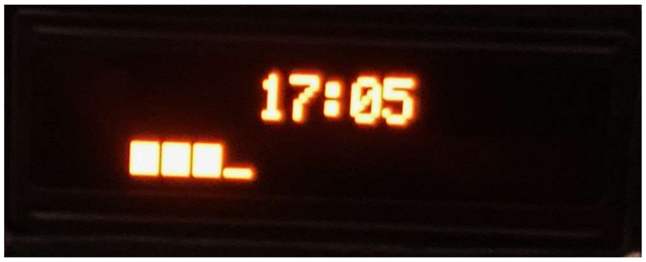
HH:MM:SS



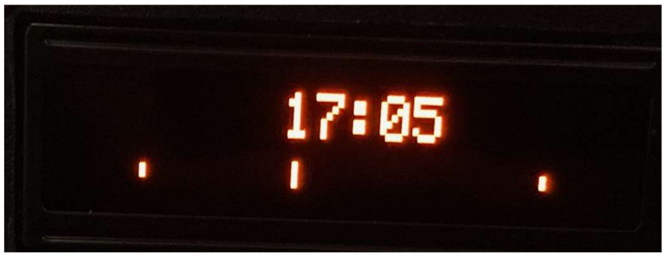
HH:MM



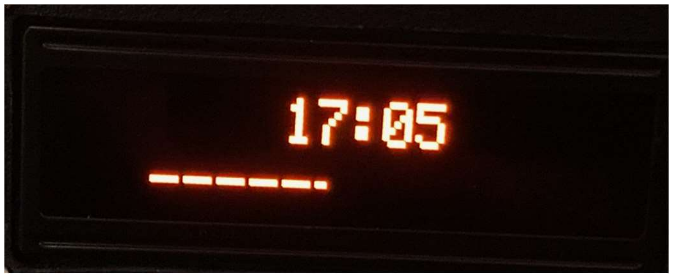
HH:MM with cumulative vertical second bar



HH:MM with horizonal second bar



HH:MM with cumulative horizonal second bar



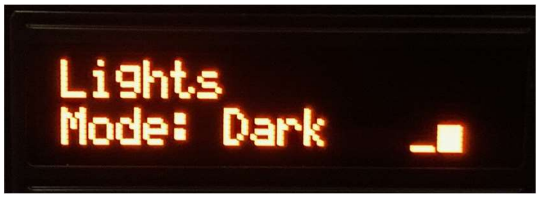
Lights

This mode is where the lights pattern can be selected and tweaked.

It has multiple subpages which are accessed by pressing the **Sub** button.

The lights can be selected from the page which has the 'Mode' text. Turning the Dial will choose between the 6 available lights:

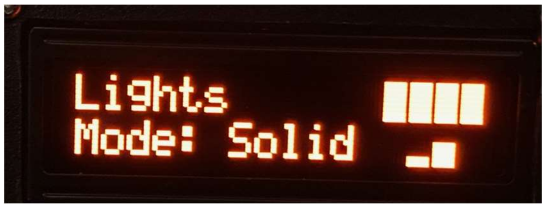
Dark



No Lights

There are no settings for this mode.

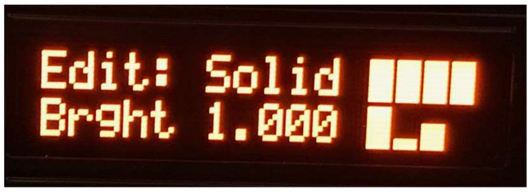
Solid



Solid Light

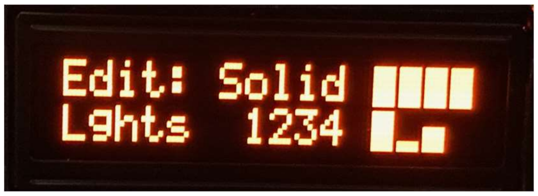
Settings:

Brightness



Sets the brightness levels.

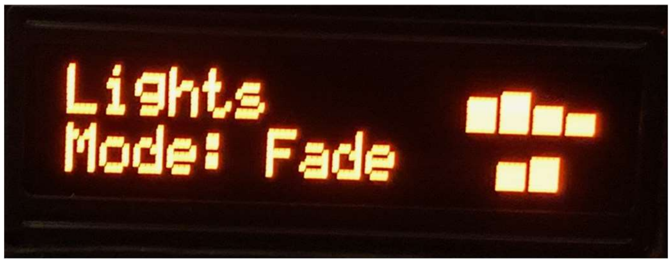
Lights



```
Edit: Solid
Lghts  1234
```

Select which of the 4 lights groups are visible.

