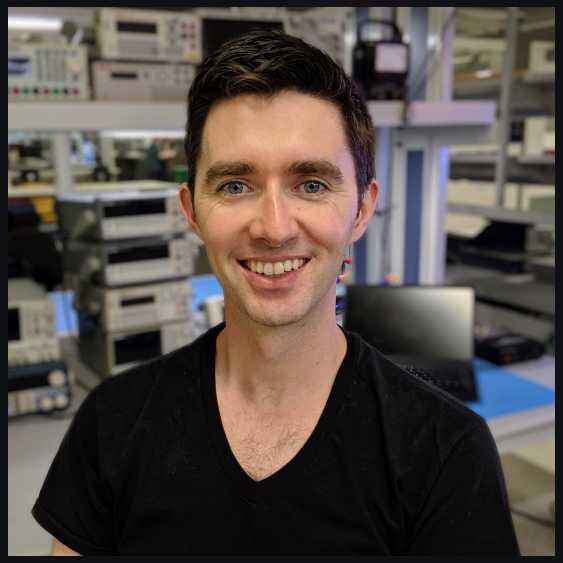
What Chip Shortage?

How We Use Zephyr for Truly Modular Hardware

Chris Gammell & Mike Szczys

About Chris



Chris Gammell

@chris_gammell on twitter

- Developer Relations Lead at Golioth
- 20 years hardware development
- Worked at places like ABB, Samsung, Supplyframe, Keithley, Hologram
- Most recently was a hardware and firmware consultant.

Golioth

About Mike

Mike Szczys

@szczys on twitter

- Developer Relations Engineer at Golioth
- 15 years of firmware experience
- Previously: Editor in Chief of Hackaday

Golioth

Golioth is a device management platform

(and more)

Golioth

Who is this talk for?

Zephyr Beginners

Golioth

Zephyr Veterans looking for help during the chip shortage

Golioth

Anyone interested in hardening their supply chain by introducing *Swappability*

Golioth

Swappability (interoperability):

- The ability to trade one temperature sensor for another temperature sensor.
- The ability to trade a temperature sensor for LIDAR.
- The ability to trade network interfaces to (cellular → WiFi) for easier testing.
- The ability to trade microcontrollers within a family or even between manufacturers.

Golioth

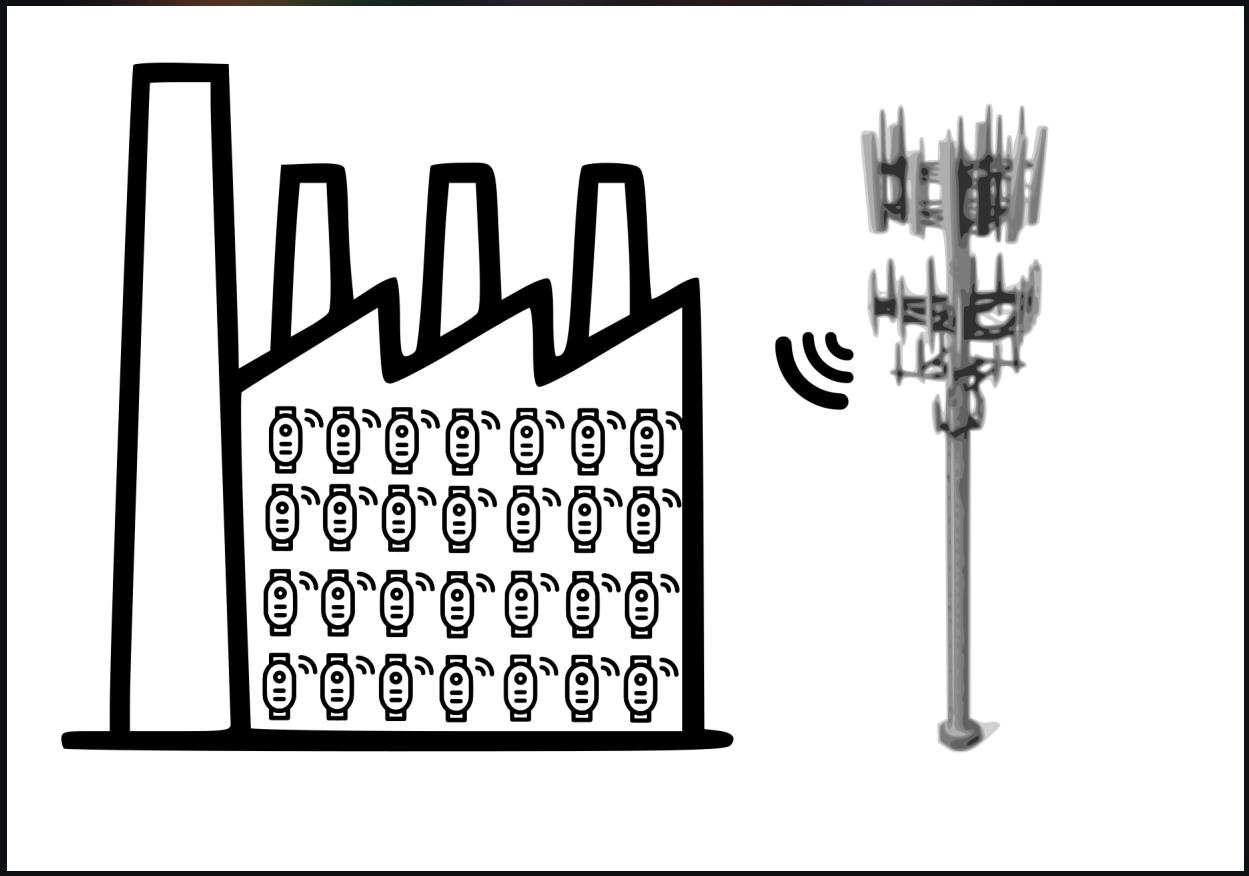
Why bother with flexibility?

Golioth

Part shortages

Golioth

On-the-ground issues



Golioth

Different customers, same design

Golioth

What is the Aludel Hardware?

