MMC Getting Started

Revision History:
Rev 1.0 01.10.2013 First Draft

Table of Contents

Illustration Index

Index of Tables

1 Overview

The Module Management Controller (MMC) Software allows quick development of AMCs without prior IPMI knowledge. It provides all the mandatory IPMI functionality required by the µTCA, MCTA.4 and AMC specifications, thus allowing easy and fast development of AMCs.

The MMC Software can implement discrete, temperature, voltage, current, fan or OEM sensors . The SDRs for the standard version of software implement only a basic set of sensors.

This document will provide a set-up example for a simple MMC application:

- 5 sensors:
  - 3 voltages: Management Power (MP +3.3V), Payload Power (PP +12V),+5V,
  - 2 temperature sensors (thermistors)
- 1 Gb Ethernet connectivity on Port 0
- PCIe connectivity on Ports 4,5,6,7
- hot swap signalled shut down for the payload
- Power requirements 36 W (3A @12V)

2 Setting up the Hardware

For a successive application the MMC has to be properly connected to all the necessary signals. This chapter will address the connectivity for the example set-up described above.

Signal	Controller Pin	Description
AN0 +3.3V (MP)	95	Analog Input 0 monitors the MP rail
AN1 +12V (PP)	93	Analog Input 1 monitors the PP rail
AN2 TEMP1	92	Analog Input 2 monitors a temperature sensor
AN3 TEMP2	91	Analog Input 3 monitors a temperature sensor
AN4 +5V	90	Analog Input 4 monitors a voltage rail +5.5V
Hot Swap	72	Input for the hot swap switch signal
Shutdown_Req#	82	Signal to inform the payload that it will be turned off
Shutdown_Ready#	59	Signal to inform the MMC that the payload is ready to be turned off
Shutdown#	83	Shutdown signal. When active disables the payload power.

Table 1: Example MMC connectivity

The maximum voltage for the analog inputs is 2.5 Volts. All monitored signals that surpass this limiting value will have to be divided.

The MMC can use the analog inputs to monitor thermistor temperature sensors. The measurements are calibrated for 2 lead NTC thermistors with a resistance of 10K and Beta = 3950(ex: EPCOS B57891M0103K000 farnell_link).

All the control signals used for the hot swap operation use open drain drivers so external pull ups are required.

2.1 Shut down protocol

The shut down of the payload is controlled using the hot swap switch. When the Hot swap handle is opened the MMC will signal the payload that there is a shut down request by asserting the Shutdown_Req# signal. When the payload is ready to shut down it will signal the MMC using Shutdown_Ready#. At this point the payload power will be disabled using the Shutdown# signal.

3 Setting up the Software

The MMC software setup is done by uploading two files over RS232: SDR and FRU. For creating such files two Graphical User Interface compiler are provided : SDR compiler, FRU compiler.

3.1 SDR Compiler

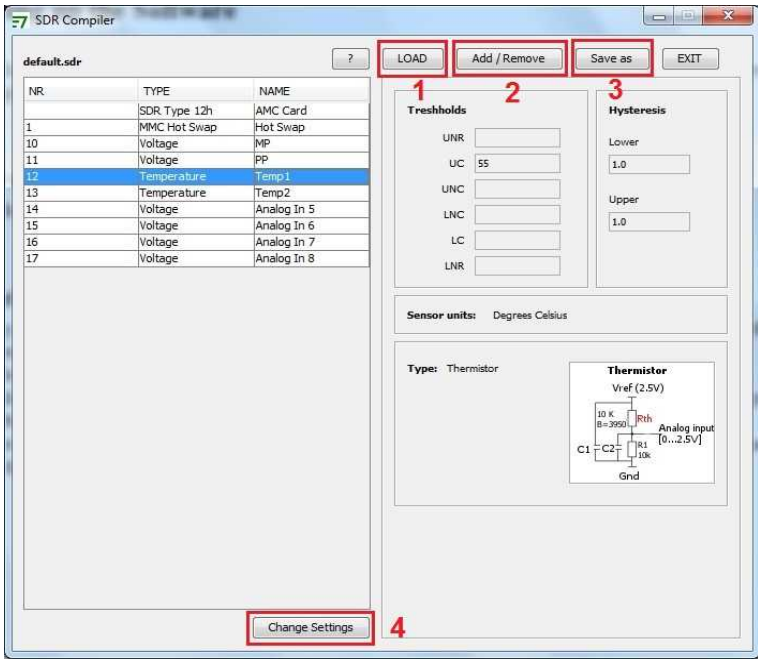

Illustration 1: SDR compiler

The interface for the SDR compiler is composed of:

- available sensors list
- sensor information panes
- buttons

This document is not intended to describe all the features of the SDR compiler, and will only address the ones necessary for the depicted example. For more details on the SDR compiler please refer to it's user manual.

In order to obtain the SDR file required for monitoring the 5 sensors(MP,PP,TEMP1,TEMP2,+5V) the following steps are required:

- define the sensor list by selecting the desired 5 sensors from all the supported sensors list
- customize the sensors
- save the sdr file
- upload the file using xmodem

3.1.1 Defining the Sensor List

By default, at start-up the SDR Compiler loads an empty sensor list. To modify the list the **ADD/Remove [2]** button from the main interface, has to be used.

