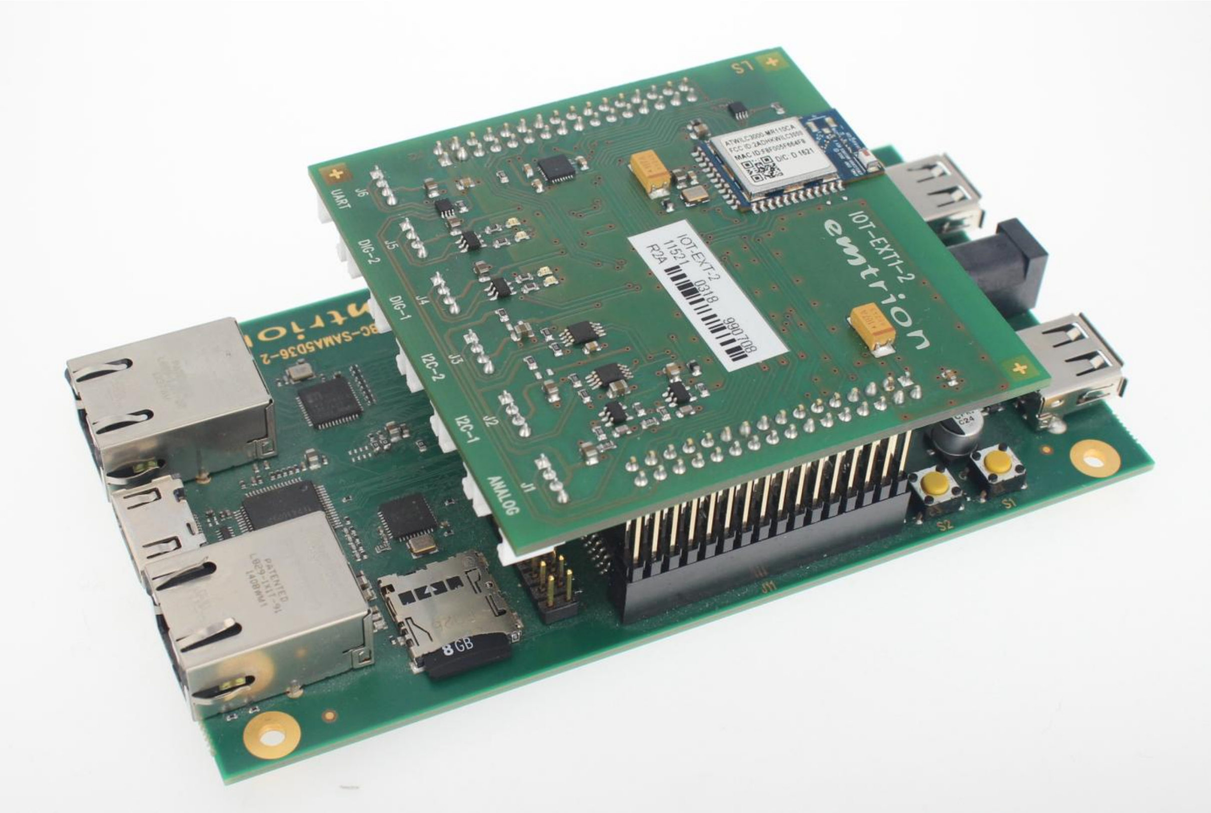
SBC-SAMA5D36 IOT_EXT Developer Kit

Software Manual

Rev02 / 27.03.2018



emtrion GmbH

Revision: **02 / 27.03.2018**

Rev	Date/Signature	Changes
1	20.03.2018/Ha	Initial release
2	27.03.2018/Ha	Add overview of the Devkit

1 Table of contents

Revision: **02 / 27.03.2018**.....2
2 Introduction.....4
3 The IoT Extension of the SBC-SAMA5D36.....5
4 Preconfigured image.....6
4.1 Device Network Setup.....7
4.1.1 Ethernet connection.....7
4.1.2 Wi-Fi connection.....7
4.2 IoT: MQTT protocol test example.....7
5 Buildroot build system.....8
5.1 Build the root file system.....8
5.2 Install the emtrion BSP.....8
5.3 Modify and Build your image.....9
5.4 Install and run the new image.....9
6 U-Boot.....10
6.1 Basic U-Boot operation.....10
6.2 Using U-Boot to change boot device or update parts of the system.....10
6.2.1 Booting.....10

2 Introduction

Welcome to SBC-SAMA5D36 IOT_EXT documentation. This manual will give you a startup software guideline of our developer kit. It will describe how to use the preconfigured image together with the IoT Extention. Further it will give you a quick introduction on how to build and modify your own RootFS using Buildroot, and to conclude a quick view of the bootloader.

