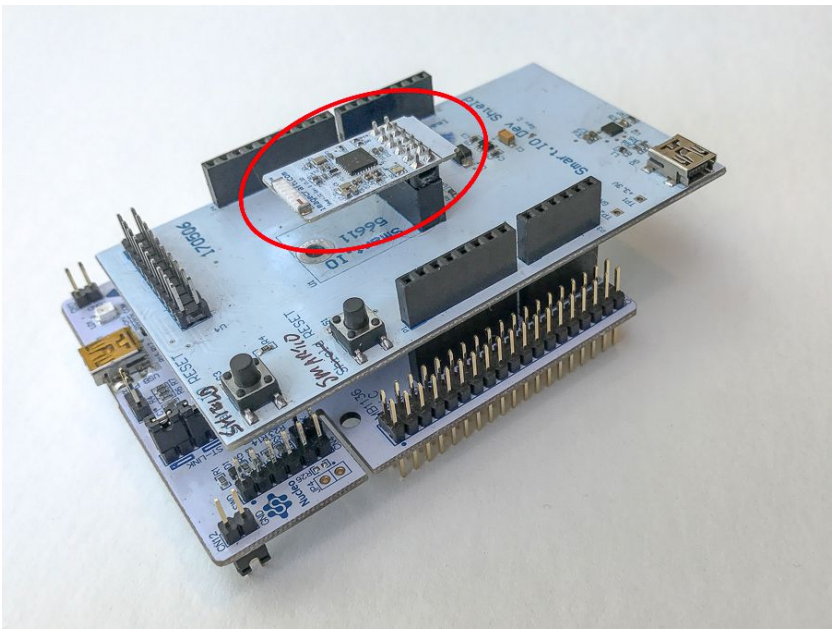
Evaluating Smart.IO using the Smart.IO Starter Kit

Richard Man, richard@imagecraft.com

The Smart.IO Starter kit is an easy way to evaluate the Smart.IO technology. It contains:

1. A Smart.IO module (denoted by the red ellipse in the photo below)
2. An Arduino compatible shield (the board in the middle where the Smart.IO module is sitting on)
3. An ST-Nucleo-F411 MCU board (the bottom board)



Note: only the Smart.IO module is an essential part of using Smart.IO. The Arduino shield and the ST-Nucleo are not necessary for using Smart.IO, and are part of the evaluation kit only. In an embedded user application, they may use any MCU of their choice, needing only to incorporate the Smart.IO module into their hardware designs.

The ST-Nucleo board is pre-programmed with test firmware. It is best to use two terminal connections to the starter kit. While the ST-Nucleo could have been programmed with a simple test program that does not involve interactions through the terminal, this is the best way to demonstrate the flexibility of the Smart.IO technology.

Setting up the terminal programs

There is one USB port on the Arduino shield, and another one on the ST-Nucleo. Drivers should be automatically installed by the OS. In case of any issues, the ST-Nucleo driver can be found on http://st.com (search for “ST-Nucleo USB driver”). The shield’s USB port uses the FTDI serial-to-USB chip and its drivers can be found on http://ftdichip.com.

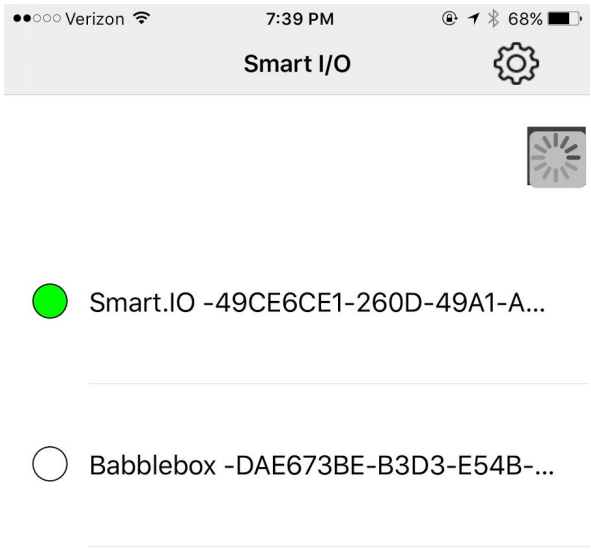
MacOS comes with built-in terminal program, and Windows users may use the “PuTTY” terminal program. Set the baudrate to 9600.

The ST-Nucleo is running an interactive program which allows you to type commands. The Smart.IO firmware is using the terminal (through the USB port on the Arduino shield) for diagnostic message display only.

Startup operations

1. Connect the USB cables to the starter kit and invoke the terminal programs. The kit gets its power from the USB cables.
2. Enable BLE (Bluetooth) access on the smartphone.
3. Invoke the Smart.IO app on the smartphone.
4. Reset the starter kit by clicking the “Shield Reset” button (note that the prototype board has the labels mis-printed, and the “top” button is the shield reset button).
5. If the terminal programs have been set up correctly, you should see sign-on messages from both boards.
6. You should see a BLE device scanning page on the Smart.IO app
7. Tap the one labeled “Smart.IO 49CE…” to connect
8. Wait until you see “BLE connected” message on the ST-Nucleo terminal window before proceeding.
9. If you do not see the “BLE connected” message after 10 to 15 seconds, click on the gear icon on the app and select “Rescan devices”, and then restart from Step 4 and reset the kit.