The “Add/Remove” button opens a new window that allows changes to the sensors list.

Adding sensors is done by selecting them from the left side column (**Available Sensors**) and pressing the **">>"** button, thus moving them in the **Selected Sensors** Column.

Removing sensors is done by selecting them from the right side column (**Selected Sensors**) and pressing the **"<<"** button, thus moving them back to the **Available Sensors** set.

For the described example we require :

- 3 voltage sensors: MP(+3.3V),PP(+12V),+5V
- 2 Temperature sensors: Temp1,Temp2

Besides the sensors required by the monitoring application the Sensor list has to also include:

- sdr type 12h: AMC card; This is required by IPMI
- sensor 1: Hot swap Handle: also required by IPMI

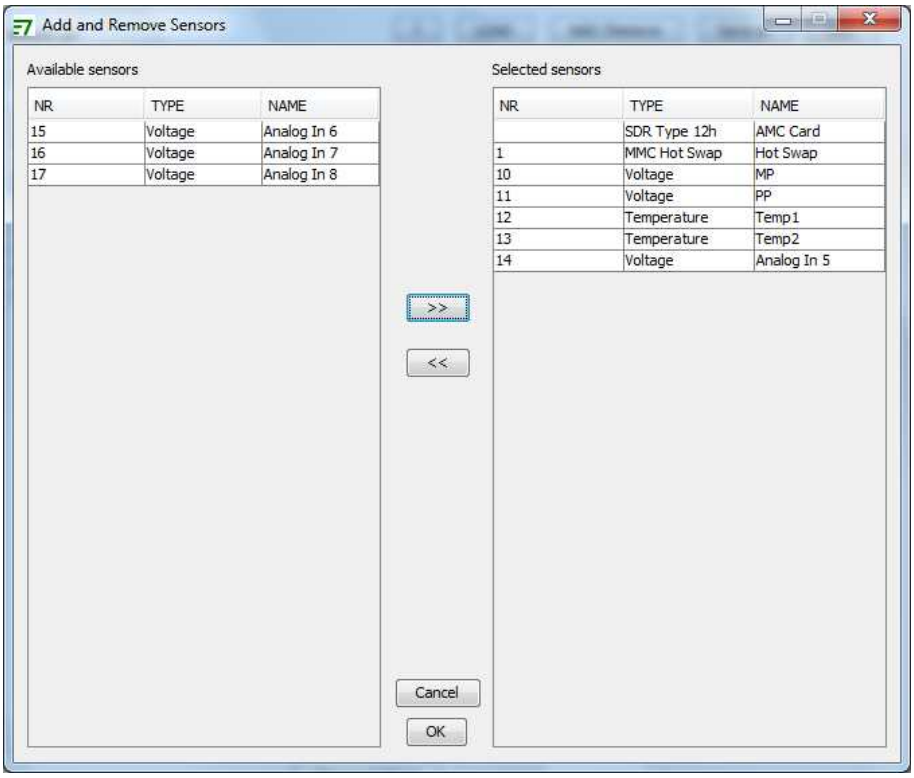

Illustration 2: Sensor List

3.1.2 Customizing the sensors

After the list is populated with the necessary sensors, each one can be customized to address it's individual requirements: thresholds, hysteresis and the sensor name can be changed.

The maximum value for the analog inputs is 2.5 Volts, so for signals that surpass this limiting value conversion circuit are required. A few standard circuits have been embedded in the SDR compiler in order to allow easy integration of popular conversion sensors: positive voltage divider, negative voltage divider, gain block, thermistor.

The current example uses 4 standard sensors: MP, PP, TEMP1, TEMP2. These 4 sensor have been initialized by default so no changes will be required.

The only sensor that needs to be customized is sensor number 14 Analog In 5. To do so select it in the sensor list, and do a double mouse click or use the **Change settings** button [4] in the main interface. At this point the Modify Sensor window will be opened:

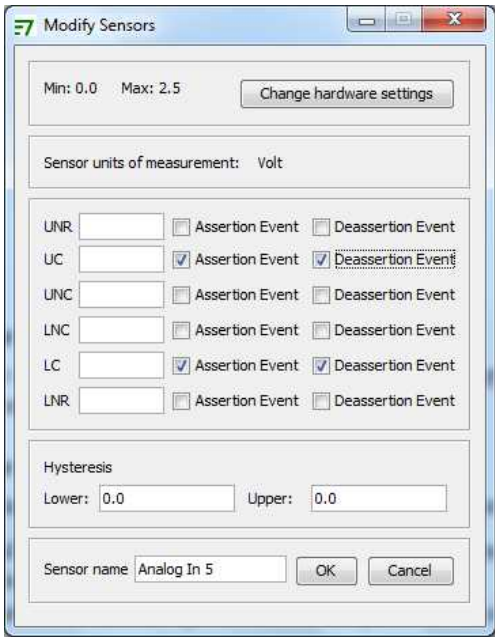

Illustration 3: Modify Sensor Window

The maximum value for the analog input is 2.5 V so we will have to use a voltage divider for the +5V voltage in order to monitor it. We will use a positive voltage divider consisting of a 2.2KΩ and a 1 KΩ resistors. The SDR for the sensor will also have to be adapted to the external hardware divider. This is acomplished using the **Change Hardware Settings** button.

