Linux SmartFusion

BSP (Board Support Package) Guide for the Emcraft Systems SmartFusion SOM Board

Release 1.8.0

Table of Contents

1. Overview

This document is a Linux SmartFusion BSP (Board Support Package) Guide for the Emcraft Systems SmartFusion SOM (System-On-Module) board, Release 1.8.0.

The BSP provides a software development environment for evaluation and development of Linux on the Cortex-M3 processor core of the Microsemi SmartFusion microcontroller using the Emcraft Systems SmartFusion SOM board in harness with the Emcraft Systems SOM-BSB baseboard as a hardware platform.

This BSP is provided as part of the Emcraft Systems SmartFusion SOM (System-On-Module) Starter Kit. The kit provides a hardware platform and Linux software development environment for the SmartFusion 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. SmartFusion SOM board;
2. SOM-BSB baseboard;
3. Mini-USB cable.

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-1A-schem.pdf` - SOM-BSB schematics in PDF format;
2. `SOM-BSB-1A-bom.xls` - SOM-BSB Bill-Of-Materials (BOM) in Excel format;
3. A2F-SOM.IntLib - Altium Designer 9.4 integrated library for the A2F500-SOM-FG484 symbol and footprint.

2.3. Downloadable Software Materials

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

1. `a2f-som-1a.pdb` - Libero `.pdb` file with the U-Boot image embedded, ready for installation onto the SmartFusion SOM board using the Microsemi FlashPro tool;
2. `a2f-som-1a.zip` - Libero project file corresponding to the `pdb` file in the item above;
3. `networking.uImage` - prebuilt kernel image ready to be loaded to the SmartFusion SOM board;
4. `linux-A2F-1.8.0.tar.bz2` - Linux SmartFusion 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 SmartFusion.

2.4. Downloadable Documentation Materials

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

1. `a2f-som-ha.pdf` - Emcraft Systems SmartFusion SOM (System-On-Module) Hardware Architecture;
2. `a2f-som-bsb-ha.pdf` - Emcraft Systems SmartFusion SOM SOM-BSB Hardware Architecture;
3. `linux-cortexm-um-1.8.0.pdf` - Linux Cortex-M User's Manual;
4. `linux-A2F-SOM-bspg-1.8.0.pdf` - Linux SmartFusion BSP (Board Support Package) Guide for the Emcraft Systems SmartFusion SOM Board (this document).

3. Software Functionality

3.1. Supported Features

The following list summarizes the features and capabilities of Linux SmartFusion, Release 1.8.0:

- 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 SmartFusion;
  - 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 SmartFusion core;

  - Hardened exception handling; an exception triggered by a process affects only the offending process;
  - Loadable kernel modules;
  - Secure shell (`ssh`) daemon;
  - Web server;
  - MTD-based Flash partitioning and persistent JFFS2 Flash file system for external Flash;
  - SPI controller master-mode device driver;
  - Device driver for the embedded NVM;
  - Framebuffer device driver for the low-cost SPI-based LCD monitor (Nokia6100 LCD);
  - Serial device driver for CoreUARTapb.
- 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. Enable tickless kernel (`CONFIG_NO_HZ`).
   ID: RT 79812.

3.3. Known Problems & Limitations

This section lists known problems and limitations of this release:

1. NFS-mounts without an `-o rsize=1024` option result in "Frame CRC errors" reported by the Linux Ethernet driver.
   ID: RT 62655.
   Workaround: Use an `-o rsize=1024` option when NFS-mounting remote directories. Example:

```
mount -o nolock,rsize=1024 <ip>:<remote_dir> <mount_point>
```

2. SPI driver doesn't work on A2F500.
   ID: RT 71635.
   Workaround: None.
3. `CONFIG_KERNEL_IN_ENVM` requires disabling `CONFIG_ARM_UNWIND` and `CONFIG_EARLY_PRINTK`.
   ID: RT 74683.

