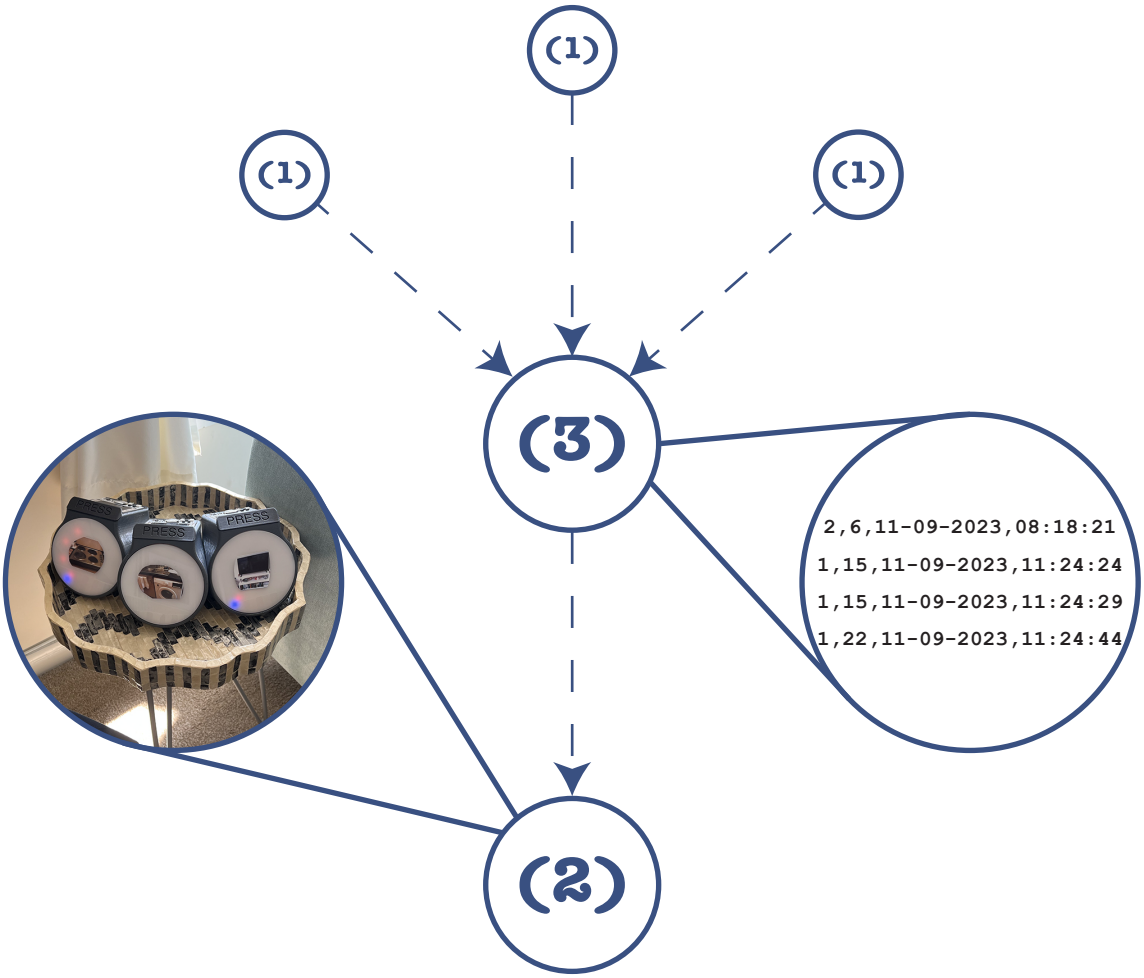
# Older Generation Toolkit

**The Older Generation Toolkit is made up of three parts: (1) a collection of self-powered sensors, (2) self-powered displays, and (3) a central hub to collect sensor activation data and transmit activity data to the corresponding displays. Each self-powered sensor is designed to be as flexible as possible to allow the widest range of applicability, where sensors can be used for many types of interaction e.g. one sensor type can detect both door and drawer use.**

**These three parts allow the tracking of activity data around the home. For example, you might like to track the amount of activity you do with your hobbies, cooking or chores. Surfacing this activity promotes self-reflections such as "I read less than I thought I did".**

**The three parts of the system work together to collect sensor activity data and display it back to the user in a meaningful way.**



# Contents

# Self-Powered Sensors

## Base Components

The ToolKit uses several off-the-shelf self-powered sensors. These sensors are configured and modified to adapt their intended use in the ToolKit. These components include:

-- Cypress CYALKIT-E03 Self-Powered BLE Sensor Beacons
    -- Used to sense when an item is being used when exposed to light. Such as storage boxes or books.

-- EnOcean PM220 Energy Converter Modules
    -- Used to detect when an object that requires a linear interaction has been used. Such as doors and cupboards

-- InPlay IN100 NanoBeacon BLE Beacon
    -- This device allows the connection of different trigger modules, such as PIR sensors, adding wireless communication to these simple sensors.

-- Epishine LEH3 Organic PV Eval Module
    -- The NanoBeacon trigger modules require constant monitoring. This PV module allows this, and allows the creation of custom self-powered sensors.

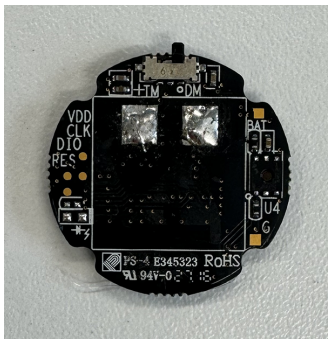
The ToolKit can be modified to accept other BLE devices, using the MAC address. The Toolkit listens for PINGS from the sensors, by simply searching to see if the device exists. No other data is recieved to preserve privacy.

# Cypress CYALKIT-E03

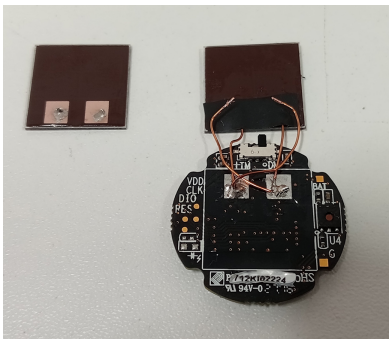
## Solar-Powered BLE Sensor Beacons


**Powered by ambient light falling on a 15 mm x 15 mm amorphous silicon solar module and charging 400 uF storage capacitor and 0.2 F supercapacitor via a S6AE103A energy power management IC. The beacon is set to demo-mode, meaning that the transmission interval varies with ambient light level (illuminance). The beacon operates at a minimum illuminance of 100 lux (e.g 100-150 lux is the illuminance in a typical domestic living room). The BLE transmit power is configured to 0 dBm (1 mW), with the time interval between BLE transmission being anywhere between 3 seconds (illuminance of 1000 lux) to 50 seconds (illuminance of 100 – 200 lux) depending on illumination.**

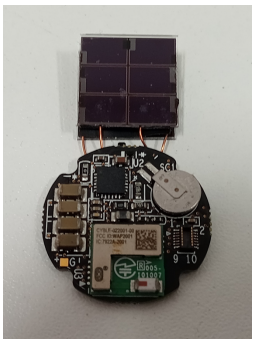
**Cypress BLE beacons are used in default configuration, except for the bookmarks, which require some modifications as detailed below.**




