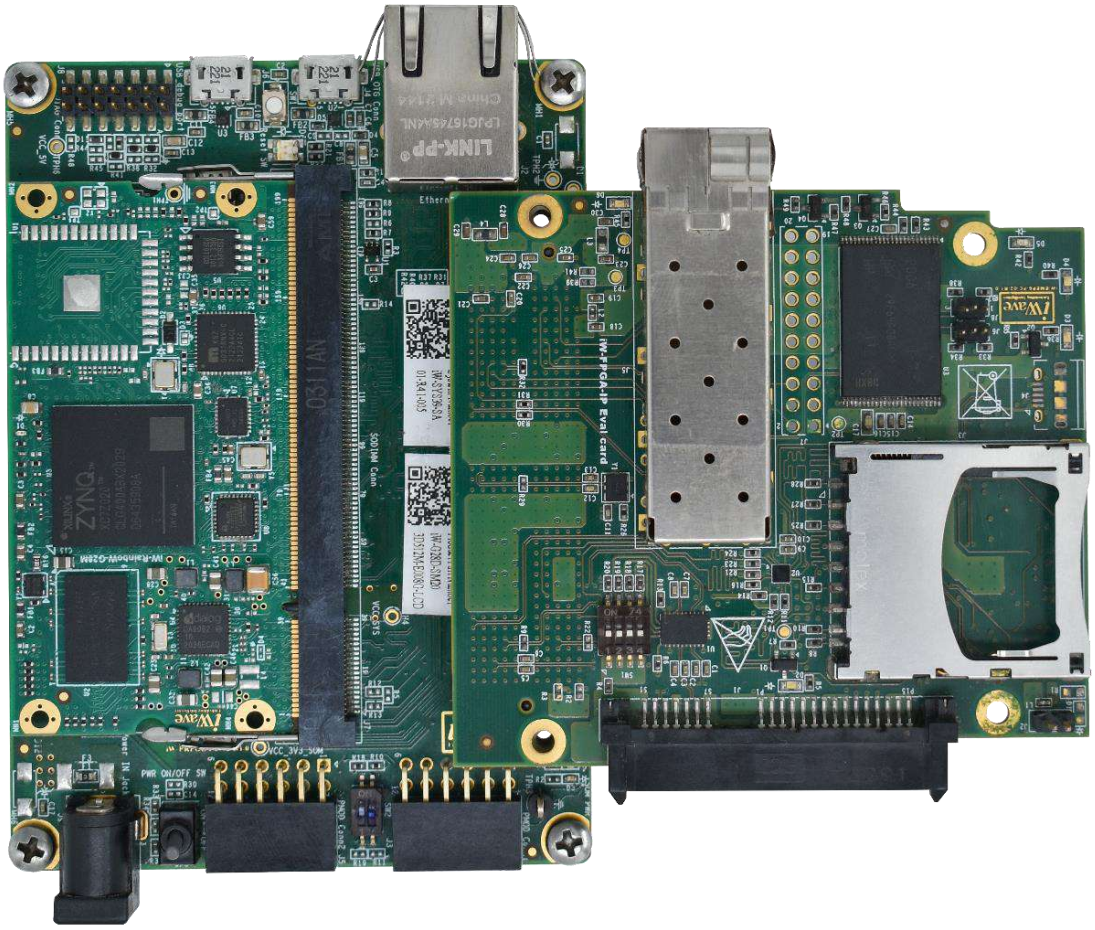
Software User Guide for NAND host controller



iWave

Disclaimer

iWave Systems reserves the right to change details in this publication including but not limited to any Product specification without notice.

No warranty of accuracy is given concerning the contents of the information contained in this publication. To the extent permitted by law no liability (including liability to any person by reason of negligence) will be accepted by iWave Systems, its subsidiaries or employees for any direct or indirect loss or damage caused by omissions from or inaccuracies in this document.

CPU and other major components used in this product may have several silicon errata associated with it. Under no circumstances, iWave Systems shall be liable for the silicon errata and associated issues.

Trademarks

All registered trademarks, product names mentioned in this publication are the property of their respective owners and used for identification purposes only.

Certification

iWave Systems Technologies Pvt. Ltd. is an ISO 9001:2015 Certified Company.

Warranty & RMA

Warranty support for Hardware: 1 Year from iWave or iWave's EMS partner. For warranty terms, go through the below web link, http://www.iwavesystems.com/support/warranty.html
For Return Merchandise Authorization (RMA), go through the below web link, http://www.iwavesystems.com/support/rma.html

Technical Support

iWave Systems technical support team is committed to provide the best possible support for our customers so that our Hardware and Software can be easily migrated and used.