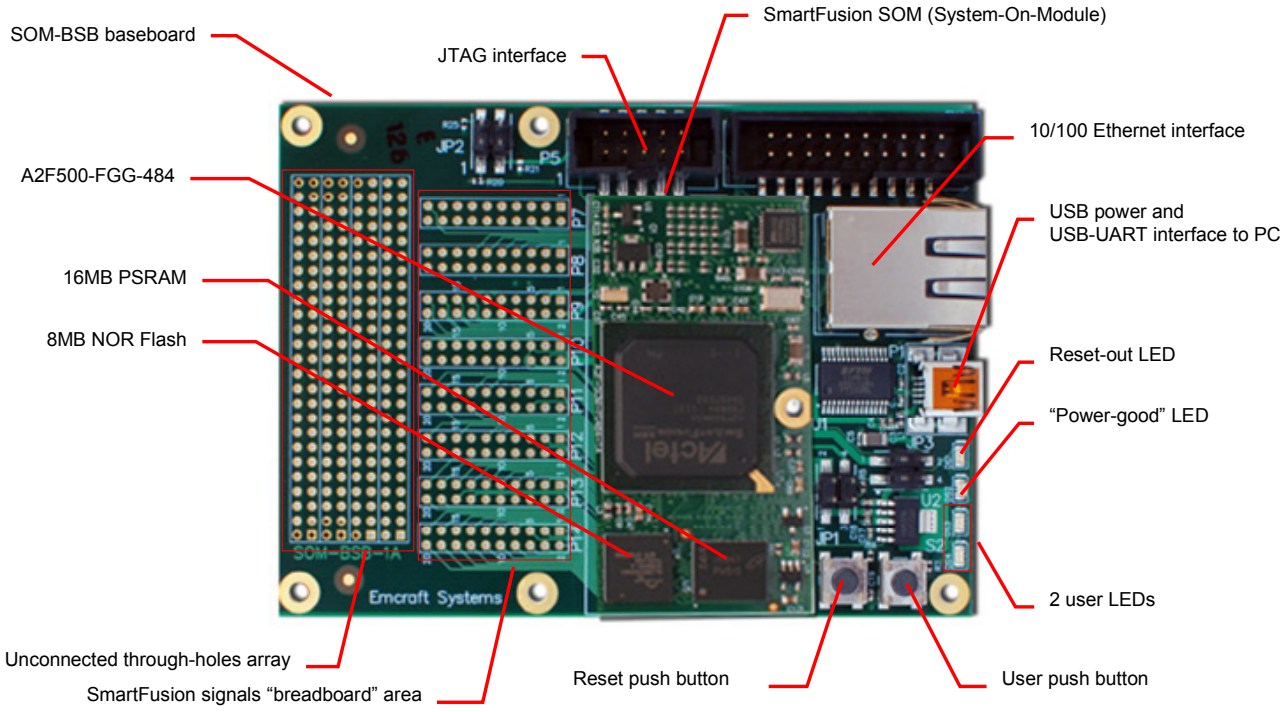
Workaround: When enabling `CONFIG_KERNEL_IN_ENVM` in the kernel, disable `CONFIG_ARM_UNWIND` and `CONFIG_EARLY_PRINTK`.

4. Hardware Setup

This section explains how to set up the Emcraft Systems SmartFusion SOM board in harness with the Emcraft Systems SOM-BSB baseboard in such a way as to allow running uClinux on this hardware platform.

4.1. Hardware Interfaces

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



4.2. Jumpers

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

Jumper	Configuration	Notes
JP1	1-2 closed, 3-4 closed	To enable power on the SmartFusion SOM (VCC3) and save the battery life when the mini-USB is connected
JP2	1-2 open, 3-4 closed	To select the necessary JTAG mode and enable power on the JTAG controller
JP3	1-3 open, 2-4 closed	To use the mini-USB port as the power source

4.3. Board Connections

To power the SOM-BSB baseboard with the SmartFusion 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 board. As soon as the connection to the PC has been made, the on-board LED `DS2` should lit, indicating that the board is up and running.

On the PC side, the USB link provides a serial console device to the board. The Linux SmartFusion software installed on the board is configured for a 115.2Kb terminal. On the Linux host, the serial console is available using a `/dev/ttyUSBn` device.

To provide network connectivity to the board, connect it into your LAN by plugging a standard Ethernet cable into the 10/100 Ethernet connector.