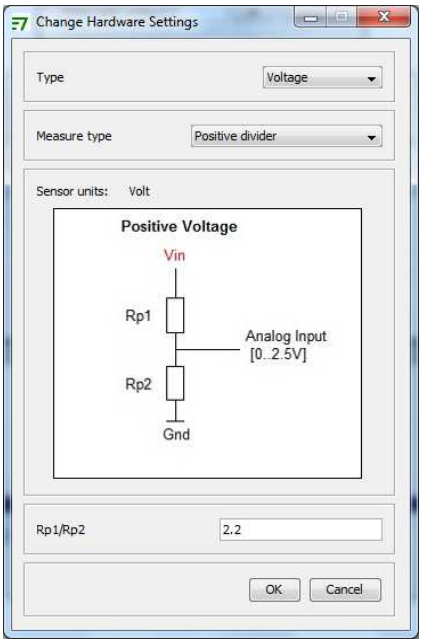

Illustration 4: Change Hardware Settings Window

In the change hardware settings window we select positive voltage divider as hardware model and input the divider value 2.2. At this point the SDR reflects the hardware circuit.

The next step will be to define the threshold values. In this example we will use only 2 thresholds: upper critical +5.5V and lower critical +4.5V. (Depending on specific requirements up to 6 thresholds can be defined). We will also define a 0.2V hysteresis for both positive and negative going events.

The final step will be changing the sensor's name to +5V Mon.

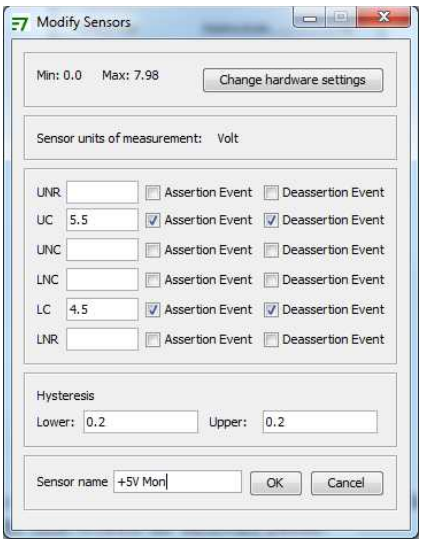

Illustration 5: Settings for +5V sensor

3.1.3 Saving / Loading a SDR file

After all the changes have been done, the file can be saved using the **Save As** [3] button in the main interface.

Existing SDR files can be loaded using the **Load** [1] button in the main interface.
For more information on the SDR Compiler software refer to the SDR Compiler User Manual.

3.1.4 Uploading the SDR file

The SDR file is loaded by sending the file to the MMC over RS232 using the xmodem protocol.
The RS232 is available on the TXD,RXD pins.
For connecting to the RS232 Command Line Interface(CLI) you will have to use a terminal program. On Windows OS we recommend the use of **Tera Term**.
By default the terminal settings are:

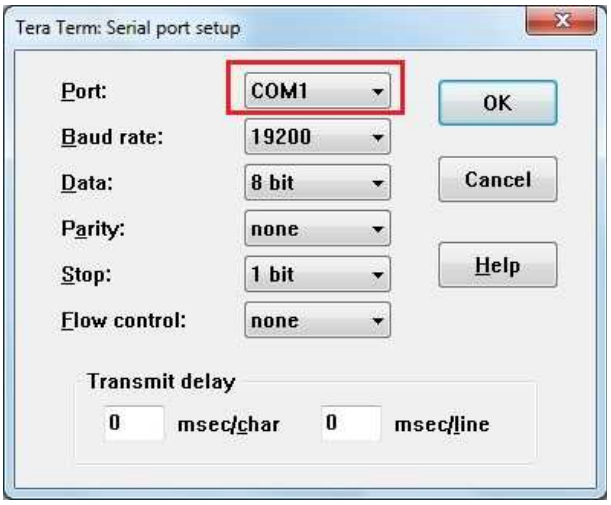

Illustration 6: Default Terminal Settings

The **Port** setting depicts the physical port of the computer on which the MMC is connected. The default Baud rate is 19200.

After the MMC firmware starts the SDR file can be uploaded using the command: **xmodem sdr**. The command will start the reception protocol and will wait for the beginning of the transfer. At this point you can send the file using the xmodem protocol.