For assistance, contact our Technical Support team at,

Email : emfew@iwavesystems.com
Website : www.iwavesystems.com
Address : iWave Systems Technologies Pvt. Ltd.
7/B, 29th Main, BTM Layout 2nd Stage,
Bangalore, Karnataka,
India – 560076

Table of Contents

List of Figures

List of Tables

1. INTRODUCTION

1.1 Purpose and scope

This document is the Software User Guide for testing NAND Flash Controller. This guide provides detailed procedure to test the NAND flash.

1.2 References

- Petalinux tool document 2020.1
- UBIFS document

1.3 List of Acronyms

The following acronyms will be used throughout this document.

Table 1 **:Acronyms & Abbreviations**

Acronyms	Abbreviations
NAND	NOT AND
MTD	Memory Technology Device
JFFS2	Journaling Flash File System version 2
UBIFS	UBI File System

2. BSP Petalinux Compilation

This section explains procedure and detailed information about compiling PetaLinux for iW-RainboW-G28M platform

2.1 Host Requirements

- 8 GB RAM.
- 2 GHz CPU clock or equivalent.
- 100 GB free HDD space.
- Linux 64bit host PC (Ubuntu 16.04 or higher).
- Root permission on the Development Host. The PetaLinux tools need to be installed as a non-root user.

2.2 Host Package Installation

PetaLinux requires a number of standard development tools and libraries to be installed on your Linux host workstation.

- Open a terminal window and install the below packages in host PC

```
$ sudo apt-get update
$ sudo apt-get install sed wget cvs subversion
$ sudo apt-get install git-core coreutils unzip
$ sudo apt-get install texi2html texinfo zlib1g:i386
$ sudo apt-get install libsdl1.2-dev docbook-utils
$ sudo apt-get install gawk python-pysqlite2 diffstat
$ sudo apt-get install help2man make gcc vim
$ sudo apt-get install build-essential g++ desktop-file-utils
$ sudo apt-get install chrpath libgl1-mesa-dev xvfb repo
$ sudo apt-get install libglu1-mesa-dev mercurial libssl-dev
$ sudo apt-get install autoconf automake groff libtool xterm
$ sudo apt-get install socat tftpd-hpa
$ sudo apt-get install gcc-multilib libsdl1.2-dev libglib2.0-dev libncurses-dev
```

2.3 PetaLinux Tool Installation

- Download PetaLinux 2020.1 Tools Installer from the below link https://www.xilinx.com/member/forms/download/xef.html?filename=petalinux-v2020.1-final-installer.run
- User need to create an account to download from Xilinx website
- PetaLinux Tools installation is straight-forward. Without any options, the PetaLinux Tools are installed into the current working directory. Alternatively, an installation path may be specified.
- Once the tool is downloaded, execute below commands

  host@host~$ chmod 755 petalinux-v2020.1-final-installer.run

  host@host~$ ./petalinux-v2020.1-final-installer.run
- The above command installs the PetaLinux Tools into the current working directory.
- Read and agree to the PetaLinux End User License Agreement to install PetaLinux.

2.4 PetaLinux Build

- To setup the Petalinux working environment, source the appropriate settings script using below command.

  host@host~$ source <path-to-installed-petalinux>/settings.sh

  host@host~$ export LANG=en_US.UTF-8
- Execute the below command to create a PetaLinux zynq project of name “zynq-iwg28m”.

  host@host $ petalinux-create --type project --template zynq --name zynq-iwg28m
- Change into the directory of your PetaLinux Project.

  host@host~$ cd <path-to-zynq-iwg28m>/zynq-iwg28m
- The PetaLinux project-spec patch from deliverables is located in the below path.

  EMFEW_Release_1.0/iW-EMFEW-PF-01-R1.0/iW-EMFEW-SF-01-R1.0/iW-EMFEW-SC-01-R1.0/PATCH000-iW-EMFEW-SC-01-R1.0-REL0.1-PL20.1_customization.patch

  Copy the PetaLinux patch file, by executing the below command.

  host@host/<Directory>/zynq-iwg28m~$ cp <path_to_Patch>/PATCH000-iW-PRFOZ-SC-01-RX.X-RELX.X-PL20.2_customization.patch ../
- To apply the patch file, execute the below command.

  host@host/<Directory>/zynq-iwg28m~$ patch -Np1 < ../PATCH000-iW-EMFEW-SC-01-R1.0-REL0.1-PL20.1_customization.patch
- The FPGA XSA file from deliverables is located in the below path.

