# i.MX8MSCALE DDR Tool

## User’s Guide

**Document Number: IMX**
**Rev. V2.0.0**
**1/23/2021**

*How to Reach Us:*

**Home Page**:
www.nxp.com

**Web Support**:
www.nxp.com/support

Information in this document is provided solely to enable system and software implementers to use Freescale products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document.

Freescale reserves the right to make changes without further notice to any products herein. Freescale makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters that may be provided in Freescale data sheets and/or specifications can and do vary in different applications, and actual performance may vary over time. All operating parameters, including “typicals,” must be validated for each customer application by customer’s technical experts. Freescale does not convey any license under its patent rights nor the rights of others. Freescale sells products pursuant to standard terms and conditions of sale, which can be found at the following address: freescale.com/SalesTermsandConditions.

# Contents

# Chapter 1
# Introduction

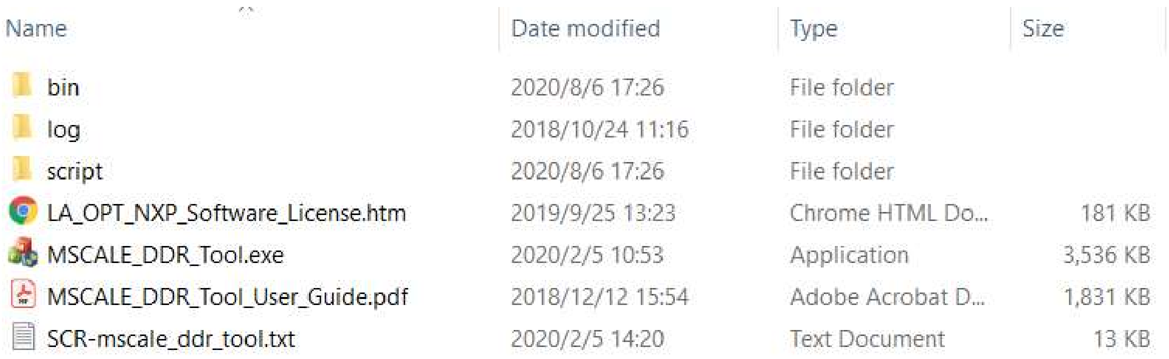
i.MX8/MSCALE DDR Tool is a software application to verify DDR performance on i.MX8/MSCALE series boards. It is a program running on the PC side which downloads a test image to the i.MX series processor’s internal RAM through a USB connection. Because it needs to access Windows Registry, user must run it in administrator mode. The test image running on the target board executes the DDR training, code generation and stress test. The result is sent to the PC via the A-core UART and is displayed in the log window. There is also an option to save the output to a log file.

i.MX8/MSCALE DDR Tool can help verify DDR stability on the board in a non-OS environment.

# Chapter 2
# Installation and Setup

## 2.1 Installation

You can run mscale_ddr_tool_installation.exe to install this application. After installation, the following directory structure will be created:



| Name | Date modified | Type | Size |
|---|---|---|---|
| bin | 2020/8/6 17:26 | File folder | |
| log | 2018/10/24 11:16 | File folder | |
| script | 2020/8/6 17:26 | File folder | |
| LA_OPT_NXP_Software_License.htm | 2019/9/25 13:23 | Chrome HTML Do... | 181 KB |
| MSCALE_DDR_Tool.exe | 2020/2/5 10:53 | Application | 3,536 KB |
| MSCALE_DDR_Tool_User_Guide.pdf | 2018/12/12 15:54 | Adobe Acrobat D... | 1,831 KB |
| SCR-mscale_ddr_tool.txt | 2020/2/5 14:20 | Text Document | 13 KB |

- MSCALE_DDR_Tool.exe: PC software
- MSCALE_DDR_Tool_User_Guide.pdf: This file
- SCR-mscale_ddr_tool.txt: copyright file
- Directory bin: Executable binaries for MX8MSCALE inside
- Directory log: test logs inside
- Directory script: board initialization script files inside

## 2.2 System Requirements

- Minimum PC Requirements – 2.0 GHz CPU, 1GB RAM with USB connection.

