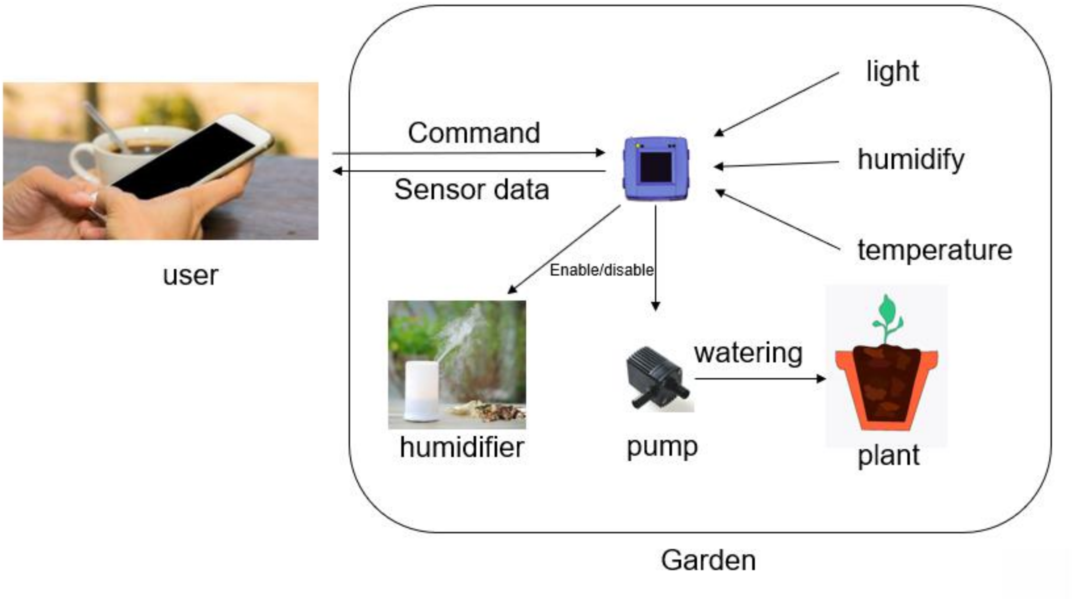
Rapid IoT Studio project example

Smart Botanic Garden

1 Project introduction

An automatic botanic garden can take care of your special plant (e.g. tropical plant); The humidity, temperature, illumination intensity can be captured then send to your phone by BLE; User can monitor if the environment is good for plant;
When the external pump and humidifier are added, user can water the plant/add the humidity remotely.



Due to the time is limited, some part of the garden is finished. Lots more functions can be added into the system, to make the garden smarter.

2 Looking forward:

2.1 A soil hygrometer can be added to monitor if the soil is dry, user can know when he should water the plant

2.2 Add WIFI module, so that user can take care of the garden not at home

3 Project video

3.1 Link:

- Please refer to the attachment:
  - Rapid IoT Studio project example Smart Botanic Garden.wmv

4 Preparation

4.1 Rapid IoT kit

- https://www.nxp.com/support/developer-resources/rapid-prototyping/nxp-rapid-iot-prototyping-kit:IOT-PROTOTYPING

4.2 Docking station to expend the pins for Rapid IoT Kit

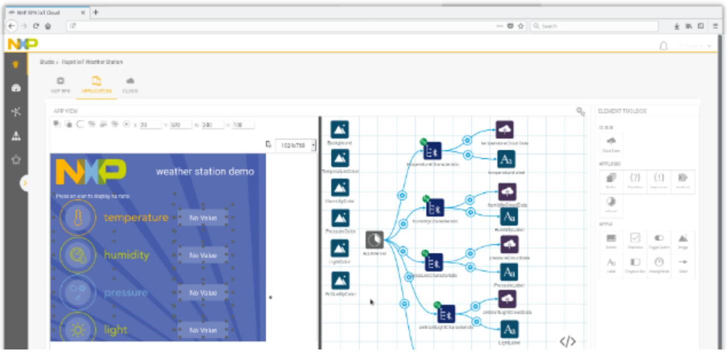
- https://www.mikroe.com/hexiwear-docking-station

4.3 Your Phone

4.4 Rapid IoT online IDE: Atmosphere, we should register a user account on it, most of the design tasks are based on this IDE

- https://rapid-iot-studio.nxp.com/

Mobile application view of the NXP Rapid IoT Studio online IDE.