  EMFEW_Release_1.0/iW-EMFEW-PF-01-R1.0/iW-EMFEW-FF-01-R1.0/iW-EMFEW-BN-01-R1.0/top_test_mod.xsa

- Import the hardware description with petalinux-config command, by giving the path of the directory containing top_mod.xsa file as follows:

  host@host/zynq-iwg28m$ petalinux-config --get-hw-description=<path-to-parent-directory-of-xsa- file>

- The above command launches the top system configuration menu as shown below;

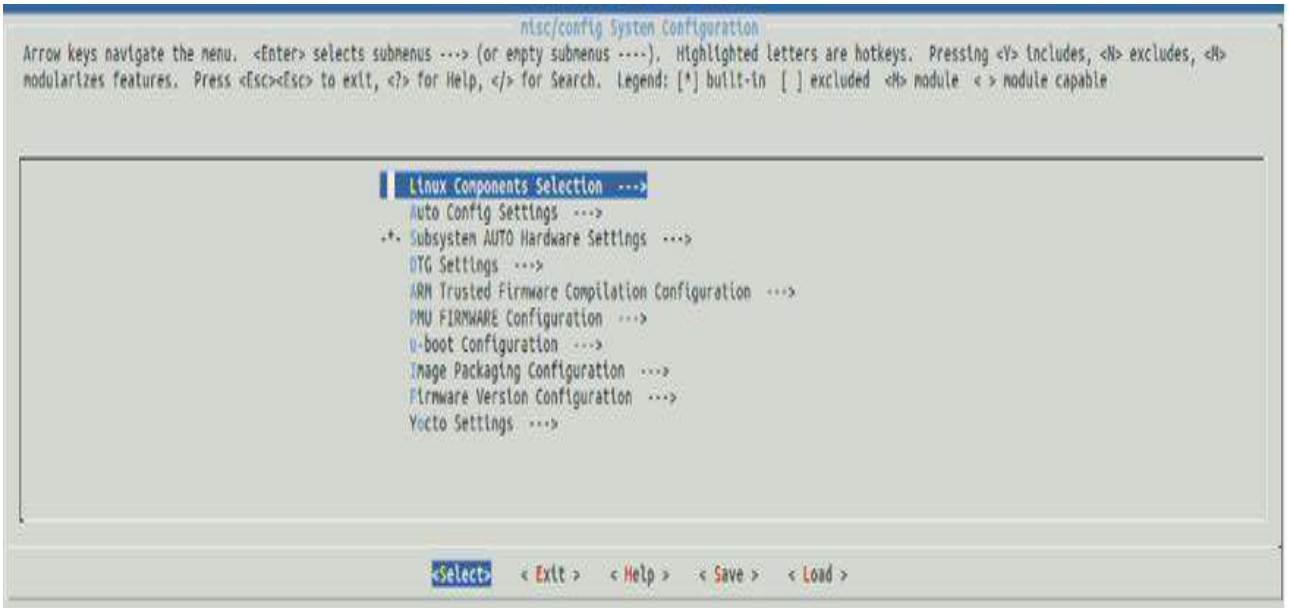


Figure 1 : Petalinux system configuration

- Click on save and exit to save the configuration.
- To build the system image, execute the below command.

  host@host/<Directory>/zynq-iwg28m~$ petalinux-build

- After the successful compilation the binaries will be placed in below path.

  ~/<path to zynq-iwg28m>/zynq-iwg28m/images/linux/

- The binary files are listed below

  zynq_fsbl.elf, u-boot.elf, system.bit.bin, image.ub

- To create BOOT.bin, execute the below command. The BOOT.bin will be created in images/linux folder.

  host@host/<Directory>/zynq-iwg28m~$ petalinux-package --boot --format BIN --fsbl images/linux/zynq_fsbl.elf --u-boot images/linux/u-boot.elf -o images/linux/BOOT.bin

Refer the **BINARY PROGRAMMING** section to update the Linux kernel binary

2.5 PL Bitstream

- Bit file is an optional file that is used when PL Programming is involved.
- Bit file will be generated when creating a Zynq FSBL as mentioned in previous section.
- Before using the PL bitfile, it has to be converted to binary. The BootGen tool can be used for this operation.
- The BootGen tool uses an input file called a “BIF” which outlines the structure of the boot image.
- An example of a BIF file is:

  all:

  {

  system.bit

  }

- Open terminal and source the Xilinx tools environment.
  host@host~$ *source <path-to-installed-vivado-2020.1>/settings.sh*
- Execute the below to create the bin file

  host@host~$ *$ bootgen -image <file>.bif -w -process_bitstream bin*

  where <file>.bif is the above created BIF file.
- system.bit.bin file will be created in the same directory as BIF file. This binary file can be used by U-Boot to directly program the Zynq PL.
- Refer the **BINARY PROGRAMMING** section to update the bitstream binary.

