Emcraft Systems STM32F4 SOM Starter Kit Guide

Release 1.11.0

Table of Contents

1. Overview

This document is the Emcraft Systems STM32F4 SOM Starter Kit Guide, Release 1.11.0.

The BSP provides a software development environment for evaluation and development of Linux on the Cortex-M4 processor core of the STMicroelectronics STM32F4 microcontroller using the Emcraft Systems STM32F4 SOM board in harness with the Emcraft Systems SOM-BSB-EXT baseboard as a hardware platform.

This BSP is provided as part of the Emcraft Systems STM32F4 SOM (System-On-Module) Starter Kit. The kit provides a hardware platform and Linux software development environment for the STM32F4 SOM (System-On-Module).

2. Product Contents

This product includes the following components.

2.1. Shippable Hardware Items

The following hardware items are shipped to customers of this product:

1. STM32F4 SOM board;
2. SOM-BSB-EXT baseboard;
3. Mini-USB cable UART/power interface.

Note that this product does not include any JTAG programmer tools or associated hardware items. Such equipment needs to be purchased directly from respective vendors.

2.2. Downloadable Hardware Materials

The following hardware materials are available for download from Emcraft's web site to customers of this product:

1. `SOM-BSB-EXT-1A-schem.pdf` - SOM-BSB-EXT schematics in PDF format;
2. `SOM-BSB-EXT-1A-bom.xls` - SOM-BSB-EXT Bill-Of-Materials (BOM) in Excel format;
3. `SOM-BSB-EXT-1A_Dimensions.pdf` - SOM-BSB-EXT mechanical drawing in PDF format.

2.3. Downloadable Software Materials

The following software materials are available for download from Emcraft's web site to customers of this product:

1. `u-boot.bin` - prebuilt U-Boot file in the format suitable for installation into embedded Flash of Cortex-M4 on the STM32F4 SOM board;
2. `networking.uImage` - prebuilt Linux image ready to be loaded to the STM32F4 SOM board;
3. `linux-STM32F4-1.11.0.tar.bz2` - Linux STM32F4 software development environment, including:
   a) U-Boot firmware;
   b) Linux kernel;
   c) `busybox` and other target components;
   d) Linux-hosted cross-development environment;
   e) Framework for developing multiple projects (embedded applications) from a single installation, including sample projects allowing to kick-start software development for Linux STM32F4.

2.4. Downloadable Documentation Materials

The following documentation materials are available for download from Emcraft's web site to customers of this product:

1. `linux-cortexm-um-1.11.0.pdf` - Linux Cortex-M User's Manual;
2. `STM-SOM-skg-1.11.0.pdf` - Emcraft Systems STM32F4 SOM Starter Kit Guide (this document).

3. Software Functionality

3.1. Supported Features

The following list summarizes the features and capabilities of Linux STM32F4, Release 1.11.0:

- U-Boot firmware:
  - U-Boot v2010.03;
  - Target initialization from power-on / reset;
  - Runs from the internal eNVM and internal SRAM (no external memory required for standalone operation);
  - Serial console;
  - Ethernet driver for loading images to the target;
  - Serial driver for loading images to the target;
  - Device driver for built-in Flash (eNVM) and self-upgrade capability;
  - Device driver for storing environment and Linux images in external Flash;
  - Autoboot feature, allowing boot of OS images from Flash or other storage with no operator intervention;
  - Persistent environment in Flash for customization of target operation;
  - Sophisticated command interface for maintenance and development of the target.
- Linux:
  - uClinux kernel v2.6.33;
  - Boot from compressed and uncompressed images;
  - Ability to run critical kernel code from integrated Flash of STM32F4;
  - Serial device driver and Linux console;
  - Ethernet device driver and networking (`ping`, NFS, Telnet, FTP, `ntpd`, etc.);
  - `busybox` v1.17;
  - POSIX pthreads;
  - Process-to-kernel and process-to-process protection using the Memory Protection Unit (MPU) of the STM32F4 core;
  - Hardened exception handling; an exception triggered by a process affects only the offending process;
  - Loadable kernel modules;
  - Support for the hardware FPU;
  - Secure shell (`ssh`) daemon;
  - Web server;
  - MTD-based Flash partitioning and persistent JFFS2 Flash file system for external Flash;

- Device driver for the DMA interface;
- SD Card device driver;
- I^2C device driver;
- RTC device driver;
- SPI controller master-mode device driver;
- GPIO device driver.