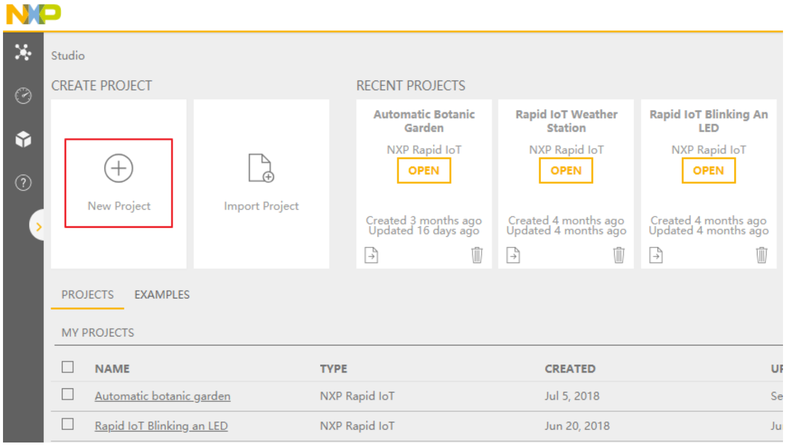
4.5 External devices

- Pump
- Humidifier

5 Design details

5.1 Design for KIT

- Firstly, we should login the online IDE account then create a new project for your design



Studio

CREATE PROJECT

New Project

Import Project

RECENT PROJECTS

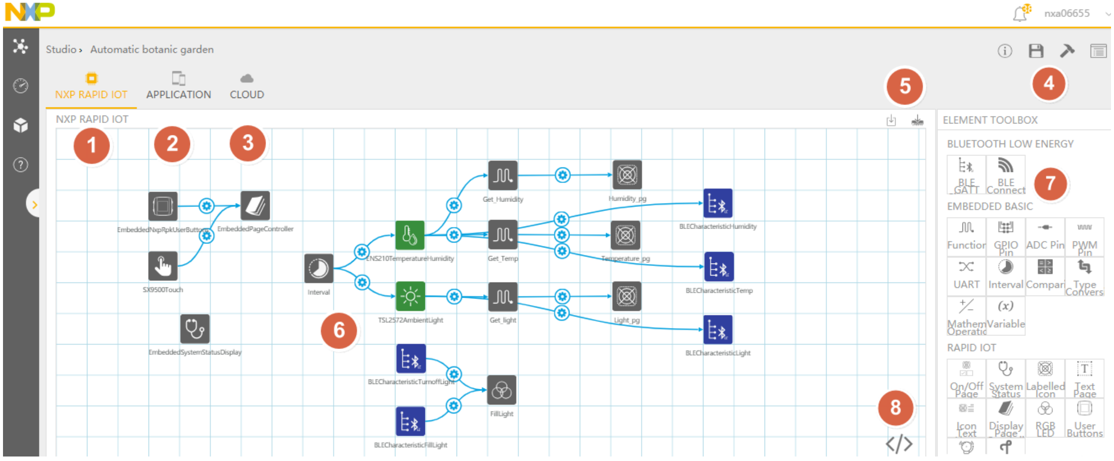
Automatic Botanic Garden	Rapid IoT Weather Station	Rapid IoT Blinking An LED
NXP Rapid IoT	NXP Rapid IoT	NXP Rapid IoT
OPEN	OPEN	OPEN
Created 3 months ago Updated 16 days ago	Created 4 months ago Updated 4 months ago	Created 4 months ago Updated 4 months ago