3. BINARY PROGRAMMING

This section explains the procedure and detailed information about programming the binaries into boot device of iW-RainboW-G28M platform.

- The prebuilt binaries are available in the deliverables in the below path.

  EMFEW_Release_1.0/iW-EMFEW-PF-01-R1.0/iW-EMFEW-SF-01-R1.0/iW-EMFEW-BN-01-R1.0

3.1 JTAG Programming

This section explains the step-by-step procedure to program the binaries into iW-RainboW-G28M platform using JTAG Programmable Cable.

3.1.1 Requirements

To program the binaries for iW-RainboW-G28M platform, following Items are required:

- JTAG Programmable Cable.
- Binaries (BOOT.bin and zynq_fsbl.elf)
- USB type A to micro B cable.
- Host PC (Windows/Linux) with Xilinx Vitis 2020.2 for JTAG Programming.
- Power Supply.

Note: Here DIGILENT JTAG programming Cable is used. For more information regarding this, check URL: www.digilentinc.com

Programming QSPI

- Programming the boot image to QSPI via JTAG is done through Xilinx Vitis 2020.2 tool.
- Refer **Xilinx Vitis** section to install Xilinx Vitis 2020.2 tool on host PC.
- Make sure that the Digilent plugin for Xilinx tools is installed in the Host PC if DIGILENT JTAG programming Cable is used.
- Digilent plugin for Xilinx Tools for Linux PC can be downloaded from https://reference.digilentinc.com/lib/exe/fetch.php?tok=329b1e&media=http%3A%2F%2Ffiles.digilent.com%2FSoftware%2FAdept2%2BRuntime%2F2.18.3%2Fdigilent.adept.runtime_2.18.3-x86_64.tar.gz

- To install Digilent plugin for linux PC, untar the downloaded file.

  $tar -xvf digilent.adept.runtime_2.18.3-x86_64.tar.gz

- Install the plugin by executing below commands

  $cd digilent.adept.runtime_2.18.3-x86_64

  $sudo ./install.sh silent=1

- BOOT.bin and zynq_fsbl.elf file from the deliverables to below path:

  EMFEW_Release_1.0/iW-EMFEW-PF-01-R1.0/iW-EMFEW-SF-01-R1.0/iW-EMFEW-BN-01-R1.0

- Connect the JTAG programmable cable between the JTAG connector in the carrier board and Host PC.
- Open the Xilinx Vitis 2020.2 tool and click on “Xilinx Tools” and the following drop-down menu appears.

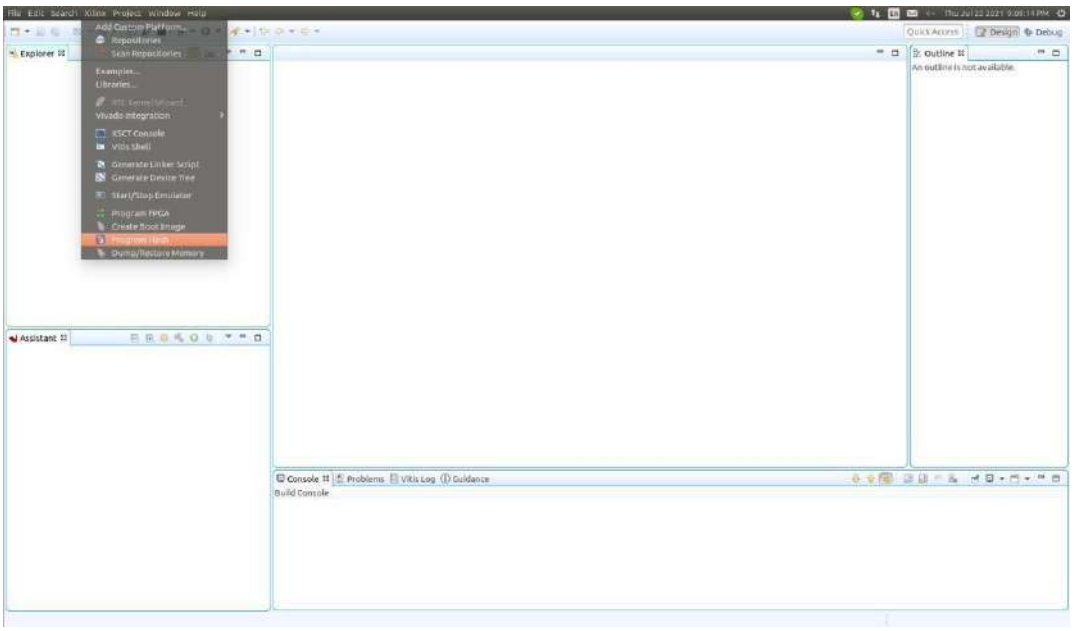


Figure 2: Xilinx Program Flash

- Select “Program Flash” and the following wizard will appear.