Fade

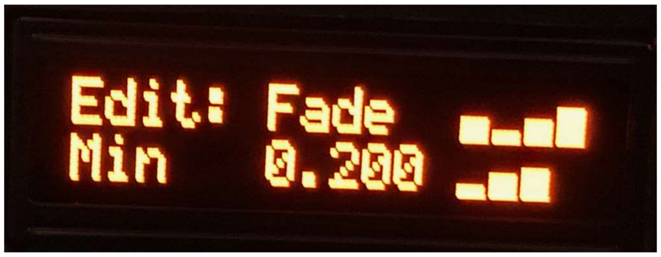


```
Lights
Mode: Fade
```

Fades between the min and max values. The light groups of out of phase, the speed can be controlled.

Settings:

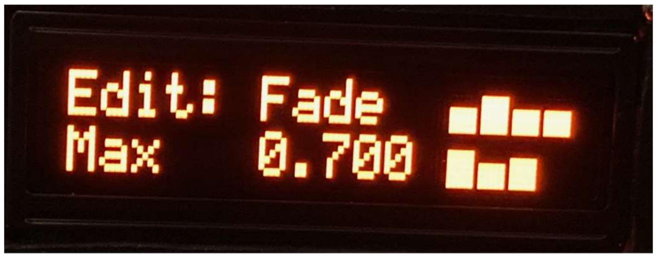
Min Brightness



```
Edit: Fade
Min   0.200
```

Sets the minimum brightness level.

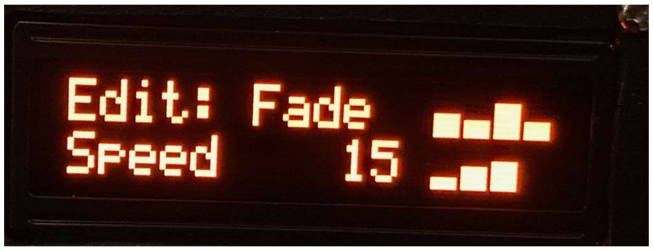
Max Brightness



```
Edit: Fade
Max   0.700
```

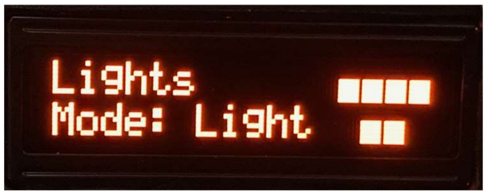
Sets the maximum brightness level.

Speed


Edit: Fade
Speed 15

Sets the speed of the animation.

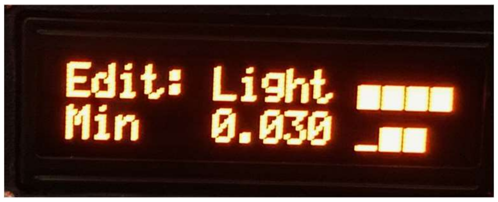
Light


Lights
Mode: Light

The lights brightness is determined by the light level in the room.

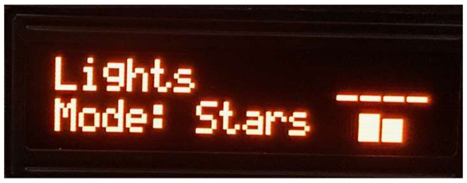
Settings:

Min Brightness


Edit: Light
Min 0.030

Sets the minimum brightness level.

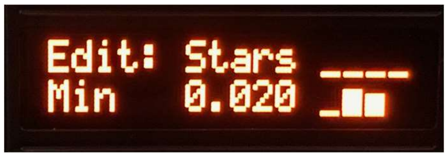
Stars


Lights
Mode: Stars

Twinkles between the min and max values. the speed can be controlled as well as the probability of a twinkle occurring.

Settings:

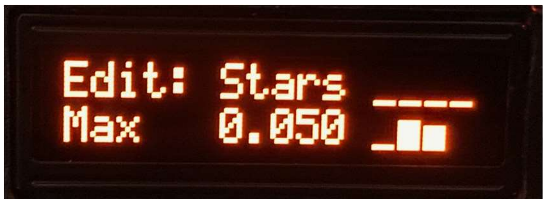
Min Brightness



```
Edit: Stars
Min   0.020
```

Sets the minimum brightness level.

