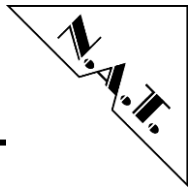
NAMC-ZYNQ-FMC
FMC AMC Module
Board Support Package Manual V1.1



N.A.T GmbH
Konrad-Zuse-Platz 9
53227 Bonn-Oberkassel

Phone: +49 / 228 / 96 58 64 – 0
Fax: +49 / 228 / 96 58 64 – 10

Internet: http://www.nateurope.com

Disclaimer

The following documentation, compiled by N.A.T GmbH (henceforth called N.A.T), represents the current status of the product´s development. The documentation is updated on a regular basis. Any changes which might ensue, including those necessitated by updated specifications, are considered in the latest version of this documentation. N.A.T is under no obligation to notify any person, organization, or institution of such changes or to make these changes public in any other way.

We must caution you, that this publication could include technical inaccuracies or typographical errors.

N.A.T offers no warranty, either expressed or implied, for the contents of this documentation or for the product described therein, including but not limited to the warranties of merchantability or the fitness of the product for any specific purpose.

In no event will N.A.T be liable for any loss of data or for errors in data utilization or processing resulting from the use of this product or the documentation. In particular, N.A.T will not be responsible for any direct or indirect damages (including lost profits, lost savings, delays or interruptions in the flow of business activities, including but not limited to, special, incidental, consequential, or other similar damages) arising out of the use of or inability to use this product or the associated documentation, even if N.A.T or any authorized N.A.T representative has been advised of the possibility of such damages.

The use of registered names, trademarks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations (patent laws, trade mark laws, etc.) and therefore free for general use. In no case does N.A.T guarantee that the information given in this documentation is free of such third-party rights.

Neither this documentation nor any part thereof may be copied, translated, or reduced to any electronic medium or machine form without the prior written consent from N.A.T GmbH.

This product (and the associated documentation) is governed by the N.A.T General Conditions and Terms of Delivery and Payment.

Table of Contents

List of Tables

List of Figures

Conventions

If not otherwise specified, addresses and memory maps are written in hexadecimal notation, identified by *0x*.
The following table gives a list of the abbreviations used in this document.

Abbreviation	Description
BSP	Board Support Package
FSBL	First Stage Bootloader
SoC	System on Chip
OS	Operating System
FMC	FPGA Mezzanine Card

1 Board Support Package Description

1.1 Introduction

The Board Support Package (BSP) is shipped with the NAMC-ZYNQ-FMC to support customers generating a complete linux distribution including FSBL, U-Boot and kernel image for the Xilinx ZYNQ FPGA SoC. The BSP ensures a basic working infrastructure including necessary communication and memory interfaces to support customers developing their own FMC drivers and applications on top of that.

The software part is based on the Petalinux build system that is officially supported by Xilinx. Before you can use the BSP you will have to install the necessary Xilinx tools and the N.A.T sources. This is going to be explained in the next chapters. Although N.A.T recommends using the BSP you are completely free whether to use your own software distributions. It is also possible to run a bare metal application without any OS on the SoC but there approaches are outside of the manuals scope.

For generating the necessary hardware description files you may use the N.A.T Vivado example project that is also included in the board support package. The example project contains a block diagram that customers could use as a template to build their logic on top of that. It ensures the basic hardware functionality such as standard communication interfaces.

Common information about the Petalinux build system can be found at "UG1144 Petalinux Tools Documentation Reference Guide" [2] or at Xilinx wiki:

http://www.wiki.xilinx.com/Petalinux

1.2 Prerequisite for using the BSP

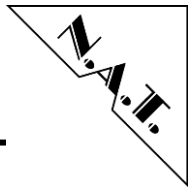
Before you can use the BSP you will have to set up a few things:

- A host system (linux or windows) with Vivado installed
- A host system (linux) containing the Petalinux toolchain and SDK

The N.A.T reference FPGA configuration image was created with Xilinx Vivado 2015.2.1 thus for best compatibility N.A.T recommends to use both the Petalinux 2015.2.1 build system and the corresponding Vivado version. Please follow the Xilinx Petalinux Documentation "UG1144 Petalinux Tools Reference Guide" on how to install the Xilinx tools on your host machine.