PROJECTS EXAMPLES

MY PROJECTS

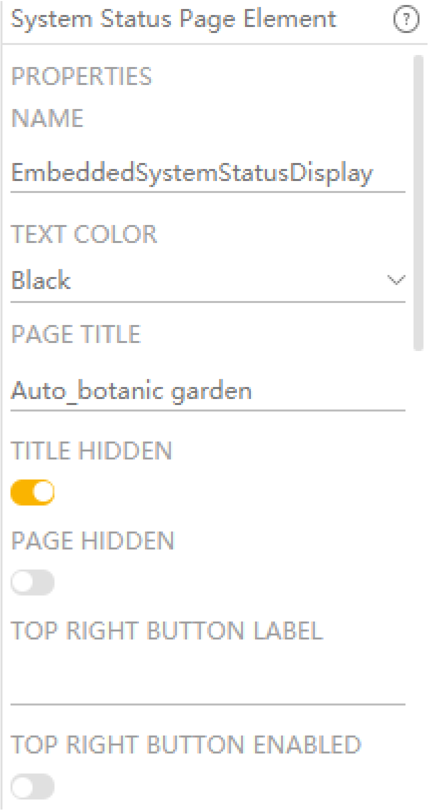
□	NAME	TYPE	CREATED	UI
□	Automatic botanic garden	NXP Rapid IoT	Jul 5, 2018	Se
□	Rapid IoT Blinking an LED	NXP Rapid IoT	Jun 20, 2018	Ju

- Then you can start your design



- ➢ Label1, design page for Kit
- ➢ Label2, design page for App
- ➢ Label3, design page for cloud

  - Label4, info, save, build, project button
  - Label5, download and firmware button
  - Label6, design grids
  - Label7, function block for your chosen
  - Label8, show the code for this project
- Let’s start to design the Kit part, here I mainly to introduce the process for design, some detail things will be dropped because they are easy to handle.
- I want to show something at the Kit, so let’s learn the rule of the display on the Kit with below link, according to the display rule, I can arrange the page easily
  - https://developer.atmosphereiot.com/documents/guides/nxpdisplayelementsguide.html
- Then I added a block ‘EmbeddedSystemStatusDisplay’ to show the main page and put the project name on the screen, by modify the element setting ‘page title’ as below



System Status Page Element ?

