Manufacturer: NI

Board Assembly Part Numbers (Refer to Procedure 1 for identification procedure)**:**

Part Number and Revision	Description
131868D-03L or later	**NI Ettus USRP X440**

Volatile Memory

Target Data	*Type*	*Size*	*Battery Backup*	*User*[1] *Accessible*	*System*[1] *Accessible*	*Sanitization Procedure*
Device operation	FPGA	Xilinx XCZU28DR	No	Yes	Yes	Cycle Power
Device operation	DRAM	12 GB	No	Yes	Yes	Cycle Power
System Controller Unit – RAM	SRAM	256 KB	No	Yes	Yes	Cycle Power

Non-Volatile Memory (*incl. Media Storage*)

Target Data	*Type*	*Size*	*Battery Backup*	*User*[1] *Accessible*	*System*[1] *Accessible*	*Sanitization Procedure*
Motherboard power-up configuration	FPGA	Intel 10M04SAU	No	Yes	Yes	Procedure 3
Motherboard power-up configuration	FPGA	Intel 10M04SAU	No	No	No	None
Daughterboard secure EEPROM	EEPROM	256 B	No	No	No	None
System Controller Unit	Flash	512 KB	No	No	Yes	None
Motherboard eMMC	eMMC Flash	16 GB	No	Yes	Yes	Procedure 2
Clock distribution	EEPROM	256 B	No	No	Yes	None
Clocking ID	EEPROM	256 B	No	Yes	Yes	Procedure 4
Power ID	EEPROM	256 B	No	Yes	Yes	Procedure 4
Digital IO ID	EEPROM	256 B	No	Yes	Yes	Procedure 4
Daughterboard ID	EEPROM	256 B	No	Yes	Yes	Procedure 4
Motherboard ID	EEPROM	256 B	No	Yes	Yes	Procedure 4

[1] Refer to *Terms and Definitions* section for clarification of *User* and *System Accessible*

Contact: 866-275-6964
support@ni.com

Procedures

Procedure 1 – Board Assembly Part Number identification:
To determine the Board Assembly Part Number and Revision, refer to the label applied to the bottom of your product. The Assembly Part Number should be formatted as "P/N: 131868x-03L" where "x" is the letter revision of the assembly (e.g. A, B, C…).

Procedure 2 – eMMC

The eMMC has four distinct memory areas: 2 boot partitions, 1 replay-protected memory block, and the 16 GB main storage area. Only the 16 GB main storage area is used by default on X440, though the other areas are system accessible. Note that the replay-protected memory block must be provisioned with a key prior to use, and that key is required to write data or clear that memory in the future.

The main 16 GB memory area is used to store the operating system and user data. This may be sanitized by writing the factory image to the X440. The latest factory image can be downloaded from http://files.ettus.com/binaries/cache/x4xx/. Refer to the manifest to locate the correct file for the latest factory image at http://github.com/EttusResearch/uhd/blob/master/images/manifest.txt.

To write the image over USB, connect both USB ports of the X440 to an external computer, and open a serial console to the PS. Reboot the X440 and stop the boot process at the bootloader by typing `noautoboot` when prompted. At the `ZynqMP>` prompt, run `ums 0 mmc 0` which will make the X440's eMMC accessible to the connected computer as a USB mass storage device. Write the factory image to the device, and power cycle the device after this is complete.

The sdimg file can be written to the device using a tool such as dd or or bmaptool. For example, on the attached host computer, run:

sudo dd if=<file.sdimg> of=/dev/<usb device name> bs=16M status=progress

Procedure 3 – MB CPLD

The motherboard CPLD can be reprogrammed by running:

```
python3 /usr/lib/python3.7/site-
packages/usrp_mpm/periph_manager/x4xx_update_cpld.py
```

Procedure 4 – ID EEPROMs (ClkAux, PwrAux, DioAux, DB0, DB1, MB)

Option A: Save and restore EEPROM contents

The contents of these EEPROMs are critical for correct operation of the device and blanking them will render the device unusable. These EEPROMs contain device-dependent information, and therefore it is recommended to create a backup before deployment and restore this backup to sanitize.

To save the initial contents of the EEPROMs, boot the X440 and log in as root. Run the following commands to back up each of the 6 EEPROMs:

```
dd if=$(eeprom-path mb_eeprom) of=mb_eeprom.bak
dd if=$(eeprom-path clkaux_eeprom) of=clkaux_eeprom.bak
dd if=$(eeprom-path dioaux_eeprom) of=dioaux_eeprom.bak
dd if=$(eeprom-path pwraux_eeprom) of=pwraux_eeprom.bak
dd if=$(eeprom-path db0_eeprom) of=db0_eeprom.bak
dd if=$(eeprom-path db1_eeprom) of=db1_eeprom.bak
```

After these commands have been run, copy the 6 .bak files off of the device (e.g., to a USB drive or via SSH) for safe-keeping.

To sanitize the EEPROMs, boot the X440 and log in as root. Copy these *.bak files back to the device, and run the following commands to write them to the EEPROM:

```
dd if=mb_eeprom.bak of=$(eeprom-path mb_eeprom)
dd if=clkaux_eeprom.bak of=$(eeprom-path clkaux_eeprom)
dd if=dioaux_eeprom.bak of=$(eeprom-path dioaux_eeprom)
dd if=pwraux_eeprom.bak of=$(eeprom-path pwraux_eeprom)
dd if=db0_eeprom.bak of=$(eeprom-path db0_eeprom)
dd if=db1_eeprom.bak of=$(eeprom-path db1_eeprom)
```

Option B: Blank EEPROMs

WARNING: THIS WILL RENDER THE DEVICE UNUSABLE

```
dd if=/dev/zero of=$(eeprom-path mb_eeprom)
dd if=/dev/zero of=$(eeprom-path clkaux_eeprom)
dd if=/dev/zero of=$(eeprom-path dioaux_eeprom)
dd if=/dev/zero of=$(eeprom-path pwraux_eeprom)
dd if=/dev/zero of=$(eeprom-path db0_eeprom)
dd if=/dev/zero of=$(eeprom-path db1_eeprom)
```

Terms and Definitions

Cycle Power:
The process of completely removing power from the device and its components and allowing for adequate discharge. This process includes a complete shutdown of the PC and/or chassis containing the device; a reboot is not sufficient for the completion of this process.

Volatile Memory:
Requires power to maintain the stored information. When power is removed from this memory, its contents are lost. This type of memory typically contains application specific data such as capture waveforms.

Non-Volatile Memory:
Power is not required to maintain the stored information. Device retains its contents when power is removed. This type of memory typically contains information necessary to boot, configure, or calibrate the product or may include device power up states.

User Accessible:
The component is read and/or write addressable such that a user can store arbitrary information to the component from the host using a publicly distributed NI tool, such as a Driver API, the System Configuration API, or MAX.

System Accessible:
The component is read and/or write addressable from the host without the need to physically alter the product.

Clearing:
Per *NIST Special Publication 800-88 Revision 1*, "clearing" is a logical technique to sanitize data in all User Accessible storage locations for protection against simple non-invasive data recovery techniques using the same interface available to the user; typically applied through the standard read and write commands to the storage device.

Sanitization:
Per *NIST Special Publication 800-88 Revision 1*, "sanitization" is a process to render access to "Target Data" on the media infeasible for a given level of effort. In this document, clearing is the degree of sanitization described.