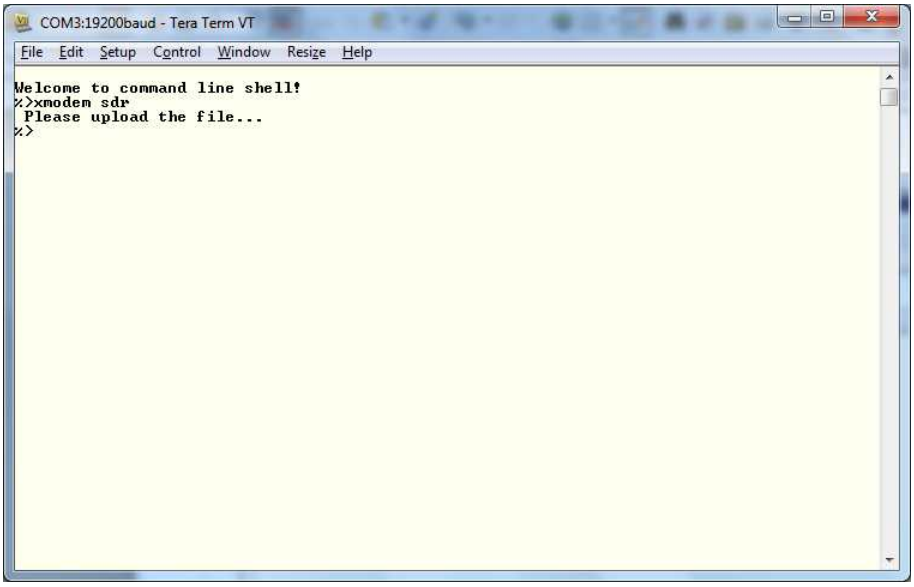

Illustration 7: xmodem sdr command

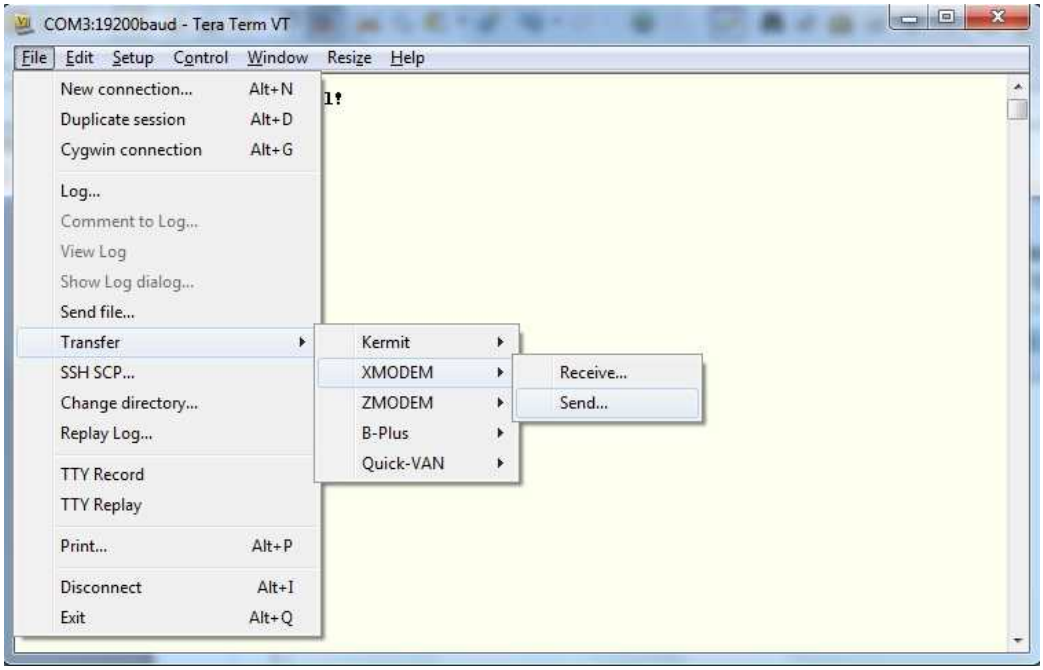

Illustration 8: Send file using xmodem

After the file transfer is complete a confirmation message will be displayed: **Done!** . The SDRs will be updated at the next restart of the MMC. You can trigger a restart by using the **reboot** command.

At the next restart, the SDRs have been changed. You can view the installed sensors using the **sensor** command:

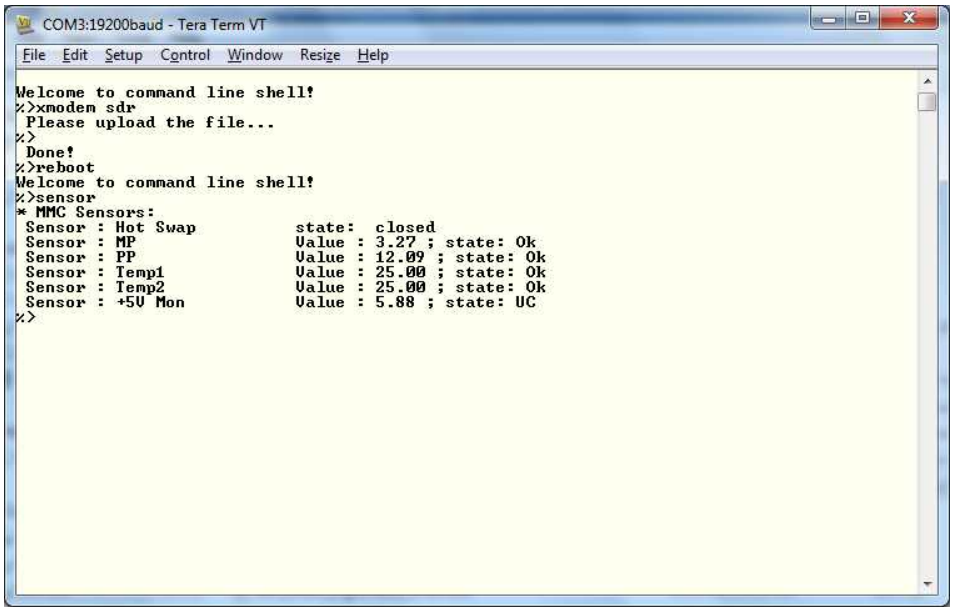

Illustration 9: Complete SDR update

3.2 FRU Compiler

For generating the FRU file the FRU compiler is used.