- Development tools:
  - ARMv7-optimized GNU toolchain from CodeSourcery (`2010q1`) is used for development of U-Boot, Linux and user-space applications (toolchain must be downloaded separately from the CodeSourcery web site);
  - Cross GDB for debugging user-space applications;
  - `mkimage` tool used by the Linux kernel build process to create a Linux image bootable by U-Boot.
- Development environment:
  - Linux-hosted cross-development environment;
  - Development of multiple projects (embedded applications) from a single installation;
  - `hello` sample project ("Hello, world!" single-process configuration);
  - `networking` sample project (basic shell, networking and Flash management tools demonstration);
  - `developer` sample project (template project that can be used to jump-start development of custom user-space applications and loadable kernel modules).

3.2. New and Changed Features

This section lists new and changed features of this release:

1. Develop an I^2C device driver for Linux STM32.
   ID: RT 86120.

3.3. Known Problems & Limitations

This section lists known problems and limitations of this release:

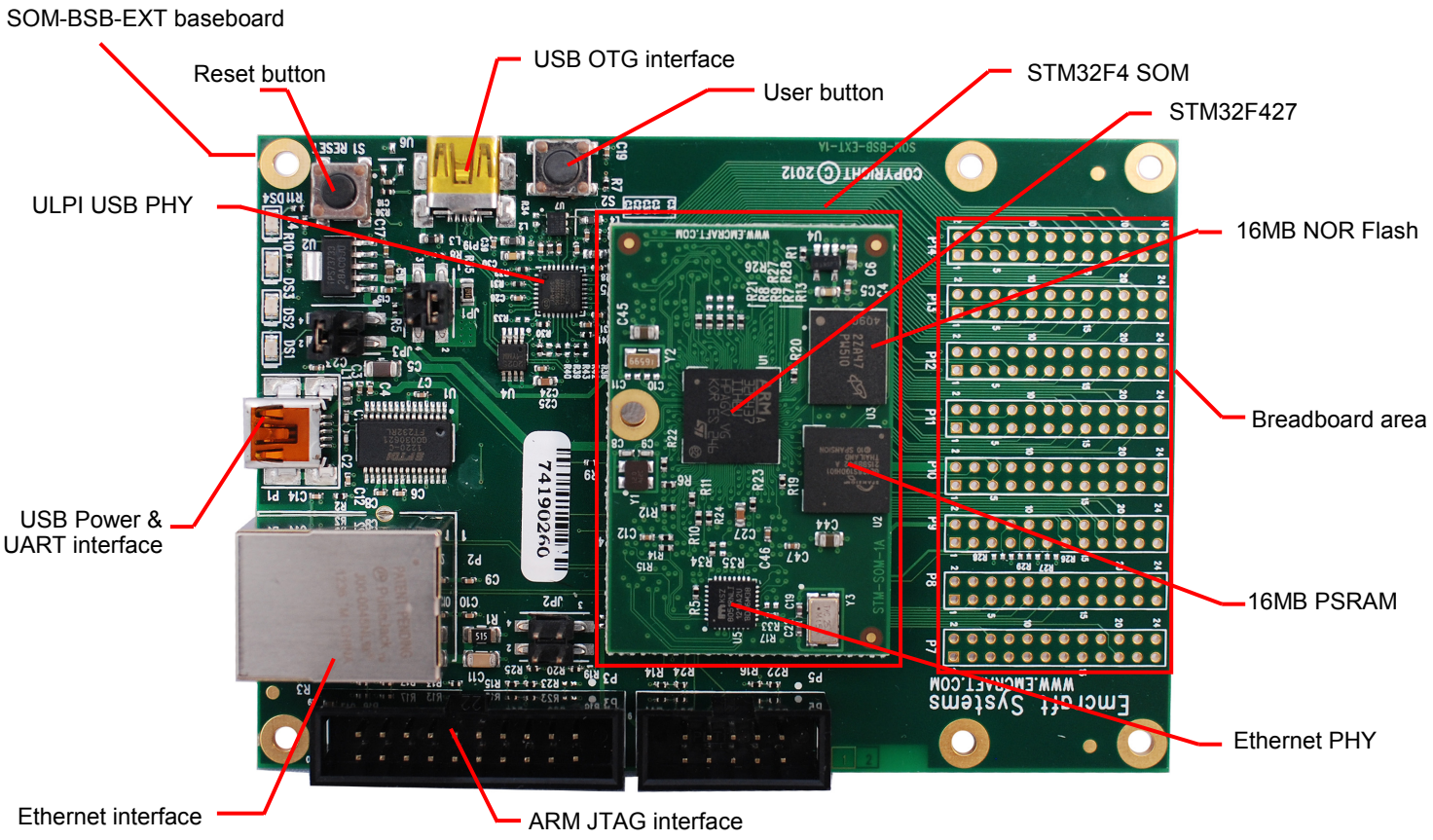
1. `CONFIG_KERNEL_IN_ENVM` requires disabling `CONFIG_ARM_UNWIND` and `CONFIG_EARLY_PRINTK`.
   ID: RT 74683.
   Workaround: When enabling `CONFIG_KERNEL_IN_ENVM` in the kernel, disable `CONFIG_ARM_UNWIND` and `CONFIG_EARLY_PRINTK`.