Golioth

Standard Enclosure

Easily purchased from multiple vendors, widely available

Golioth

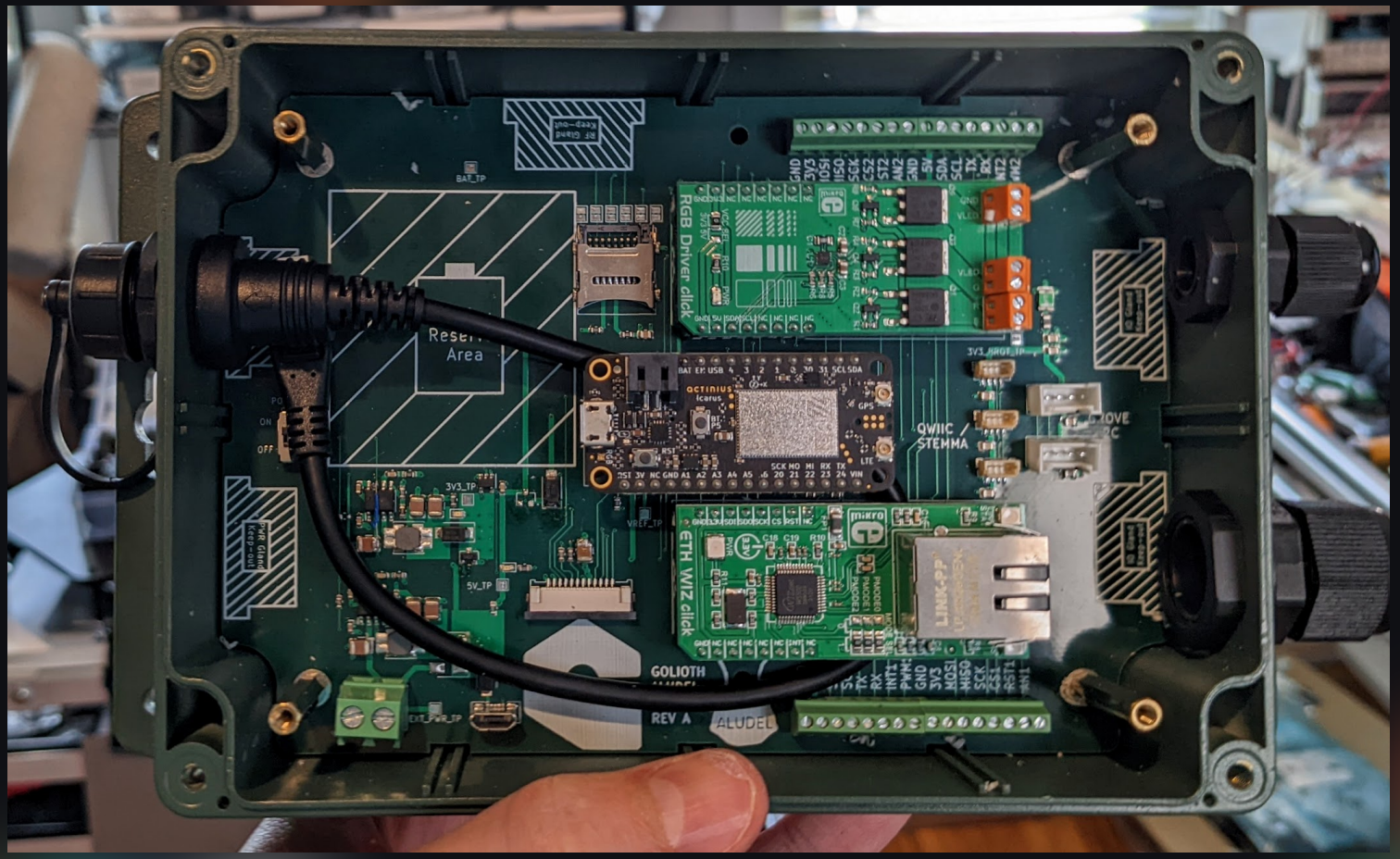
Swappable faceplates

Each customer gets a tailor-made hardware look

Golioth

Internal Standards

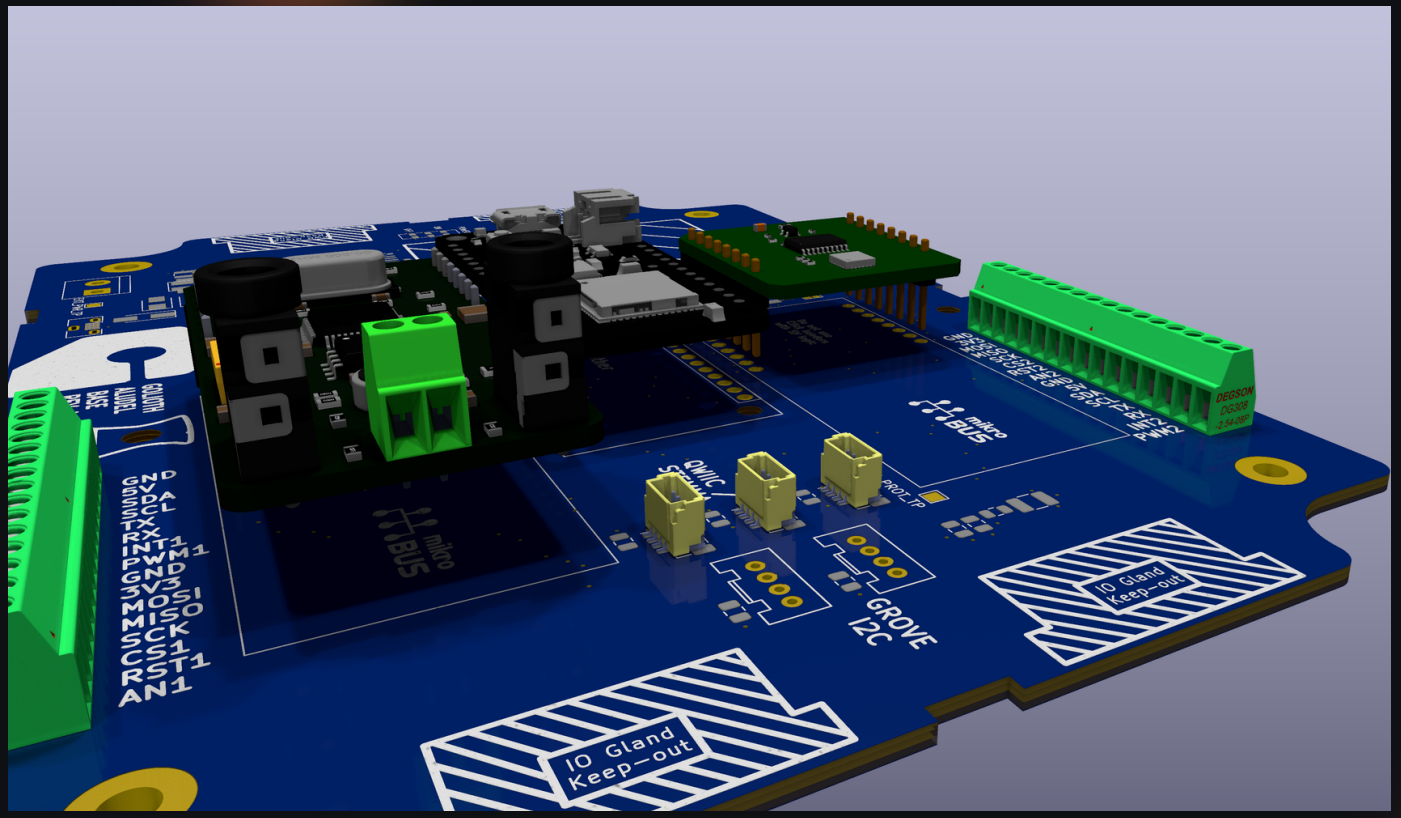
Off-the-shelf modules made interchange possible