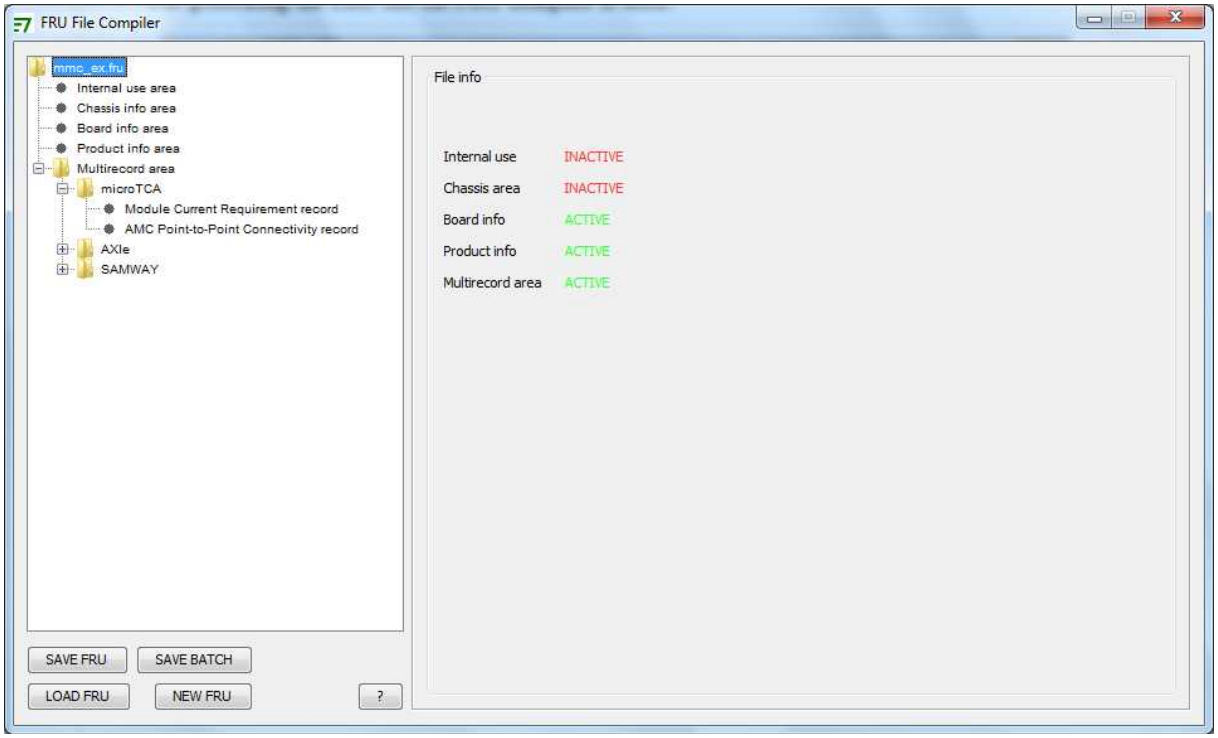

Illustration 10: Fru Compiler

The GUI FRU compiler allows the user to edit the contents of all the FRU file areas:

- Internal Use Area
- Chassis Info Area:type, part number, serial number
- Board info Area: manufacturing date and time, board manufacturer name, board product name,board part number, board serial number
- Product Info Area: product manufacturer name, product name, product part number, product version, product serial number.
- Multirecord Area: module current requirement record, AMC point to point conectivity records (ekeying records)

This document is meant to describe a set-up example for a board that consumes 36 W on the Payload Power rail, and uses GbE connectivity on port 0 and PCIe connectivity on ports 4,5,6,7, so it will not use all the features of the FRU compiler. For more details you can refer to the FRU compiler User Manual.

The Board Info Area contains information regarding the physical board. For simple modules the board and the product depict the same physical entity so only one of the board and product info areas can be used. For

more complex modules where the product is a mixture of carrier, mezzanine and ad-on cards both board and product info areas may be necessary to fully identity the complete module.

Depending on your specific requirements you can use only one, both or none of the two info areas to describe your module. The current example will use both.

3.2.1 Board Info Area

By default, for a new FRU file this area is disabled, so before editing it you have to enable it by using the Activate/Deactivate button in the upper right corner. Filing the Board information is really straight forward as all the area options are implemented as text boxes.