**Desolder PV Module using hot air**



**Solder wires to PV module and add insulation tape**



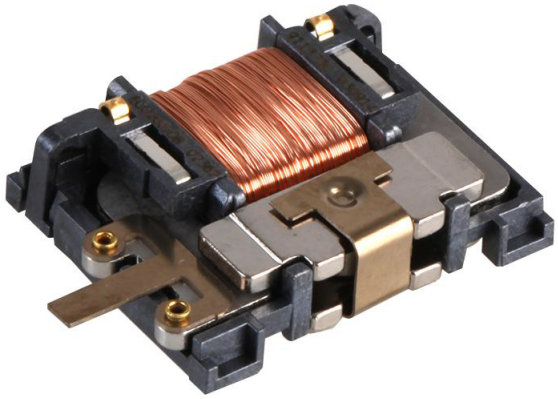
**Solder wires to second PV module**

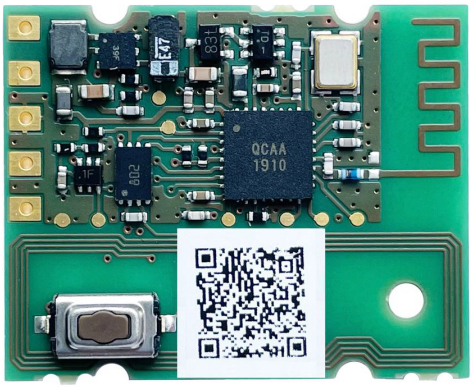
**Cypress BLE beacon used to make the bookmarks; the silicon PV module is carefully desoldered from the beacon PCB using a hot air re-work soldering station. Two (back-to-back) silicon PV modules soldered in place with connecting wires, so that the bookmark works either side up.**

[1]Configured via Tera Term using the supplied BLE-USB BRIDGE AND DEBUG BOARD

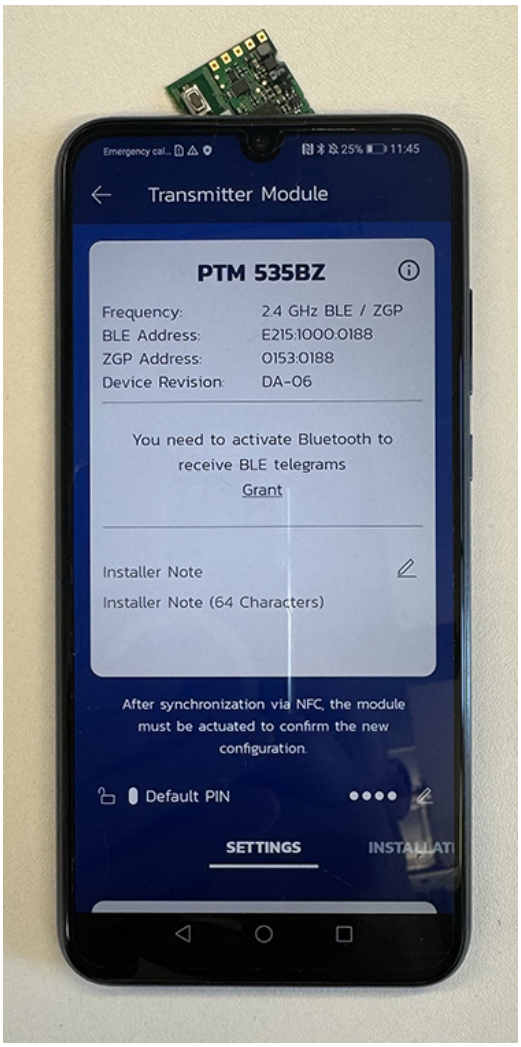
# EnOcean PM220

## Energy Converter Module






**EnOcean PM220 Energy Converter Module for Motion Energy Harvesting to Power PTM 535BZ BLE. Each PM220 activation generates between 120 to 210 µJ @ 2 V – which is sufficient to power the PM535BZ BLE transmitter module for a single wireless transmission.**




**PM535BZ configuration using a mobile App[2]**

[2]https://www.enocean.com/en/product/enocean-tool/

# InPlay IN100 NanoBeacon™

## Bluetooth® Low Energy Beacon

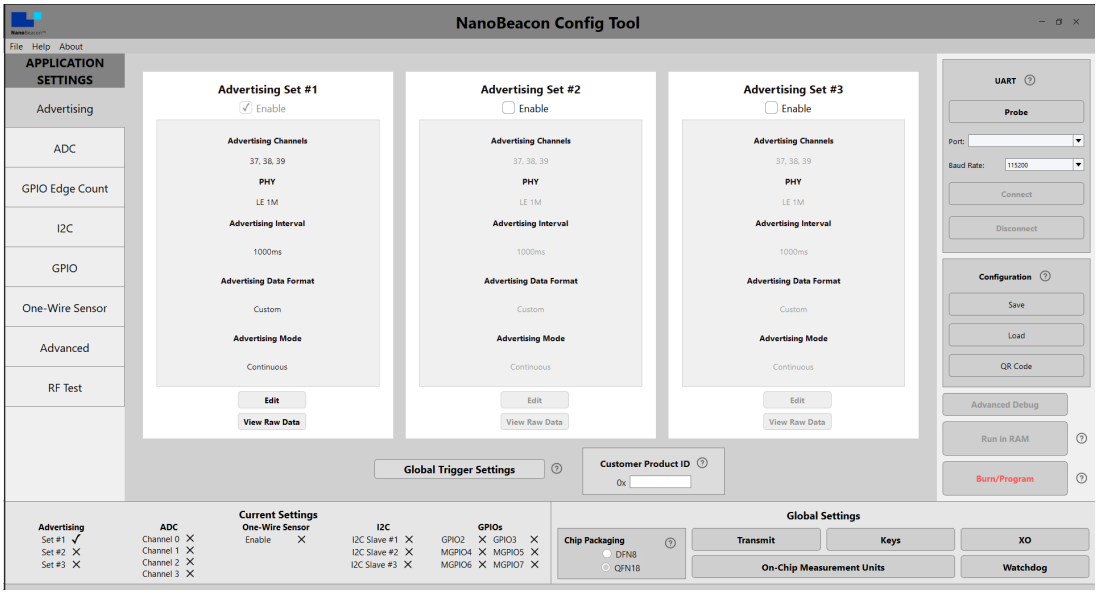
IN100 Development Board

Programmer Board

**InPlay IN100 NanoBeacon™ Bluetooth® Low Energy Beacon allows the attachment of different forms of trigger modules. We used a Panasonic PIR sensor EKMB1107112 to detect the presence of a person and trigger the NanoBeacon. The NanoBeacon is powered by an Epishine Organic Photovoltaic module**

**The NanoBeacon consumes less than 500 nA in sleep mode while the EKMB1107112 has a quiescent current consumption of only 1 uA. IN100 transmit power configured to 0dBm (1 mW). The IN100 is configured to transmit as an iBeacon.**

**NanoBeacon is configured using NanoBeacon Config Tool, via the plug in OTP programmer. The PIR sensor is connected to MGPIO4, which is configured as the trigger source for triggered advertising.**