4. Hardware Setup

This section explains how to set up the Emcraft Systems STM32F4 SOM board in harness with the Emcraft Systems SOM-BSB-EXT baseboard.

4.1. Hardware Interfaces

The STM32F4 SOM board in harness with the Emcraft Systems SOM-BSB-EXT baseboard provides the following components and interfaces:



4.2. Jumpers

The following jumpers must be configured on the SOM-BSB-EXT board:

Jumper	Configuration	Notes
JP1	1-2 closed, 3-4 open	Enable power on the STM SOM (VCC3)
JP3	1-3 open, 2-4 closed	Use the mini-USB port as the power source

4.3. Board Connections

To power the SOM-BSB-EXT baseboard with the STM32F4 SOM up, simply connect it to a PC / notebook by plugging a mini-USB cable into the P1 mini-USB connector on the SOM-BSB-EXT board. As soon as the connection to the PC has been made, the LED `DS2` should lit up, indicating that the board is up and running.

A single USB connection provides a 500 mA power to the STM32F4 SOM, which is sufficient for basic functionality. Note however that some advanced operations, such as WiFi connectivity using the USB WiFi module, may require more than 500 mA for reliable operation. Use the second link of the mini-USB Y-cable to connect to the PC for such configurations.

On the PC side, the USB link provides a serial console device to the STM32F4 SOM. The software installed on the board is configured for a 115.2 K terminal. On the Linux host, the serial console is available using a `/dev/ttyUSB`*n* device.

To provide network connectivity to the board, connect it into your LAN by plugging a standard Ethernet cable into the J1 connector. The board is pre-configured with an IP address of 192.168.0.2.

4.4. Extension Interfaces

For description of the extension interfaces provided by the Emcraft Systems STM32F4 SOM board refer to `Emcraft Systems STM32F4 SOM (System-On-Module) Hardware Architecture`.

For description of the extension interfaces provided by the Emcraft Systems SOM-BSB-EXT baseboard refer to `Emcraft Systems SOM-BSB-EXT Baseboard Hardware Architecture`.

The above mentioned documents can be downloaded from the following page:

http://www.emcraft.com/som/stm32f4

5. STM32F4 SOM Board Linux Software Set-up

5.1. U-Boot Environment

When the STM32F4 SOM board is reset, the Linux bootstrap will proceed to boot the U-Boot firmware from the built-in Flash printing the following output to the serial console:

```
U-Boot 2010.03-linux-cortexm-1.11.0 (July 5 2013 - 13:17:48)

CPU  : STM32F4 (Cortex-M4)
Freqs: SYSCLK=168MHz,HCLK=168MHz,PCLK1=42MHz,PCLK2=84MHz
Board: STM-SOM Rev Rev 1.A, www.emcraft.com
DRAM:  16 MB
Flash: 16 MB
In:    serial
Out:   serial
Err:   serial
Net:   STM32 MAC
Hit any key to stop autoboot:  0
STM-SOM>
```

U-boot makes use of the so-called environment variables to define various aspects of the system functionality. Parameters defined by the U-boot environment variables include: target IP address, target MAC address, address in RAM where a Linux bootable images will be loaded, and many more. To examine the current settings of the environment variables, run `printenv` from the U-Boot command interface.