PROPERTIES

NAME

EmbeddedSystemStatusDisplay

TEXT COLOR

Black

PAGE TITLE

Auto_botanic garden

TITLE HIDDEN

●

PAGE HIDDEN

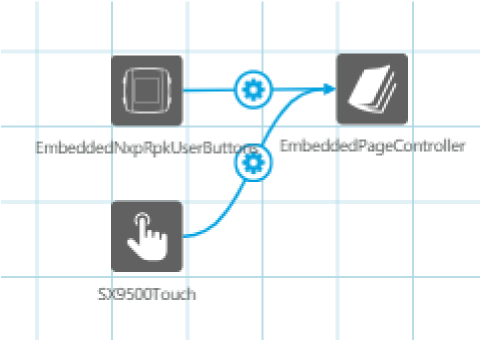
○

TOP RIGHT BUTTON LABEL

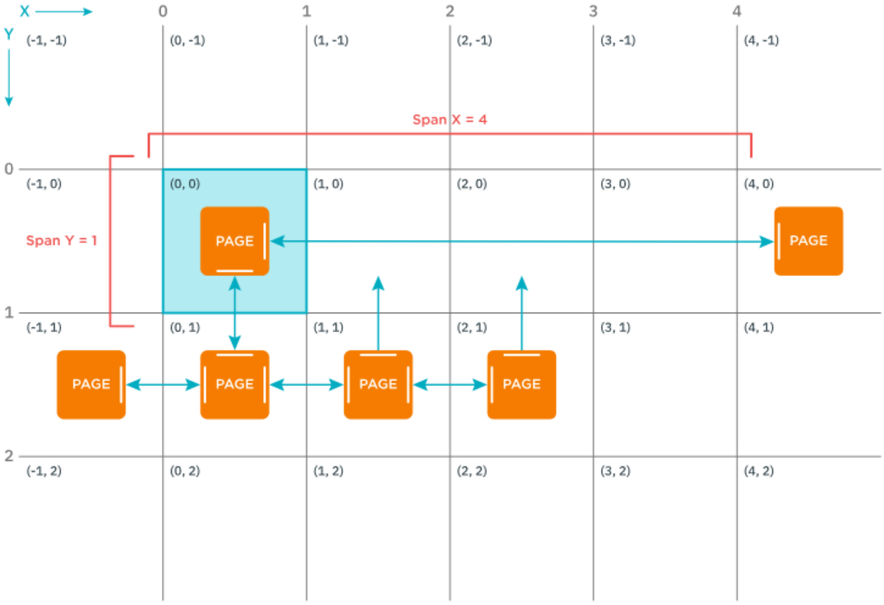
TOP RIGHT BUTTON ENABLED

○

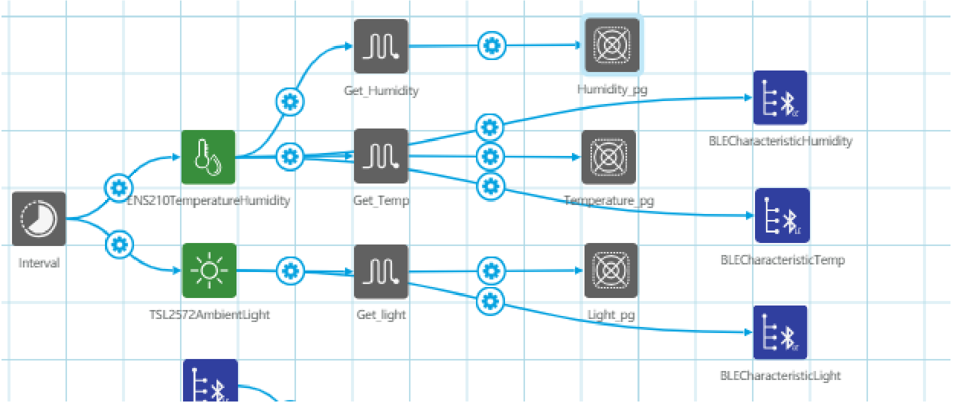
- I know that several pages should be added into the Kit when we finish the design, but how to flip over to the next page we want to show, page controller can help



- Add the button and touch element to trigger the page turning (Notice: each element may have more than 1 trigger actions. Different action triggers different event, and they work at the same time)
  - For example, we want to move to page (0,1) from page (0,0), we need to set the event: when we touch the down touch button, turn to the page under (0.0)

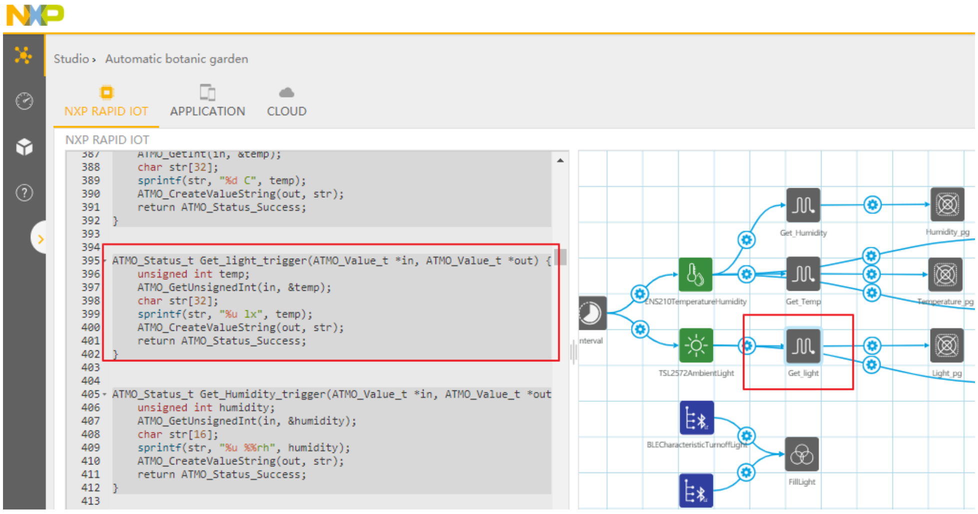


- Next, let's add the sensor part



- Interval is needed to trigger the data reading for sensors, here I set to read the data every 1000ms

- When the data is read, the format is hex, we need a function element to do the convert, code as below:
  - ➢ Get_light data as an example, the coding should be finished by user, but we can refer to the demo project to do this, click the Label8, you can see the code