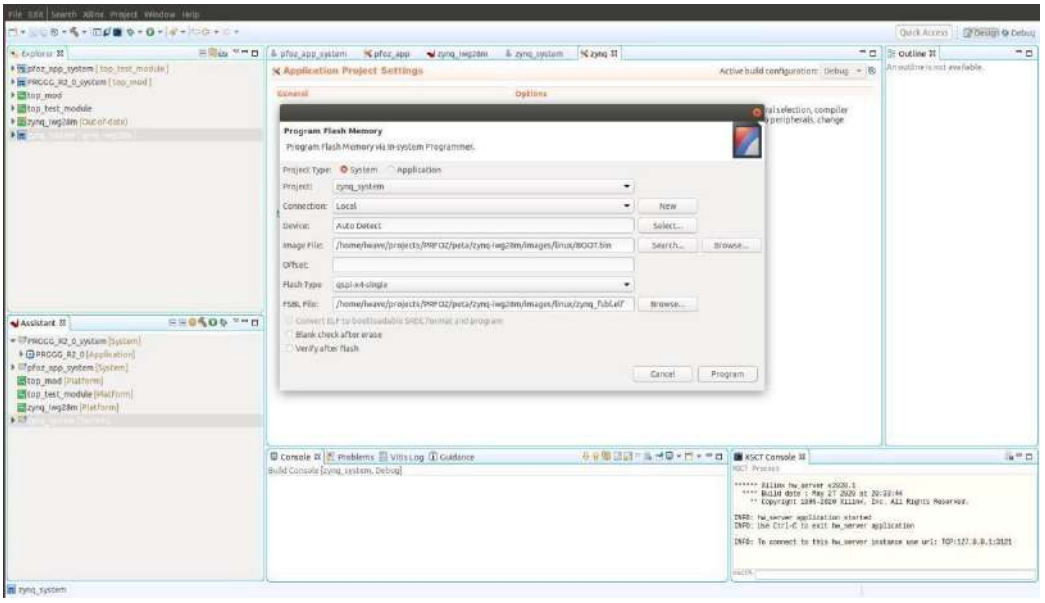


Figure 3: Xilinx Program Flash - Settings

- Select the settings as shown in the image above.
- Click on Image file browse and select BOOT.bin binary that has to be programmed.
- Click on FSBL File browse and select zynq_fsbl.elf file
- Click on program button to program the binaries to the iW-RainboW-G28M platform and programming will be done as shown below.

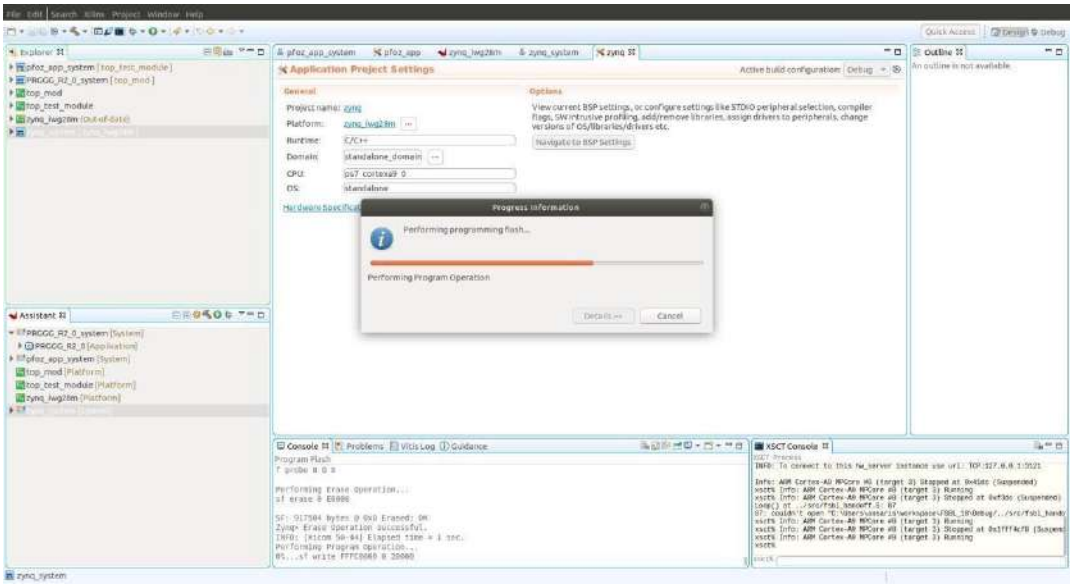


Figure 4: Program QSPI

- Once the flash programming is done, power off the board, remove the JTAG programming cable and the board will boot with the flashed binary when powered on again.