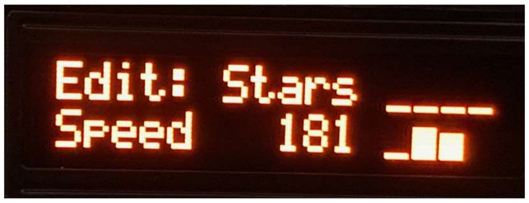
Max Brightness



```
Edit: Stars
Max   0.050
```

Sets the maximum brightness level.

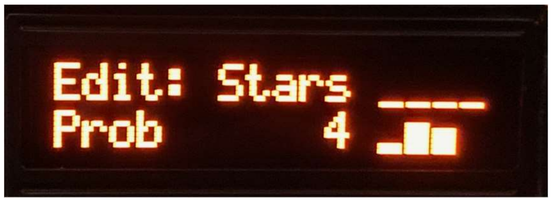
Speed



```
Edit: Stars
Speed   181
```

Sets the speed of the animation.

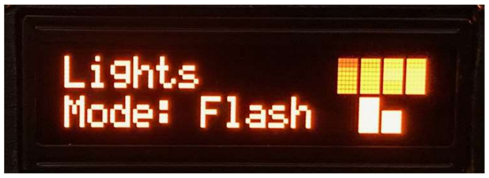
Probability



```
Edit: Stars
Prob      4
```

Sets the probability of a transition occurring.

Flash

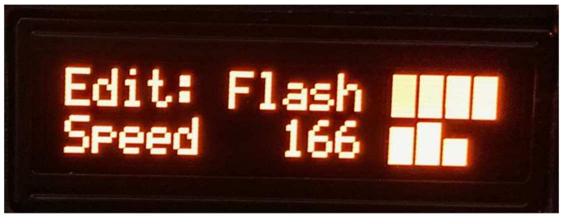


```
Lights
Mode: Flash
```

Rude on/off flashing. This is ideal for the alarm.

Settings:

Speed



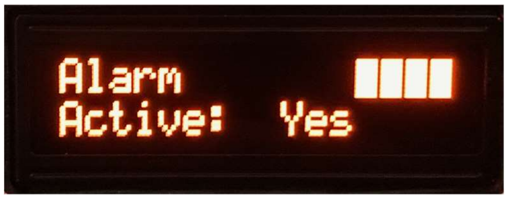
```
Edit: Flash ████
Speed   166 ██▄
```

Sets the speed of the animation.

Alarm

This is where the alarm can be set. There are 2 screens. Pressing the sub button cycles between them and turning the dial will change the value.

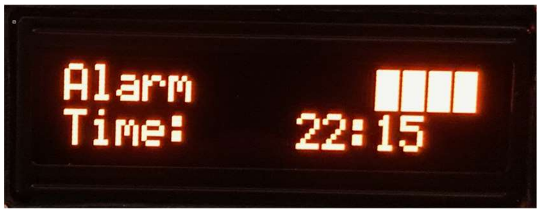
Active



```
Alarm        ████
Active:  Yes
```

This is where the alarm can be toggled on and off.

Time



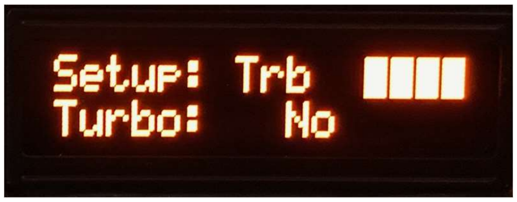
```
Alarm        ████
Time:      22:15
```

This is where the time for the alarm can be set.

Setup

There are approximately 20 user configurable settings. This mode is where they can be set. Pressing the sub button cycles between them and turning the dial will change the value.

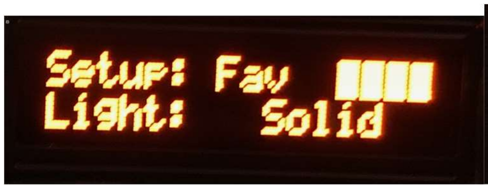
Turbo

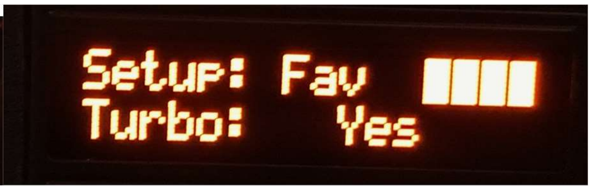


```
Setup: Trb ████
Turbo:   No
```

This toggles turbo mode.