4.4. Extension Interfaces

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

For description of the extension interfaces provided by the Emcraft Systems SmartFusion SOM SOM-BSB baseboard refer to *Emcraft Systems SOM-BSB Hardware Architecture*.

5. SmartFusion SOM Board Software Set-up

5.1. U-Boot Environment

When the SmartFusion SOM board is reset, U-Boot comes up from the built-in Flash printing the following output to the serial console:

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 SmartFusion SOM board.

This is how you can write the current U-Boot environment to the external Flash:

```
A2F-SOM> saveenv
Saving Environment to Flash...
...
A2F-SOM>
```

5.2. Ethernet MAC Address

In Linux SmartFusion, 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:

```
A2F-SOM> printenv ethaddr
ethaddr=C0:B1:3C:88:88:88
A2F-SOM>
```

The default U-Boot environment for the SmartFusion SOM board sets `ethaddr` to a fixed MAC address. This address should work for you in a general case, however if you have more than two SmartFusion SOM boards in your LAN, use of the same address on multiple boards may result in packet collisions in your LAN and overall may render your LAN mal-functioning.

If you have more than one SmartFusion SOM boards in your LAN, you have to assign a unique MAC address to each board.

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

```
A2F-SOM> setenv ethaddr C0:B1:3C:88:88:89
```

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

```
A2F-SOM> saveenv
Saving Environment to Flash...
...
```

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`:

```
A2F-SOM> setenv ipaddr 192.168.0.2
A2F-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:

```
A2F-SOM> saveenv
Saving Environment to Flash...
...
A2F-SOM>
```

5.4. Installation of Linux Images to Flash

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 in the auto-boot mode.

On the host, activate the Linux SmartFusion 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:

```
A2F-SOM> setenv image vlad/networking.uImage
A2F-SOM> run netboot
Auto-negotiation...completed.
Core10/100: link UP (100/Full)
Using Core10/100 device
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 Mon Mar 12 15:43:44 MSK 2012
```

```
...
```

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

```
A2F-SOM> setenv image vlad/networking.uImage
A2F-SOM> run update
Auto-negotiation...completed.
Core10/100: link UP (100/Full)
Using Core10/100 device
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
A2F-SOM>
```

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

```
A2F-SOM> reset
resetting ...

U-Boot 2010.03-linux-cortexm-1.8.0 (Sep 07 2012 - 17:19:37)
...
Starting kernel ...
...
init started: BusyBox v1.17.0 (Sep 07 2012 - 17:19:37)
~ #
```

5.5. U-Boot Build

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

1. Install the Linux SmartFusion distribution to the development host, as described in the Linux Cortex-M User's Manual.
2. From the top of the Linux SmartFusion installation, activate the Linux SmartFusion 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 a2f-som_config
Configuring for a2f-som board...
[psl@pvr u-boot]$ make -s
[psl@pvr u-boot]$ make -s u-boot.hex
```

5.6. U-Boot Installation

The Emcraft Systems SmartFusion SOM board arrives with the U-Boot firmware pre-installed into the on-chip Flash of the SmartFusion. 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 SmartFusion, U-Boot can be re-installed using the Emcraft-provided Linux SmartFusion Libero project and a Microsemi FlashPro tool (not included in the kit). Here is an example of how this can be done:

1. Start FlashPro on a Windows host;
2. From the FlashPro IDE, create a new project with an arbitrary name;
3. From the main FlashPro window, push `Configure Device`;
4. Push `Browse` next to `load existing programming file`. Browse to the Linux SmartFusion project file `a2f-som-1a.pdb` and choose it;
5. Push `Program` at the top of the main window to program the project onto the SmartFusion device and wait for the programming procedure to complete. If the programming completes successfully, a next reset should bring the U-Boot start-up messages and the command line interface onto the serial console interface.