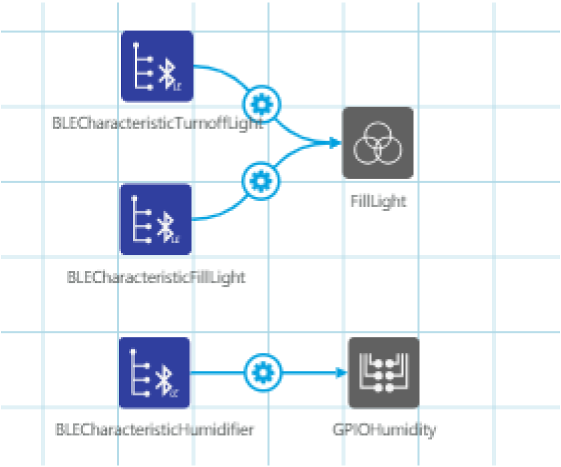


```
    ATMO_GetInt(in, &temp);
    char str[32];
    sprintf(str, "%d C", temp);
    ATMO_CreateValueString(out, str);
    return ATMO_Status_Success;
}


ATMO_Status_t Get_light_trigger(ATMO_Value_t *in, ATMO_Value_t *out) {
    unsigned int temp;
    ATMO_GetUnsignedInt(in, &temp);
    char str[32];
    sprintf(str, "%u lx", temp);
    ATMO_CreateValueString(out, str);
    return ATMO_Status_Success;
}


ATMO_Status_t Get_Humidity_trigger(ATMO_Value_t *in, ATMO_Value_t *out
    unsigned int humidity;
    ATMO_GetUnsignedInt(in, &humidity);
    char str[16];
    sprintf(str, "%u %%rh", humidity);
    ATMO_CreateValueString(out, str);
    return ATMO_Status_Success;
}
```

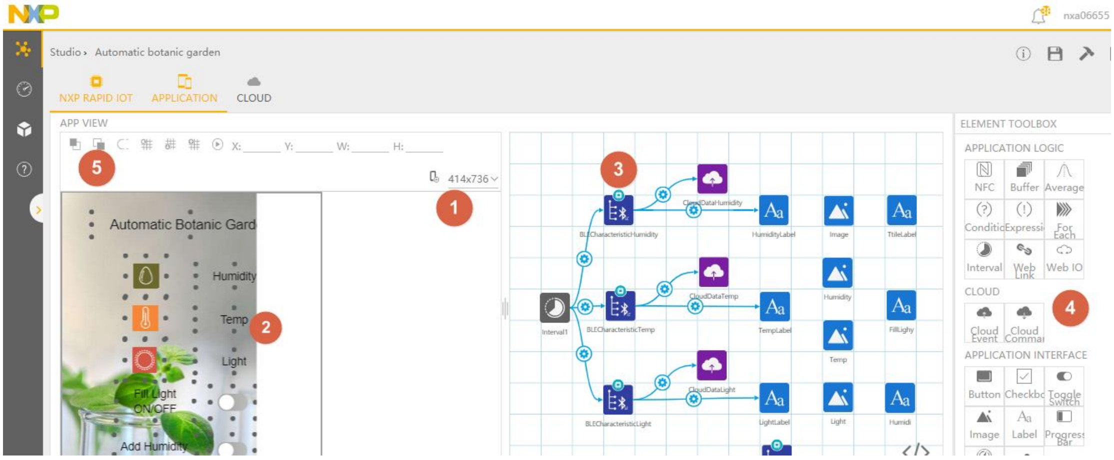
- After the convert, we can send the data to a label and show it on the Kit
- Another thing is that, we need send another copy of data to BLE, so that App on the phone can get it
- The last part for the Kit is to deal with the BLE data send from App. Set the event to turn on/off the LED on the kit or control the external device by GPIO/I2C/SPI and so on.
  - ➢ Notice: for BLE element, the max number is 60, and every 10 can share a service UUID and each of them has a special characteristic UUID



BLECharacteristicTurnoffLight
FillLight
BLECharacteristicFillLight
BLECharacteristicHumidifier
GPIOHumidity

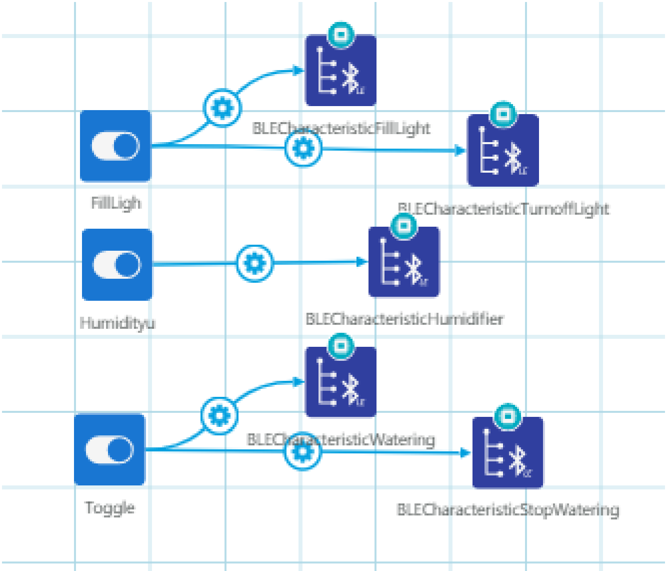
5.2 Design for App on the phone

- Click the application button, we can start to do the design for App, it is really an easy way to design a UI on the phone, amazing.