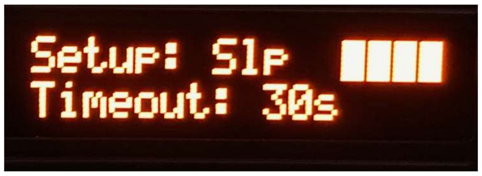
Favorite Light / Favorite Turbo


Setup: Fav
Light: Solid


Setup: Fav
Turbo: Yes

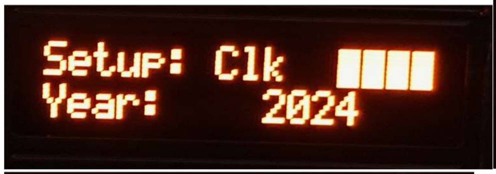
This set the favorite light and turbo value.

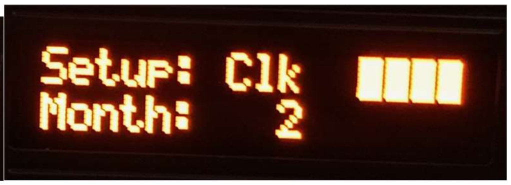
Sleep Timeout

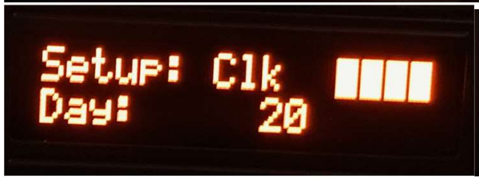

Setup: Slp
Timeout: 30s

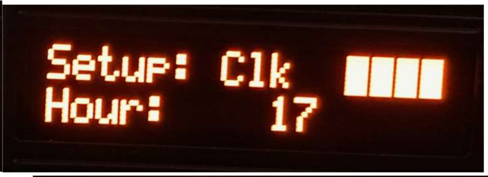
Sets the numbers of seconds till the screen dims, if set to infinite it never dims

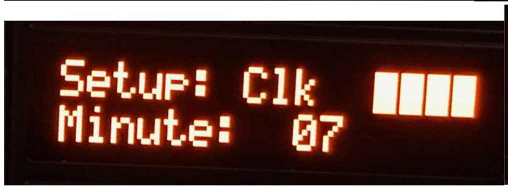
Clock set – Year/Month/Day/Hours/Minutes/Seconds

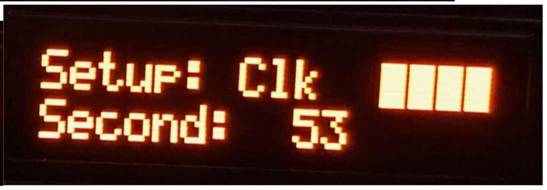

Setup: Clk
Year: 2024


Setup: Clk
Month: 2


Setup: Clk
Day: 20

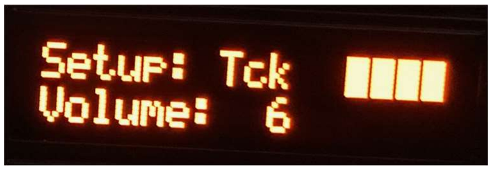

Setup: Clk
Hour: 17


Setup: Clk
Minute: 07


Setup: Clk
Second: 53

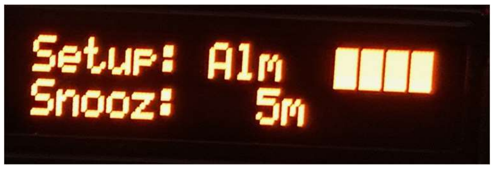
Six screen to set the clock time.

Second Tick


Setup: Tck
Volume: 6

Set the volume of the second’s tick.

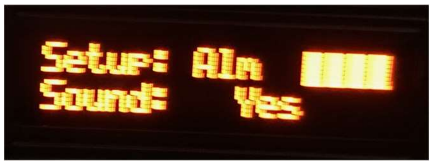
Alarm Snooze



Setup: Alm ████
Snooz: 5m

Set the how long the alarm snoozes for in minutes, can be set to No

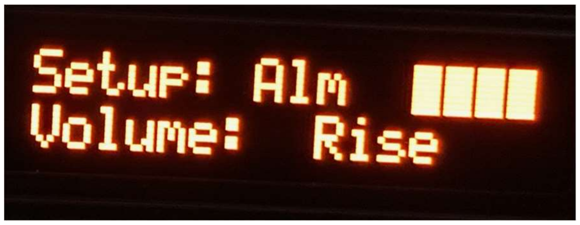
Alarm Sound



Setup: Alm ████
Sound: Yes

Sets whether the alarm makes a noise.