3.2 Manual Programming

This section explains the step-by-step procedure to program the Boot.bin to QSPI Flash and Bitstream and Linux binaries into iW-RainboW-G28M platform manually to eMMC using a USB memory stick/Micro SD. Refer this section only if any of the fsbl and u-boot binaries are already programmed and bootable from QSPI Flash.

3.1.1 Requirements

To program the binaries into iW-RainboW-G28M platform, following Items are required:

- USB memory stick & Host Cable or Micro SD or Ethernet Cable
- Host PC (Linux) for manual binary programming

Note: In iW-RainboW-G28M SOM, Micro SD and Wifi/BT are multiplexed in same pins and so either Micro SD or WiFi/BT is supported based on the SOM configuration.

3.1.2 Bootloader Programming to QSPI Programming Via U-boot

- To program the bootloader to QSPI from U-Boot console, you can either use USB memory stick or Micro SD or TFTP.
- Power on the board and enter into U-boot console. Make sure that the device with BOOT.bin binary is connected to the board.
- If USB is used, execute the below commands to copy the BOOT.bin from USB to RAM.

iWave-G28M> usb start

iWave-G28M> fatload usb 0 <ram_address> BOOT.bin

3.2.1.1 Example:

iWave-G28M> fatload usb 0 0x15000000 BOOT.bin

- If Micro SD is used, execute the below commands to copy the BOOT.bin from Micro SD to RAM.

iWave-G28M> mmc dev 0

iWave-G28M> fatload mmc 0 <ram_address> BOOT.bin

3.2.1.2 Example:

iWave-G28M> fatload mmc 0 0x15000000 BOOT.bin

- In case of TFTP, execute the below commands to load BOOT.bin from remote PC to RAM.

  iWave-G28M> tftpboot <ram_address> <server_ip>:BOOT.bin

- Once the binary is copied to RAM, execute the below commands to erase the QSPI flash and write the binary.

  iWave-G28M> sf probe

  iWave-G28M> sf erase 0x0 0x200000

  iWave-G28M> sf write <ram_address> 0x0 <file_size>

3.2.1.3 Example:

iWave-G28M> sf write 0x15000000 0x0 0xD2D90

- Now restart the board to boot the newly flashed bootloader.

Programming Via Linux

- To program the bootloader to QSPI flash from Linux console, make sure that the BOOT.bin file is present in either USB memory stick or Micro SD.
- Power on the board and enter into Linux console.
- The below cat command will list out all the partitions in QSPI flash.

  $ cat /proc/mtd

- To copy the file from USB to QSPI flash using command 'flashcp'

  $ flashcp /run/media/sda1/BOOT.bin /dev/mtd0

- To copy the file from Micro SD to QSPI flash, execute the below command

  $ flashcp /media/sd-mmcblk0pX/BOOT.bin /dev/mtd0

- Now restart the board to boot the newly flashed bootloader.

4. Environment configuration

4.1 Minicom/Tera Term configuration

1. Connect a serial cable between the UART connector on board and PC.
2. Run Minicom application on the PC and configure it with the following parameters.

***COM number may change**

```
A -    Serial Device      : /dev/ttyUSB0
B - Lockfile Location     : /var/lock
C -   Callin Program      :
D -  Callout Program      :
E -    Bps/Par/Bits       : 115200 8N1
F - Hardware Flow Control : No
G - Software Flow Control : No

   Change which setting?
```