It is assumed that users of emtrion Linux developer kits are already familiar with Linux. General Linux and programming knowledge are out of the scope of this document. emtrion will gladly assist you in acquiring this knowledge. If you are interested in training courses or getting support, please contact the emtrion sales department.

The examples in this manual are demonstrated on specific hardware but if not mentioned otherwise they all work on all supported emtrion devices.

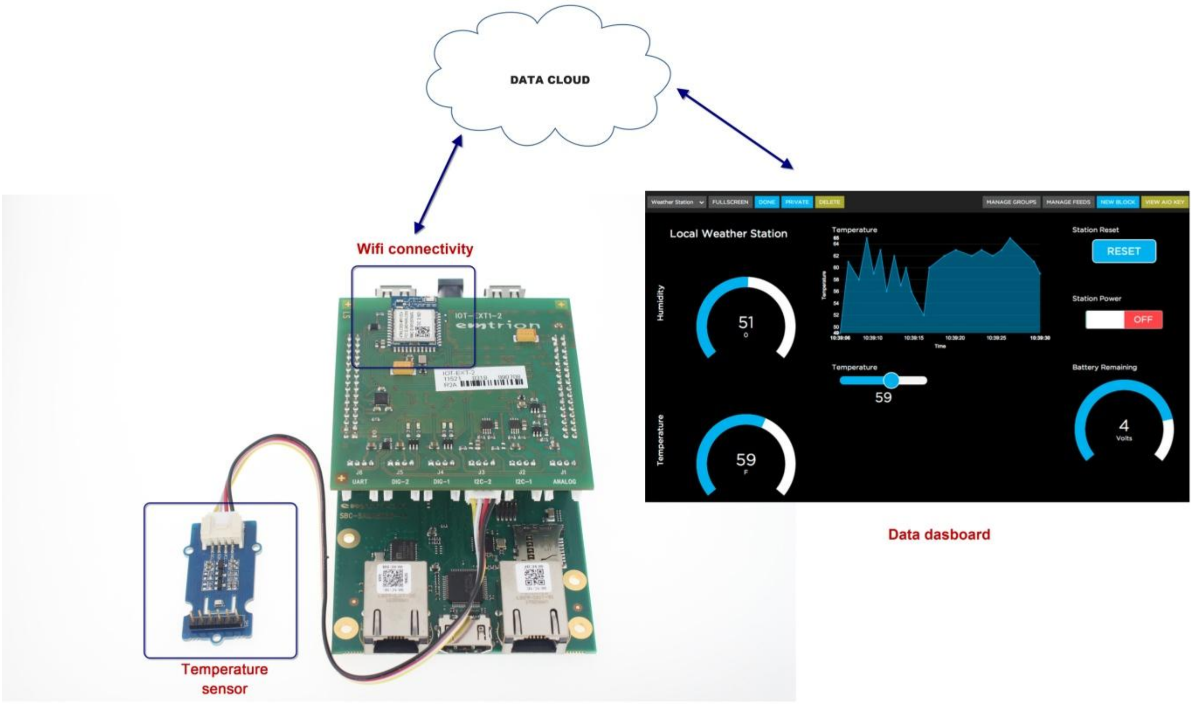
Please refer to the “Hardware Description” of IOT_EXT available on the emtrion support website (http://support.emtrion.de) for more detailed info of the capability of the product.