Golioth

Terminals Blocks

Standard industrial option for connecting components



Golioth

Glands, Glands, Glands

Industrial IoT needs to think about water/dust ingress

Golioth

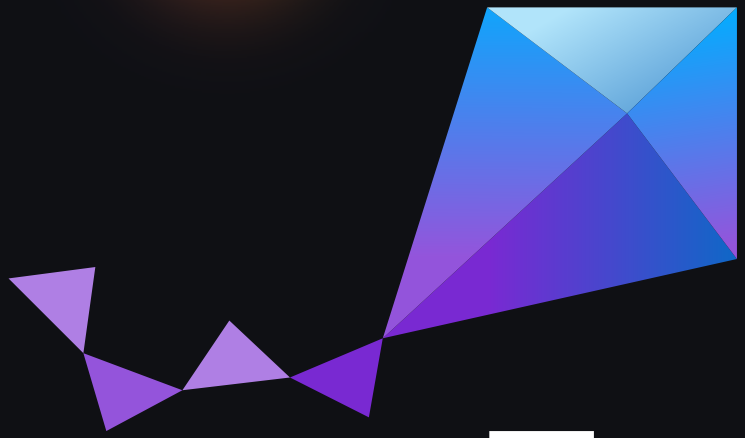
Plans, Plans, Plans

Future plans include more power options and fine control

Golioth

What is the Aludel Software?

Golioth



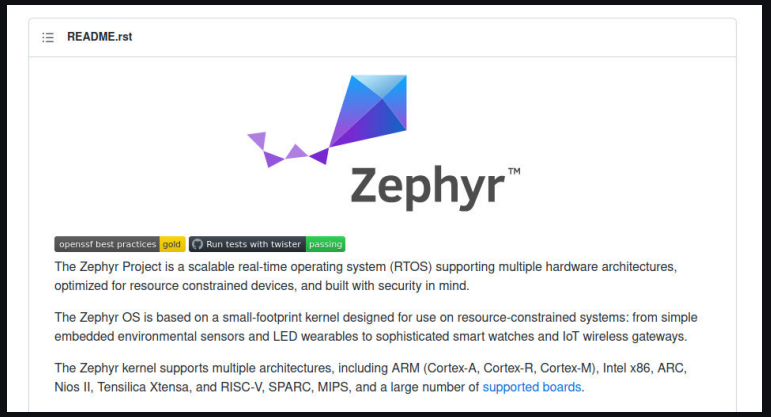
Zephyr™

Golioth

Flavors of Zephyr

Zephyr

(vanilla)



nRF Connect

(still Zephyr, but it tastes a bit like Nordic)

Golioth

Device Tree

boards > adafruit_feather_stm32f405.overlay > ...

```
&i2c1 {
    bme280@76 {
        compatible = "bosch,bme280";
        reg = <0x76>;
        label = "BME280_I2C";
    };
};

&spi2 {
    compatible = "st,stm32-spi";
    status = "okay";
    cs-gpios = <&gpioc 5 GPIO_ACTIVE_LOW>, <&gpiob 9 GPIO_ACTIVE_LOW>, <&gpioc 7 GPIO_ACTIVE_LOW>;
    test_spi_w5500: w5500@0 {
        compatible = "wiznet,w5500";
        label = "w5500";
        reg = <0x0>;
        spi-max-frequency = <10000000>;
        int-gpios = <&gpioc 4 GPIO_ACTIVE_LOW>;
        reset-gpios = <&gpioa 6 GPIO_ACTIVE_LOW>;
    };
};

/ {
    aliases {
        aludeli2c = &i2c1;
        pca9539int = &interrupt_pin0;
    };
    gpio_keys {
        compatible = "gpio-keys";
        interrupt_pin0: int_pin_0 {
            gpios = < &gpiob 8 GPIO_ACTIVE_LOW >;
            label = "Port Expander Interrupt Pin";
        };
    };
};
```

Everything kind of just works, but...