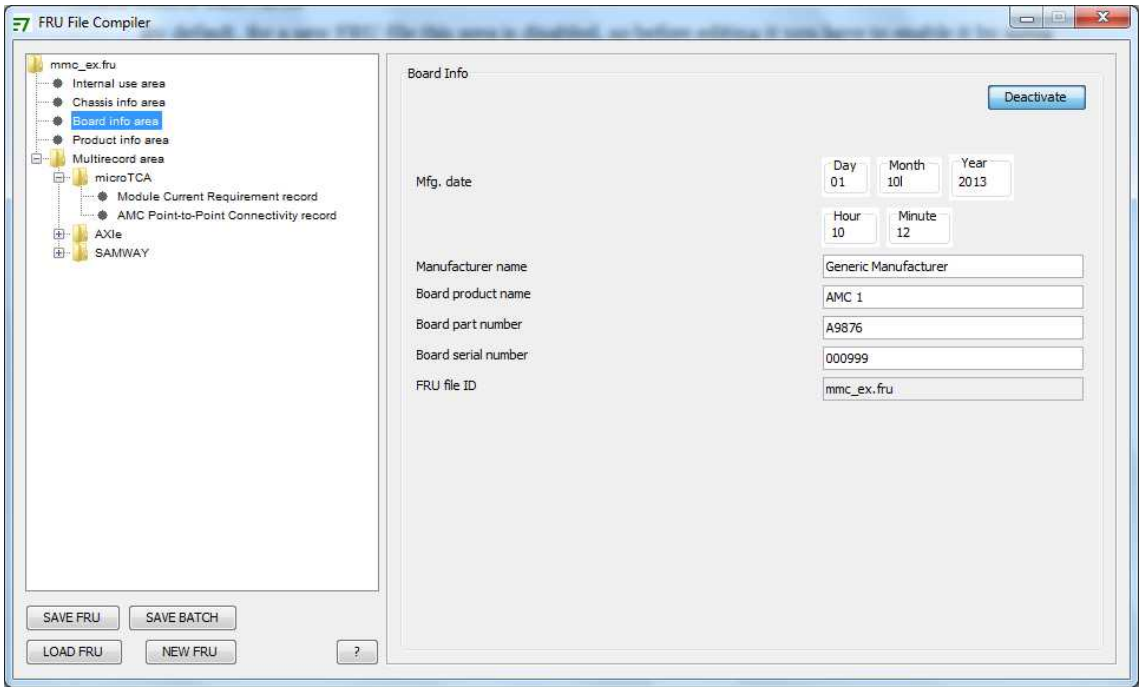

Illustration 11: Board Info Area

3.2.2 Product Info Area

By default, for a new FRU file this area is disabled, so before editing it you have to enable it by using the Activate/Deactivate button in the upper right corner.

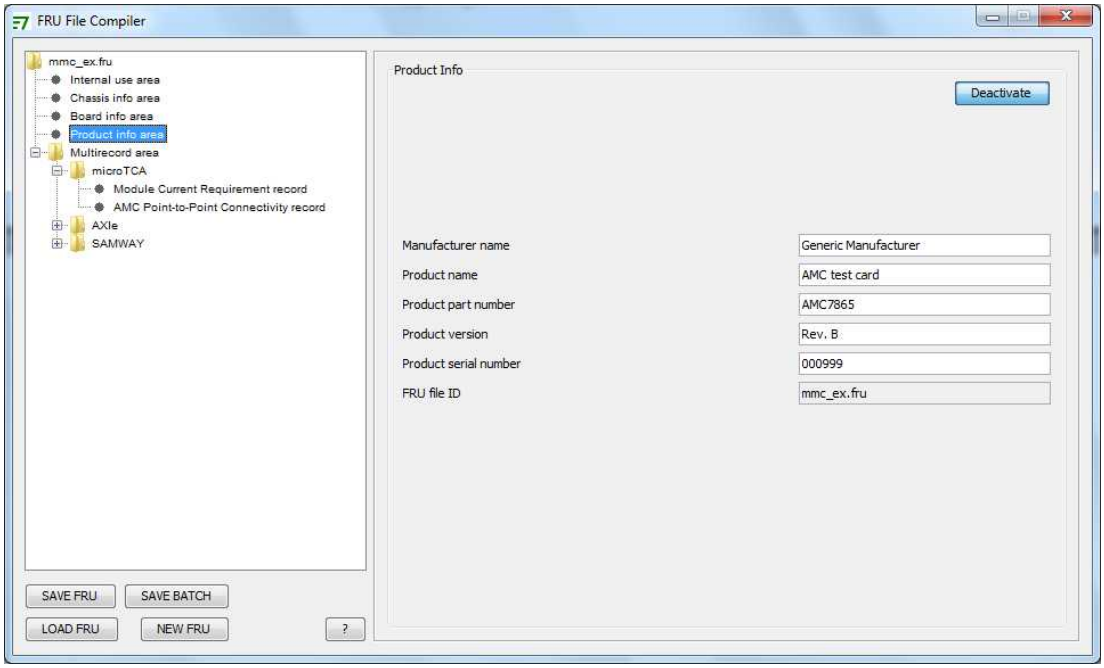

Illustration 12: Product Info Area

3.2.3 Module Current Requirement record

The record has to be activated before it can be edited. The current example's payload consumes 36W. That translates into a (36W/12V) 3A current consumption.