Figure 5: Minicom settings

4.2 Hardware Setup

1. Copy the binaries image.ub and system.bit.bin to the required boot media
2. Connect the 5V @ 2500mA power supply.
3. Power on the board.
4. When powered on, you will see boot sequence messages on the Minicom / Tera Term console as shown in figure below. Observe NAND specific messages of detection, part number and size.

```
nand: device found, Manufacturer ID: 0x2c, Chip ID: 0xda
nand: Micron MT29F2G08ABAEAWP
nand: 256 MiB, SLC, erase size: 128 KiB, page size: 2048, OOB size: 64
nand: WARNING: MT29F2G08ABAEAWP: the ECC used on your system is too weak compared to the one required by the NAND chip
Bad block table found at page 131008, version 0x01
Bad block table found at page 130816, version 0x01
nand_read_bbt: bad block at 0x00000ffa0000
nand_read_bbt: bad block at 0x00000ffc0000
1 fixed-partitions partitions found on MTD device MT29F2G08ABAEAWP
Creating 1 MTD partitions on "MT29F2G08ABAEAWP":
0x000000000000-0x00000c800000 : "iwave-nand"
spi-nor spi0.0: is25lp016d (2048 Kbytes)
libphy: Fixed MDIO Bus: probed
CAN device driver interface
libphy: MACB_mii_bus: probed
Generic PHY e000b000.ethernet-ffffffff:03: attached PHY driver [Generic PHY] (mii_bus:phy_addr=e000b000.ethernet-ffffffff:03, irq=POLL)
macb e000b000.ethernet eth0: Cadence GEM rev 0x00020118 at 0xe000b000 irq 28 (00:01:02:03:04:05)
```

Figure 6: NAND detection print on boot log

5. NAND Utility Test Procedure

5.1 Block Erase Test

1. To test the block erase, type the command as below
   root@zynq-iwg28m:$ mtd_debug erase /dev/mtd0 0x00000000 0x400000
2. On completing erase observe the following
   Erased 4194304 bytes from address 0x00000000 in flash

```
root@NAND_BSP:~# mtd_debug erase /dev/mtd0 0x00000000 0x6800000
Erased 109051904 bytes from address 0x00000000 in flash
```

Figure 7: Block erase

5.2 Data Transfer Test

5.2.1 Write Page Test

1. Create a file with randomized data using the command below
   root@zynq-iwg28m:$ dd if=/dev/urandom of=/media/sd-mmcblk1p1/write.img bs=1024 count=8
2. Write the above created file to NAND using the below command
   root@zynq-iwg28m:$ mtd_debug write /dev/mtd0 0x00000000 0x2000 /media/sd-mmcblk1p1/write.img
3. On completion of write number of bytes copied are displayed
   Copied 8192 bytes from /media/sd-mmcblk1p1/write.img to address 0x00000000 in flash

```
root@NAND_BSP:~# mtd_debug write /dev/mtd0 0x00000000 0x2000 /media/sd-mmcblk1p1/write.img
Copied 8192 bytes from /media/sd-mmcblk1p1/write.img to address 0x00000000 in flash
root@NAND_BSP:~#
```

Figure 8: Page write

5.2.2 Read Page and Data Integrity Test

1. Read the file from NAND and store into a new file using the below command
   root@zynq-iwg28m:$ mtd_debug read /dev/mtd0 0x00000000 0x2000 /media/sd-mmcblk1p1/read.img
2. On read completion ensure that the number of bytes written are read back
   Copied 8192 bytes from address 0x00000000 in flash to /media/sd-mmcblk1p1/read.img
3. Compare the written and read files, ensure no differences are present, using the command
   root@zynq-iwg28m:$ cmp /media/sd-mmcblk1p1/write.img /media/sd-mmcblk1p1/read.img
4. Compare the file checksum using the below command and ensure both files have the same

root@zynq-iwg28m:$ md5sum /media/sd-mmcblk1p1/write.img /media/sd-mmcblk1p1/read.img

```
root@NAND_BSP:~# mtd_debug read /dev/mtd0 0x00000000 0x2000 /media/sd-mmcblk1p1/read.img
Copied 8192 bytes from address 0x00000000 in flash to /media/sd-mmcblk1p1/read.img
root@NAND_BSP:~#
root@NAND_BSP:~# cmp /media/sd-mmcblk1p1/write.img /media/sd-mmcblk1p1/read.img
root@NAND_BSP:~#
root@NAND_BSP:~#
root@NAND_BSP:~# md5sum /media/sd-mmcblk1p1/write.img /media/sd-mmcblk1p1/read.img
074cb148340913dc4cfbc3a2bb9d6d09  /media/sd-mmcblk1p1/write.img
074cb148340913dc4cfbc3a2bb9d6d09  /media/sd-mmcblk1p1/read.img
root@NAND_BSP:~#
```

Figure 9: Page read and compare

5.3 Data Integrity Using NANDTEST utility

1. To start the NAND Test Utility use the below command
   root@zynq-iwg28m:$ nandtest /dev/mtd0
2. This test performs erase, write, read and checks for data integrity for the full NAND device. Additionally scans for Bad Blocks and updates the Bad Block Table
3. On completion of the test, check whether it passed or failed 1.

```
root@NAND_BSP:~# nandtest /dev/mtd0
ECC corrections: 0
ECC failures   : 0
Bad blocks     : 0
BBT blocks     : 0
0c7e0000: checking...of 4)...
Finished pass 1 successfully
root@NAND_BSP:~#
```

