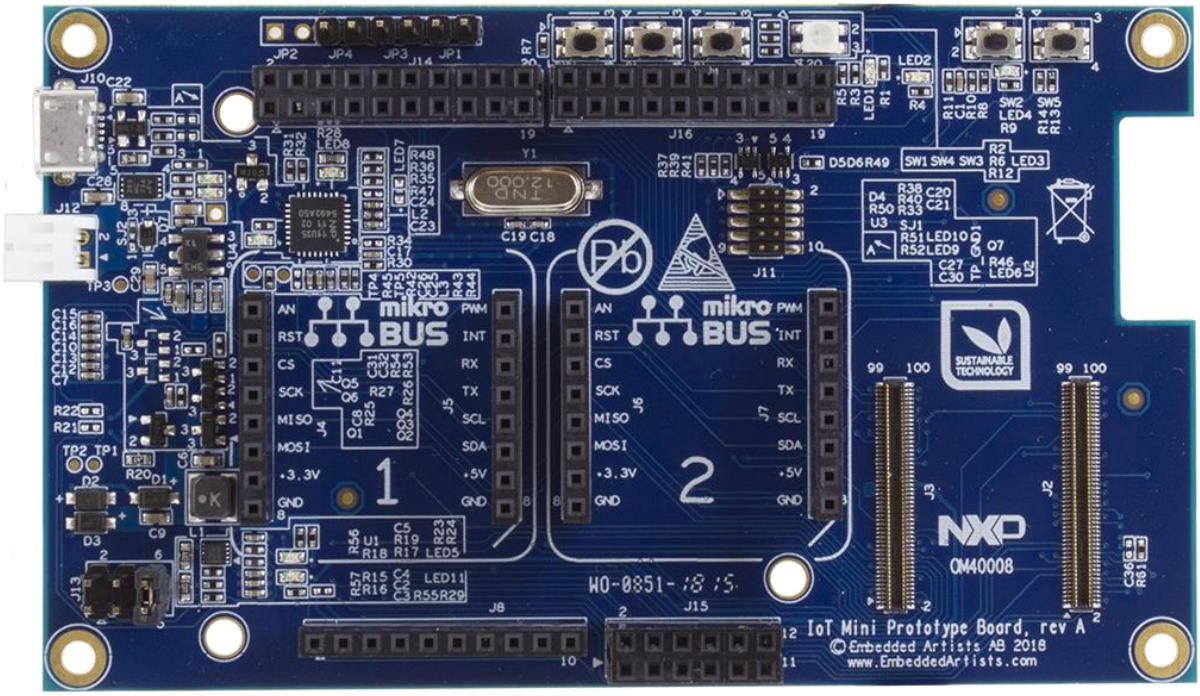
IoT Mini Prototype Board with LPC54018 IoT Module



Embedded Artists AB
Jörgen Ankersgatan 12
211 45 Malmö
Sweden

http://www.EmbeddedArtists.com

Disclaimer

Embedded Artists AB makes no representation or warranties with respect to the contents hereof and specifically disclaim any implied warranties or merchantability or fitness for any particular purpose. Information in this publication is subject to change without notice and does not represent a commitment on the part of Embedded Artists AB.

Feedback

We appreciate any feedback you may have for improvements on this document. Please send your comments to support@EmbeddedArtists.com.

Trademarks

All brand and product names mentioned herein are trademarks, services marks, registered trademarks, or registered service marks of their respective owners and should be treated as such.

Table of Contents

1 Document Revision History

Revision	*Date*	*Description*
PA1	2018-05-15	Initial version with software instructions.
PA2	2018-06-04	Added information about hardware design.
PA3	2018-06-11	Added correct pictures and information about ePaper mounting.

2 Board Overview

The ***IoT Mini Prototype Board*** has been designed to be a easy-to-use prototyping platform for the ***LPC54018 IoT Module*** - to goal is to get up-and-running quickly with your prototyping involving!

The board has the following features:

- Arduino compatible expansion connector
  There is a wide range of Arduino shields that can be used for prototyping.
- Dual Click module compatible expansion connectors
  There is a wide range of Click modules that can be used for prototyping.
- 3-axis accelerometer
- RGB-LED and LEDs
- Pushbuttons for controlling ISP boot mode of the IoT Module
- DAP-LINK compatible debug interface
- Possibility to measure current consumption of the IoT Module
- Li-ion battery charger
- Connector to ePaper display (note that this is an optional feature - the ePaper display is not included. It must be ordered separately from the display manufacturer or a catalog distributor.)

The picture below illustrates the top side of the board and the location of the main features.

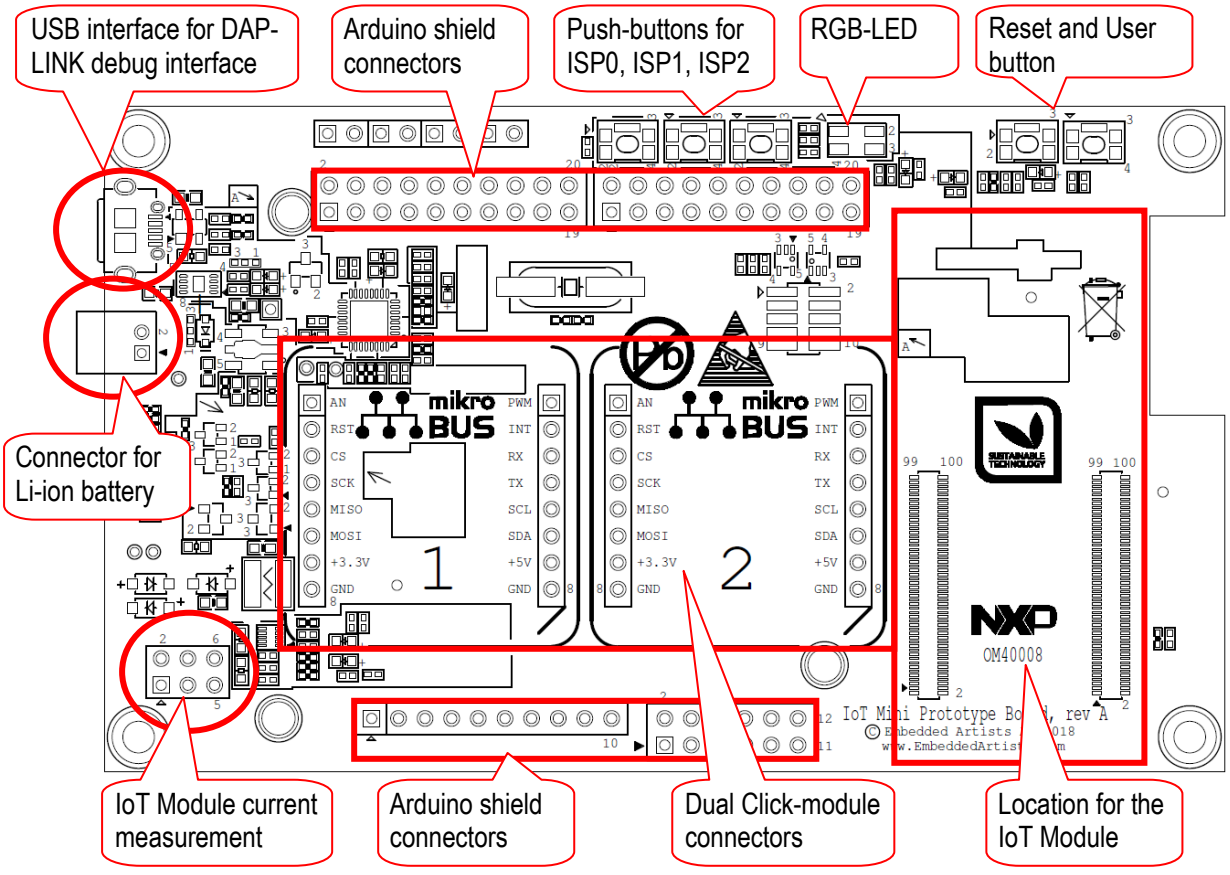


Figure 1 – IoT Mini Prototype Board, Top Side

On the bottom side, the optional ePaper display can be mounted.