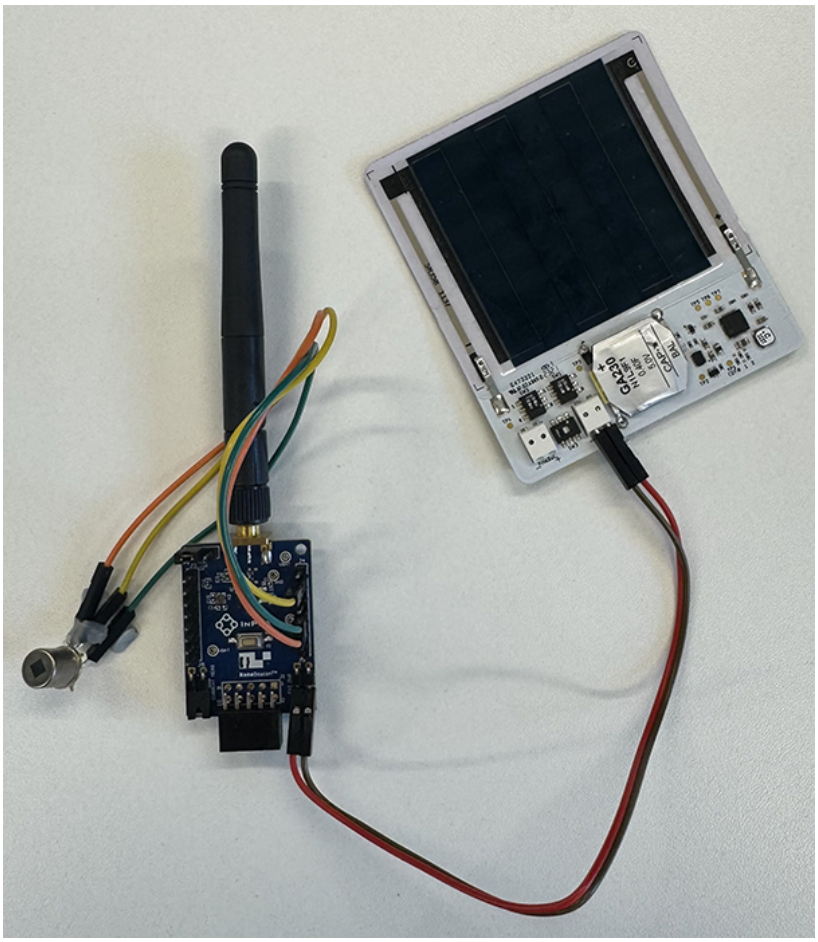

**InPlay IN100 is configured using NanoBeacon Config Tool[3]. Triggered advertising is set and MGPIO4 is used as the trigger source.**

[3]https://inplay-tech.com/nanobeacon-config-tool

# InPlay IN100 NanoBeacon™

## Bluetooth® Low Energy Beacon

**The InPlay IN100 BLE beacon is powered by ambient light energy harvested by an organic PV module. Beacon transmissions are triggered by a PIR sensor (with the plastic Fresnel lens removed to reduce the PIR sensor's field of view) connected to MGPIO4**



**EKMB1107112 PIR module**

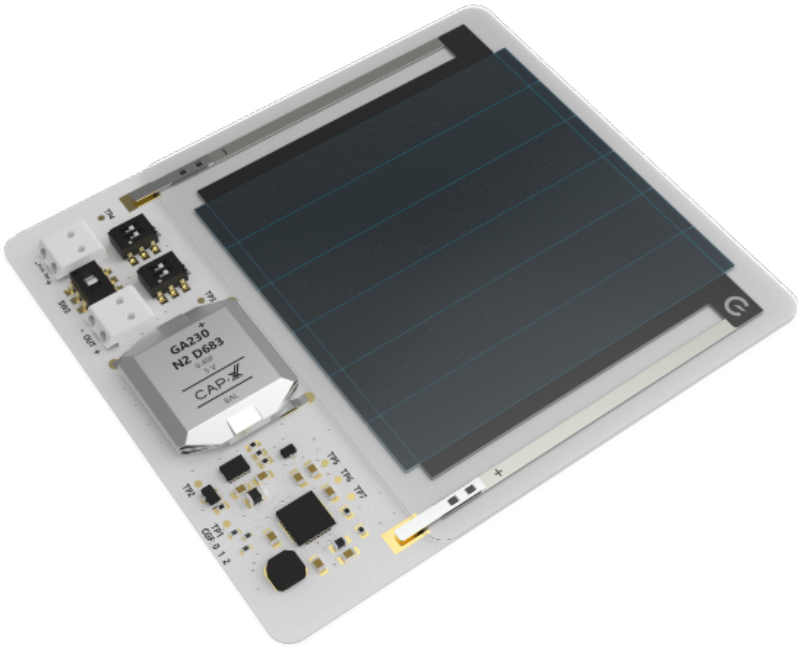
| | |
|---|---|
| **Detection Range** | **5 m** |
| **Field of view** | **110 deg x 110 deg** |
| **Standby current consumption** | **1 uA** |

# Epishine LEH3 Eval Module

## Organic PV Module

**Epishine modules are optimized for use indoors under illumination intensities ranging from 20 to 1000 lux.**

**The Epishine module powers the self-powered displays, and the NanoBeacon sensors due to their trigger events requiring constant monitoring.**




| | |
|---|---|
| Open Circuit Voltage | 3.8 V |
| Short Circuit Current | 147 uA |
| Output Power | 418 uW |

**Epishine LEH3_50x50_6 Organic PV module (at 500 lux warm white LED on white background)**

**5 V, 0.4 F super capacitor fully charged under 500 lux illuminance in approximately 5 hours. The total charge stored in the supercapacitor is 0.4 F * 5 V = 2 C.**

**Each IN100 trigger and BLE transmission, consumes approximately 13.65 uC (measured using Nordic NRF PPK2 Power Profiler Kit). This figure is negligible in terms of our deployment scenarios.**

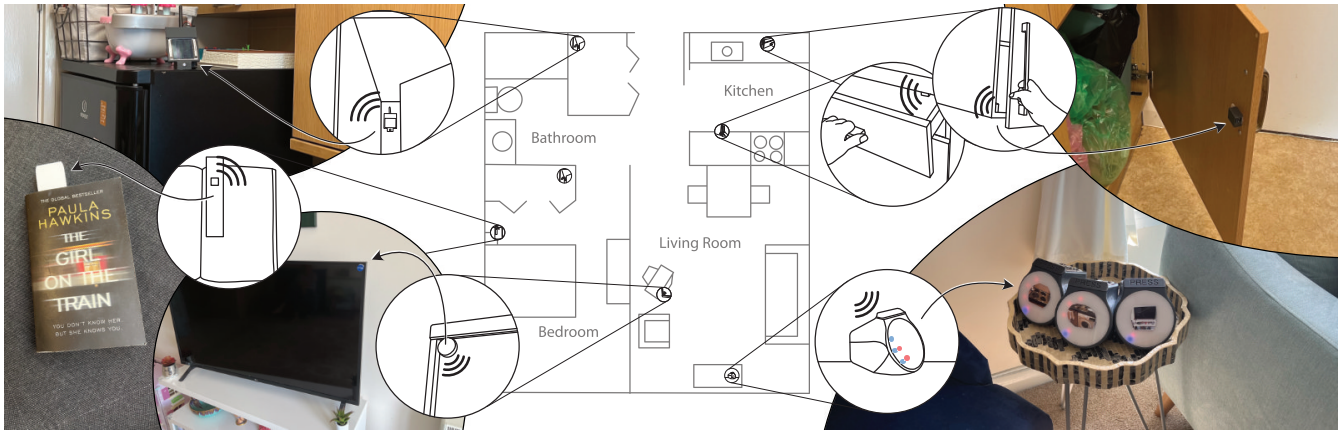
# Sensor Design
## and Placement

**After configuring the sensors and energy harvesters, we created a set of housings using 3D printing. Our models can be downloaded <LINK REDACTED FOR REVIEW>. Or you can choose to create your own.**