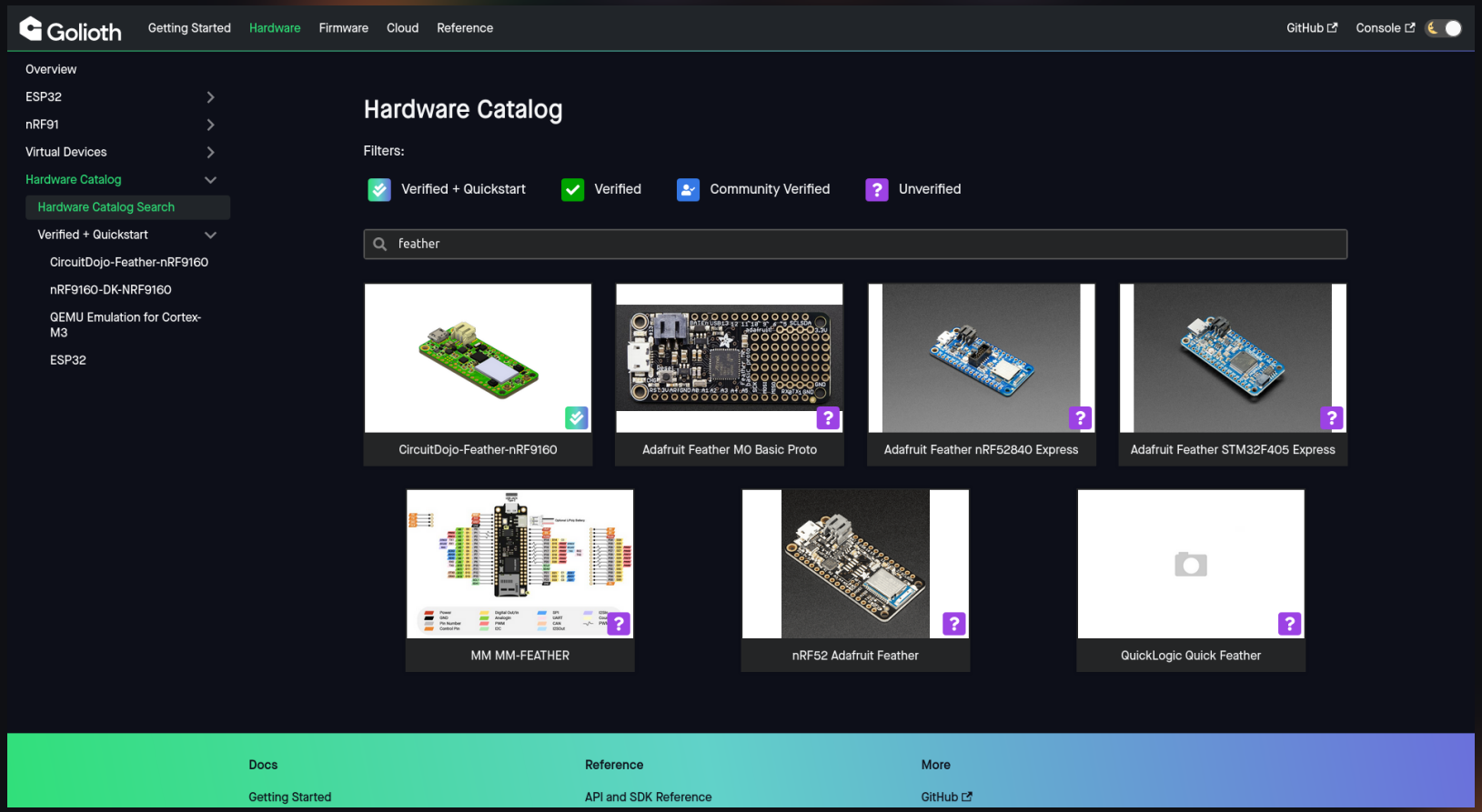
```
if (IS_ENABLED(CONFIG_GOLIOTH_SAMPLE_WIFI)) {
    LOG_INF("Connecting to WiFi");
    wifi_connect();
}

#if IS_ENABLED(CONFIG_NET_L2_ETHERNET)
if (!IS_ENABLED(CONFIG_WIFI_ESP32))
{
    LOG_INF("Connecting to Ethernet");
    struct net_if *iface;
    iface = net_if_get_default();
    net_dhcpv4_start(iface);
}
#endif
```

Some conditionals for specialized code (Ethernet DHCP, ESP32 as AT modem)

Golioth

Zephyr support for 'Feather'



Golioth

Overlay and Conf Files

Chosen on the board level

- The Feather footprint has good support
  - Circuit Dojo nRF9160 (Cellular)
  - Adafruit Feather nRF52840 (Ethernet)
  - Sparkfun Thing+ STM32f405 (Ethernet)
- Others have related board support:
  - Adafruit Feather ESP32 (WiFi)

Golioth

<board>.conf files

boards > adafruit_feather_nrf52840.conf

```
CONFIG_SPI=y
CONFIG_NET_L2_ETHERNET=y
CONFIG_ETH_W5500=y
CONFIG_ETH_W5500_TIMEOUT=1000
CONFIG_NET_DHCPV4=y
CONFIG_NET_MGMT=y

CONFIG_GPIO=y
CONFIG_I2C=y

CONFIG_UART_CONSOLE=n
CONFIG_RTT_CONSOLE=y
CONFIG_USE_SEGGER_RTT=y
```

<board>.overlay

boards > circuitdojo_feather_nrf9160_ns.overlay

```
1  &i2c1 {
2      bme280@76 {
3          compatible = "bosch,bme280";
4          reg = <0x76>;
5          label = "BME280_I2C";
6      };
7  };
8
```

Golioth

Overlay files can remap for board variations

```
15  &i2c0 {
16      status = "okay";
17      clock-frequency = <I2C_BITRATE_STANDARD>;
18      sda-pin = <23>;
19      scl-pin = <22>;
20
21      bme280@76 {
22          compatible = "bosch,bme280";
23          reg = <0x76>;
24          label = "BME280_I2C";
25      };
26  };
```

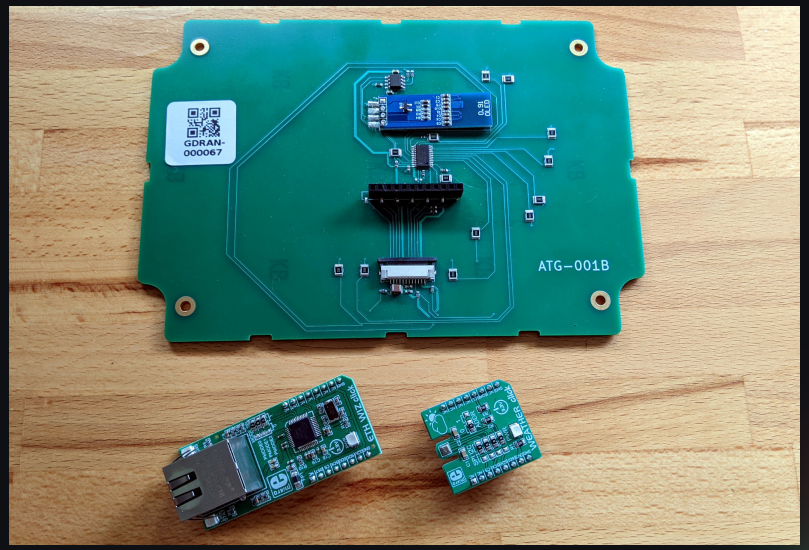
Remap pins

Or you can define your own DTS files for custom boards

Golioth

Sensors and Add-Ons

- Examples:
  - BME380 Weather sensor (built-in)
  - W5500 Ethernet module (built-in)
  - AW9523 i2c port expander (direct)