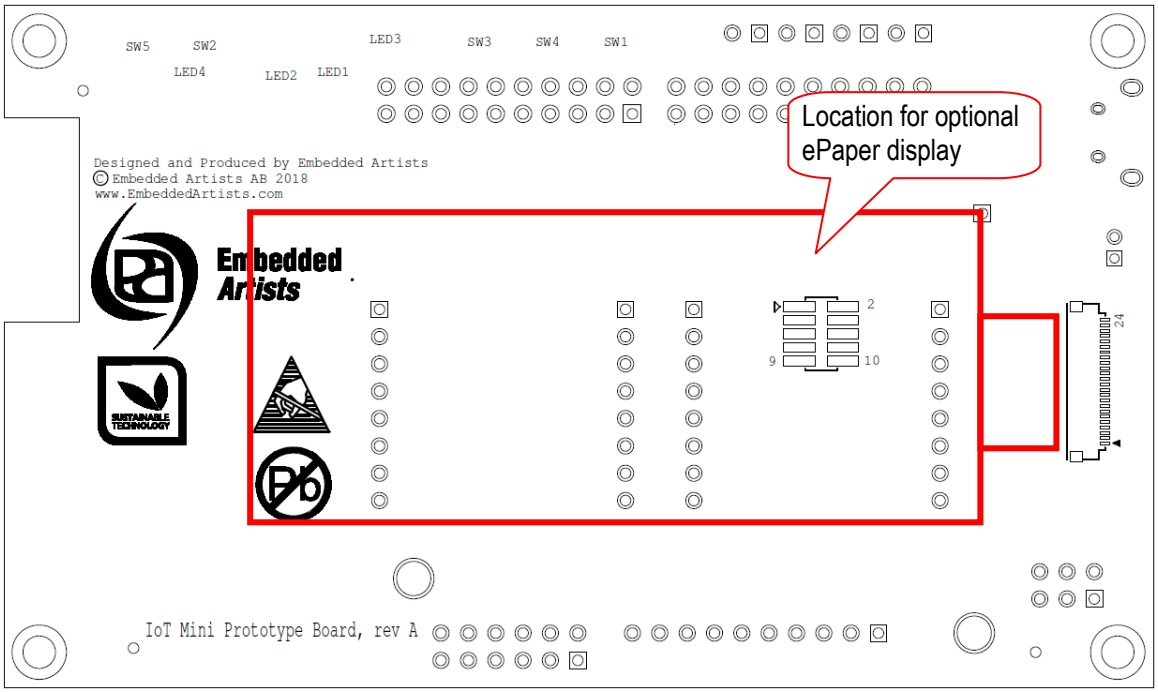


Figure 2 – IoT Mini Prototype Board, Bottom Side

3 Hardware Design

This chapter describes the hardware design, i.e., the schematic in more detail. Each separate function will be addressed in a separate section.

3.1 LPC54018 IoT Module Connectors

Page 6 of the schematic contains the two DF40C connectors, J2/J3, that mates to the ***LPC54018 IoT Module***. These are low-profile connectors with only 1.5mm space between the boards.

> Caution - be very careful when you mount and unmount/remove the ***LPC54018 IoT Module*** to/from the ***IoT Mini Prototype Board***.
>
> When mounting the ***LPC54018 IoT Module***:
>
> 1) Place the ***LPC54018 IoT Module*** over J2/J3, but do not press. Just find the location where the connectors lock by moving around the ***LPC54018 IoT Module*** a bit with very small movements. When the locking position has been found, continue with next step.
>
> 2) Place both your thumbs on top of the ***LPC54018 IoT Module***, just over J2 and J3, respectively, and press firmly on both sides at the same time. A click sound will be heard when the two pairs of connectors lock.
>
> When unmounting/removing the ***LPC54018 IoT Module*** - first consider if you really need to do that. If you definitively need to remove the module, then start wiggling the ***LPC54018 IoT Module*** with very small movements. A very small screw with flat head can also be used to very gently lift up the module. Note that it is easy to damage the DF40C connectors with improper handling..

The picture below illustrates how it looks like when the ***LPC54018 IoT Module*** has been mounted.

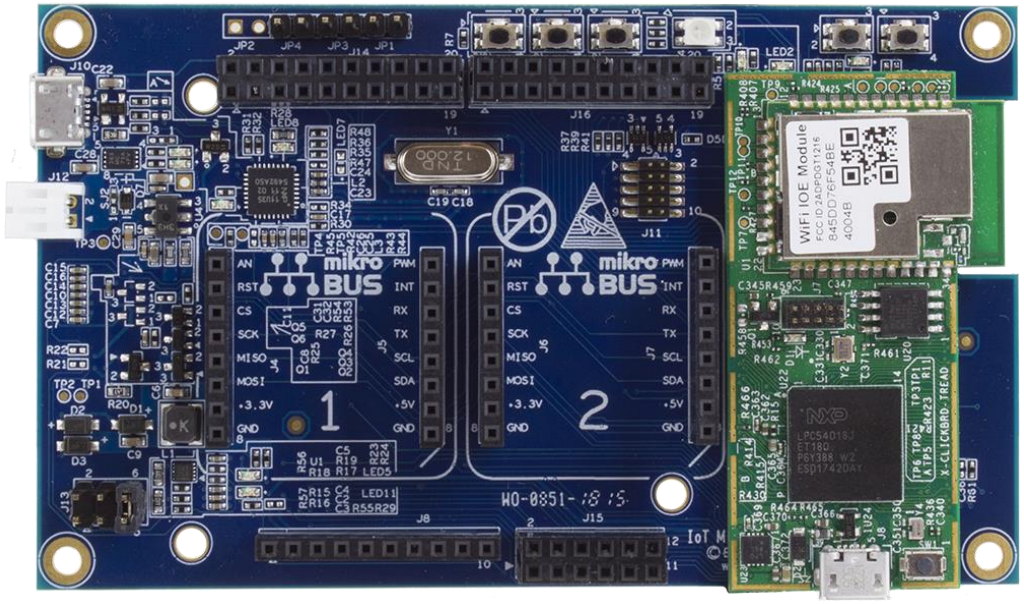


Figure 3 – IoT Mini Prototype Board with LPC54018 IoT Module Mounted

3.2 Arduino Compatible Expansion Interface

Page 6 of the schematic contains the Arduino shield connectors, J8, J14, J15 and J16. Three of these (J14, J15 and J16) are double row connectors, which are not part of the Arduino shield connector standard, but these extra pins have been added to be compatible with some of NXP's LPCXpresso and related add-on boards.

The table below lists the main Arduino shield pins and the corresponding LPC54018 signal they are connected to. Note that some signals are also connected to the Click module expansion connectors.