**Our models include:**

1. **A housing for the EnOcean PM220, which allows it to be attached to doors and drawers.**
2. **A laser cut template to create a Bookmark using the modified Cypress E03**
3. **A clip like housing for the Cypress E03, to allows attachment to storage boxes**
4. **A housing for the InPlay N100 and Epishine module to hold all the components together for a self-powered PIR sensor module.**


**The image below shows several examples of sensor placement**



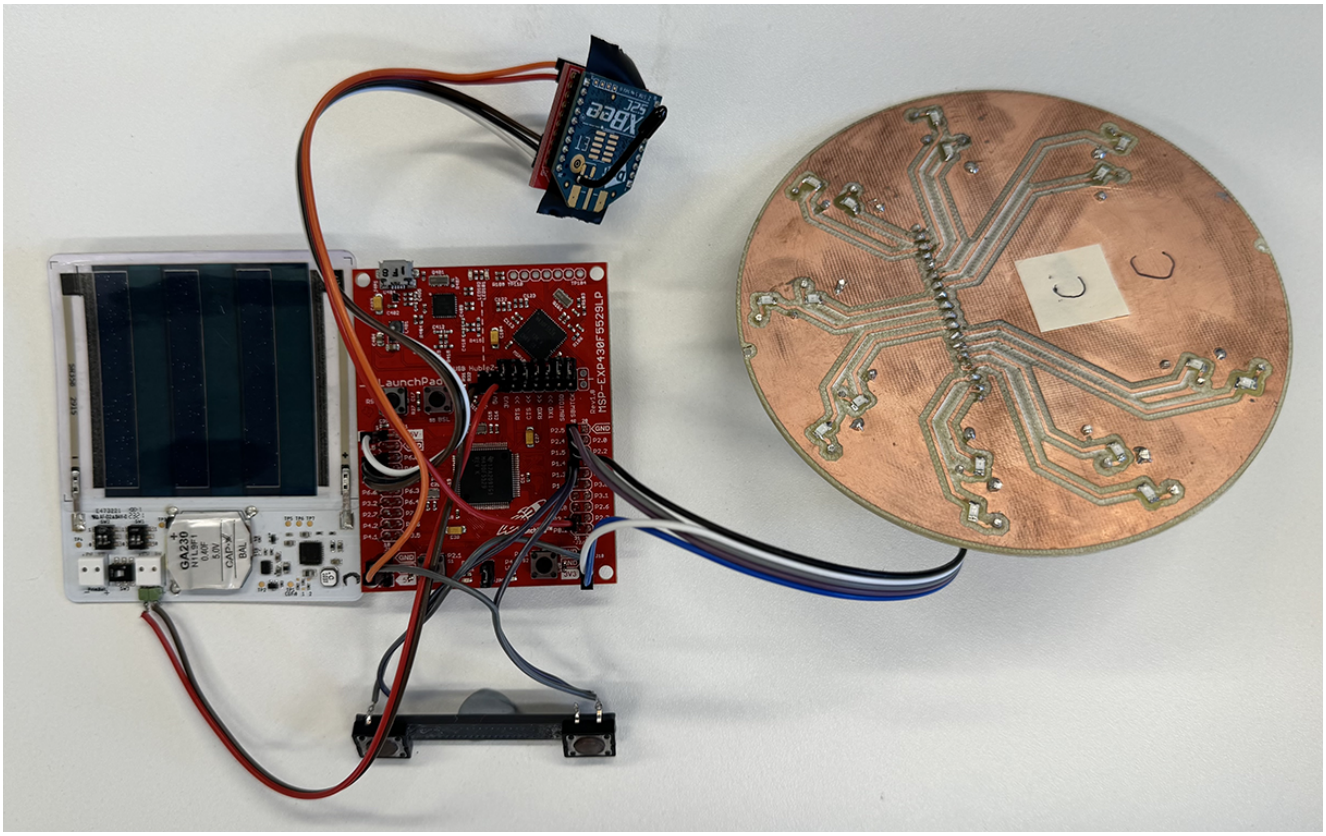
# Self-Powered Displays

**Two user interface form factors were created, “pods” and “fridge magnets”, which consist of a custom-made LED display board, driven by an ultra-low power microcontroller MSP-EXP430F5529 and a pair of 74HC595 shift registers. Wireless communications between the pods and central hub is via S2C Zigbee module. The pods are self-powered from ambient light energy harvested by Epishine Light Energy Harvesting (LEH3) evaluation kits. Total current consumption of the pods/fridge magnets in sleep mode is approximately 4.29 uA @ 3v3. The microcontroller launchpad is programmed via Energia IDE.**

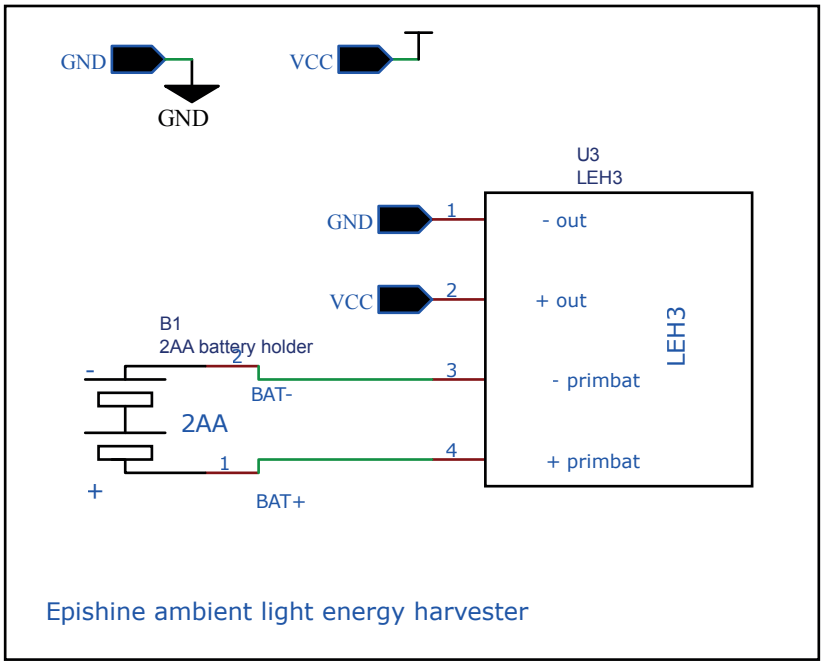
**The display of each interface consists of two circular rings of LED lights. The inner ring shows the average of an activity calculated since activation, and the outer ring shows the count of an activity for the current day. Each interface has an acivation button, which retrieves the activity information to display on the LEDs from the hub.**

**Each pod/fridge magnet activation consumes between 0.24 and 0.41 C, depending on the display data (i.e. number of LEDs illuminated). This means that a fully charged supercapacitor can power the pod/magnets for typically no more than 4 or 5 activations.**