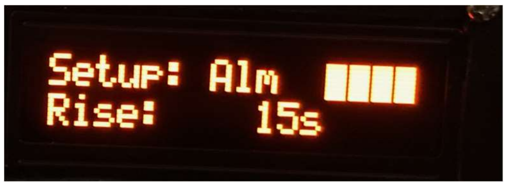
Alarm Volume



Setup: Alm ████
Volume: Rise

Sets the volume of the alarm. Between 1 and 9. If Rise is selected, the volume will get louder as it sounds.

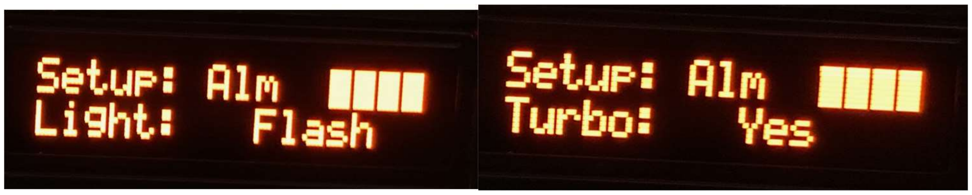
Alarm Rise



Setup: Alm ████
Rise: 15s

This sets the number of seconds until an increase in alarm volume occurs.

Alarm Light / Turbo

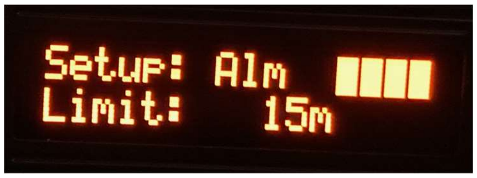


Setup: Alm ████
Light: Flash

Setup: Alm ████
Turbo: Yes

These 2 settings set whether a light will be triggered by an alarm. Setting light to ‘No mod’ means no light change will occur.

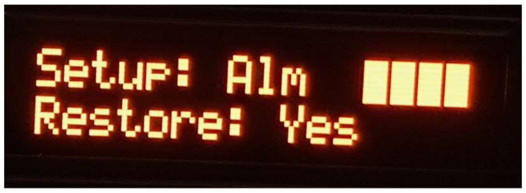
Alarm Limit



```
Setup: Alm  ████
Limit:    15m
```

This limits how long the alarm will sound if not stopped.

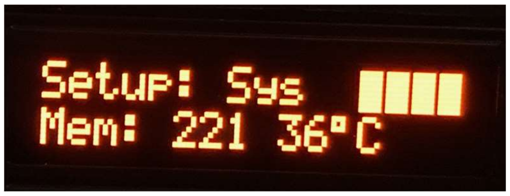
Alarm Restore



```
Setup: Alm  ████
Restore: Yes
```

If this is set to yes, after the alarm is stopped, then the light will go back to the previous state before the alarm started.

System Info



```
Setup: Sys  ████
Mem: 221 36°C
```

This displays the amount of free memory and the temperature of the RTC module.

Controls

Alarm

If the alarm is sounding, then the control takes on a different function.

Either the Main or the Sub button will stop the alarm and not trigger the snooze feature if enabled

Turning the dial by more than a 1/3 of it’s travel will trigger the snooze feature is enabled, or stop the alarm if snooze is not enabled.

Long Presses

If the button are held in for more than a second, then they perform different functions, and this depending on which screen is displayed.

Clock and Lights screen

Long Press Main

This will toggle the favorite function.

If there are no lights on (i.e. Dark) then the configured favorite will be enabled

If there is a light on, then Dark mode will be selected.

Long Pres Sub

This will toggle the turbo features (same as the setup screen)

Alarm Screen

Long Press Main

This will start the alarm sounding, useless to check the volume level

Setup Screen

Long Press Main

This will save all the current settings to EEPROM, so they will be available after power cycling.