Arduino signal	*Expansion connector and pin number*	*Corresponding LPC54018 pin*
AIN0	J15, pin 2	P0_16-ADC0IN4 Note: Signals is also connected to Click Module #1
AIN1	J15, pin 4	P0_31-ADC0IN5 Note: Signals is also connected to Click Module #2
AIN2	J15, pin 6	NOTE: Grounded on rev A boards!
AIN3	J15, pin 8	P2_0-ADC0IN7-PMOD2_INTR Note: Signals is also connected to Click Module #2
AIN4	J15, pin 10	P3_4-ACCL_INTR
AIN5	J15, pin 12	P1_1-USER_PB
RX (D0)	J16, pin 15	P3_26-FC4_RXD Note: Signals is also connected to both Click Modules
TX (D1)	J16, pin 13	P3_27-FC4_TXD Note: Signals is also connected to both Click Modules
D2	J16, pin 11	P3_2-FC9_MOSI-CT1MAT2
D3	J16, pin 9	P4_5-CT4MAT3 Note: Signals is also connected to Click Module #1
D4	J16, pin 7	P3_10-CT3MAT0
D5	J16, pin 5	P3_14-CT3MAT1-USR_LED1
D6	J16, pin 3	P3_28 Note: Signals is also connected to Click Module #2
D7	J16, pin 1	P2_2-CT1MAT1-USR_LED3
D8	J14, pin 19	P3_29
D9	J14, pin 17	P2_1-CT1MAT0
D10	J14, pin 15	P3_30-FC9_SSELN0 Note: Signals is also connected to Click Module #1
D11	J14, pin 13	P3_21-FC9_MOSI Note: Signals is also connected to both Click Modules
D12	J14, pin 11	P3_22-FC9_MISO Note: Signals is also connected to both Click Modules
D13	J14, pin 9	P3_20-FC9_SCK Note: Signals is also connected to both Click Modules

SDA	J14, pin 3	P3_23-FC2_SDAX Note: Signals is also connected to both Click Modules
SCL	J14, pin 1	P3_24-FC2_SCLX Note: Signals is also connected to both Click Modules

Note that due to a schematic error on rev A boards, pin 6 of J15 is grounded. This means that the Arduino signal AIN2 cannot be used - and is grounded. Make sure the connected Arduino shield works and is not damaged when AIN2 is grounded.

3.3 Click Module Compatible Expansion Interface

There are two Click module expansion interface connectors. Another name for this open interface standard is mikroBUS™ (see https://www.mikroe.com/mikrobus). microBus is said to be "the add-on board standard that offers maximum expandability with the smallest number of pins". The specification can be downloaded from:
https://download.mikroe.com/documents/standards/mikrobus/mikrobus-standard-specification-v200.pdf

Page 6 of the schematic contains the two Click Module expansion interface connectors, J4/J5 and J6/J7. The two Click module interfaces share several pins for the SPI, UART and I2C busses. They also share several pins with the Arduino shield expansion connectors.

The table below lists the connector pins and the corresponding LPC54018 signal they are connected to.

Click Module #1 signal	*Direction related to connector socket*	*Expansion connector and pin number*	*Corresponding LPC54018 pin*
AN	Input	J4, pin 1	P0_16-ADC0IN4 Note: Signals is also connected to Arduino interface
RST	Output	J4, pin 2	NOTE: Grounded on rev A boards!
CS	Output	J4, pin 3	P3_30-FC9_SSELN0 Note: Signals is also connected to Arduino interface
SCK	Output	J4, pin 4	P3_20-FC9_SCK Note: Signals is also connected to Click Module #2 and to the Arduino interface
MISO	Input	J4, pin 5	P3_22-FC9_MISO Note: Signals is also connected to Click Module #2 and to the Arduino interface
MOSI	Output	J4, pin 6	P3_21-FC9_MOSI Note: Signals is also connected to Click Module #2 and to the Arduino interface
PWM	Output	J5, pin 1	P4_5-CT4MAT3 Note: Signals is also connected to Arduino interface
INT	Input	J5, pin 2	P1_4
RX	Input	J5, pin 3	P3_26-FC4_RXD Note: Signals is also connected to Click Module #2 and

			to the Arduino interface
TX	Output	J5, pin 4	P3_27-FC4_TXD Note: Signals is also connected to Click Module #2 and to the Arduino interface
SCL	Output	J5, pin 5	P3_24-FC2_SCLX Note: Signals is also connected to Click Module #2 and to the Arduino interface
SDA	Bidirectional	J5, pin 6	P3_23-FC2_SDAX Note: Signals is also connected to Click Module #2 and to the Arduino interface

Note that due to a schematic error on rev A boards, pin 2 of J4 is grounded. This means that the Click module #1 signal RST cannot be controlled - and is grounded. Make sure the connected Click module does not use the RST input - many Click modules do not use this signal, but some do..

Click Module #2 signal	*Direction related to connector socket*	*Expansion connector and pin number*	*Corresponding LPC54018 pin*
AN	Input	J6, pin 1	P0_31-ADC0IN5 Note: Signals is also connected to Arduino interface
RST	Output	J6, pin 2	P2_0-ADC0IN7-PMOD2_INTR Note: Signals is also connected to Arduino interface
CS	Output	J6, pin 3	P3_30-FC9_SSELN0 Note: Signals is also connected to Arduino interface
SCK	Output	J6, pin 4	P3_20-FC9_SCK Note: Signals is also connected to Click Module #1 and to the Arduino interface
MISO	Input	J6, pin 5	P3_22-FC9_MISO Note: Signals is also connected to Click Module #1 and to the Arduino interface
MOSI	Output	J6, pin 6	P3_21-FC9_MOSI Note: Signals is also connected to Click Module #1 and to the Arduino interface
PWM	Output	J7, pin 1	P3_28 Note: Signals is also connected to Arduino interface
INT	Input	J7, pin 2	P1_5
RX	Input	J7, pin 3	P3_26-FC4_RXD

			Note: Signals is also connected to Click Module #1 and to the Arduino interface
TX	Output	J7, pin 4	P3_27-FC4_TXD Note: Signals is also connected to Click Module #1 and to the Arduino interface
SCL	Output	J7, pin 5	P3_24-FC2_SCLX Note: Signals is also connected to Click Module #1 and to the Arduino interface
SDA	Bidirectional	J7, pin 6	P3_23-FC2_SDAX Note: Signals is also connected to Click Module #1 and to the Arduino interface

3.4 3-Axis Accelerometer

Page 3 of the schematic contains the 3-axis accelerometer, MMA8652FCR1. The sensor has an I2C interface and is connected to the following I2C signals of the LPC54018: P3_24-FC2_SCLX and P3_23-FC2_SDAX. There is also an interrupt output that is connected to signal: P3_4-ACCL_INTR.

The MMA8652FCR1 has I2C address: 0x3A/0x3B (read/write, in 8-bit format) and 0x1D (in 7-bit format).

3.5 RGB-LED, LEDs and Push-buttons

Page 2 of the schematic contains an RGB-LED, two user-controlled LEDs and number push-buttons.

The table below lists the LED connections.

LED	*LED color*	*Corresponding LPC54018 pin*
LED1, Single LED	Red	P3_3-USR_LED2
LED2, Single LED	Red	P3_14-CT3MAT1-USR_LED1
LED3, RGB-LED	Red	P0_17-SCT0_OUT0
LED3, RGB-LED	Green	P0_19-SCT0_OUT2
LED3, RGB-LED	Blue	P0_18-SCT0_OUT1

The table below lists the push-button connections. Three of the push-buttons are used to control the ISP mode pins, to control boot source for the LPC54018.

Push-button	*Usage*	*Corresponding LPC54018 pin*
SW1	ISP0 control	P0_4-EMC_D2
SW2	Reset	RESET_B + Reset-LED (LED4)
SW3	ISP1 control	P0_5-EMC_D3
SW4	ISP2 control	P0_6-EMC_D4 (signal name is IF_ISPEN, but connects to signal P0_6-EMC_D4 on schematic page 7)
SW5	User button	P1_1-USER_PB

The push-buttons only drive the signals low via high-ohm resistors. For the ISP control signals this is because the signals are be outputs under regular operation (for example when using an external SDRAM)

on the ***LPC54018 IoT Module***. For the user button, SW5, connected to signal P1_1-USER_PB, the 100 ohm resistor is just to protect if the application code by accident sets the signals as an output.

3.6 DAP-LINK Debug Interface

Page 7 of the schematic contains a DAP-LINK debug interface, implemented in the LPC11U35 microcontroller. The MCUXpresso IDE supports this debug interface.

3.7 Powering, Current Measurement and Battery Charger

Page 7 of the schematic contains the powering solution. There is a 3.3V power supply, capable of supplying up to 700mA to the ***IoT Mini Prototype Board*** and ***LPC54018 IoT Module*** together.

There are four ways to power the board:

- Via the USB interface, J10, of the debug interface.
  This is the normal/most common way of powering the board during development.
- Via a (rechargeable) Li-ion battery, connected to J12.