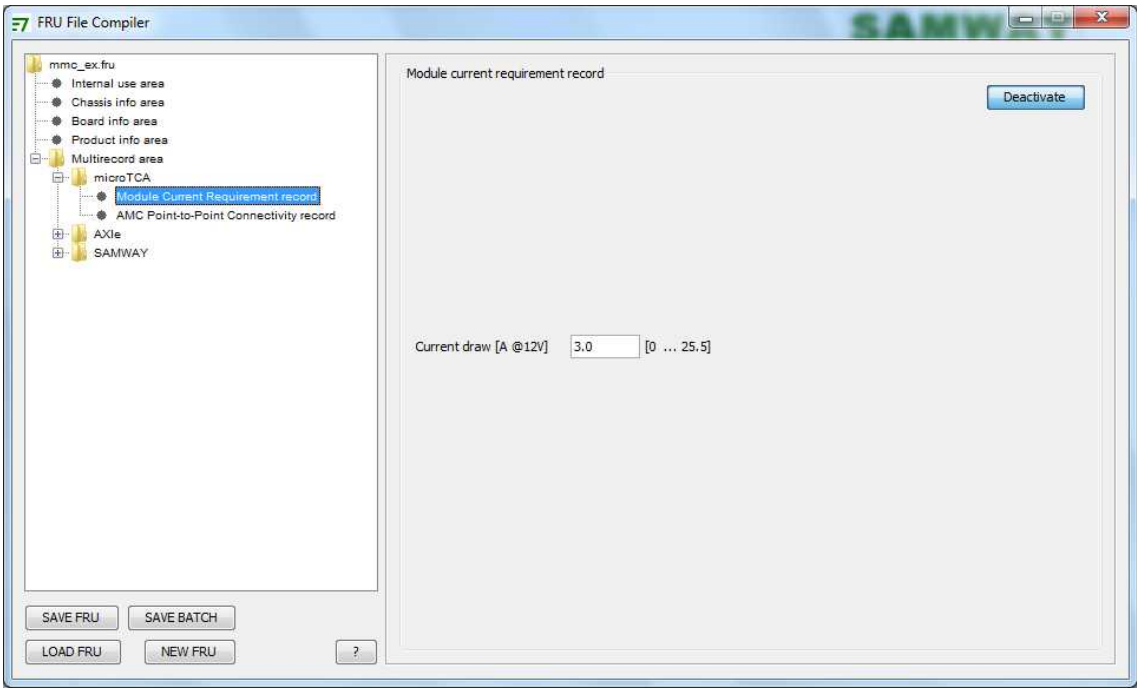

Illustration 13: Module Current Requirement

3.2.4 AMC Point To Point Connectivity Records

This record type holds information about the connectivity that is implemented on an Advanced Mezzanine Card (AMC).

In MicroTCA the interconnectivity is accomplished using Ports, Channels and Links. The FRU compiler provides an intuitive way of specifying the connectivity features for an AMC.

The links available for an AMC are displayed as a high priority first list. This means that if two links use the same ports, the one that is higher in the list has a higher priority in the E-keying process.

The current example uses 1 Gb Ethernet connectivity on Port 0, and PCIe connectivity on ports 4,5,6,7. For PCIe the example assumes that both Gen 1 and Gen 2 links are supported by the AMC, and that it is also able to successfully negotiate x1, x2, x4 connections. Assuming that a Gen2 x4 link is the most desirable connectivity case, and a Gen 1 x1 is the least desirable one we will populate the list with links.

Before adding links the record has to be activated using the activate/deactivate button in the upper right corner.

Adding links resumes at using the ADD button, choosing the link type,link type extension and the ports used.

First we will add the 1 Gb Ethernet link:

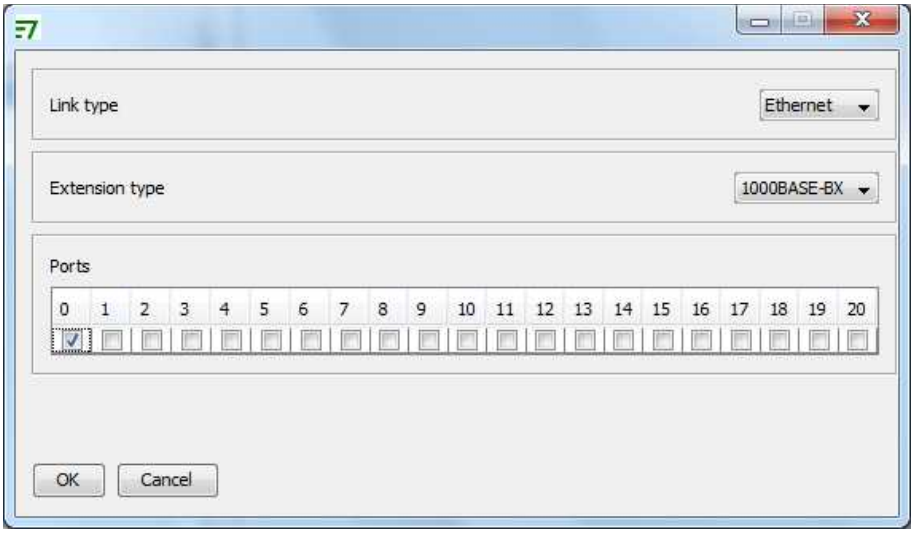

Illustration 14: Setting up a 1GbE link on port 0

Next, we will start adding the PCIe links, Gen2 x4, Gen2 x2, Gen2 x1, Gen1 x4, Gen1 x2, Gen1 x1. For the PCIe link also the Asymmetric Match field can be chosen. For our case, AMC slave board, we will choose the secondary (slave) asymmetric match.