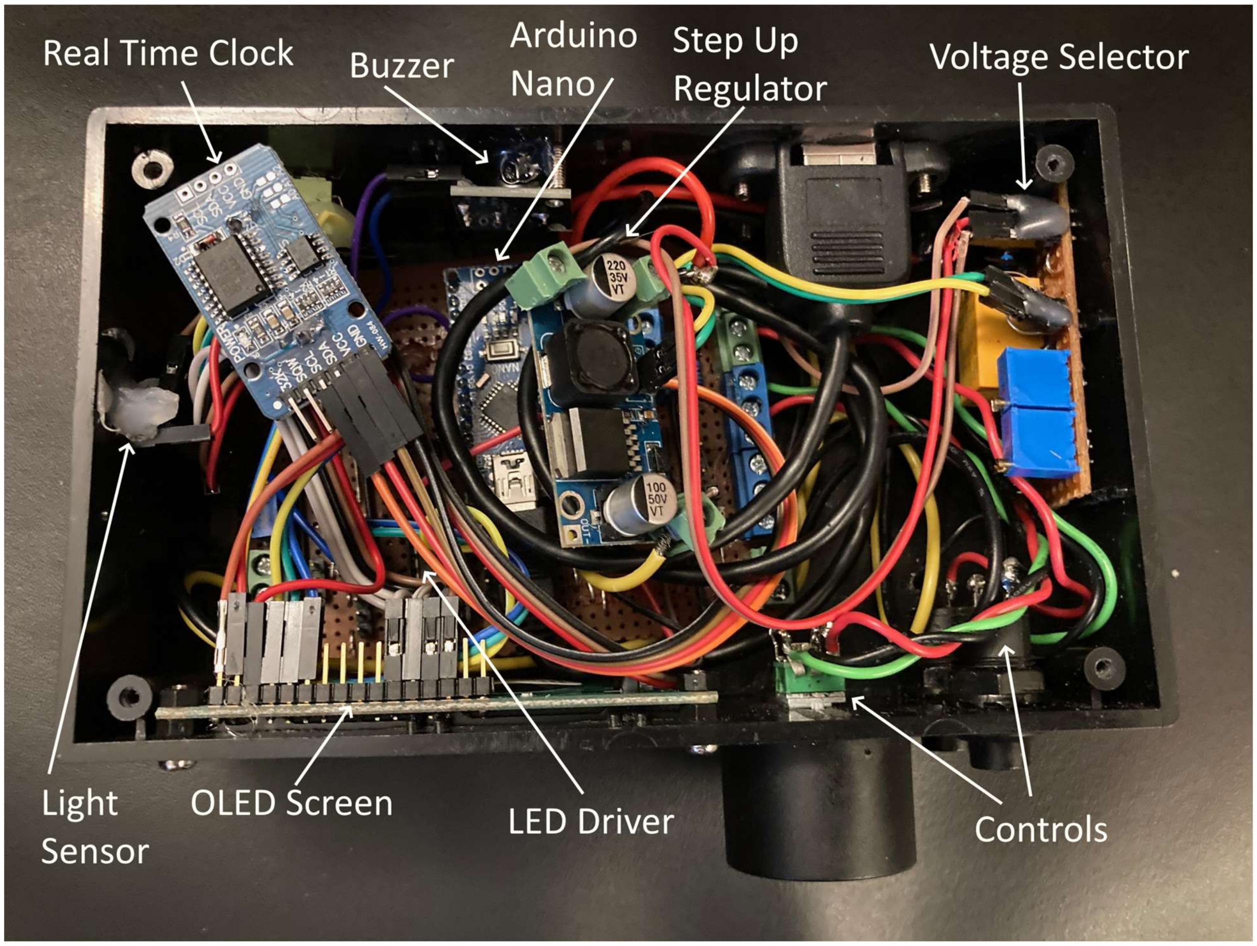
Long Press Sub

This will restore the factory default.

Note: Factory defaults can also be restored at boot time by holding down the Sub button as the device starts