  Note that SJ2 must be changed to be in 1-2 position. SJ2 can be found close to the battery connector, J12.

  Note that the battery must always be connected the board is operating. The Li-ion charger cannot alone supply the current needed during peaks, for example when the Wi-Fi module is active. Current peaks must be supplied directly from the battery.

  Note that the Li-ion battery can only be charged via the debug interface USB connector, J10.

  Note that the external Li-ion battery shall be a single cell battery with a battery voltage in the 3.7-4.2V range.

  Note that the Li-ion battery charger is set to 200mA/50mA constant/trickle charge current.
- Via the USB interface connector on the ***LPC54018 IoT Module***.
- Via the +5V rail on the Click module or Arduino shield expansion connectors, J5 pin 7, J7 pin 7 or J8 pin 7.

It is possible to measure current into LPC54018 of the ***LPC54018 IoT Module*** via J13, the 2x3 pos pin header in the lower left corner. Two series resistors (each 1 ohm) can be inserted in series with the current to the LPC54018. By measuring the voltage drop over the series resistor(s), the current can be calculated.

Series resistor to measure over	***J13 setup***	***Note***
0 ohm	insert 3 jumpers, in 1-2, 3-4 and 5-6 pos	No series resistor to measure voltage over.
1 ohm	insert 2 jumpers, in 3-4 and 5-6 pos	Measure the voltage drop (over the 1 ohm resistor) on J13 pos 1-2.
2 ohm	insert 1 jumper, in 5-6 pos	Measure the voltage drop (over the 2 ohm resistor) on J13 pos 1-3.

		This is the default setup.
Use current meter instead	insert 2 jumpers, in 1-2 and 3-4 pos	Measure current through J13 pos 5-6. Note: Use a current meter with low internal resistance

3.8 ePaper Display Interface

Page 4 of the schematic contains the ePaper display interface. It is designed for Pervasive Display Inc panel E2266FS092. That is a 2.66 inch, 3-ColorsTFT EPD Panel with 296 x 152 pixels resolution. That gives a pixel density of 152 dpi. For more information about the display, see:
http://www.pervasivedisplays.com/products/266

Note that this is an optional feature - the ePaper display is not included. It must be ordered separately from the display manufacturer (http://www.pervasivedisplays.com/products/266) or a catalog distributor.
https://www.digikey.com/products/en?keywords=E2266FS092

When mounting the ePaper display, first place thick double-sided tape on the bottom side. Cover most of the display area with tape. In the picture below, the tape used is 20 mm wide and three rows are needed to cover most the display. As seen in the picture, two layers of the tape are applied. The tape is 1mm thick so the resulting thickness is about 2mm. This is the minimum recommended thickness.

Figure 4 – ePaper Display, Two Layers of Double-Sided Tape on Bottom Side

Open the connector's flip lock carefully. It is found on the bottom side of the ***IoT Mini Prototype Board***. Insert the flat cable of the ePaper display carefully. Close the connector's flip lock carefully. Remove the upper protection layer of the double-sided tape, align the display and press the display carefully to the PCB. Do not use too much force when pressing the ePaper display. The material is glass so it can break when applying too much pressure.

When ready, it will look something like the picture below when the ePaper display has bene mounted. Note that there will be no content on the display

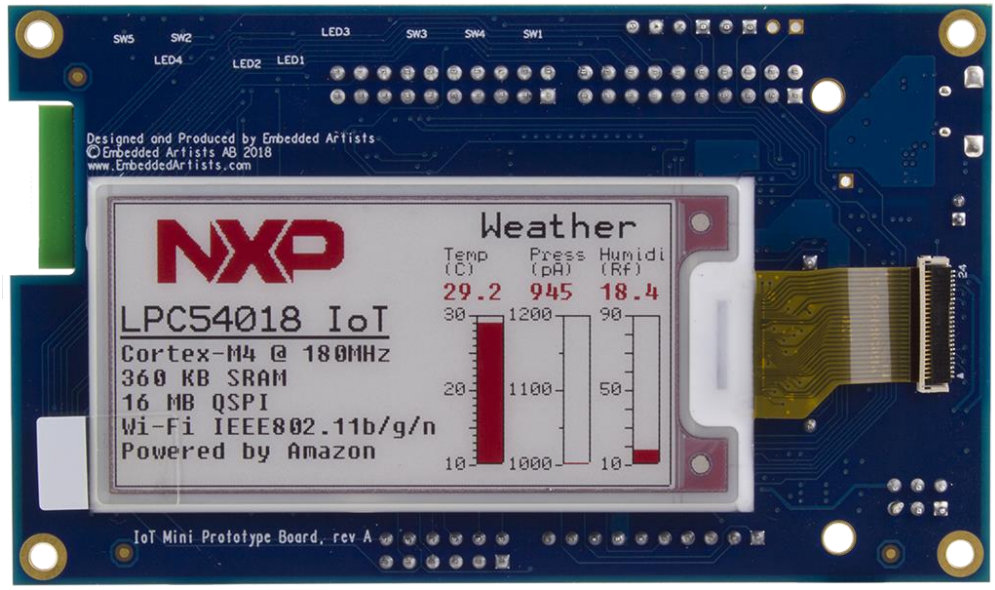


Figure 5 – ePaper Display Mounted on Bottom Side

Note that ePaper displays must be refreshed regularly. Over time the contrast will become lower. The picture below illustrates how the display looks like after about 24 hours. The red color starts to “bleed” in the display area. This specific ePaper display needs to be refreshed at least 2-4 times a day to keep the contrast acceptable.

Figure 6 – ePaper Display Turning Pink

4 Getting Started with NXP SDK

This chapter contains instructions to get started with software development work in two steps:

- First, follow the instructions from NXP to setup the software development environment on your computer. This part is covered in this chapter.
- Secondly, follow the instructions in this document to download and compile the code specific for the ***IoT Mini Prototype board***. This part is covered in chapter 5 .

4.1 Initial Instructions from NXP to Get Started

To start with, you need to download and install:

- MCUXpresso IDE, latest version. This is currently the only supported IDE.
- The SDK for the ***LPC54018 IoT Module***.

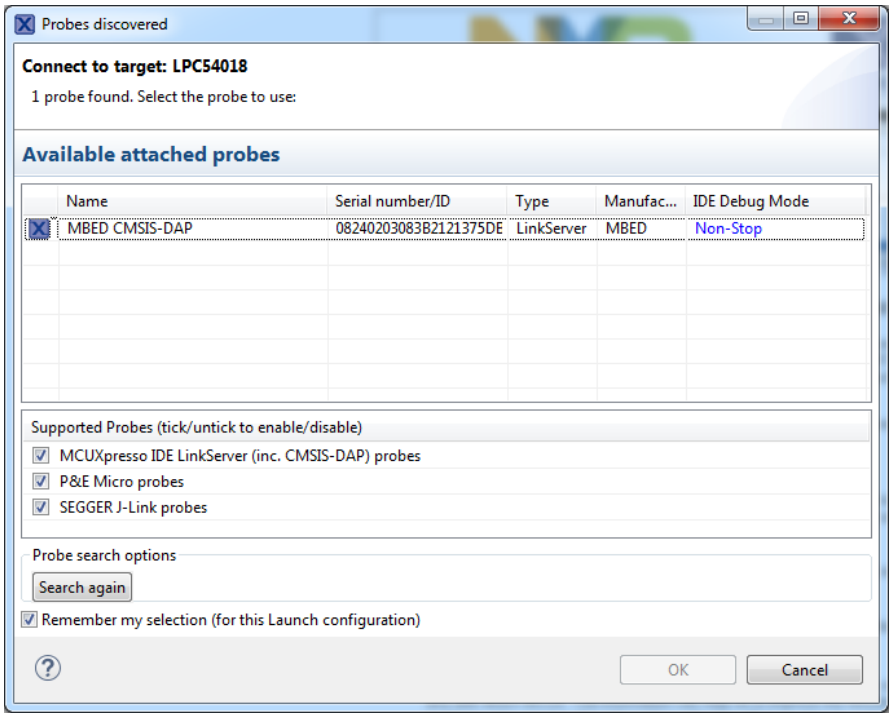
You also need to create an Amazon AWS account.