3 The IoT Extension of the SBC-SAMA5D36

The purpose of this product is to provide a highly customizable solution for the Internet of Things development. This solution provides not only a ready to use wireless connectivity with WiFi/Bluetooth but also connections to the Grove sensor system from the company Seeed. Those sensors are very wide in terms of variety and can be easily purchased to many distributors.

The preconfigured image available for free on the emtrion support website is preloaded with an IoT example. This example is a very simple demonstration that sends temperature data over a Data Cloud using MQTT protocol (http://mqtt.org/). Afterwards, the data can be seen on a dashboard on any device connected to the Internet with a web browser after logging to the Data Cloud web interface.

The Linux version running the system is the kernel v4.9 from linux4sam (https://github.com/linux4sam/linux-at91/tree/linux-4.9-at91)



In order to get the Preconfigured demo up and running you simply need to:

- Flash the preconfigured image

- Setup the WiFi connectivity with your credential

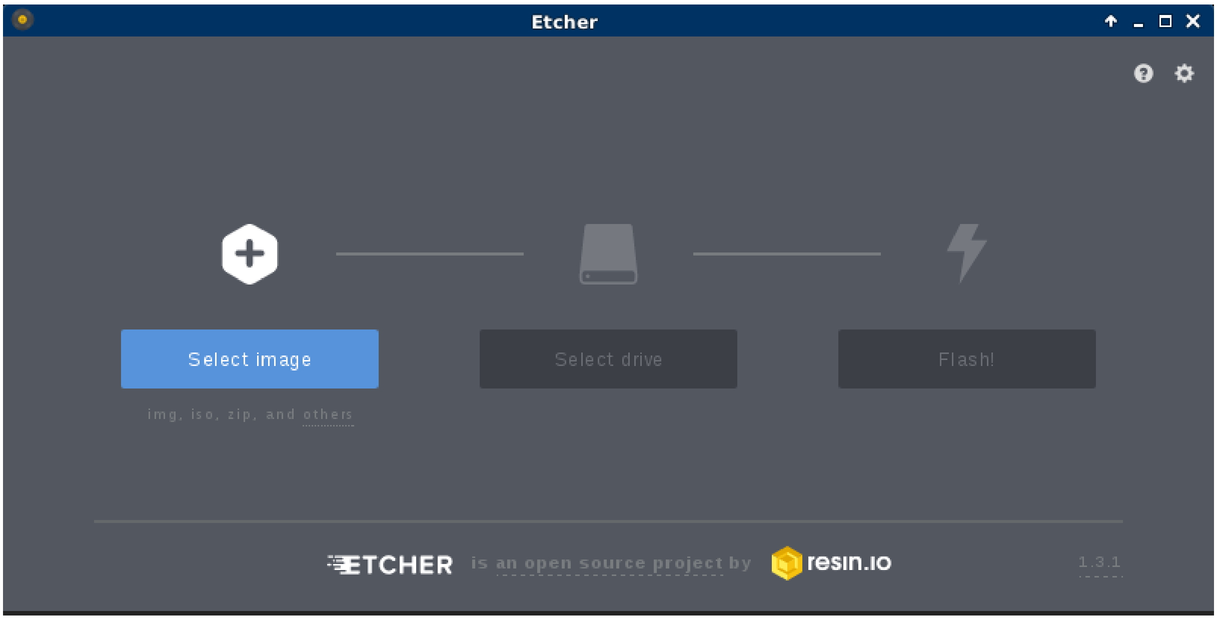
- Setup the MQTT agent with your credential

All information about those steps are explained in this document.

4 Preconfigured image

You can download the latest preconfigured image from our support pages (http://support.emtrion.de) and burn into a micro SD card (4GBytes minimum) using etcher tool (www.etcher.io) for example or using dd command line in Linux.

Etcher tool:


Etcher
Select image
img, iso, zip, and others
Select drive
Flash!
ETCHER is an open source project by resin.io
1.3.1

Linux dd command line example with /dev/sdX your micro SD card device:

```
hico@halinux:~/work/SAMA5D2x$ sudo dd if=emtrion.img of=/dev/sdX bs=512
```

Once your SD card ready you can plug it in your SBC-SAMA5D36 micro SD card socket.

You may plug the FTDI USB/Serial to your computer in order to get the Terminal output.