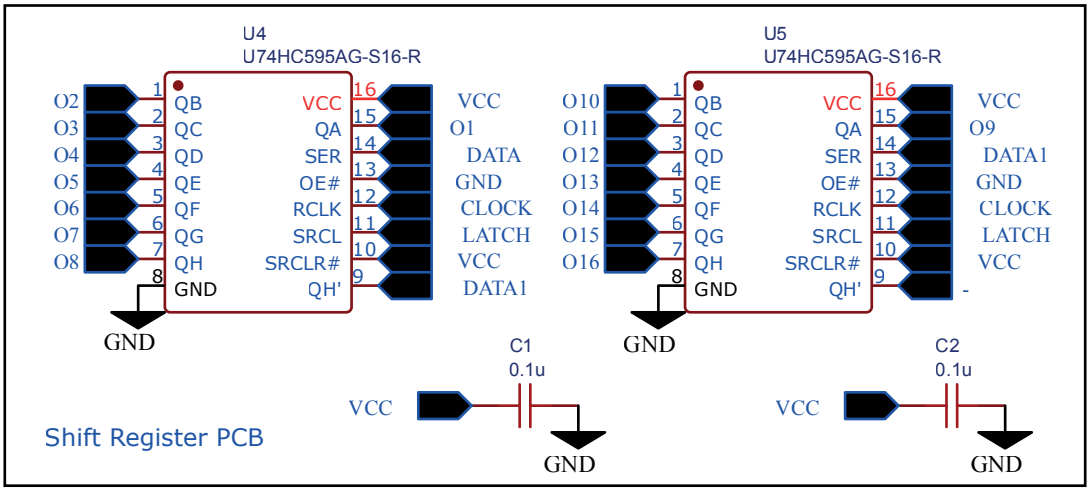
**The internal electronics are made to be identical, making production faster and easier. Then the form factor of housing is created depending on where the interfaces could be placed. We have created "pods", a desktop interface, and "fridge-magnets", which are intended to attach to refrigerators. The image below shows the internal components.**