1.3 BSP structure

Referring to the table below, the board support package contains four folders. The bsp folder includes the software project containing the build output of the kernel image and other components. The image folder contains all images that are needed for the device to operate including the FPGA bitstream, u-boot and kernel binaries. The subsystem folder is used for holding user configuration (e.g. device-tree). In the /doc folder you can find any necessary documentation. The /src folder is intended to be used by the bsp for holding user application data and patch files in order for the software components to work with the hardware that were made by N.A.T. These files are patched by using the



makefile delivered with the bsp. The /vivado folder contains the hardware reference project that can be used as a template for bitstream generation. The overview of the folder structure is shown at the table below:

bsp		**Petalinux project folder (bsp=project name)**
>	build	*Object files from build process*
>	components	*Contains FSBL sources*
>	hw-description	*Contains FPGA configuration file*
>	images	*Contains images for booting the device*
>	subsystems	*Several configuration files (device tree, startup configs)*
doc		**Documentation**
src		**User defined sources**
>	patches	*Patch files*
>	components	*Linux-Kernel and U-boot sources*
>	apps	*User applications*
vivado		***Vivado project folder containing reference hardware design for NAMC-ZYNQ-FMC***

Table 1: BSP folder structure

1.4 Vivado reference design

The BSP has a folder /vivado witch contains the reference design for the ZYNQ FPGA that customers may use as a template for developing further logic on top of that. The design was created with Vivado 2015.2.1. Opening and compiling the design requires a license file for the ZYNQ FPGA device (xc7z045ffg900-2) that can be purchased at Xilinx store.

The design brings a block diagram (see figure below) with basic communication and memory interfaces, such as Ethernet, AXI PCIe and a DDR3 memory controller. There is also an instance that serves 100 MHz telecom clocks TCLKB, TCLKD to the backplane. To use that feature, just change the constant value from "0" to "1".
For interfacing with the FMC using SPI or I2C protocol you may use the FPGA EMIO to route these protocols directly into the ARM core. That will enable you to use the peripherals quickly within the standard linux drivers. All you need to do is to assign the FMC I/O pin to the IIC_1 and SPI_0 interface before implementing the design.

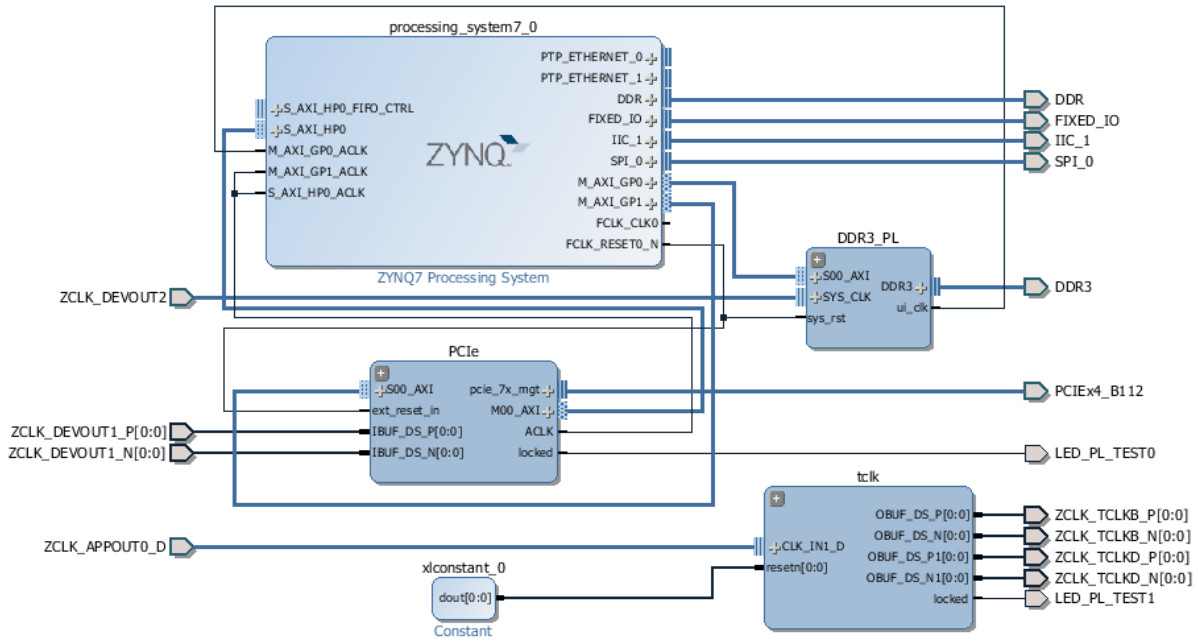


Figure 1: Vivado Block Diagram

Before you can use the design with the Petalinux build system you will have to export the hardware description file once you have implemented and generated a bitstream. To do so, select **"File->Export->Export Hardware"**.