- ➢ Label1, select your phone and screen size
- ➢ Label2, UI
- ➢ Label3, design grids
- ➢ Label4, function element
- ➢ Label5, operation button for UI

- ■ Create an interval to trigger the BLE transfer
- ■ Create the labels to show and humidity, light, temperature data from BLE
- ■ Create the labels for uploading local pictures, try to make the UI looks nice
- ■ Add the button on the UI, set the events for them
    - ➢ For example, when user click the button, an trigger signal will send to the Kit though BLE, then the GPIO will be triggered to enable/disable the external devices.

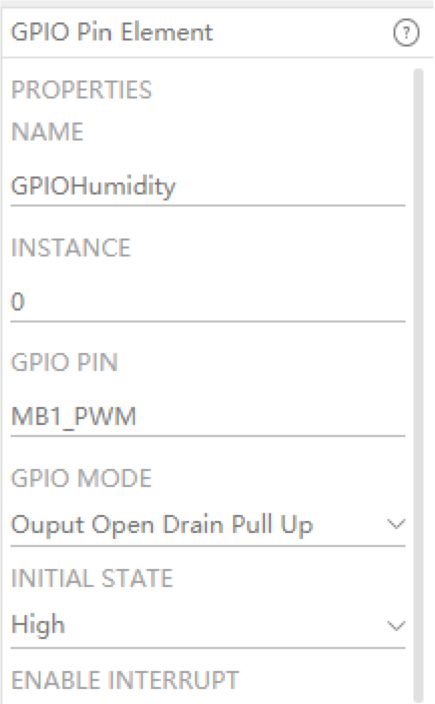


FillLigh
BLECharacteristicFillLight
BLECharacteristicTurnoffLight
Humidityu
BLECharacteristicHumidifier
Toggle
BLECharacteristicWatering
BLECharacteristicStopWatering

5.3 Add external devices

- ■ Add the external devices, so that you can do lots things, such as add the humidity, water the plant, turn on/off the light, even open/close your curtain if you add a motor.

- Here we need to put the kit into Docking station, then you can easily connect the pins to the controllers for your external devices
- But how to find the pins, we need to check the setting for GPIO



GPIO Pin Element

PROPERTIES

NAME

GPIOHumidity

INSTANCE

0

GPIO PIN

MB1_PWM

GPIO MODE

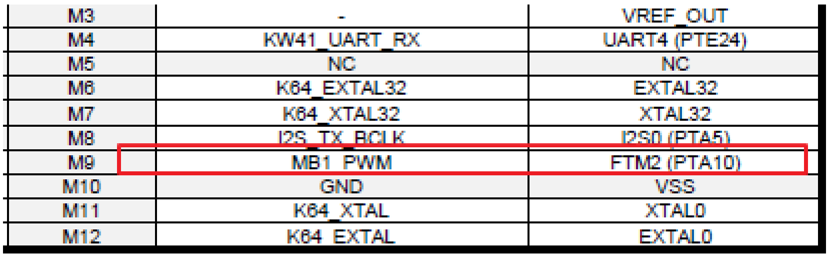
Ouput Open Drain Pull Up

INITIAL STATE

High

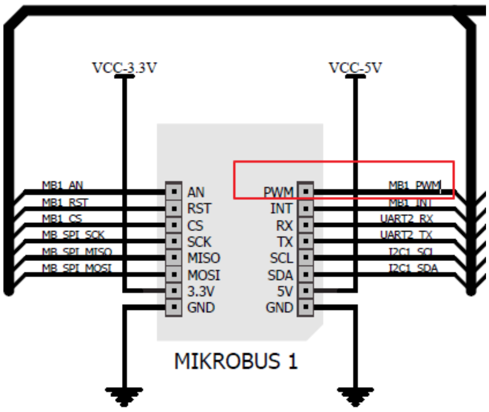
ENABLE INTERRUPT

- Pin name is MB1_PWM, then open the user guide for Rapid IoT, here I'm using 'SLN-RPK-NODE_UG v0.21.pdf', turn to chapter 3.2.2 pin assignment, you can know that the pin is PTA10