The best way to get started is to use the great online guide that NXP has written here: https://www.nxp.com/support/developer-resources/reference-designs/lpc54018-iot-module-for-the-lpc540xx-family-of-mcus:OM40007?tab=In-Depth_Tab.

This document covers the following steps:

- Download/Installation of the IDE
- Download of the SDK
- Creation of an Amazon account
- Registering the IoT Module in the AWS platform
- Build, download and execution of the **lpc54018iotmodule_aws_shadow_wifi_qspi_xip** example.

The only step that differs from the online guide is step **3.4 Connect a debugger probe**. The ***IoT Mini Prototype Board*** has an onboard debug interface:



Follow all the steps in the online guide up to **3.6 Success!** You are now ready to install and use the software specific to the ***IoT Mini Prototype Board***, see next chapter.

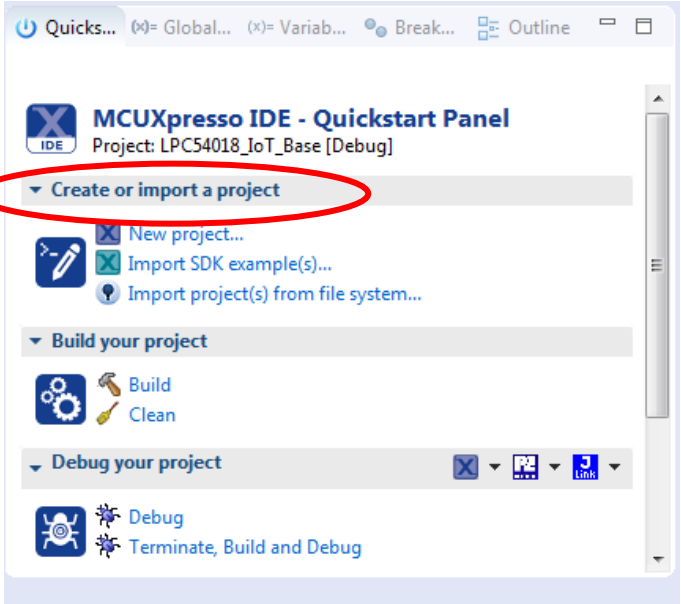
5 Explore the IoT Mini Prototype Board SW

This chapter contains instructions to download and compile the code specific for the ***IoT Mini Prototype board***.

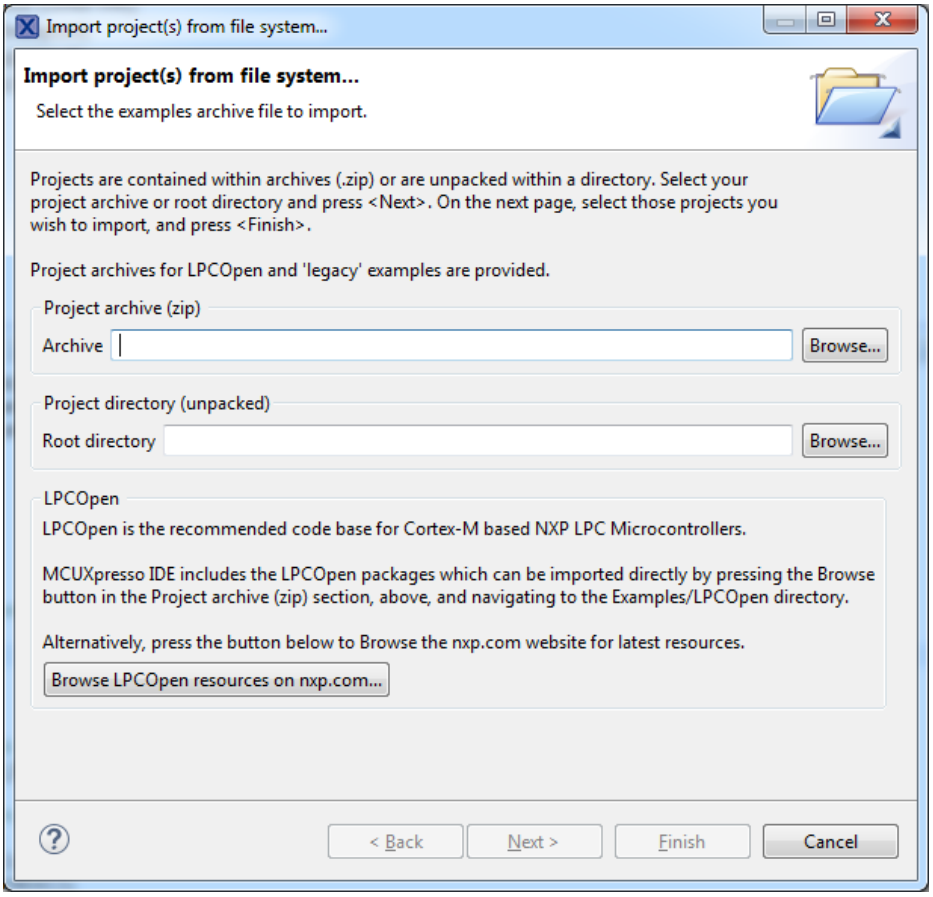
5.1 Installation of Embedded Artists' Software Bundle

After verifying that everything works by following the guide in the previous chapter it is now time to install the example projects specific for the ***IoT Mini Prototype Board***.

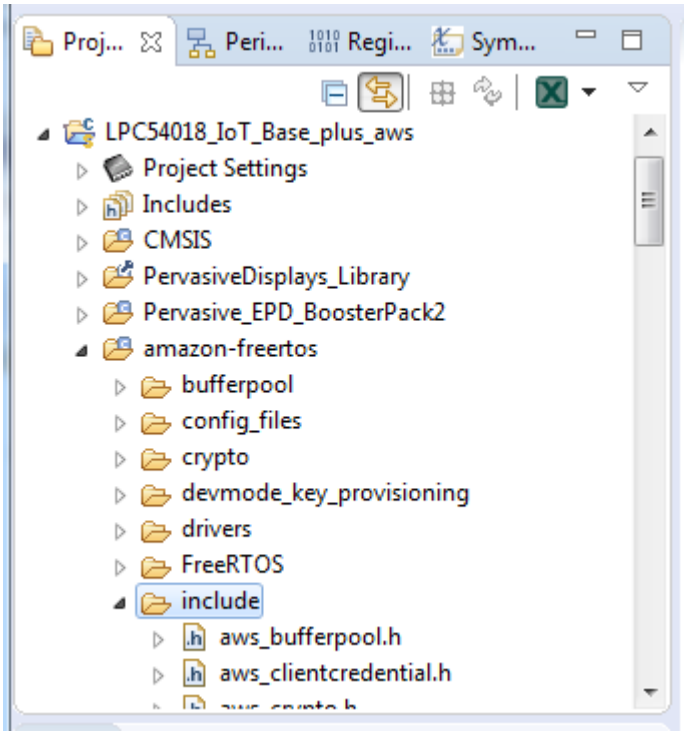
1. Start by clicking the **Import project(s) from file system…** link in the Quickstart Panel in MCUXpresso.



2. In the dialog that appears either point to the zip file or, if you have unpacked it, to the directory where you unpacked it. Click Finish to import all projects.



3. In the online guide that was followed in section 4.1 one file was generated (aws_clientcredential_keys.h) and one was updated (aws_clientcredential.h). The files contain the wifi settings and the certificates that are needed to connect to AWS. Copy those files from the amazon-freertos/include/ directory in the **lpc54018iotmodule_aws_shadow_wifi_qspi_xip** project into the same location in the newly imported LPC54018_IoT_Base_plus_aws project.