You may now power on the developer kit. You should see it booting U-Boot and Linux from Flash.

After the developer kit booted you are prompted for login:

- user: **root**
- password: **hico**

4.1 Device Network Setup

4.1.1 Ethernet connection

If your network offers a DHCP server the developer kit can be triggered to use it by using following command in a terminal:

```
#dhcpcd eth0
```

You can check the result with the command **ifconfig**.

4.1.2 Wi-Fi connection

In the rootFS of your device there is a script file provided as example to set-up quickly your Wi-Fi.

The example is for a WPA/WPA2 WiFi access with DHCP server example.

You may want to edit the file **wlan-setup.sh** on target (located in /root) or directly on your computer with the rootFS of your micro SD card mounted, and change to your SSID and your PASSWORD:

Change YOUR_SSID in this line:

```
Wpa_cli -p/var/run/wpa_supplicant set_network 0 ssid '"YOUR_SSID"'
```

Change YOUR_PASSWORD in this line:

```
wpa_cli -p/var/run/wpa_supplicant set_network 0 psk '"YOUR_PASSWORD"'
```

Then run the script:

```
#./wlan-setup.sh
```

Once finished, you can check the result with the command "**ifconfig wlan0**".

4.2 IoT: MQTT protocol test example

The rootFS is provided with an example of MQTT protocol from the company Adafruit (https://io.adafruit.com/). The example is free of charge and can be tested for free as well after a simple registration on the website. Any support related to this can be found in this website https://io.adafruit.com/support .

/!\ The software is provided "as is", without warranty, for test purpose only.

In the device, the folder of the example is located here: **/root/io-client-python-master**

In this folder, you may found **mqtt-test.py** , a simple python script that will run the following test: connect to the Adafruit MQTT broker, read the sensor data, sent it to the cloud and sleep for few seconds. This process is running in loop.

Don’t forget to that you need to change ADAFRUIT_IO_KEY, ADAFRUIT_IO_USERNAME, ADAFRUIT_MQTT_KEY values in the script to your own credential!

5 Buildroot build system

5.1 Build the root file system

The Linux system running on our device was created using the embedded Linux tool called Buildroot (www.buildroot.org).

The Buildroot user manual is available online:

https://buildroot.org/downloads/manual/manual.html

Or in PDF format:

https://buildroot.org/downloads/manual/manual.pdf

The best way to get this build system is to go the following page (https://buildroot.org/download.html) and clone the repository:

```
hico@halinux:~/work/SAMA5D3x$ git clone git://git.buildroot.net/buildroot
Cloning into 'buildroot'...
remote: Counting objects: 278679, done.
remote: Compressing objects: 100% (87607/87607), done.
remote: Total 278679 (delta 194250), reused 273184 (delta 189912)
Receiving objects: 100% (278679/278679), 59.81 MiB | 5.25 MiB/s, done.
Resolving deltas: 100% (194250/194250), done.
Checking connectivity... done.
hico@halinux:~/work/SAMA5D3x$
```

5.2 Install the emtrion BSP

Once you have retrieved the buildroot source code, you can add the emtrion BSP by extracting the archive into the board directory:

```
hico@halinux:~/work/SAMA5D2x/buildroot/board$ tar zxvf emtrion-
buildroot-bsp.tar.gz
emtrion/
emtrion/sbc-sama5d36-iot-ext/
emtrion/sbc-sama5d36-iot-ext/patches/
emtrion/sbc-sama5d36-iot-ext/patches/linux/
emtrion/sbc-sama5d36-iot-ext/patches/linux/0001-sbc-sama5d36-
wilc1000-change-pin-to-IOT_EXT.patch
emtrion/sbc-sama5d36-iot-ext/linux_iot_ext_defconfig
emtrion/sbc-sama5d36-iot-ext/rootfs_overlay/
emtrion/sbc-sama5d36-iot-ext/rootfs_overlay/etc/
emtrion/sbc-sama5d36-iot-ext/rootfs_overlay/etc/ssh/
… (more lines)
emtrion/sbc-sama5d36-iot-ext/rootfs_overlay/root/io-client-python-
master/tests/test_errors.py
```

```
emtrion/sbc-sama5d36-iot-ext/rootfs_overlay/root/wlan-setup.sh
emtrion/sbc-sama5d36-iot-ext/buildroot_iot_ext_defconfig
emtrion/sbc-sama5d36-iot-ext/sbc-sama5d36.dts
hico@halinux:~/work/SAMA5D2x/buildroot/board$
```