BLE Device scanning page



Sign-on message on the ST-Nucleo terminal window

```
ImageCraft STM32F411 ST-Nucleo... System running at 84Mhz
Command driver for Smart.IO 1.01
> BLE connected
```

Sign-on message from the Arduino shield+Smart.IO module

```
Smart.IO (version: 1.01)
GATT Init succeeded
GAP Init succeeded
Handshaking Service added. Handle 0x000C, Handshaking Characteristic handle: (W)
 0x000D (R) 0x0010
Set General Discoverable Mode.
aci_gap_set_discoverable() --> SUCCESS
aci_gap_update_adv_data() --> SUCCESS
Sent Hello
OK Sent. Ready for commands
```

Running the test program

Once connected, hit RETURN on the ST-Nucleo terminal, and you should see a prompt “> “ at which you may type in a command followed by carriage return. For the purposes of this demo, all commands are in the form of “128 <number>” where <number> is 0 to 8[1].

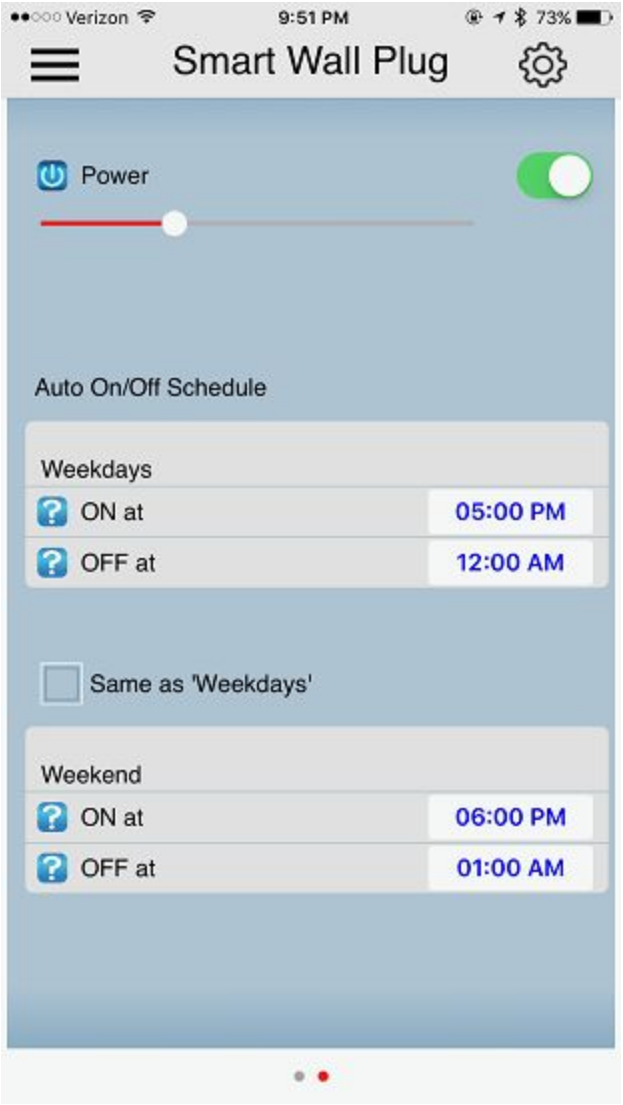
[1] Future revisions of the test program may allow more test samples and the allowable number may increase.

The test programs are summarized in the following table [2]. The most complex one is “128 4”. You may either run the commands with or without resetting the boards and restarting the process. If you do not, most tests create a new page with sample UI, and you can swipe sideways to access different pages.

Test Command	**Description**
128 0	Creates a “user application menu”
128 1	Demonstrates a textbox with font control
128 2	Smart Wall Plug UI, slightly simplified
128 3	Demonstrates checkboxes, radio buttons, and multiline textbox
128 4	Smart Wall Plug UI
128 5	Sample UI for a battery charger
128 6	UI for a Pokemon panel
128 7	Shows two text entry boxes for text input
128 8	Demonstrate non-UI Smart.IO API: generating random number, unique ID, and EEPROM storage

[2] Again, this is subject to future revisions

Running “128 4” should produce a screen looking like this:



Changing the sliders or the time values send the updated values to the ST-Nucleo test program.

Where to find downloads

The main page for Smart.IO is here: https://imagecraft.com/smartio/

The software download is here: https://imagecraft.com/download/smart-io-downloads

The documentation is here: https://imagecraft.com/documentation/smart-io-documentation

Modifying the test program

The source code of the test program and the “Host Interface Layer” (see the “Quick Start Guide to Smart.IO” in the documentation page) are available on the download page. To modify the test

program as is, you will need the JumpStart C for Cortex compiler (Windows only), available on https://imagecraft.com/download/demo-software. The demo is fully functional for 45 days.

To use a different compiler, or to port the test program and Smart.IO API to a different MCU, please refer to the document “Host Interface Layer” located on the documentation page.