5.2 Software Common to all Projects

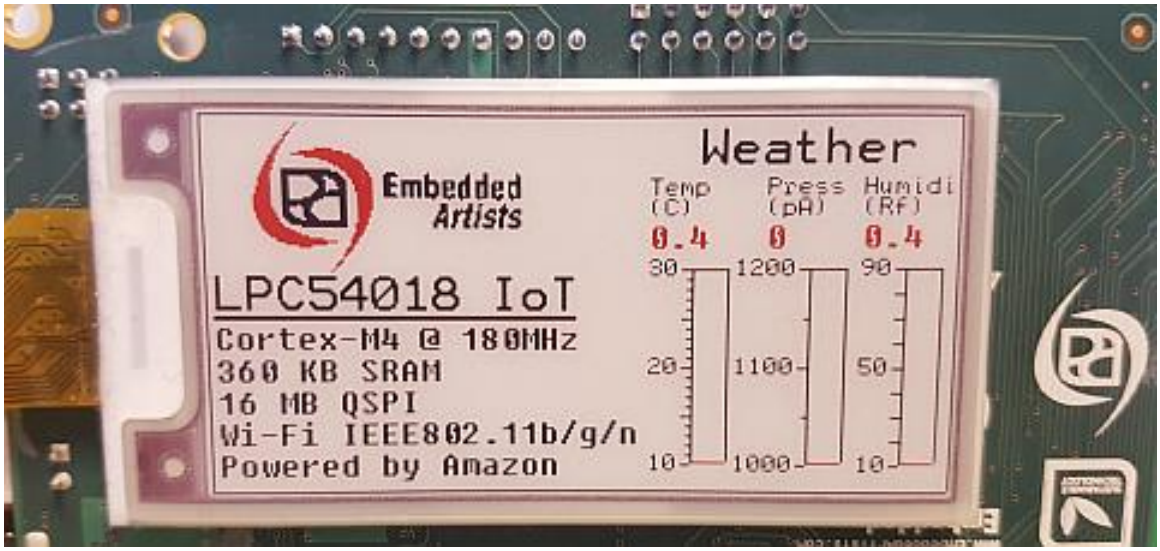
The goal with the example projects is to make it easier to use the unique features of the ***IoT Mini Prototype Board***.

5.2.1 Support for ePaper Display

The ***IoT Mini Prototype Board*** has a connector to an optional (bought separately) 2.66" ePaper display that can display red in addition to the standard black/white. The display requires two different libraries:

- Pervasive_EPD_BoosterPack2 which is the hardware abstraction layer
- PervasiveDisplays_Library which has drawing primitives

The displayEPD() function in TaskEPD.cpp contains one example of how to use the display. It draws one of three images, some text and a couple of bar charts. It looks something like this:



The example implementation waits for one or more values to change in the DataStore (see 5.2.3), allowing it to be redrawn only when needed.

5.2.2 Support for Click Modules and Arduino Shields

The ***IoT Mini Prototype Board*** has two connectors for Click modules and one for Arduino shield. The example projects come with skeleton code to make it easier to plug in software for any of the expansions.

To add for example a Click module to socket 2 on the ***IoT Mini Prototype Board***:

- Open expansion.h and change the corresponding define to a "1" to enable it.

```
#define EXPANSION_USE_CLICK_1    0
#define EXPANSION_USE_CLICK_2    1
#define EXPANSION_USE_ARDUINO    0
```

- Add the needed code to the exp_click2 directory
- Add initialization code (that need to be executed before the RTOS is started) to the expansion_click2_prepare() function in expansion_click1.c
- Add the click control code (executes in its own task) to the click2_task_entry() function in expansion_click2.c
- If the click module has some data to share, for example sensor readings, then add it to the DataStore (see 5.2.3)

5.2.3 DataStore

The DataStore is a collection of functions to allow multiple FreeRTOS tasks to share data and to be notified when it changes. This can normally be accomplished with queues but queues are limited to one consumer. In our case both the ePaper task and the AWS task are interested in the same data so another solution was needed. The DataStore uses a mutex to prevent different tasks from manipulating the data at the same time and event groups to notify "subscribers" of changes.

The actual data is stored in an instance of the data_store_t struct:

```
typedef struct
{
        char msg[DS_MAX_MSG_LEN+1];
        bool button_pressed;
} data_store_t;
```

It should be modified to suit the real application. By default it contains the status of the User button (the button closest to the corner of the ***IoT Mini Prototype Board***).

Next is a set of defines for the changes that can be made to the shared data:

```
#define DS_MSG_CHANGED    (1<<0)
#define DS_BUTTON_CHANGED (1<<1)
```

The defines will be used as bitmasks so it is important that the value is a single bit.

The read/write functions should be expanded/modified based on what is needed and the existing ones can be used for reference.

The last part of the DataStore is the ds_subscribe() and ds_unsubscribe() functions. Subscribe basically means that a unique event group is created and returned. The event group can then be used when calling the FreeRTOS function xEventGroupWaitBits to wait for a specific combination of the DS_*_CHANGED defines. For an example of how to use it, see the TaskEPD.cpp file.

5.2.4 Other Tasks

There is a task in main.cpp that handles button presses and turns some of the LEDs on and off. It uses the DataStore to save the state of the User button.

5.3 LPC54018_IoT_Base

A project with unused skeleton code for Click1 / Click2 / Arduino expansion.

The project has two tasks:

- **led_accel_task_entry** blinks with the LEDs and if the User Button is pressed it gets reported to the DataStore. The task also echoes any character typed in the terminal back to the user.
- **epd_task_entry** waits for the DataStore to report a button press and then updates the ePaper display. If there has been no updates in 60s the ePaper display is refreshed anyway.

5.4 LPC54018_IoT_Base_plus_click2

Based on LPC54018_IoT_Base but with the following changes:

- The project is configured to have a Weather Click module (https://www.mikroe.com/weather-click) inserted in socket 2 on the ***IoT Mini Prototype Board***.
- The code for the click module is in the exp_click2/ folder and it has its own task that publishes the weather data in the DataStore every 5 seconds
- The epaper task updates the display with the weather data from the DataStore

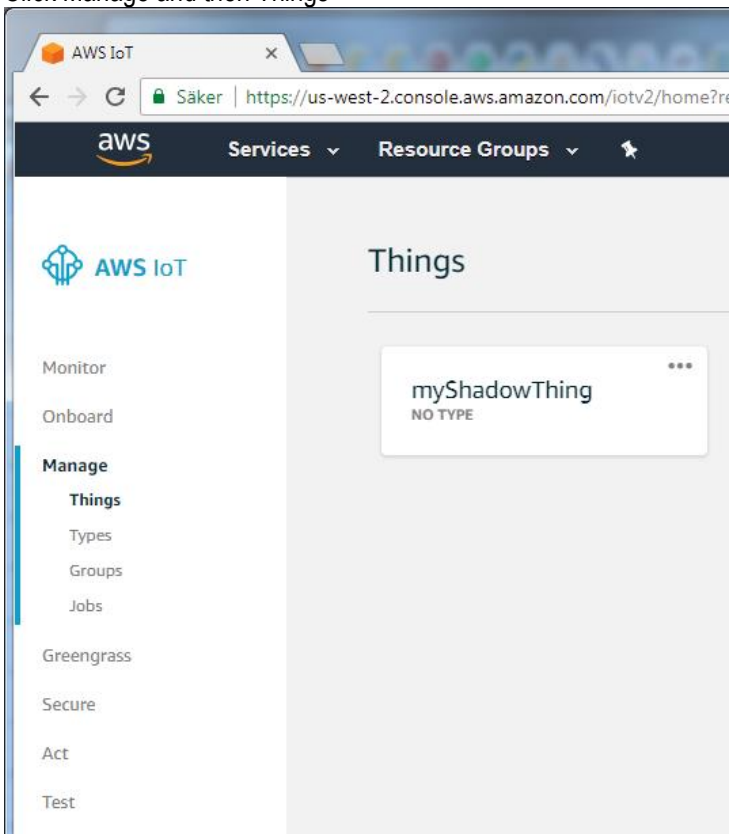
5.5 LPC54018_IoT_Base_plus_aws

Based on LPC54018_IoT_Base but with the following changes:

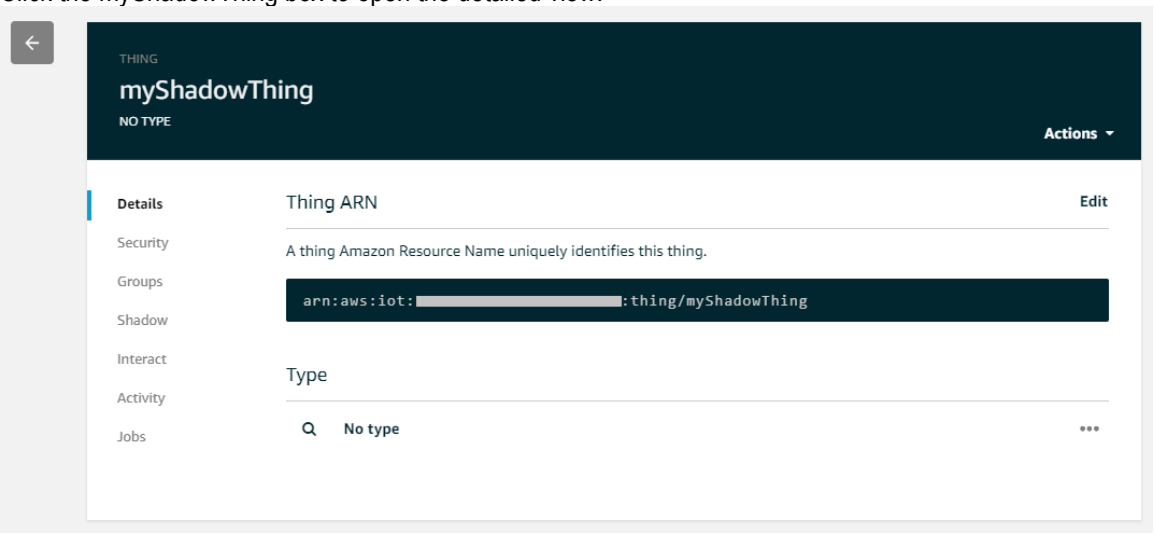
- FreeRTOS replaced with the setup from the SDK's shadow_wifi_qspi_xip example
- The wifi module on the IoT board is enabled and used. SSID and password must be configured in amazon-freertos/include/aws_clientcredential.h for it to work
- A connection to AWS must be configured according to the NXP/Amazon instructions which will generate the amazon-freertos/include/aws_clientcredential_keys.h file that is needed to compile this project.
- The led_accel_task_entry task no longer echoes typed characters and is mainly focused on detecting button clicks
- The epd_task_entry task also detects changes in a "message" that can be set from the AWS and pushed to the device. This message will be shown in the lower left corner of the ePaper display.
- The aws_shadow_lightbulb_on_off.c file has been modified to subscribe to changes from both the DataStore and AWS. The button presses come from the DataStore and are published in AWS. The message to display on the ePaper comes from AWS and is published in the DataStore.

To see the "shadow" of the IoT module online:

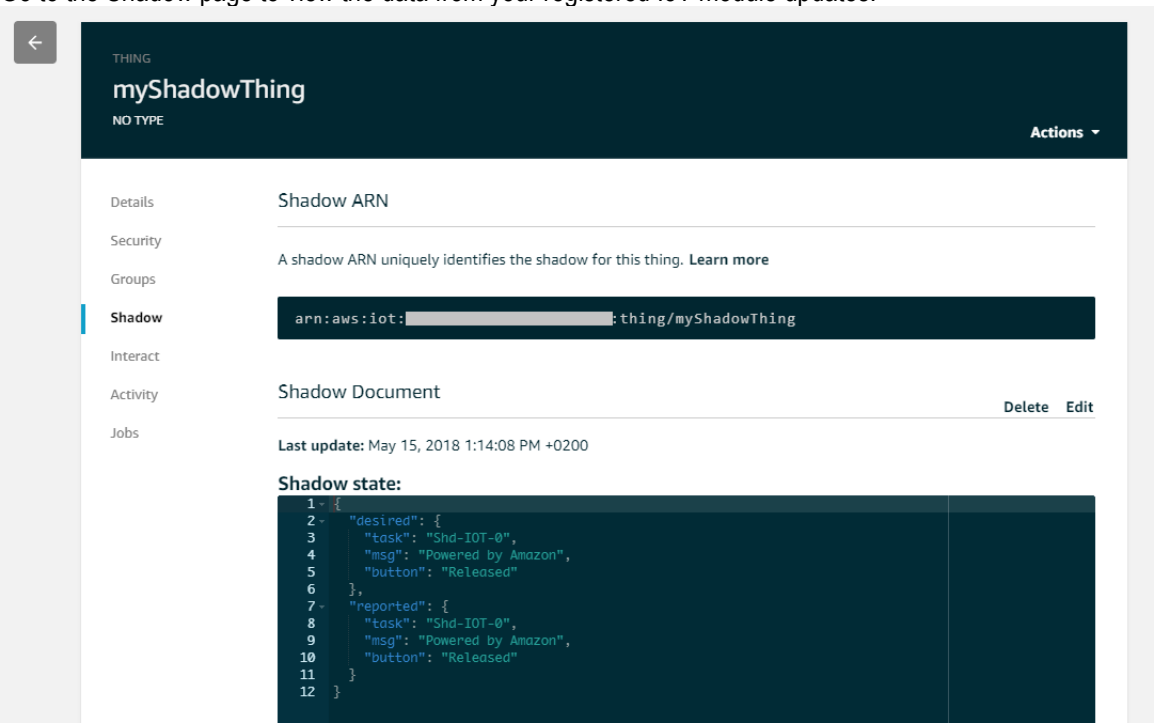
1. Login to AWS
2. Navigate to the AWS IoT page
3. Click Manage and then Things



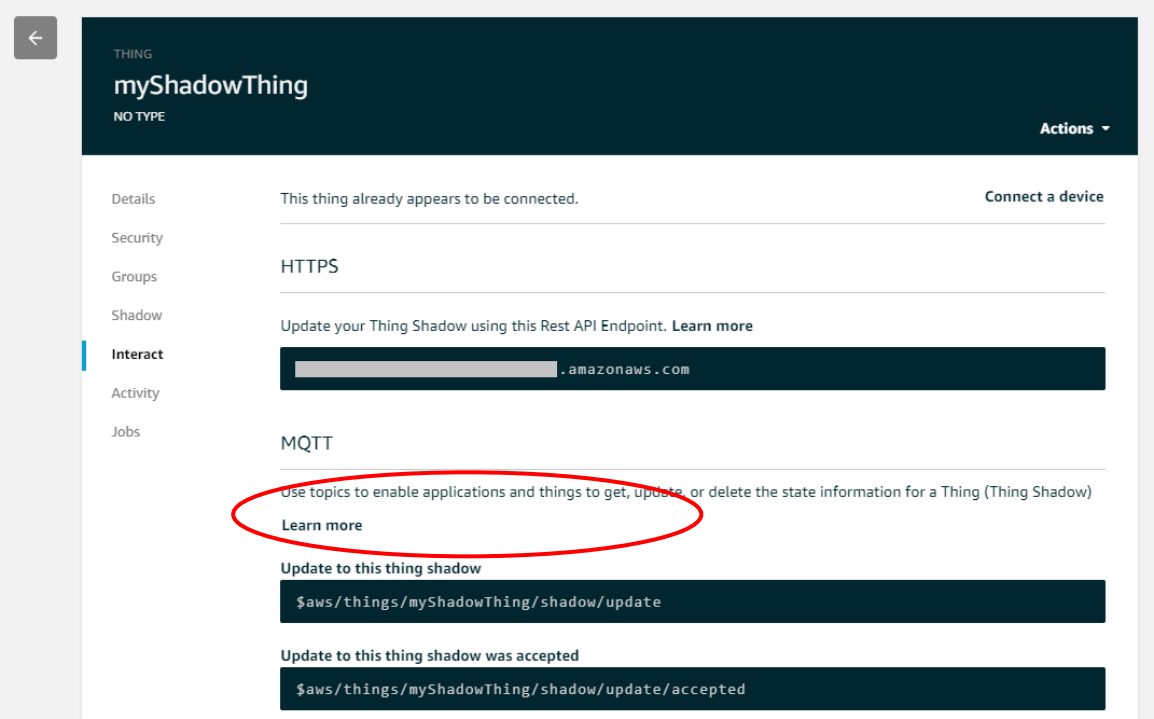
4. Click the myShadowThing box to open the detailed view:



5. Go to the Shadow page to view the data from your registered IoT module updates:

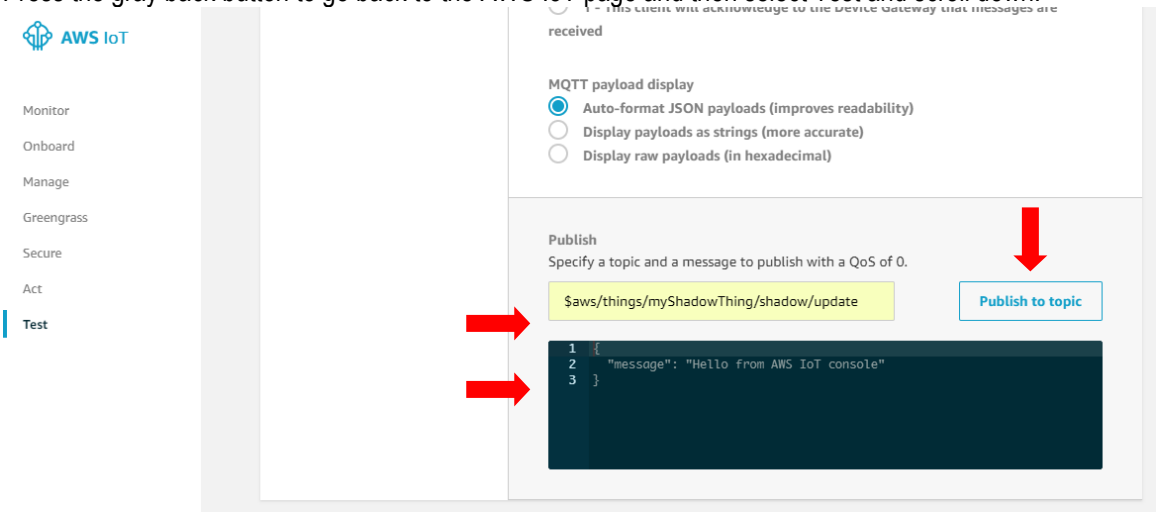


6. Go to the Interact page to see some different URLs to use when communicating with the Thing. To send a string to the IoT module we need the update URL:



The URL is $aws/things/myShadowThing/shadow/update

7. Press the gray back button to go back to the AWS IoT page and then select Test and scroll down:



8. Write the update url in the In the Topic field and then copy-paste the following into the black area:

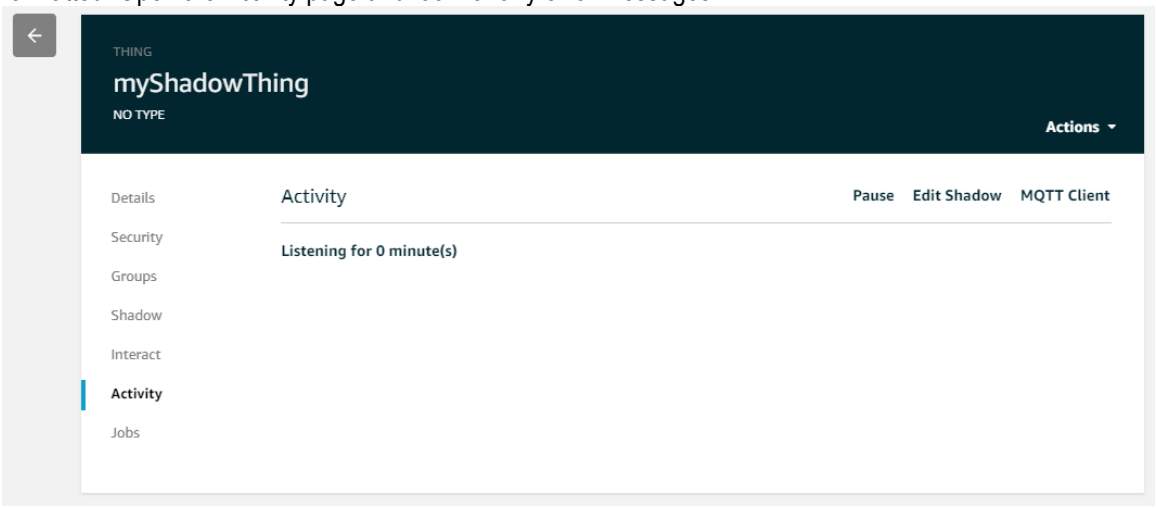
```
{
   "state": {
      "desired" : {
         "msg" : "Hello World!"
      }
   }
}
```

9. Press the Publish to topic.

10. The message is now saved in AWS and the IoT device will detect the change and start updating the ePaper display.

Some notes.

- You can have the Test view open in one browser tab and the Shadow view in another. That way it is easy to see the occurring changes
- If nothing happens as you press publish it can be because the message you sent was not correctly formatted. Open the Activity page and look for any error messages



- Pressing the User Button on the IoT Mini Prototype Board should result in an update on the Shadow page showing that the button is pressed.

6 AWS CLI

The AWS CLI is an open source tool built on top of the AWS SDK for Python (Boto) that provides commands for interacting with AWS services. With minimal configuration, you can start using all of the functionality provided by the AWS Management Console from your favorite terminal program.

6.1 Install

Instructions can be found here: https://docs.aws.amazon.com/cli/latest/userguide/installing.html but if you already have python installed then the aws cli can be installed with:

pip install awscli

Verify that it has been installed by running:

$ aws --version

aws-cli/1.15.22 Python/2.7.13 Windows/7 botocore/1.10.22

6.2 Configure

The aws cli tool can be used it must be configured with the security credentials to use. Instructions here: https://docs.aws.amazon.com/cli/latest/userguide/cli-chap-getting-started.html

6.3 Test

To list all available "Things":

$ aws iot list-things

```
{
    "things": [
        {
            "thingArn": "arn:aws:iot:us-west-2:------:thing/myShadowThing",
            "version": 1,
            "thingName": "myShadowThing",
            "attributes": {}
        }
    ]
}
```

To get the current shadow for a "Thing":

$ aws iot-data get-thing-shadow --thing-name myShadowThing outfile

The result is a file named outfile that contains the json data. With some formatting it looks like this:


```
{
	"state":{
		"desired":{
			"task":"Shd-IOT-0",
			"msg":"Hello World!",
			"button":"Released"
		},
		"reported":{
			"task":"Shd-IOT-0",
			"msg":"Hello World!",
			"button":"Released"
		}
	},
	"metadata":{
		"desired":{
			"task":{
				"timestamp":1526565090
			},
			"msg":{
				"timestamp":1526565090
			},
			"button":{
				"timestamp":1526565090
			}
		},
		"reported":{
			"task":{
				"timestamp":1526562748
			},
			"msg":{
				"timestamp":1526562996
			},
			"button":{
				"timestamp":1526563785
			}
		}
	},
	"version":2709,
	"timestamp":1526565090
}
```


To update the shadow with a new message ("Hello again!"):

$ aws iot-data update-thing-shadow --thing-name myShadowThing --payload "{\"state\": {\"desired\": {\"msg\": \"Hello again!\" }}}" outfile2