U-Boot provides a command called `saveenv` that stores the up-to-date run-time environment to the persistent storage, which will be the external Flash for the U-Boot configuration used on the STM32F4 SOM board. You need to call `saveenv` any time when you want to copy current settings of the environment variables to the persistent storage in Flash. This is how you can write the current U-Boot environment to the external Flash:

```
STM-SOM> saveenv
Saving Environment to Flash...
...
STM-SOM>
```

5.2. Ethernet MAC Address

In Linux STM32F4, the MAC address of the Ethernet interface is defined by the `ethaddr` U-Boot environment variable. The value of the MAC address can be examined from the U-Boot command line monitor as follows:

```
STM-SOM> printenv ethaddr
ethaddr=C0:B1:3C:88:88:88
STM-SOM>
```

The STM32F4 SOM board comes with `ethaddr` set to a MAC address uniquely allocated for the specific board. Given that each STM32F4 SOM board has a unique MAC address allocated to it, there is no need to update the `ethaddr` variable (although it is possible to do so).

The MAC address can be changed by modifying the `ethaddr` variable as follows:

```
STM-SOM> setenv ethaddr C0:B1:3C:88:88:89
```

Don't forget to store your update in the persistent storage using `saveenv` so it is remembered across resets and power cycles.

5.3. Network Configuration

You will have to update the network configuration of your board to match settings of your local environment.

Typically, all you have to allow loading images over network from a TFTP server is update the U-Boot environment variables `ipaddr` (the board IP address) and `serverip` (the IP address of the TFTP server). Here is how it is done.

Update `ipaddr` and `serverip`:

```
STM-SOM> setenv ipaddr 192.168.0.2
STM-SOM> setenv serverip 192.168.0.1
```

and then save the updated environment to the external Flash so that your changes are persistent across resets/power cycles.

5.4. Running Pre-installed Linux Image

The STM32F4 SOM board comes with a Linux bootable image for the `networking` project installed into external Flash. To boot this Linux configuration onto the STM32F4 SOM board just reset the board and let U-Boot perform the autoboot sequence.

Detailed information on functionality of the pre-installed Linux image can be found in *Linux Cortex-M User's Manual*, Section 3.

5.5. Loading Linux Images

At this point, you are able to load Linux bootable images to the board over TFTP and either boot them directly or install them to the external Flash to allow booting Linux from Flash on power-up/reset.

On the host, activate the Linux STM32F4 development environment and build the `networking` project:

```
-bash-3.2$ . ACTIVATE.sh
-bash-3.2$ cd projects/networking/
-bash-3.2$ make
...
-bash-3.2$
```

Copy the Linux bootable image to the TFTP download directory:

```
-bash-3.2$ cp networking.uImage /tftpboot/vlad/
-bash-3.2$
```

To load the image directly, use the `netboot` U-Boot macro:

```
STM-SOM> setenv image vlad/networking.uImage
STM-SOM> run netboot
...
TFTP from server 172.17.0.1; our IP address is 172.17.5.100
Filename 'vlad/networking.uImage'.
...
Loading: #################################################################
         #################################################################
         #############
done
Bytes transferred = 2084704 (1fcf60 hex)
...
   Image Name:   Linux-2.6.33-arm1
```

```
   Image Type:   ARM Linux Kernel Image (uncompressed)
...
   Verifying Checksum ... OK
   Loading Kernel Image ... OK
OK

Starting kernel ...

Linux version 2.6.33-arm1 (vlad@ocean.emcraft.com) (gcc version 4.4.1 (Sourcery G++ Lite
2010q1-189) ) #1 Fri July 5 15:43:44 MSK 2013
...
```

To load the image into the Flash, use the U-Boot `update` macro:

```
STM-SOM> setenv image vlad/networking.uImage
STM-SOM> run update
...
TFTP from server 172.17.0.1; our IP address is 172.17.5.100
Filename 'vlad/networking.uImage'.
...
Loading: ################################################################
         ################################################################
         #############
done
Bytes transferred = 2084704 (1fcf60 hex)
................................ done
Un-Protected 32 sectors

................................ done
Erased 32 sectors
Copy to Flash... done
STM-SOM>
```