# Schematic

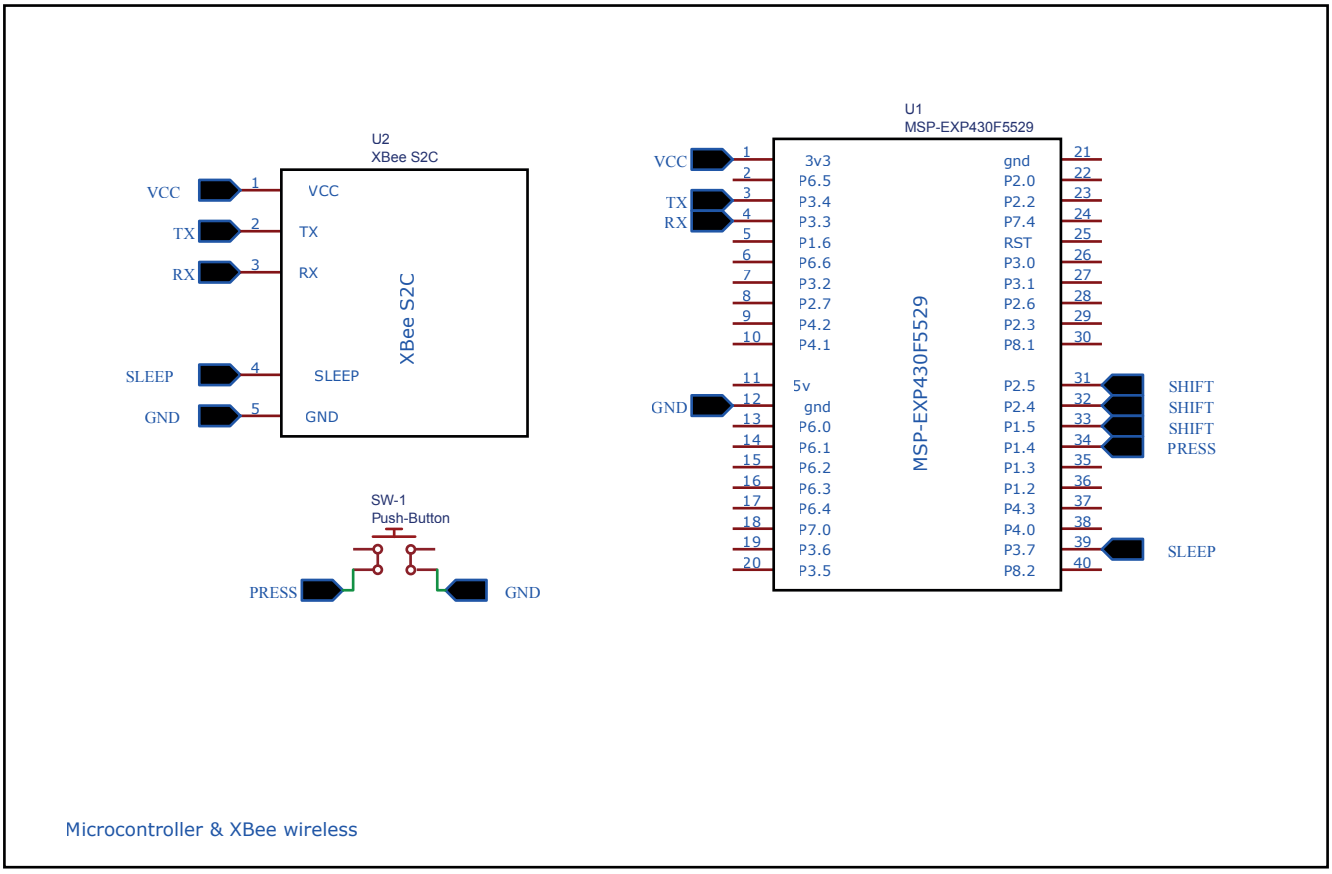



Epishine ambient light energy harvester

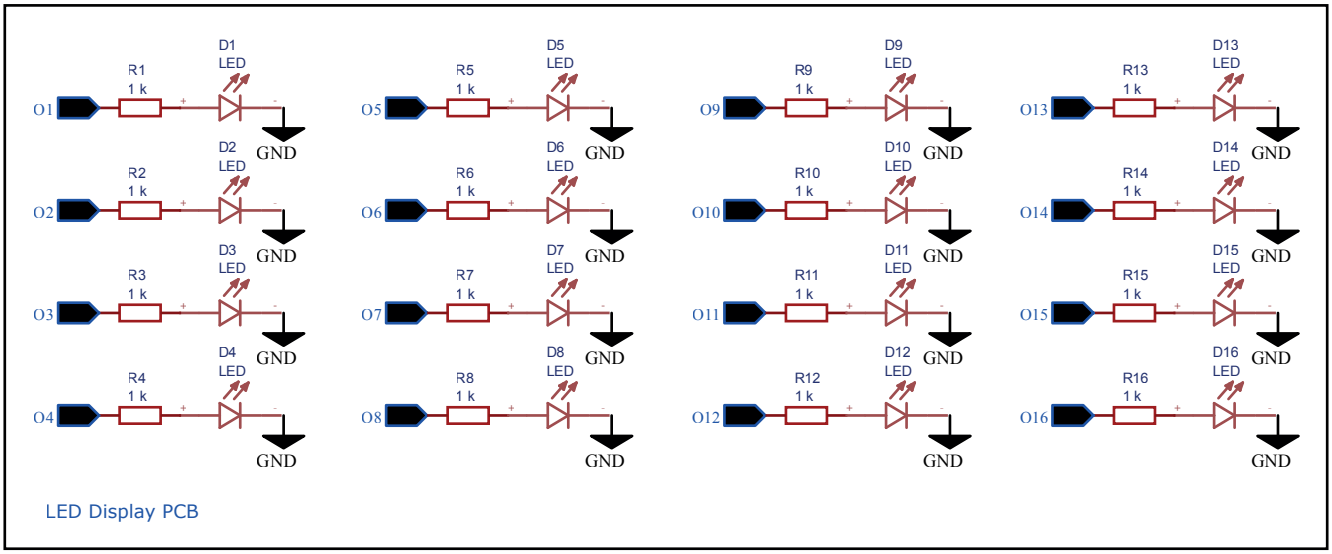



Shift Register PCB

# Schematic



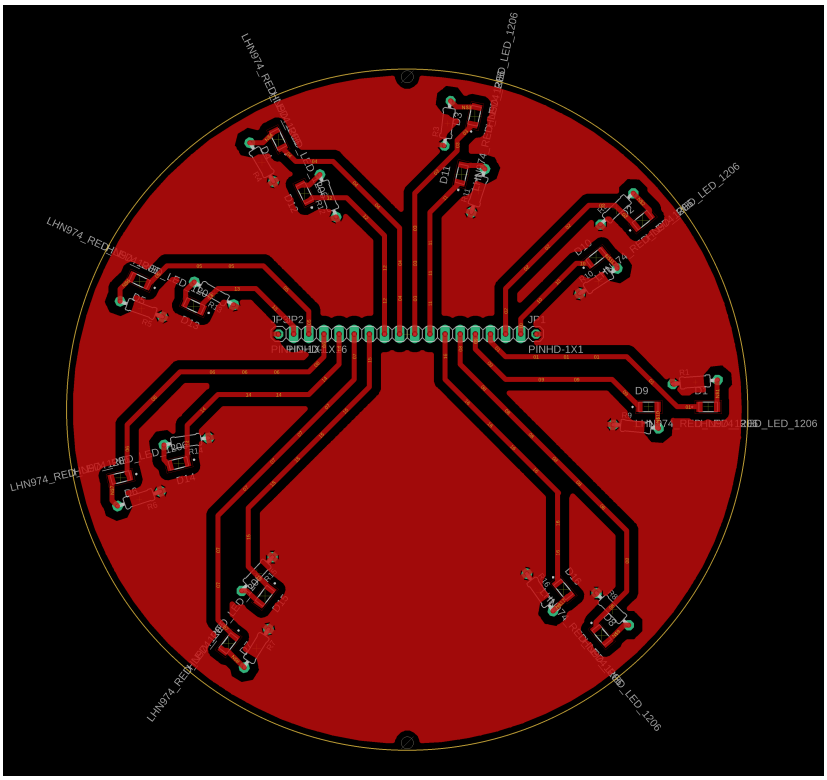

Microcontroller & XBee wireless



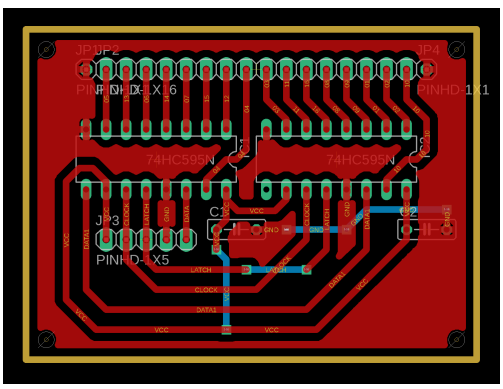

LED Display PCB

# PCB Design

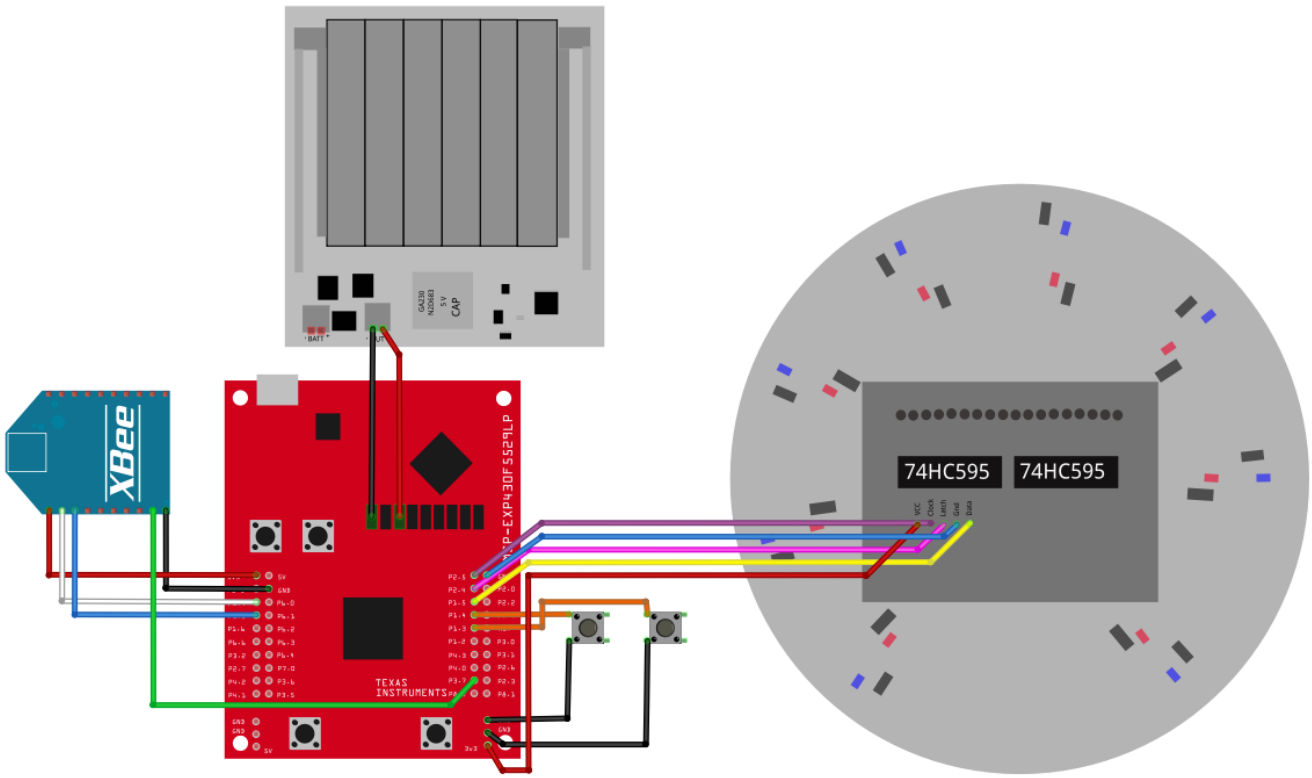
## LED Board Design




## Shift Register Board Design



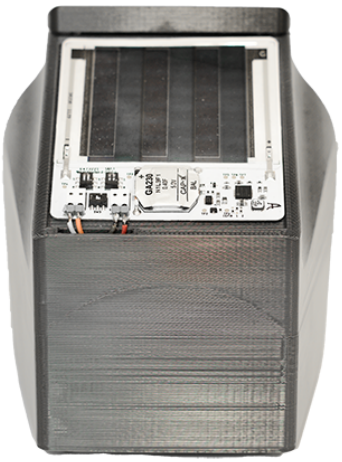
# Diagram



# Form-Factors

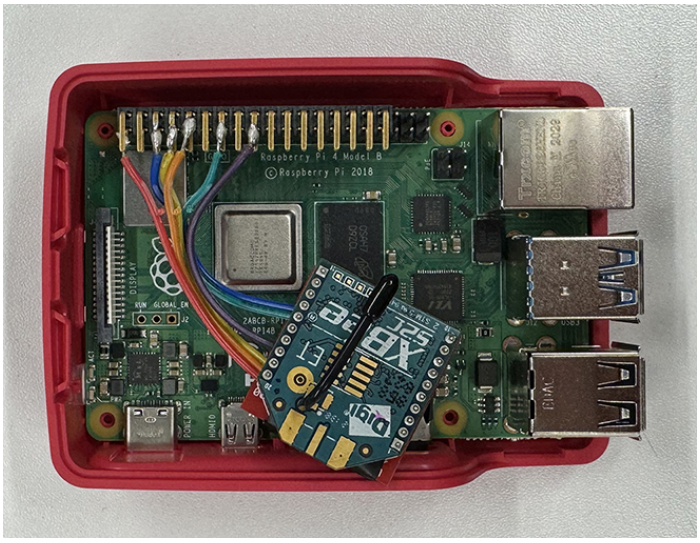
We created two form factors for the interface. The Pod and the Fridge-Magnet. The Pods were created using 3D printing and the Fridge-Magnets were made using a laser cutter. The files for both can be downloaded <LINK REDACTED FOR REVIEW>.

The Pod is designed to be placed on counter tops or coffee tables, in a similar fashion that we see home assistant devices. The LED display is angled toward the user, while the Epishine module is angled towards the light. There is a large activation button on the top.

The Fridge-magnet is designed to be placed on flat surfaces, such as refrigerators or kitchen cupboards. Both the LED display and Epishine module are facing outwards.

# Central Hub



**The central hub is a Raspberry Pi model 4B+ using the Raspberry Pi OS Lite. Within its housing, an XBee X2C module is interfaced to allow communication to the self-powered displays (See diagram for pinout connections).**

**An image of the OS with all required code libraries can be found <LINK REDACTED FOR REVIEW>**

**To create the central hub yourself, you will need:**

1. **A Raspberry Pi OS, we used the Lite version as the user interface was not required.**