6. FPGA Interfaces

6.1. FPGA IP Programming Interfaces

The Libero project included with the Linux SmartFusion distribution installs the following IP blocks to the FPGA fabric of the SmartFusion:

Address Range	IP	Tiles	Description	Mandatory/Optional
0x40050000-0x400500FF	CoreInterrupt_0	68*	Flexible interrupt controller for AMBA–Based Systems, supporting up to 4 IRQ sources	Optional - is required only if there are multiple FPGA slaves triggering interrupts to the MSS core
0x40050100-0x400501FF	VersionROM_0	20	Provides an APB register-based interface ROM used for storing FPGA design type and version information	Mandatory
0x40050200-0x400502FF	CoreGPIO_0	180**	Provides an APB register-based interface to up to 32 GPIO signals	Optional – is required only if the FPGA-based GPIOs are needed
0x40050300-0x400503FF	PSRAM_CR_0	470	Custom IP used to configure external PSRAM and to place it into Page Mode	Mandatory

** This is an approximate value for the Number of IRQ Sources = 4, taken from the CoreIterrupt datasheet. The exact number of used tiles may vary in different projects.*

*** - This is an approximate value for configuration of the CoreGPIO with number of used GPIOs = 3, based on the CoreGPIO datasheet. The exact number of used tiles may vary in different projects.*

6.2. FPGA Development in Linux SmartFusion

To facilitate development involving FPGA modifications, the `developer` project includes the `iap_tool` utility designed for run-time FPGA fabric upgrades of the SmartFusion. Such FPGA self-upgrades are performed on a running system and do not require additional hardware, such as a FlashPro3/4 programmer device.

The `iap_tool` utility resides in the `/bin` directory of the `developer` project's root filesystem. It can be invoked from the command line as follows:

```
~ # /bin/iap_tool
```

```
Program FPGA Array of the SmartFusion
Usage: /mnt/iap_tool [options] <programming_data_file>
Options:
    -h, --help
        display this help and exit
    -l, --lock
        specify the lock file to use (default is /var/run/iap_tool)
    -a, --action
        specify the IAP action to run (default is PROGRAM_ARRAY)
        available actions (case insensitive):
         1 DEVICE_INFO
         2 READ_IDCODE
         3 ERASE
         4 ERASE_ALL
         5 PROGRAM
         6 VERIFY
         7 ENC_DATA_AUTHENTICATION
         8 ERASE_ARRAY
         9 PROGRAM_ARRAY
        10 VERIFY_ARRAY
        11 ERASE_FROM
        12 PROGRAM_FROM
        13 VERIFY_FROM
        14 ERASE_SECURITY
        15 PROGRAM_SECURITY
        16 PROGRAM_NVM
        17 VERIFY_NVM
        18 VERIFY_DEVICE_INFO
        19 READ_USERCODE
        20 PROGRAM_NVM_ACTIVE
        21 VERIFY_NVM_ACTIVE
        22 IS_CORE_CONFIGURED
```

Behavior of the commands listed above is the same as implemented by the corresponding commands supported by the Actel FlashPro programming tool. Refer to:

http://www.actel.com/documents/flashpro_ug.pdf

for further details.

For example, to upgrade the FPGA array from a `file.dat` file the utility should be invoked as follows:

```
~ # /bin/iap_tool --action PROGRAM_ARRAY file.dat
```

As soon as the above command completes, the new image has been installed into the FPGA fabric and is running. No reset or Linux reboot is required.

To create a `.dat` file for FPGA upgrades using the `iap_tool` utility, load corresponding `.pdb` file into the Microsemi FlashPro application, select the `File -> Export -> Export Single Programming File` item of the main menu, choose the "DirectC File (*.dat)" in the "Output formats" list, type in the resulting file name and location, and press button "Export".

Note that the FPGA fabric is hold in reset during execution of IAP operations and cannot be accessed by software drivers. Due to this, it is strongly recommended to configure kernel drivers for FPGA-based controllers as modules and unload all such modules before running the `iap_tool` utility.

The `iap_tool` utility must be used with caution since some actions (e.g. `ERASE_ARRAY`) can render the A2F non-functional (in such cases, a FlashPro3/4 device will be required to restore the SmartFusion device to a functional state).

7. Further Materials

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

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

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

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

8. 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 6 months since the product purchase;
- Free upgrade to new releases of the downloadable materials included in the product during 6 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.