Technical Details

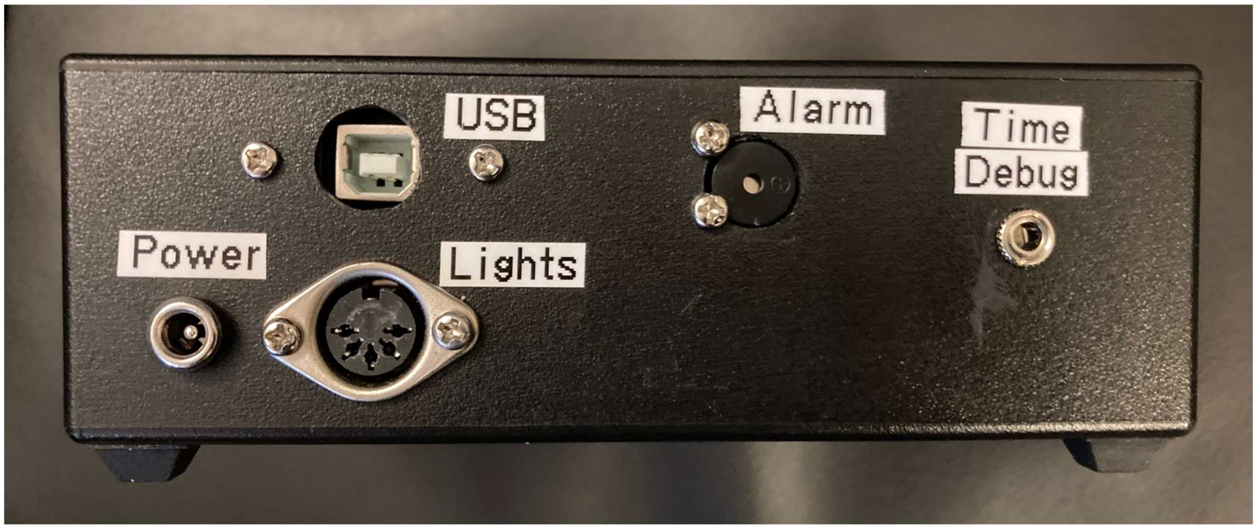
Internals



The Arduino Nano is the brain of the device. This is connected to all the other parts:

- The controls inputs – the Dial and the two switches (All use analog input pins)
- The OLED screen uses the 4bit parallel interface mode.
    - The OLED was modified to allow its brightness to be controlled, this is attached to a one of the digital output of the Nano and uses PWM to set the brightness
- The LED driver chips (L293D) is used to drive the fairy lights.
- The Step-Up Regulator converts the 12V input into either 29V or 31V (depending on the relay setting of the Voltage Selector board which is controlled by a pin on the Nano). This is the normal vs turbo.
- The Real Time Clock module uses I2C to talk to the Nano
- The Light sensor an LDR connected directly to an analog input on the Nano

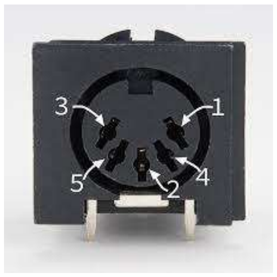
Rear Panel



Power

Power is center positive 12V

Lights



This is 5 pib DIN socket.

Pins 3 and 5 connect to the 1st Fairy lights strip

Pins 1 and 4 connect to the 2nd Fairy lights strip

Pin 2 is unused

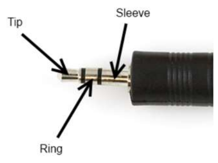
Alarm

This is a buzzer. It can get very loud!

USB

This is used for firmware updates. The device can be powered of this, but then the fairy lights do not work

Time Debug



This is used for timing the device (See details in Theory of Operation).

Sleeve is ground.

Ring is the signal which controls the polarity of the LED driver.

Tip is the active signal sent to the LED driver to enable the LEDs.

To make use of this, connected Ring and Tip to an oscilloscope.

Theory of Operation

The application runs in a loop, the first part of which is acting as a PWM timing circuit for the LEDs and the remaining time in which cycle is spent doing other tasks, such as the clock, alarm, button handling, effects updates etc.

The main loop does the following:

- Exec chunk (each chunk is round robined across different frames)
    - Chunk 1:
        - Get Time from RTC chip (slow!)
    - Chunk 2:
        - Check Buttons
        - Check Analog inputs
    - Chunk 3:
        - Create Time/Date string
        - Sleep Checks
        - Handle button presses
        - Check Alarm Conditions
        - Fade LCD update
        - Splash screen sequencing
        - Screen manager updates
- Play alarm if sounding
- Update Current Effect
- Update OLED