Golioth

Sensors need to be added to the overlay file for every microcontroller variant

```
&i2c0 {
    apds9960: apds9960@39 {
        compatible = "avago,apds9960";
        status = "okay";
        reg = <0x39>;
        label = "APDS9960";
        int-gpios = <&gpio0 26 (GPIO_ACTIVE_LOW | GPIO_PULL_UP)>;
    };
};
```

```
&i2c1 {
    apds9960: apds9960@39 {
        compatible = "avago,apds9960";
        status = "okay";
        reg = <0x39>;
        label = "APDS9960";
        int-gpios = <&gpio0 13 (GPIO_ACTIVE_LOW | GPIO_PULL_UP)>;
    };
};
```

```
&i2c1 {
    apds9960: apds9960@39 {
        compatible = "avago,apds9960";
        status = "okay";
        reg = <0x39>;
        label = "APDS9960";
        int-gpios = <&gpioa 4 (GPIO_ACTIVE_LOW | GPIO_PULL_UP)>;
    };
};
```

```
&i2c0 {
    apds9960: apds9960@39 {
        compatible = "avago,apds9960";
        status = "okay";
        reg = <0x39>;
        label = "APDS9960";
        int-gpios = <&gpio0 4 (GPIO_ACTIVE_LOW | GPIO_PULL_UP)>;
    };
};
```

Golioth

```
arduino_header: connector {
        compatible = "arduino-header-r3";
        #gpio-cells = <2>;
        gpio-map-mask = <0xffffffff 0xffffffc0>;
        gpio-map-pass-thru = <0 0x3f>;
        gpio-map = <0 0 &gpioa 0 0>,    /* A0 */
                   <1 0 &gpioa 1 0>,    /* A1 */
                   <2 0 &gpioa 4 0>,    /* A2 */
                   <3 0 &gpiob 0 0>,    /* A3 */
                   <4 0 &gpioc 1 0>,    /* A4 */
                   <5 0 &gpioc 0 0>,    /* A5 */
                   <6 0 &gpioa 3 0>,    /* D0 */
                   <7 0 &gpioa 2 0>,    /* D1 */
                   <8 0 &gpioa 10 0>,   /* D2 */
                   <9 0 &gpiob 3 0>,    /* D3 */
                   <10 0 &gpiob 5 0>,   /* D4 */
                   <11 0 &gpiob 4 0>,   /* D5 */
                   <12 0 &gpiob 10 0>,  /* D6 */
                   <13 0 &gpioa 8 0>,   /* D7 */
                   <14 0 &gpioa 9 0>,   /* D8 */
                   <15 0 &gpioc 7 0>,   /* D9 */
                   <16 0 &gpiob 6 0>,   /* D10 */
                   <17 0 &gpioa 7 0>,   /* D11 */
                   <18 0 &gpioa 6 0>,   /* D12 */
                   <19 0 &gpioa 5 0>,   /* D13 */
                   <20 0 &gpiob 9 0>,   /* D14 */
                   <21 0 &gpiob 8 0>;   /* D15 */
};
```

Zephyr Shields as a workaround

- Shield file calls out the &arduino_header abstraction
- Caveat: the header must be mapped in the board.dts file (can't be mapped in an overlay)

```
&i2c0 {
    apds9960: apds9960@39 {
        compatible = "avago,apds9960";
        status = "okay";
        reg = <0x39>;
        label = "APDS9960";
        int-gpios = <&arduino_header 0 (GPIO_ACTIVE_LOW | GPIO_PULL_UP)>;
    };
};
```

```
# From the root of the zephyr repository
west build -b None your_app -- -DSHIELD="x_nucleo_idb05a1 x_nucleo_iks01a1"
```

Golioth

Don't forget to Document

nrf52840 feather with w5500 Ethernet

```
west build -b adafruit_feather_nrf52840 . -D OVERLAY_CONFIG=credentials.conf -p
```

1. Connect via the J-Link programmer
2. `west flash`
3. View the serial output using JLinkRTTViewer

stm32f405 feather with w5500 Ethernet

```
west build -b adafruit_feather_stm32f405 . -D OVERLAY_CONFIG=credentials.conf -p
```

This one cannot be programmed when plugged into the Aludel.

1. Remove the feather from Aludel
2. Plug into USB, hold the `BT0` button and tap the reset button
3. `west flash`

Circuit Dojo Feather (nrf9160)

Make sure to use the Golioth **NCS** SDK (the Nordic "flavor") to build for this board.

```
west build -b circuitdojo_feather_nrf9160_ns . -D OVERLAY_CONFIG=credentials.conf -p
```

Golioth

Case Study 1:

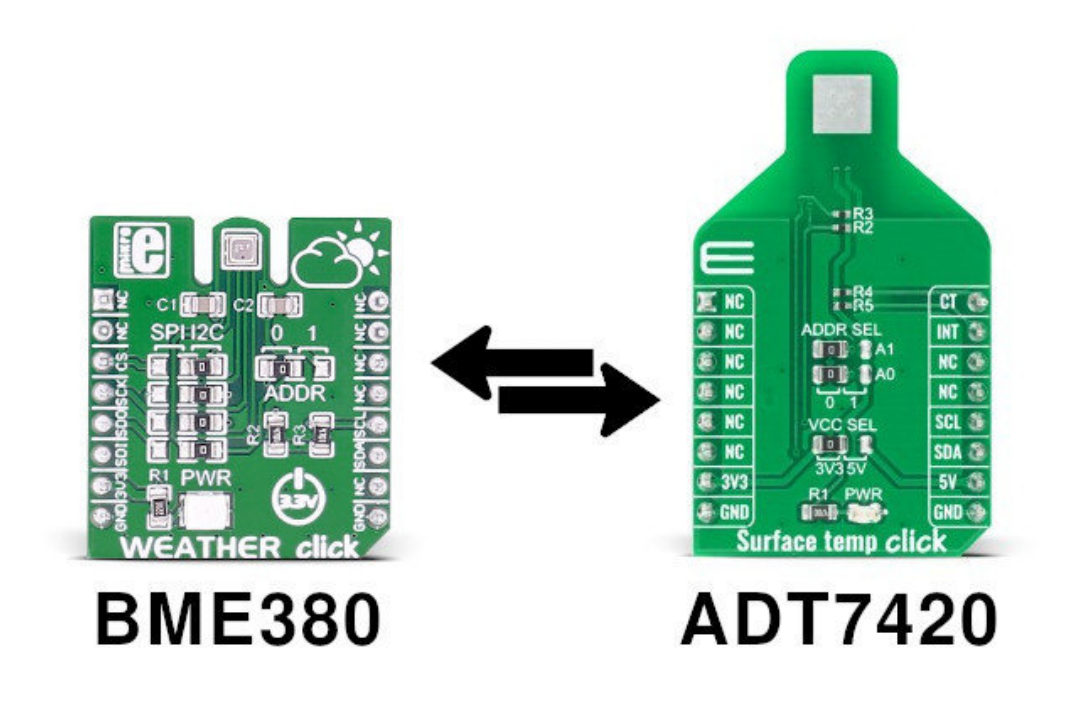
Your Sensor Is Out Of Stock

Golioth

Prototype

Golioth

Switch out the click board



Golioth

Switch out a bit of the code

- <yourboard>.overlay - if there are new pins
- prj.conf - if there are new settings
- main.c - or wherever calls the sensor

Apply Blockchain AI algorithm in order to generate NFT and apply using 5G programming methods

Golioth

Nah, we're done

(also, people talk like that?)

Golioth

Assumptions

- You checked that you can actually buy the new sensor
- Your chosen replacement is in-tree and has the same output variable like "ambient temperature"

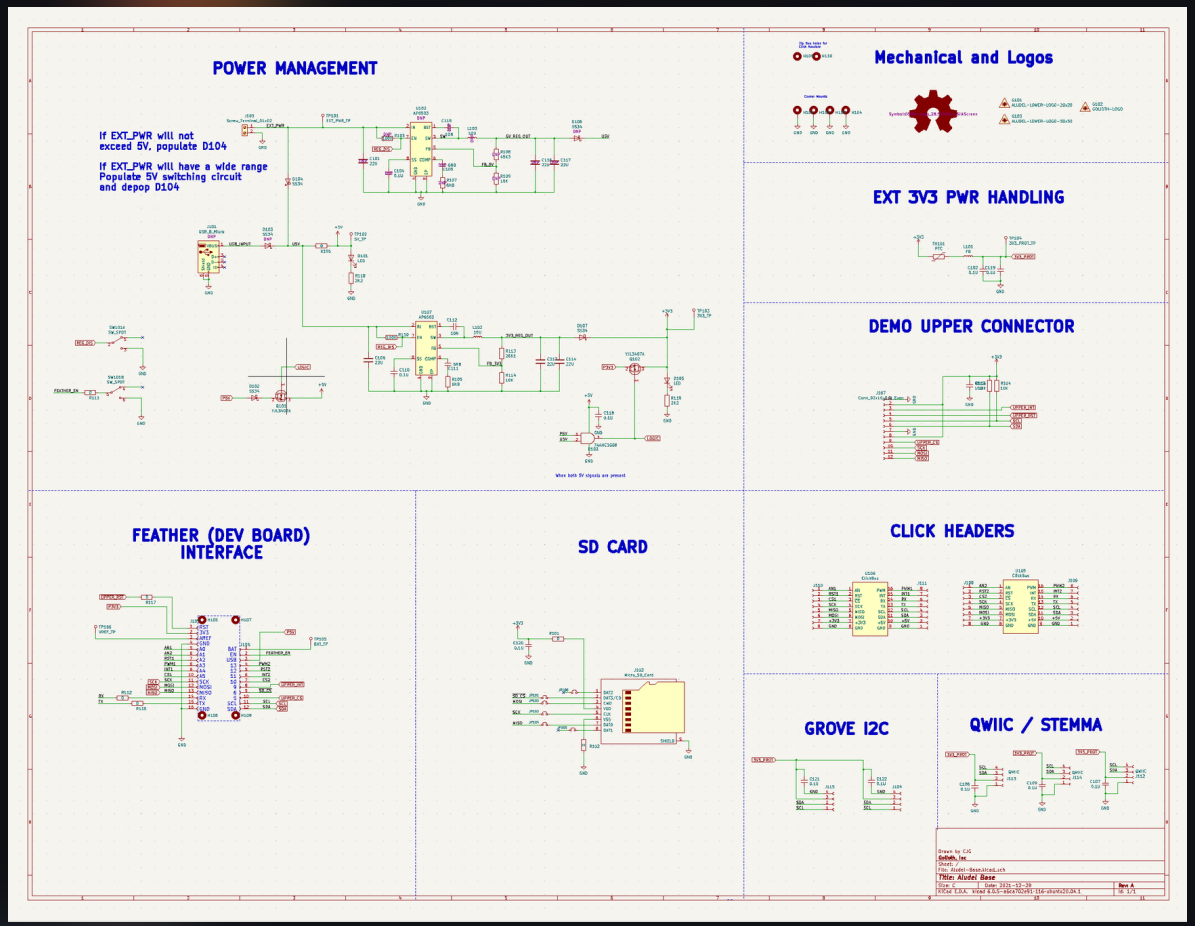
Golioth

Taking it to production

Golioth

Design in the new sensor

(it's really really in-stock...right?)



Golioth

Problem: Manage different versions of FW with devices already in the field

Golioth

Organize your devices using a device management platform

- At Golioth we use a flat tagging system to organize different hardware versions.
- Great for targeting firmware updates.
- Understand what your device contains at a glance.

Golioth

Case Study 2:

Your Micro Is Out Of Stock

SparkFun Thing Plus - STM32

DEV-17712 ROHS

$32.50

Volume sales pricing

We do not currently have an estimate of when this product will be back in stock. **Notify Me**

Note: If this item is available for backorder it is subject to price changes at any

guarantee time frame for shipping or availability.