5.3 Modify and Build your image

Once the emtrion BSP folder is ready, you can build simply by copying the defconfig into the root folder of Buildroot, and run make menuconfig and make !

```
hico@halinux:~/work/SAMA5D2x/buildroot$ cp board/emtrion/sbc-
sama5d36-iot-ext/buildroot_iot_ext_defconfig .config
hico@halinux:~/work/SAMA5D2x/buildroot$ make menuconfig
hico@halinux:~/work/SAMA5D2x/buildroot$ make
```

Note that the first build will last approximately 45minutes to 1hour depending of your internet bandwidth and your computer capacity.

The next build will be way quicker thanks to the incremental build way of Buildroot.

5.4 Install and run the new image

Once built, your image is in the ouput/image/ folder of the Buildroot directory.
You can extract it directly into your RootFS folder:

```
hico@halinux:~/work/SAMA5D2x/buildroot/output/image/$ sudo tar xvf
rootfs.tar -C /your/rootfs/folder/
```

6 U-Boot

The basic task of U-Boot is to load the operating system from bulk memory into RAM and then start the kernel. You can also use it to initiate an update of the kernel, the root file system and of U-Boot itself. Furthermore you can configure directly from the medium the operating system is to be booted from, for example MMC, NFS or a serial terminal.

6.1 Basic U-Boot operation

To work with U-Boot, first use a terminal program to connect to the serial line of the board. As soon as the U-Boot prompt appears in the terminal, U-Boot is ready to receive commands. To interrupt the automatic boot mechanism press a keyboard key while using the terminal. The general U-Boot documentation can be found here: http://www.denx.de/wiki/U-Boot/Documentation

U-Boot has a set of environment variables which are used to store information needed for booting the operating system. Variables can contain information such as IP addresses, but they can also contain a whole script of actions to perform sequentially. The following commands explain the basic handling of environment variables:

U-Boot command	**Explanation**
printenv [variable]	This shows the value of the specified variable. If no variable is specified, the whole environment is shown.
setenv [variable] [value]	Set a variable to a specific value. If no value is specified, the variable gets deleted.
saveenv	Make your changes permanent, so they remain after power off or reboot.

6.2 Using U-Boot to change boot device or update parts of the system

This chapter describes how U-Boot has to be setup for updating and booting. The exact variable assignment is board specific. The variable serverip has to be set with the IP address of the virtual machine. You can get the IP address by entering **sudo ifconfig** in the terminal of your virtual machine.

Take the IP address of the corresponding network adapter and assign it to the variable serverip in the U-Boot console. The format of [IP address] is dot decimal notation.

```
U-Boot # setenv serverip [IP address]
```

6.2.1 Booting

The default boot device in U-Boot is determined by the variable “bootcmd”. If you want to set up one of the following boot options as a default you have to set “bootcmd” to the command mentioned below.

Boot from on-board flash

This is the default boot option configured when you receive the developer kit from emtrion. To start it manually simply use this command:

```
U-Boot # boot
```

Boot via network using a NFS share

Make sure you exported a directory via NFS in your virtual machine. Of course the directory must contain an extracted Root File System of the Debian BSP for the board. Updated Root File System tar balls can be downloaded from our support pages (http://support.emtrion.de):

```
U-Boot # setenv serverip        [ip-address of virtual machine]
U-Boot # setenv nfsroot         [path to the root file system]
U-Boot # setenv ip-method       [dhcp or static]
U-Boor # setenv ipaddr [ip-address for device, only needed for static ip]
U-Boot # setenv netmask [netmask for device, only needed for static ip]
U-Boot # saveenv
U-Boot # run net_boot
```

Now the board should boot via network using the NFS share in the virtual machine.