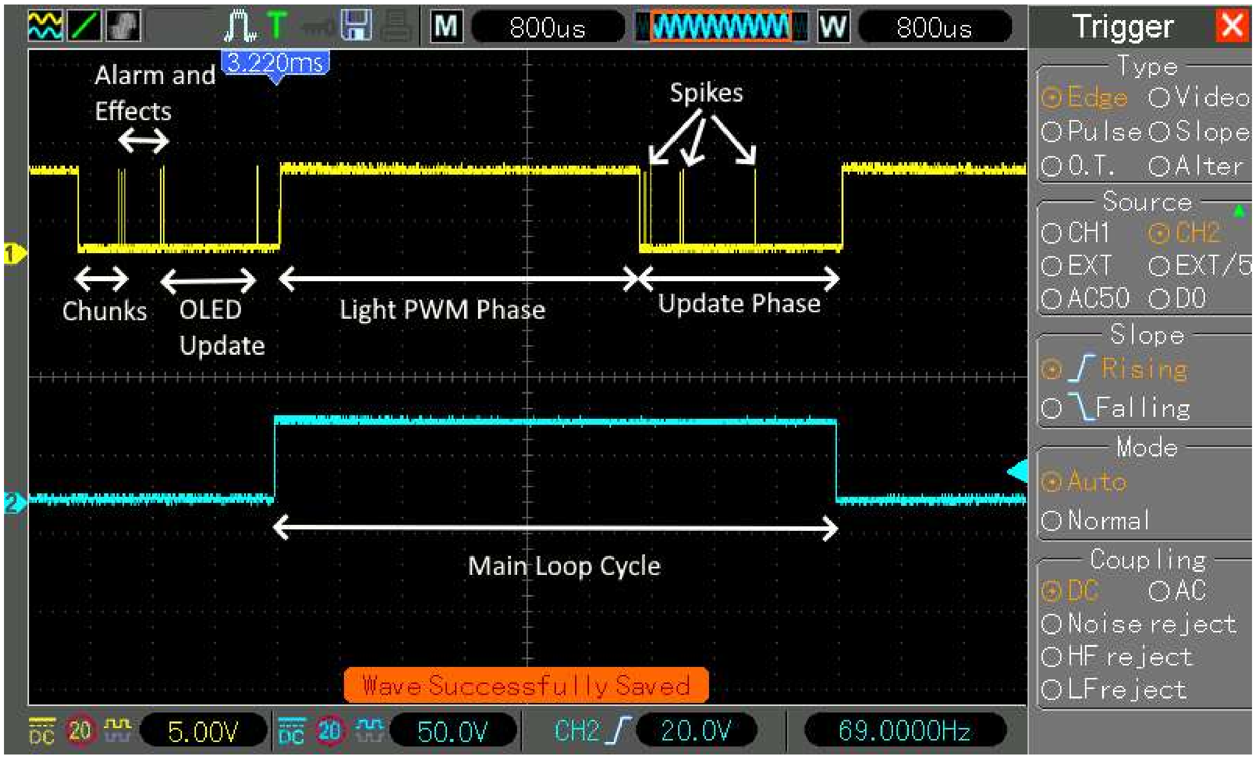
The screen have a screen manager which has a list of screen sets, and each screen set has screen.

A screen is defined with a draw() and tick() method and the manager controls change the screen sets, and the screen sets control the screens.

Each screen also has a list of setting, which have draw() and tick() methods

Timing

Each cycle is as long as the blue state below.



(The yellow is the active signal (tip), and blue the polarity (ring) and for the above **#define SPIKE** was uncommented in the code.)

Looking at the yellow trace, the solid block as the start is where the PWM phase is handled, in the above brightness is full (so about 2/3 of the full duty cycle is achievable), the remaining 1/3 is where all other operations occur.

In the code, this is broken in different chunks which can cycled through and handled on different frames (for non-time critical things)

The only things which happen per frame are the LED updates, and the updates to the OLED.

The OLED is so slow that it can't be sent more than 1 character per frame (or for font changes, the font data is send in groups of 4 bytes per frame)

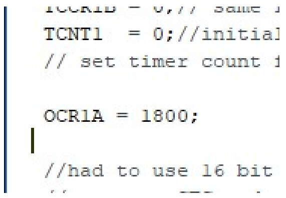
The 'Spikes' visible on the yellow trace are put into the code at various places:

- After the chunk processing
- After the alarm/effect update
- After The OLED update

On the left of the trace the spikes are clearly visible, and the 3rd one is almost at the start of the new cycle. This is the optimal balance point, where it almost touches. The worst thing to do is change the view on the minimal screen as this does font changes which is the slowest op.

Letting the 3rd spike bleed into the next frame will cause the whole frame to be skipped and flickering will be noticeable on the LEDS.

In BunningFairyDriver2.ino the period of the cycle can be set here:



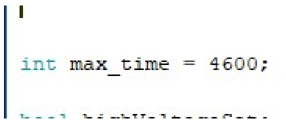
```
  TCCR1B = 0;// same f
  TCNT1  = 0;//initial
  // set timer count f

  OCR1A = 1800;

  //had to use 16 bit
```

Values much larger than this will cause visible flickering on the LEDs and strobing on the OLED backlight.

If code is added which cause flickering then the maximum time the lights can be reduced (essentially shortening the initial yellow period, which gives more time to updates etc. by decreasing the number in L293DLED.cpp The downside is the maximum brightness of the LEDs is reduced)



```
int max_time = 4600;
```

If possible, it is better to move code into a new chunk to spread out the load across frames.