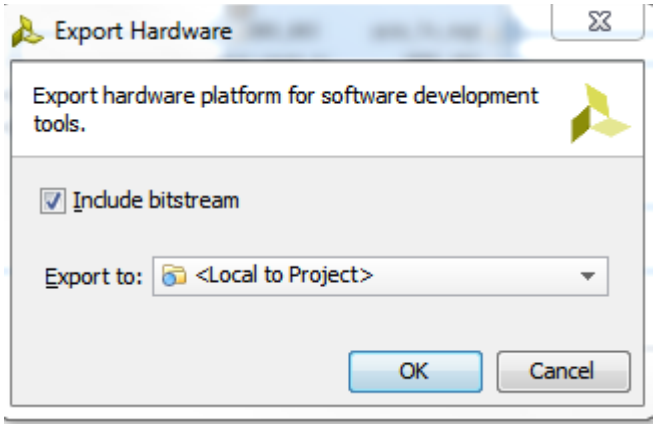
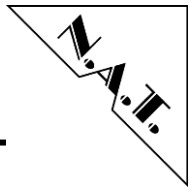


Figure 2: Export Hardware

This step is necessary to enable you to import the hardware description file with the Petalinux build system.

1.5 Installing the BSP

With having both Xilinx Vivado and Petalinux tools installed the next step is to install the BSP sources and set up several paths. First you have to copy the main BSP folder to any working directory (e.g. \home\myname\namc-zynq-fmc\). To make the installation easy N.A.T provides a makefile that does all of the jobs for you. To use the makefile you have to set up two environment variables that are noted at the first lines in the makefile. To edit the file chose any text editor of your choice. If your installation paths of Petalinux and Vivado are set to default you do not have to change anything.



- **PETALINUX_INSTALL = /opt/xilinx/petalinux-v2015.2.1**
- **VIVADO_INSTALL = /opt/xilinx/Vivado/2015.2**

When you are done with that navigate to the top level of your project folder and execute the makefile with:

- ***make import-components***

This command will copy the u-boot and linux-kernel sources that came with the Petalinux installation to your project into the directory /src/components/. This step is necessary as we want to leave the original sources untouched. The next thing to do is to patch the kernel and u-boot sources with the files located in /src/patch/. To do so just execute the makefile again with the following target:

- ***make patch-components***

The last thing to do is to create symbolic links at the place where the components are stored in the petalinux installation path. This is mandatory because the petalinux system uses this path as reference to find any components to be built. The symbolic link ensures that we will be able to select the patched sources located in the project folder later during the build system configuration. The link gets created using the makefile:

- ***make setup-links***

[NOTE] Depending on the owner of the petalinux installation it may be the case that you have to execute this command as a root user.

With these steps executed you are ready to build the first boot image. The next chapter will tell how to do that.

1.6 Building a boot image for ZYNQ

This chapter will instruct you how to build an executable build image for the NAMC-ZYNQ-FMC device. The first thing you have to do is to create a new petalinux project. To this with the makefile using:

- ***make create-project***

[NOTE] The default project name is "bsp". If you wish to create a project with a different name set the makefile environment "PROJECT_NAME".

The next step is export the hardware image from within the Vivado Project (see Chapter 1.4). Then you will be able import the hardware description from Vivado into the petalinux project. Simply do this using the makefile:

- ***make import-hw***

After importing is done the configuration for the build system should appear and you should be able to choose the components (u-boot and kernel) we have initially created with the makefile in the chapter before. See figures below as a reference:

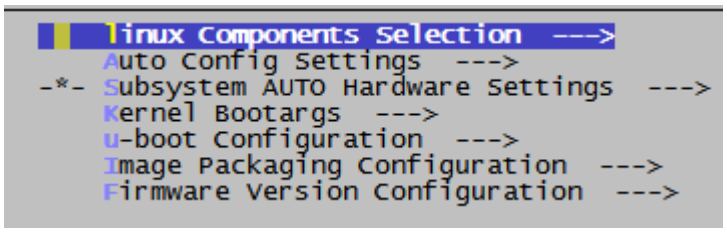


Figure 3: Petalinux main configuration menu

Figure 4: Petalinux components configuration

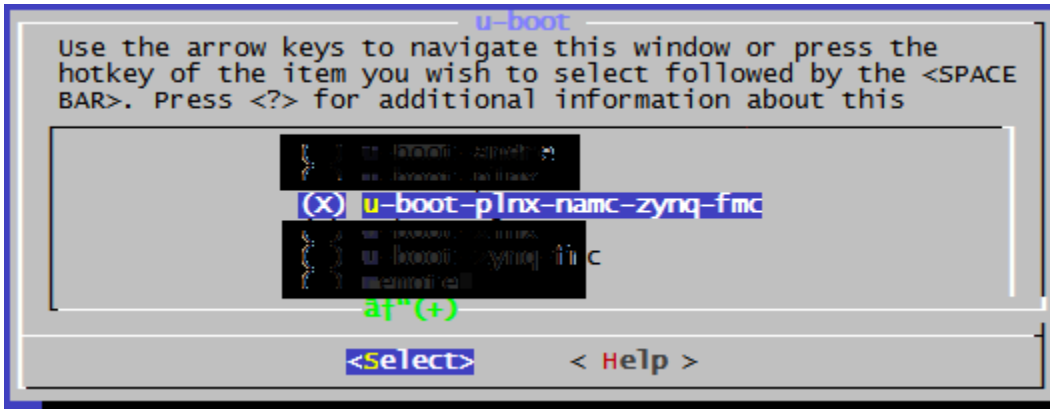


Figure 5: U-boot selection

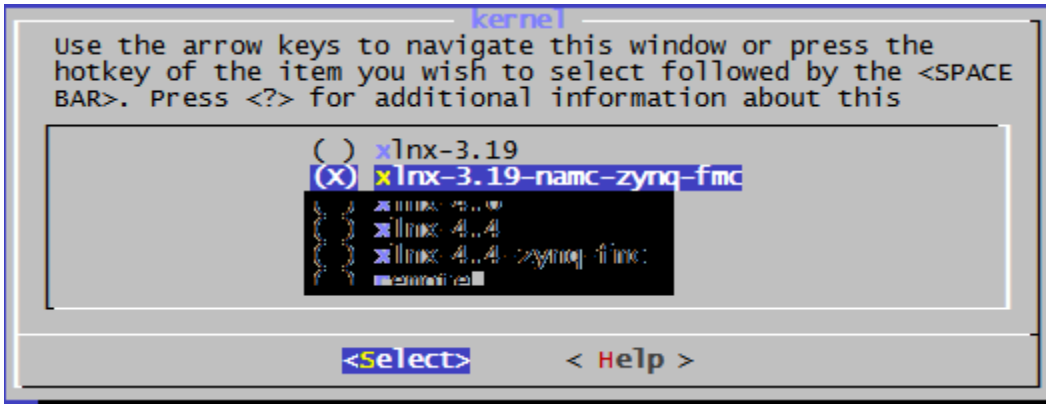
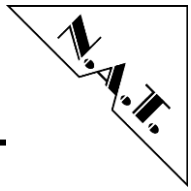


Figure 6: Kernel selection

This is the only necessary configuration you need to do to build the reference design as all other configuration is preconfigured by N.A.T. Feel free to check several other configuration options. Everything is well documented in the petalinux reference manuals.



To exit the configuration press "ESC" until it asks if you want to change your changes.

After executing the command you are able to build the distribution. Use the command:

- ***make build-all***

This command will build the FSBL, U-Boot and Kernel. The binaries will then be packaged to a single file (BOOT.bin) that is needed for booting the device with SD card. It is copied together with the kernel image (image.ub) to the folder /bsp. Copy both files to a micro SD card and populate it to the board. Depending on the speed of you host machine this may take a while so please be patient. When you are finished with that the last command you need to execute is:

If you do not want to execute all commands step by step you might use the command