2. **Install prerequisite libraries using their tutorials**
    a. **PYSerial - https://pypi.org/project/pyserial/**
    b. **Bluepy - https://github.com/IanHarvey/bluepy**
3. **Download the ToolKit code from <LINK REDACTED FOR REVIEW>**
4. **A list of all the MAC addresses of the sensors you are going to use.**
5. **(Optional) Add the code as a SYSTEMD service to run when the Pi boots. See: https://gist.github.com/emxsys/a507f3cad928e66f6410e7ac28e2990f. Or you can start the code manually if you'd prefer.**

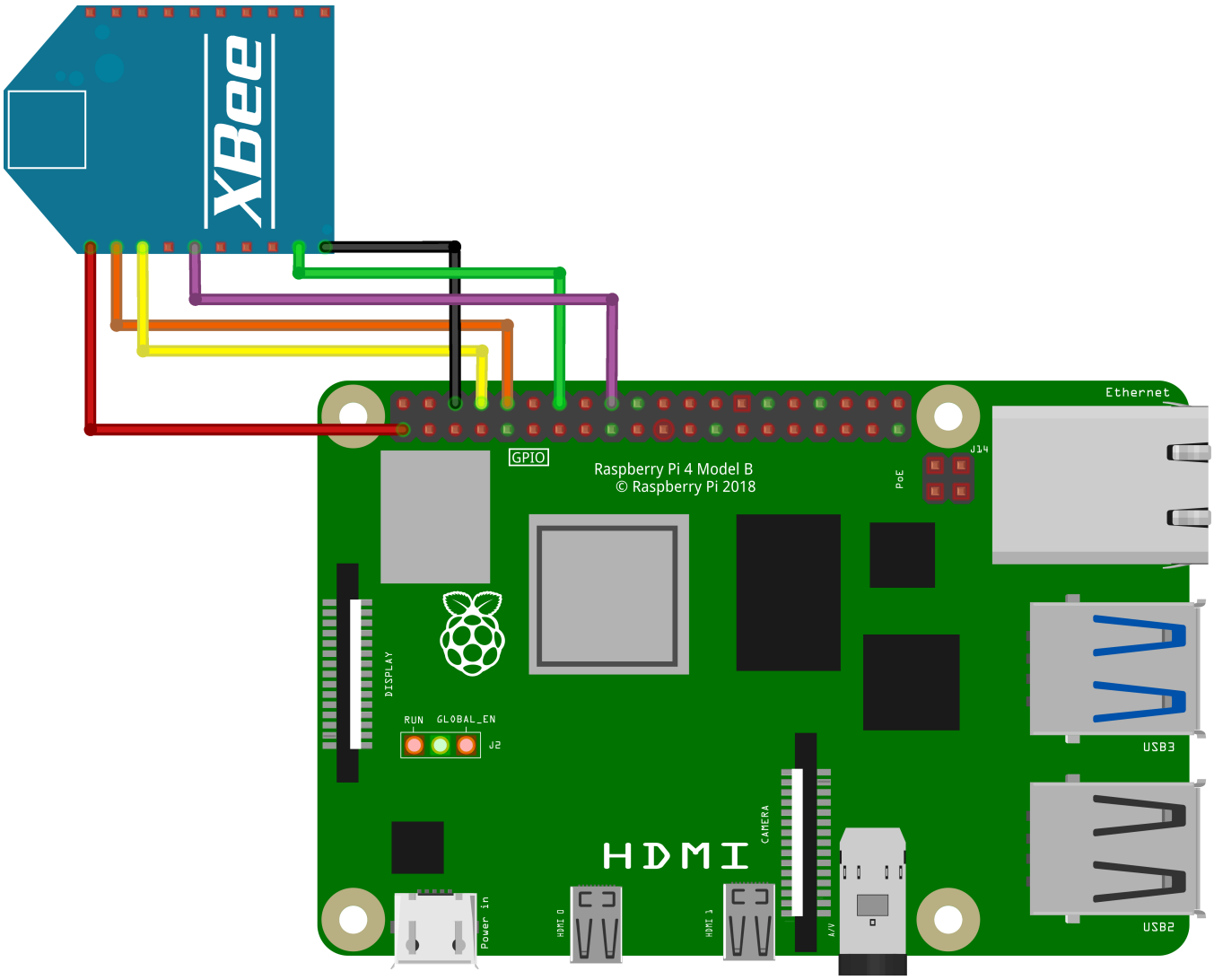
# Hub Configuration

**In lines 38-60 (snippet below), you will see the initialisation of the individual sensors. Each type is stored within an array, having an ID, a MAC address and an associated activity.**

**Here, you must add each of your sensors, including an ID, and MAC address and associate them with an activity by replacing the example sensors with your own. The snippet below shows two examples for each sensor type. More sensors can be added to each array if you wish.**

```
# [ID, MAC Address, Activity, Time of Ping, Accumulated Time, Session]
bookmarks = [[0, "00:a0:50:0e:18:19", cooking, 0, 0, 0],
             [1, "00:a0:50:0e:1c:15", chores, 0, 0, 0]
]


# [ID, MAC Address, Activity, Time of Ping, Accumulated Time, Session]
pvs = [[0, "00:a0:50:14:07:1f", hobbies, 0, 0, 0],
       [1, "00:a0:50:17:2c:2a", cooking, 0, 0, 0]
]



# [ID, MAC Address, Activity]
pirSensors = [[6, "06:05:04:03:02:01",  cooking],
              [10, "10:05:04:03:02:01", cooking]
]

# [ID, MAC Address, Activity]
clickers = [
    [0,  "e2:15:10:00:01:c3", hobbies],
    [1,  "e2:15:10:00:01:d2", chores]
]
```