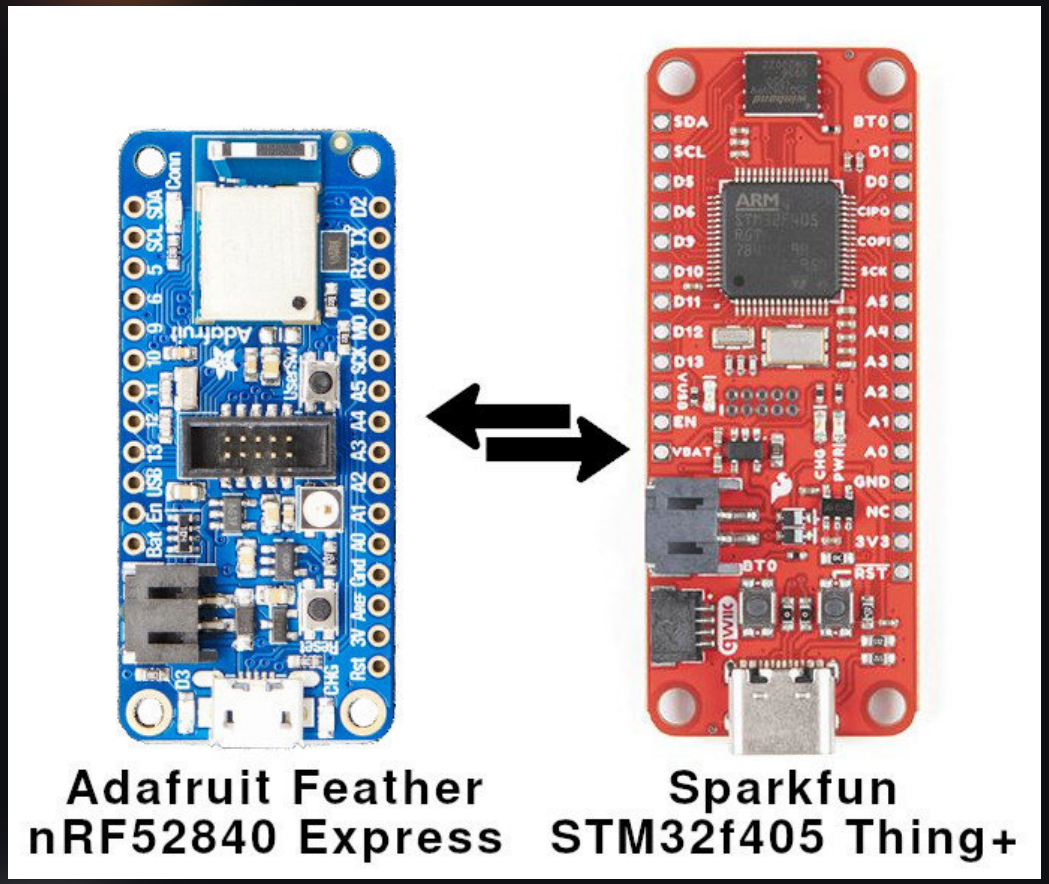
\- 1 +

BACKORDER

Quantity discounts available

Golioth

Switch out the Feather board



Golioth

If no Feather exists...

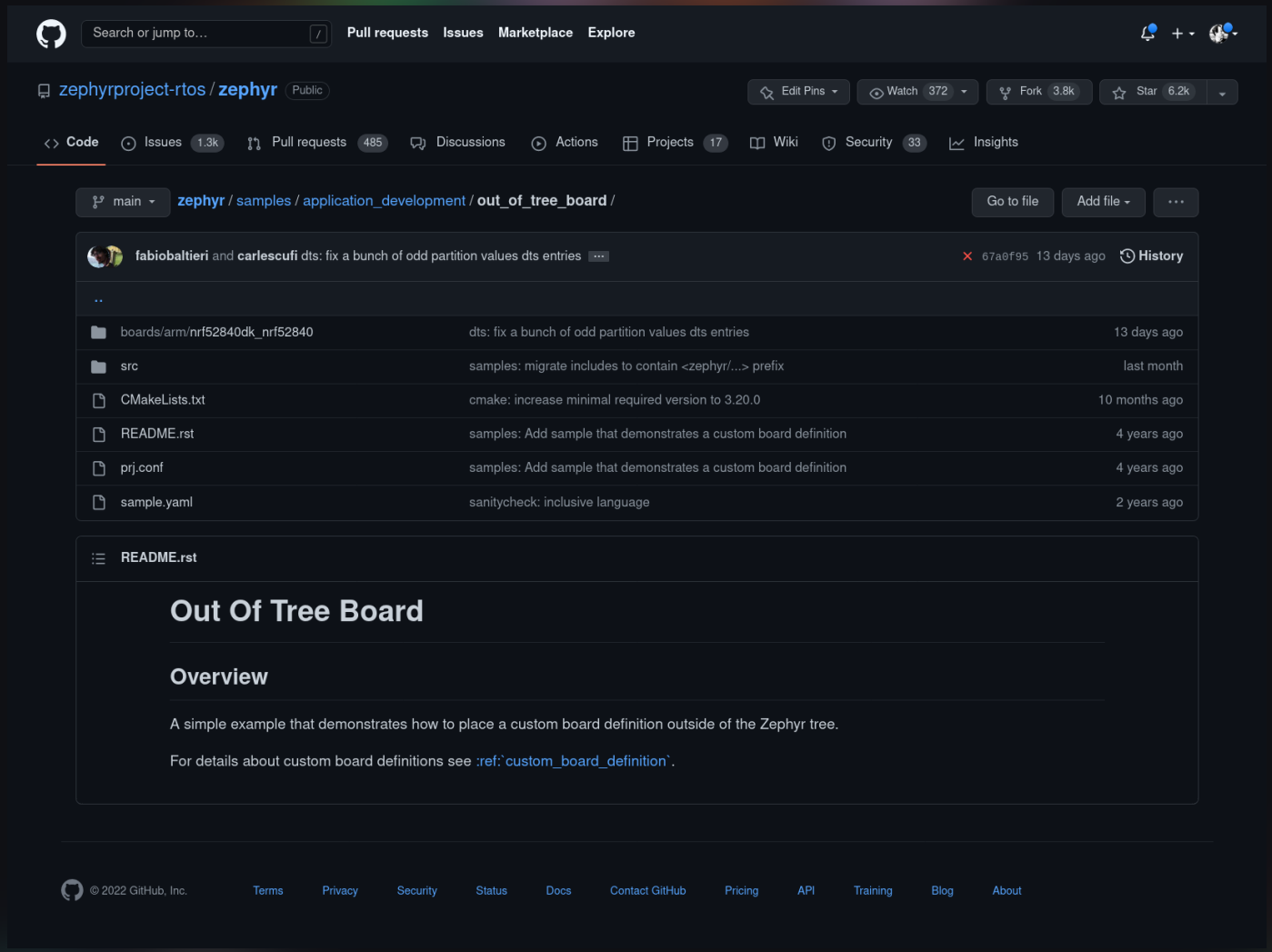
make one!