- ***make all***

As conclusion the following table summarized all targets that can be executed by the makefile

make <target>	**Description**
make all	Cleans current project, imports hardware description from vivado, patches FSBL files, builds kernel and u-boot and creates boot image.
make build-all	Builds the FSBL, kernel, and u-boot and packages it to a single binary
make package	Packages FSBL, bitstream and u-boot. BOOT.bin and image.ub are stored at /bsp
make patch-fsbl	Patches FSBL sources as they are overwritten each time a new hardware gets imported or the build system configured
make config-system	Configures the build systems generally
make config-kernel	Configures the linux kernel
make config-rootfs	Configures the root file system
make import-components	Imports Components from Petalinux installation path into project
make setup-links	Creates project component link in Petalinux installation path. Root rights might be necessary
make patch-components	Patches components with N.A.T sources to ensure hardware operability
make import-hw	Imports the hardware from Vivado project
make clean	Cleans build output
make create-project	Creates a new petalinux project

Table 2: Makefile description

1.7 Programming FMC FRU EEPROM data

It is mandatory for the FMC to contain a valid FRU data image including the minimum set of FRU records required by VITA 57.1 FMC specification section 5.5.1 (IPMI Support) as the carrier needs to adjust some configuration items depending on these records. This mainly affects the direction of the bidirectional clocks (CLKx_BIDIR) and the power supply requirements such as supported voltage range and current load. Depending on

the presence and validation of these records the carrier will decide to enable payload power domains for the FMC module.

As it may be annoying to use FMC modules that may be in beta state not containing any FRU records the carrier brings the functionality to generate and program records according user demands. To use these feature simply connect a micro USB cable to the USB input at the front panel and open a virtual terminal emulation program such as "TeraTerm Pro". You will have to choose the MMC COM port with the higher number as it will release two COM ports (ZYNQ and MMC serial console port). The following picture shows an example containing three serial ports including the MCH serial console and the NAMC-ZYNQ-FMC COM ports (numbers may be different).

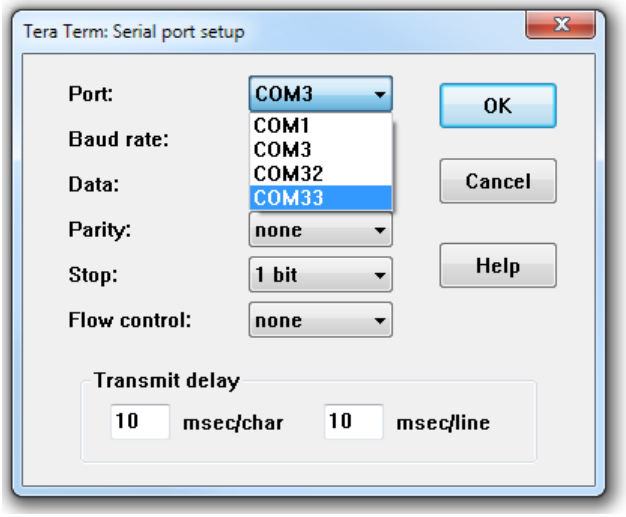


Figure 7: TeraTerm Serial Port Setup

When inserting the AMC with connected USB cable into an operational MicroTCA chassis you should see the following output ensuring being connected to the correct COM port. Ensure you are highlighting the TeraTerm application window in order to prepare for pressing any key to enter the FMC FRU image generation wizard.

```
----------------------
Welcome to NAT-AMC-ZYNQ-FMC !
V1.3 || SVN (13634) || (Feb 15 2017, 12:41:42)
----------------------

CPU frequency: 32000000
Starting power sequence ...

Press any key to generate FMC FRU file   ... 1.0
```