# Diagram


XBee
GPIO
Raspberry Pi 4 Model B
© Raspberry Pi 2018
Ethernet
J14
PoE
DISPLAY
RUN
GLOBAL_EN
J2
HDMI
CAMERA
A/V
USB3
USB2
Power in
HDMI 0
HDMI 1

# Data

**All the sensor data is saved internally on the Raspberry Pi in CSV format. The data is saved by sensor type, Activity and by day, to allow easier processing. Each data file is stored in its subfolder, and a new CSV is created each day. The file structure is as follows:**

```
-Root
      - activityZero
      - activityOne
      - activityTwo
      - bookmarks
      - clickers
      - pirs
      - pvs
      - pods
      - dayByDay
```

**Each sensor activation is saved into its corresponding CSV, where the data includes the sensor type, its ID and the timestamp of the activation. As seen below:**

```
2,6,11-09-2023,08:18:21
1,15,11-09-2023,11:24:24
1,15,11-09-2023,11:24:29
1,22,11-09-2023,11:24:44
```

**To make the data more readable, it will need to be processed using <LINK REDACTED FOR REVIEW>, which adds meaning to the numbers. As Below:**

```
23-08-2023, 08:21:59, Utensil Drawer
23-08-2023, 08:25:07, Kitchen
23-08-2023, 08:26:16, Toybox
23-08-2023, 08:26:24, Kitchen
```

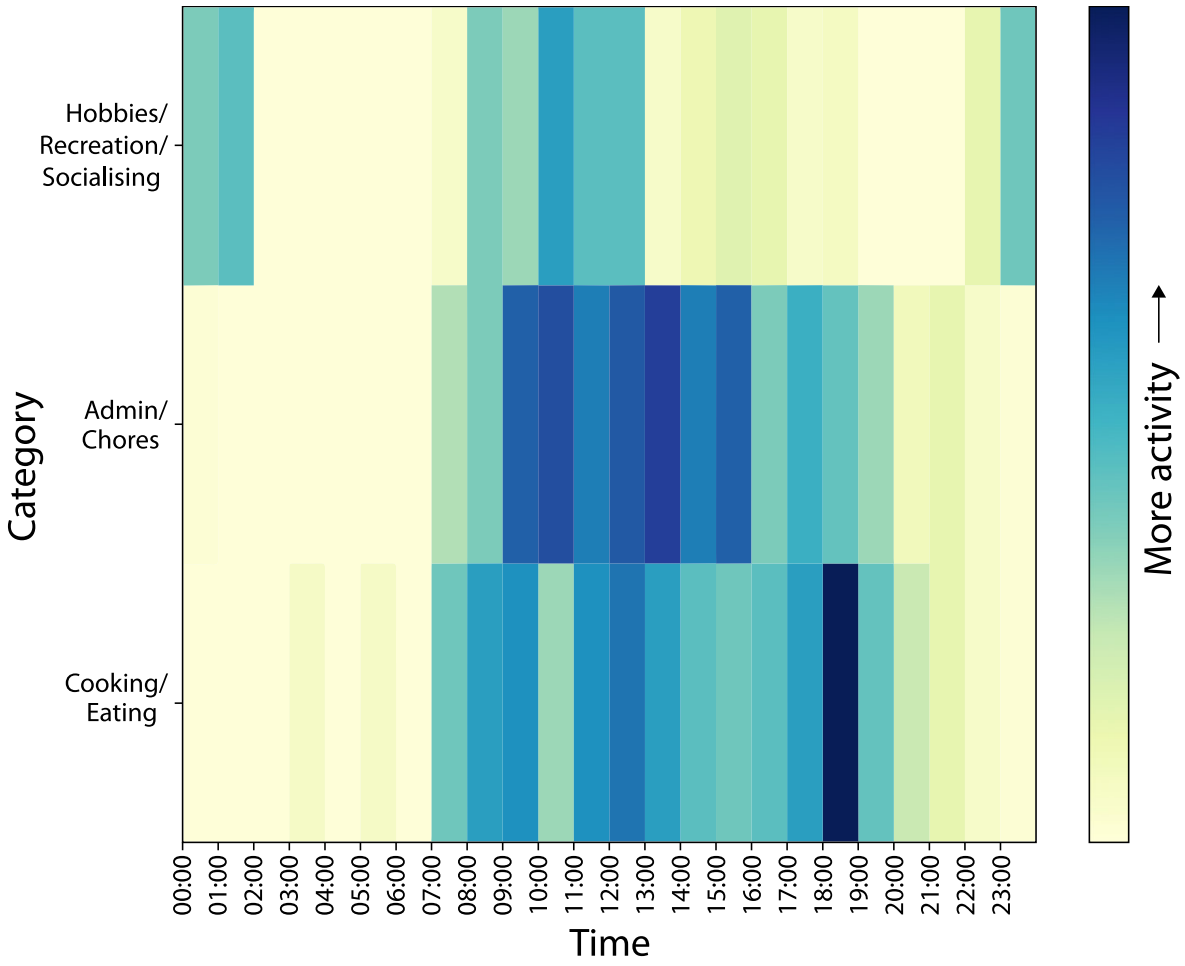
**This data is then able to be processed further to create heatmaps of activity, or even interpreted by Large Language Models to create stories and diaries.**

# Heatmaps

**Using the processed data with the Python scripts located <REDACTED FOR REVIEW>, Heatmaps can be generated. Simply copy the processed data into the code.**

**There are several scripts to choose from.**
**-- Activities averaged over a number of days specified**
**-- Activity data separated each day**

**Below is an example heatmap of activity data averaged over several days.**



# ChatGPT

**Using the processed data and Large Language Models, we can generate stories and diaries of the sensor activities. In our case, we used ChatGPT 4.0.**

**Firstly, give the LLM some context:**

```
"I am going to give you timestamped data from household sensors"

"The house is a three-story victorian townhouse belonging to a married couple Jane and John Doe"
```

**Then give it the sensor data (here is some example data)**

```
"21-08-2023, 09:57:57, Kitchen Light",
"21-08-2023, 10:00:27, Cutlery Cupboard",
"21-08-2023, 16:43:27, Cutlery Cupboard",
"21-08-2023, 17:06:51, Cutlery Cupboard",
"21-08-2023, 18:24:18, Pan Draw",
"21-08-2023, 23:07:14, Kitchen Light"
```

**Finally, ask the LLM to output the data in a format you would like. Such as:**

```
"Tell me a story about Jane and John on this day"
```

**The output from those inputs to an LLM can then look a little like this:**

```
.......................

As the clock struck 9:57 am, Jane found herself in the kitchen, brewing a pot of aromatic coffee. The soft hum of the coffee machine harmonized with the distant sounds of morning activity outside. John, with a newspaper in hand, joined her in the kitchen. The two shared a brief but meaningful moment over the morning ritual, basking in the quiet joy of each other's company.

.......................
```

**Extract and promts given to ChatGPT to recreate our diaries can be seen <REDACTED FOR REVIEW>**