M3	-	VREF_OUT
M4	KW41_UART_RX	UART4 (PTE24)
M5	NC	NC
M6	K64_EXTAL32	EXTAL32
M7	K64_XTAL32	XTAL32
M8	I2S_TX_BCLK	I2S0 (PTA5)
M9	MB1_PWM	FTM2 (PTA10)
M10	GND	VSS
M11	K64_XTAL	XTAL0
M12	K64_EXTAL	EXTAL0

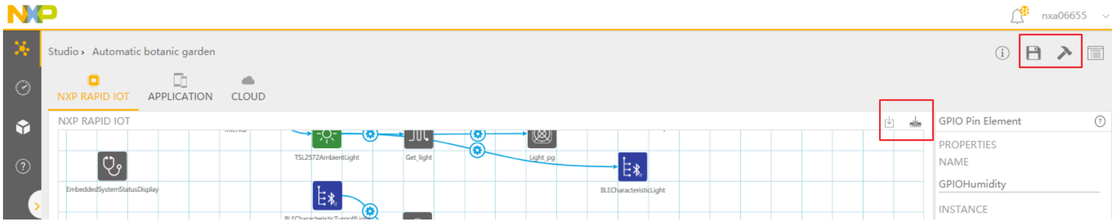
- Open the schematic for docking station, you will find the right pin



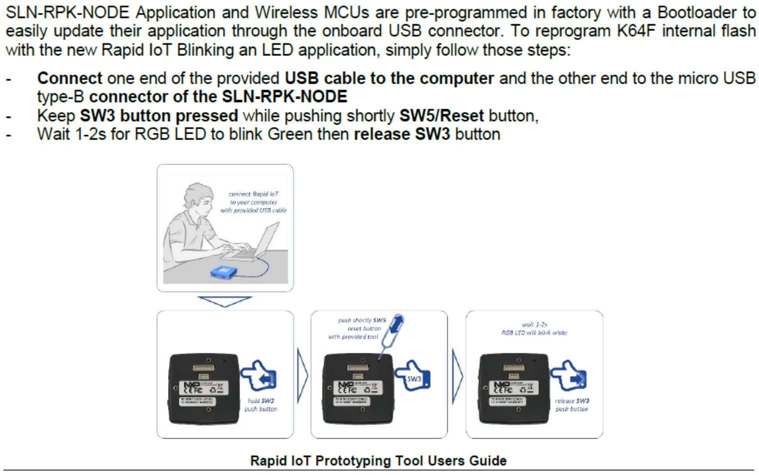
- For this project I add a relay for the humidifier and pump, turn on/off the GPIO to control them. For some other complex devices, we can use I2C/SPI or some other protocol to control them.

6 Run the project

6.1 All then design done, let's build the project and download it, don't forget to save your project



6.2 Update your firmware for the Kit according to the page 28 of the user guide



SLN-RPK-NODE Application and Wireless MCUs are pre-programmed in factory with a Bootloader to easily update their application through the onboard USB connector. To reprogram K64F internal flash with the new Rapid IoT Blinking an LED application, simply follow those steps:

- **Connect** one end of the provided **USB cable to the computer** and the other end to the micro USB type-B **connector of the SLN-RPK-NODE**
- Keep **SW3 button pressed** while pushing shortly **SW5/Reset** button,
- Wait 1-2s for RGB LED to blink Green then **release SW3** button

connect Rapid IoT to your computer with provided USB cable
hold SW3 push button
push shortly SW5 reset button with provided tool
SW3
wait 1-2s RGB LED will blink white
release SW3 push button

Rapid IoT Prototyping Tool Users Guide

6.3 The kit should be working now

6.4 Open your phone download the NXP Rapid IoT app, install and open it, login in with your account

6.5 Operate according to page 10 of the User guide, you can search your Kit and connect with it.

6.6 The UI you designed will show up on your phone.

6.7 All things done, try to play it.