Figure 8: MMC Boot Output

You will have 1 second of time to enter the wizard. When pressing any key the output the wizard will guide you through editing any record of your choice. Skip any record using the enter key. All record fields either have a standard return value (RET) when no valid record file was found in the EEPROM or a value that was parsed of the existing FRU image. Once you are finished with the wizard it will ask you to program the FMC EEPROM. As a reference the output may look like in Figure 9: FRU wizardFigure 9.

```
Press any key to generate FMC FRU file   ... 0.7

Edit Board Area ? (RET=n): y
  Manufacturer: N.A.T GmbH
  Product Name: N.A.T FMC01
  Serial Number: 0000000001
  Part Number: NFMC0815
  FRU file ID: -

Edit Subtype0 record ? (RET=n): y
  P1 Connector Size (LPC=0, HPC=1):  (RET=1|0x1): 1
  CLKx_BIDR (M2C=0, C2M=1):  (RET=0|0x0): 0
  P1 Bank A Signals needed:  (RET=0|0x0): 10
  P1 Bank B Signals needed:  (RET=0|0x0): 10
  P1 Transceivers needed:  (RET=4|0x4): 4
  Max clock for TCK (MHz):  (RET=100): 100

Edit DC_LOAD(0), (VADJ) ? (RET=n): y
  Nominal voltage (mV*10):  (RET=250|0xfa): 250
  Spec'd minimum voltage (mV*10):  (RET=190|0xbe): 190
  Spec'd maximum voltage (mv*10):  (RET=330|0x14a): 330
  Ripple and Noise pk-pk (mV):  (RET=0|0x0):
  Minimum current load (mA):  (RET=0|0x0):
  Maximum current load (mA):  (RET=0|0x0):

Edit DC_OUT(3), (VIO_B_M2C) ? (RET=n):

Edit DC_OUT(4), (VREF_A_M2C) ? (RET=n):

Edit DC_OUT(5), (VREF_B_M2C) ? (RET=n):

Generated 188 bytes binary data.
 buf 0x3e12 len 256
----------
 01 00 00 01 00 08 00 f6   01 07 00 00 00 00 ca 4e
 2e 41 2e 54 20 47 6d 62   48 cb 4e 2e 41 2e 54 20
 46 4d 43 30 31 ca 30 30   30 30 30 30 30 30 30 31
 c8 4e 46 4d 43 30 38 31   35 c1 00 00 00 00 00 ea
 fa 02 0b 84 75 a2 12 00   00 10 0a 0a 00 00 40 64
 02 02 0d fd f2 00 fa 00   be 00 4a 01 00 00 00 00
 00 00 02 02 0d b4 3b 01   4a 01 00 00 00 00 00 00
 00 00 00 00 02 02 0d cd   22 02 b0 04 64 00 14 05
 00 00 00 00 00 00 01 02   0d 03 ed 03 fa 00 00 00
 00 00 00 00 00 00 00 00   01 02 0d fc f4 04 00 00
 00 00 00 00 00 00 00 00   00 00 01 82 0d fb 75 05
 00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00
 00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00
 00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00
 00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00
 00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00

Save file to EEPROM? (RET=n): y

................................................................................
................................................................................
.............................................................. Done
```

Figure 9: FRU wizard

Any FRU record gets generated and parsed according the IPMI Storage Specification V1.0. See this specification to get more detailed information about the records.

2 Troubleshooting

Description

After sourcing settings64.sh to run Vivado, the awk executable fails with the following error:

```
symbol lookup error: awk: undefined symbol: mpfr_z_sub
```

The strace output from awk indicates that Vivado/201x.y/lib/lnx64.o/libgmp.so.Z is the source of this runtime linker breakage.

Solution

The issue occurs because the LD_LIBRARY_PATH is being set in the settings64.sh file and points to the Vivado/201x.y/lib/lnx64.o directory.

Because LD_LIBRARY_PATH is a global setting, it affects all binaries run within the shell where it is set. In this case a library file is conflicting with a version required by awk.

The LD_LIBRARY_PATH is specifically added to the script to support the AXI BFM IP which needs to use Vivado libraries.

If you are not using the AXI BFM IP, you can remove the LD_LIBRARY_PATH setting from settings64.sh.

Neither Vivado IDE nor Vivado Tcl shell require the LD_LIBRARY_PATH variable to be set.

This issue has only been reported on unsupported versions of the Ubuntu operating system. However, to avoid this issue on any OS, the Vivado 2016.3 install will no longer set the LD_LIBRARY_PATH variable.

Known bugs and restrictions

There are no known bugs or restrictions yet.

Appendix A: Reference Documentation

[1]	Petalinux official Wiki	http://www.wiki.xilinx.com/Petalinux
[2]	UG1144 Petalinux Tools Reference Guide	https://www.xilinx.com/support/documentation/sw_manuals/petalinux2014_4/ug1144-petalinux-tools-reference-guide.pdf
[3]	IPMI FRU Storage Specification V1.0	http://www.intel.com/content/dam/www/public/us/en/documents/product-briefs/platform-management-fru-document-rev-1-2-feb-2013.pdf
[4]	FMC Vita 57.1	

Appendix B: Document's History

Revision	Date	Description	Author
1.0	24.10.2016	initial release	mm
1.1	15.05.2017	Added section : Programing FMC FRU EEPROM data	mm