Figure 10: NANDTEST utility status

5.4 JFFS2 filesystem

- **Erase the flash partition: flash_erase -j <device node> 0 0 EX:**
  root@zynq-iwg28m:$ flash_erase -j /dev/mtd0 0 0
- **Mount the partition: mount -t jffs2 mtd<X> /mnt EX:**
  root@zynq-iwg28m:$ mount -t jffs2 mtd0 /mnt
- **Perform file read/write and compare operation:**
  1. Create any random large file (less than 200MB) on already mounted eMMC partition.
  2. Copy the file from eMMC partition to NAND mtd mounted portion as shown below root@zynq-iwg28m:$ cp /media/sd-mmcblk1p1/<file - 1> /mnt
  3. Copy back the data from mtd mounted partition to eMMC mounted partition root@root:$ cp /mnt/<file - 1> /media/sd-mmcblk1p1/<file - 2>
  4. Compare the written and read files, ensure no differences are present, using the command root@zynq-iwg28m:$ cmp /media/sd-mmcblk1p1/<file - 1> /media/sd-mmcblk1p1/<file - 2>
  5. Compare the file checksum using the below command and ensure both files have the same root@zynq-iwg28m:$ md5sum /mnt/<file - 1> /media/sd-mmcblk1p1/<file - 2>
- **Unmount the partition:**
  umount <directory>

  EX:
  root@root:$ umount /mnt

```
root@NAND_BSP:~# mount -t jffs2  /dev/mtdblock0  /mnt
root@NAND_BSP:~#
root@NAND_BSP:~# cp /media/sd-mmcblk1p1/image.ub /mnt/image.ub
root@NAND_BSP:~# sync
root@NAND_BSP:~#
root@NAND_BSP:~# cmp /media/sd-mmcblk1p1/image.ub /media/sd-mmcblk1p1/image_read.ub
root@NAND_BSP:~# sync
root@NAND_BSP:~#
root@NAND_BSP:~# cmp /media/sd-mmcblk1p1/image.ub /media/sd-mmcblk1p1/image_read.ub
root@NAND_BSP:~# sync
root@NAND_BSP:~#
root@NAND_BSP:~# md5sum /media/sd-mmcblk1p1/image.ub /media/sd-mmcblk1p1/image_read.ub
acbc5b2e9494c49e0999029e6cee654b  /media/sd-mmcblk1p1/image.ub
acbc5b2e9494c49e0999029e6cee654b  /media/sd-mmcblk1p1/image_read.ub
```

Figure 11: Mounting JFFS2 and Read/Write

5.5 UBIFS filesystem

UBIFS may be considered as the next generation of the JFFS2 file-system.

- **Erase your flash partition while preserving your erase counters:**
  ubiformat /dev/mtdX
  here 'X' is the mtd partition number
  EX:
  root@zynq-iwg28m:$ ubiformat /dev/mtd0

- **Attach UBI to one (of several) of the MTD partitions:**
  ubiattach /dev/ubi_ctrl -m X

5.5.1 EX:

root@zynq-iwg28m:$ ubiattach /dev/ubi_ctrl -m 0

- **Volume creation with ubimkvol:**
  root@zynq-iwg28m:$ ubimkvol /dev/ubi0 -N test -s 116MiB
- **When a UBI volume is created, creating an empty UBIFS: filesystem is just a matter of mounting it**
  root@zynq-iwg28m:$ mount -t ubifs ubi0:test /mnt/
- **Perform file read/write and compare operation:**
  6. Create any random large file (less than 200MB) on already mounted eMMC partition.
  7. Copy the file from eMMC partition to NAND mtd mounted portion as shown below
     root@zynq-iwg28m:$ cp /media/sd-mmcblk1p1/<file - 1> /mnt
  8. Copy back the data from mtd mounted partition to eMMC mounted partition
     root@zynq-iwg28m:$ cp /mnt/<file - 1> /media/sd-mmcblk1p1/<file - 2>
  9. Compare the written and read files, ensure no differences are present, using the command
     root@zynq-iwg28m:$ cmp /media/sd-mmcblk1p1/<file - 1> /media/sd-mmcblk1p1/<file - 2>
  10. Compare the file checksum using the below command and ensure both files have the same
      root@zynq-iwg28m:$ md5sum /media/sd-mmcblk1p1/<file - 1> /media/sd-mmcblk1p1/<file - 2>
- **Unmount the partition:**
  umount
  <directory> EX:
  root@zynq-iwg28m:$ umount /mnt

Figure 12: Mounting UBIFS


iWave
Embedding Intelligence