Switch out more of the code

If a similar board exists in the Zephyr tree:

- <yourboard>.conf - apply new board settings
- <yourboard>.overlay - map the pin functions
- prj.conf and main.c should not need to change

zephyr/samples/application_development/out_of_tree_board



Golioth

"But I'm not using the Aludel"

or

I'm making a single board product

Golioth

Most devices bound for production will be cost, space, and power optimized.

Golioth

Each of the interfaces on the Aludel represents a block on a finished product

Golioth

Your schematic/layout blocks can approximate these systems

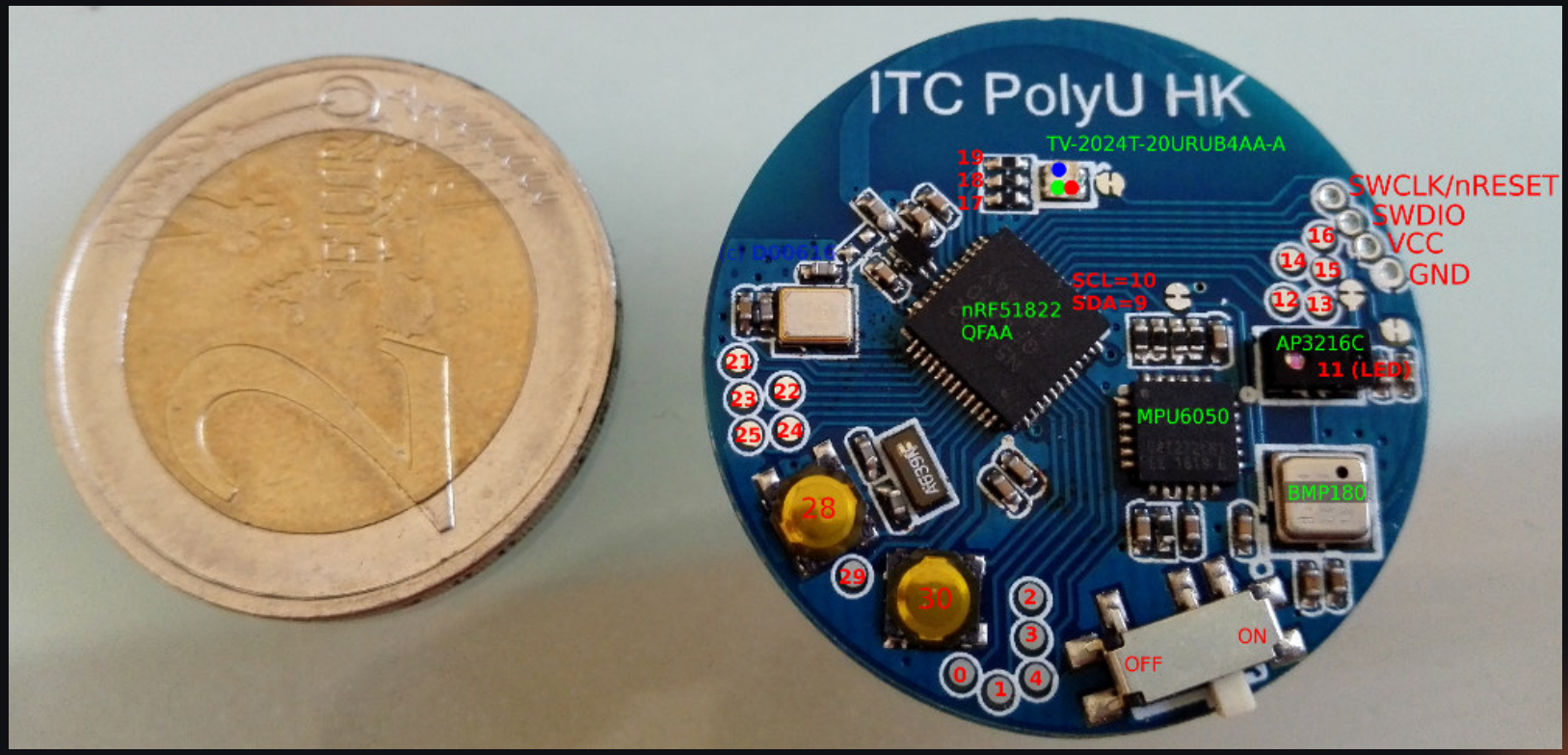


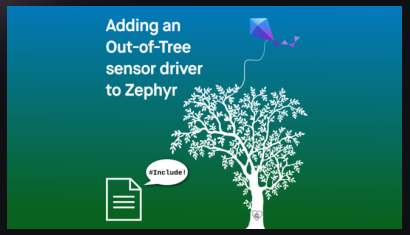
Image Source: https://bit.ly/3mkucGR

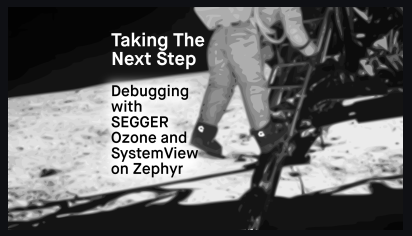
Golioth

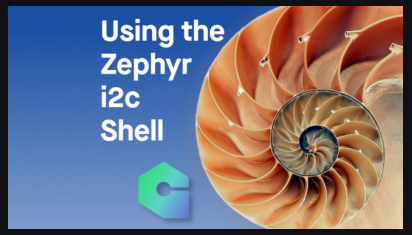
What have we learned?

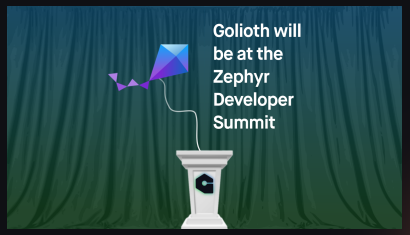
- Plan for the worst by designing in flexibility and modularity.
- Balance flexibility and modularity with the reality of your design constraints.
- Zephyr's abstracted interfaces allows HW and FW engineers to react to problems in the supply chain.

We write about Zephyr a LOT









Golioth

Find us online

Golioth.io

@golioth_iot

Golioth.io/Discord

Golioth