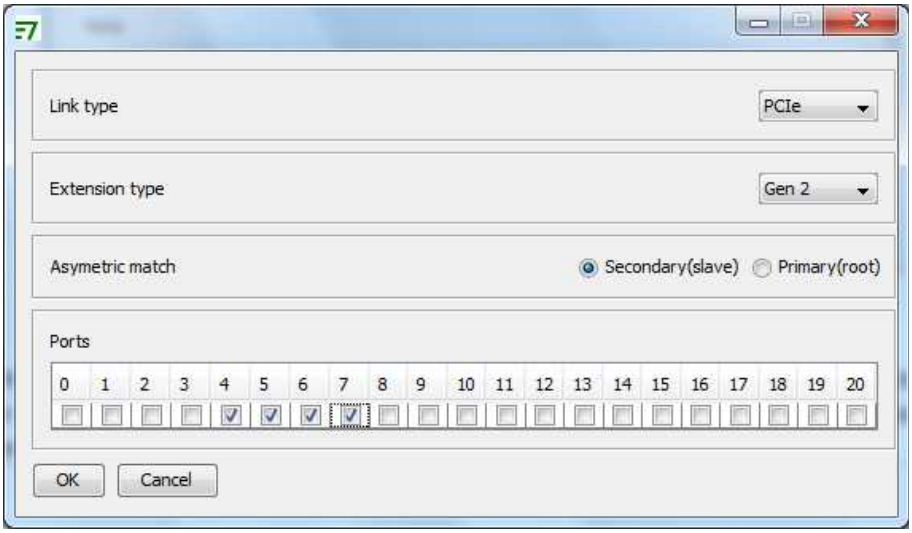

Illustration 15: Setting up a PCIe Gen2 x4 slave link

After we define all the links the link list will have 7 entries:

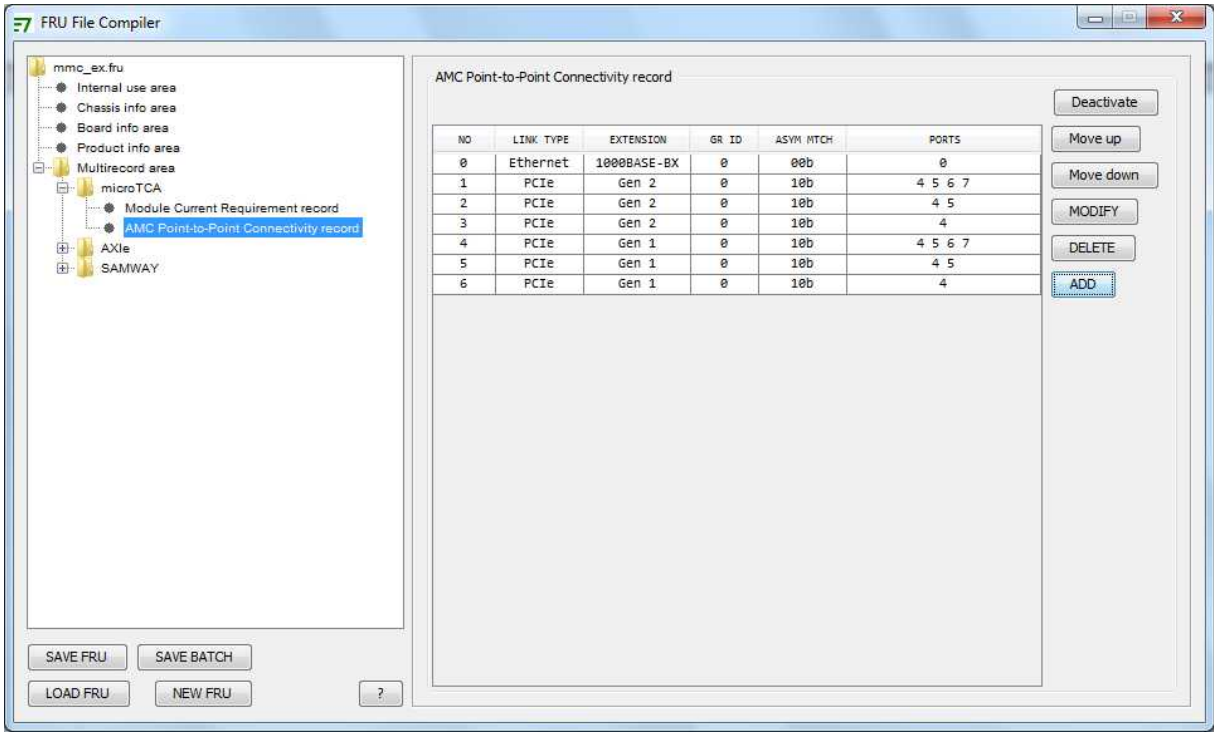

Illustration 16: AMC connectivity example

The order of the links can be changed using the move up and move down buttons. The already defined links can also be modified or deleted.

3.2.5 Saving / Loading a FRU file

After all the changes have been done, the file can be saved using the **Save FRU** button in the main interface.

Existing FRU files can be loaded using the **Load FRU** button.

The FRU compiler also supports a feature for saving the FRU files for a whole batch of boards using a single click. It uses the FRU file for the current board as a starting point and creates the files for the other boards in the batch by simply copying the data that doesn't change (manufacturer name, product name, product version, product part number, multi record area, internal area) and incrementing all the necessary serial numbers.

For more information on the FRU Compiler software refer to the FRU File User Manual.

3.2.6 Uploading the FRU file

The FRU file is loaded in the same manner as the SDR file. The only difference is that the CLI command used is **xmodem fru**. All the other upload steps are the same.