Reset the board and verify that the newly programmed image boots on the target in the autoboot mode:

```
STM-SOM> reset
resetting ...

U-Boot 2010.03-linux-cortexm-1.11.0 (July 5 2013 - 17:19:37)
...
Starting kernel ...
...
init started: BusyBox v1.17.0 (July 5 2013 - 17:19:37)
~ #
```

5.6. U-Boot Build

The BSP distribution comes with U-Boot pre-built for the STM32F4 SOM board. If however you need to re-build U-Boot for your board, please follow the instructions below:

1. Install the Linux STM32F4 distribution to the development host, as described in the Linux Cortex-M User's Manual.
2. From the top of the Linux STM32F4 installation, activate the Linux STM32F4 cross-compile environment by running `. ACTIVATE.sh`.
3. Go to the U-Boot source directory (`cd u-boot/`).
4. Run the following commands:

```
[psl@pvr u-boot]$ make stm-som config
Configuring for stm-som board...
[psl@pvr u-boot]$ make -s
```

5.7. U-Boot Installation

The Emcraft Systems STM32F4 SOM board arrives with the U-Boot firmware pre-installed into the on-chip Flash of the STM32F4. The U-Boot command line interface provides commands that allow upgrading U-Boot on the running target in self-upgrade mode.

However, should you program a faulty U-Boot image into STM32F4, U-Boot can be re-installed using an ST-LINK programmer device (not included in the kit) and the STMicroelectronics ST-LINK/V2 software. Please follow the procedure described below:

1. On the Windows development host download and install the ST-LINK/V2 software from the web site (`Design Support` in http://www.st.com/internet/evalboard/product/251168.jsp). The following software and documents are required:
2. Driver: `ST-LINK/V2 USB driver for Windows 7, Vista and XP`;
3. ST-LINK utility: `STM32 ST-LINK utility`;
4. ST-LINK User Manual: `UM0892: STM32 ST-LINK Utility`.
5. Connect an ST-LINK programmer device to the Windows development host over the USB cable and to the SOM-BSB-EXT board over the flat ribbon cable;
6. Power on the SOM-BSB-EXT board by attaching the mini-USB cable to the mini-USB connector P1. On the PC side, the USB link provides a serial console device to the board (`/dev/ttyUSB`n on Linux, `COM`n on Windows).;
7. On the Windows development host run the ST-LINK utility:
   a) Connect to the target the `Target -> Connect` menu;
   b) Open the `Target -> Option Bytes...` menu and select the `WDG_SW`, `nRST_STDBY` and `nRST_STOP` check-boxes, if not already selected;
   c) Program the SOM-BSB-EXT board as described in Section 3.4, "Device Programming" of the *ST-LINK Utility User Manual*. File to program is `u-boot.bin`, the start address is the default 0x08000000;
8. Run a terminal program (e.g. `HyperTerminal` on Windows, or `kermit` on Linux) and create a serial connection to the SOM-BSB-EXT board with the following COM-port settings: `115200 8N1`. The terminal device ;
9. Reset the SOM-BSB-EXT board and see the U-Boot start-up banner in the terminal program.

6. Further Materials

Refer to *Emcraft Systems STM32F4 SOM (System-On-Module) Hardware Architecture* for detailed information on the hardware architecture of the Emcraft Systems STM32F4 SOM board.

Refer to *Emcraft Systems SOM-BSB-EXT Baseboard Hardware Architecture* for detailed information on the hardware architecture of the Emcraft Systems SOM-BSB-EXT baseboard.

Refer to Linux *Cortex-M User's Manual* for detailed information on the software architecture of the Linux STM32F4 distribution.

Visit Emcraft Systems' web site at www.emcraft.com to obtain additional materials related to Linux STM32F4.

7. Support

We appreciate your review of our product and welcome any and all feedback. Comments can be sent directly by email to:

a2f-linux-support@emcraft.com

The following level of support is included with your purchase of this product:

- Email support for installation, configuration and basic use scenarios of the product during 3 months since the product purchase;
- Free upgrade to new releases of the downloadable materials included in the product during 3 months since the product purchase.

If you require support beyond of what is described above, we will be happy to provide it using resources of our contract development team. Please contact us for details.