- Windows® XP w/Service Pack 2 or later

# Chapter 3
# Running DDR Tool

To run the DDR Stress Tester, perform the following steps:

1. Connect the target board to PC host

   a. Configure the i.MX target board to boot in Serial Download mode/Manufacture mode and power up the board.

   b. Connect a UART cable from the host computer to the MX8MSCALE debug UART.

   c. Connect a USB cable from the host computer to the USB OTG port on the MX8MSCALE target board. A “HID-compliant device” or “USB Input Device” will be shown in the Device Manager as Figure 3-1:

   *Note: For the MX8MSCALE USB OTG connection, the USB cable must be connected directly to the Host PC USB port and not through a USB HUB.*

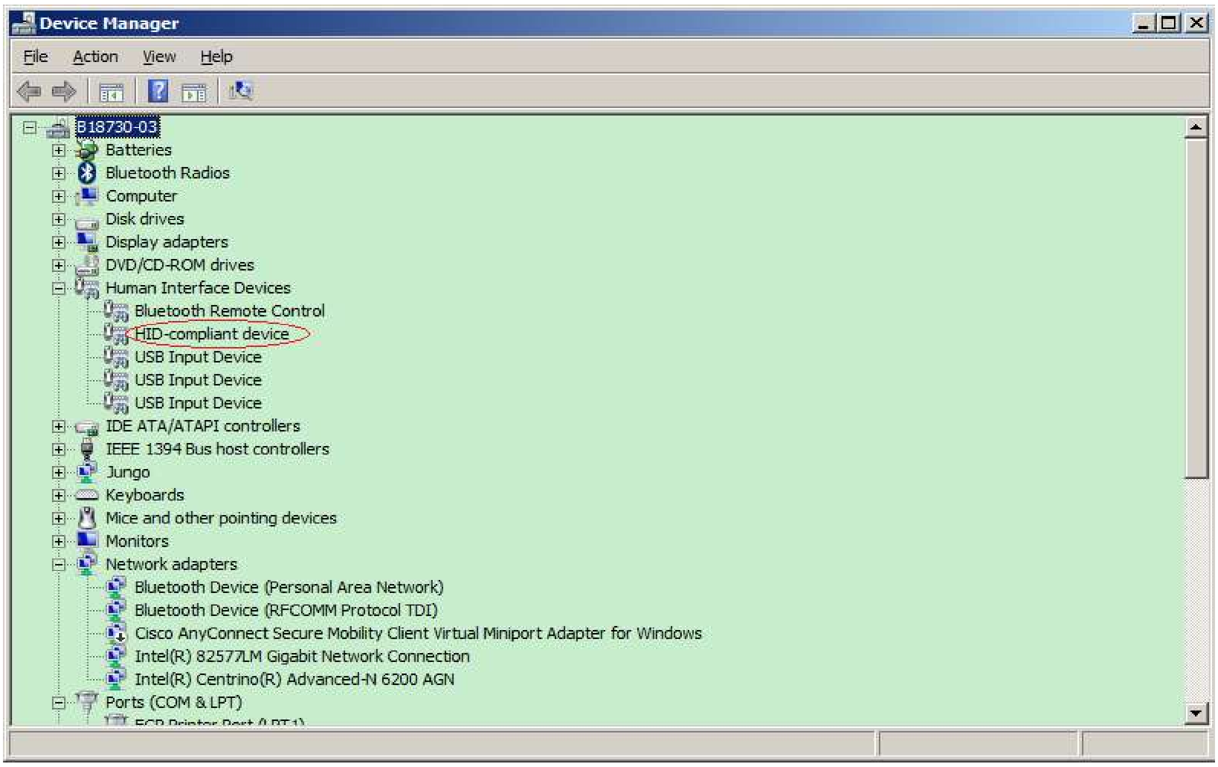
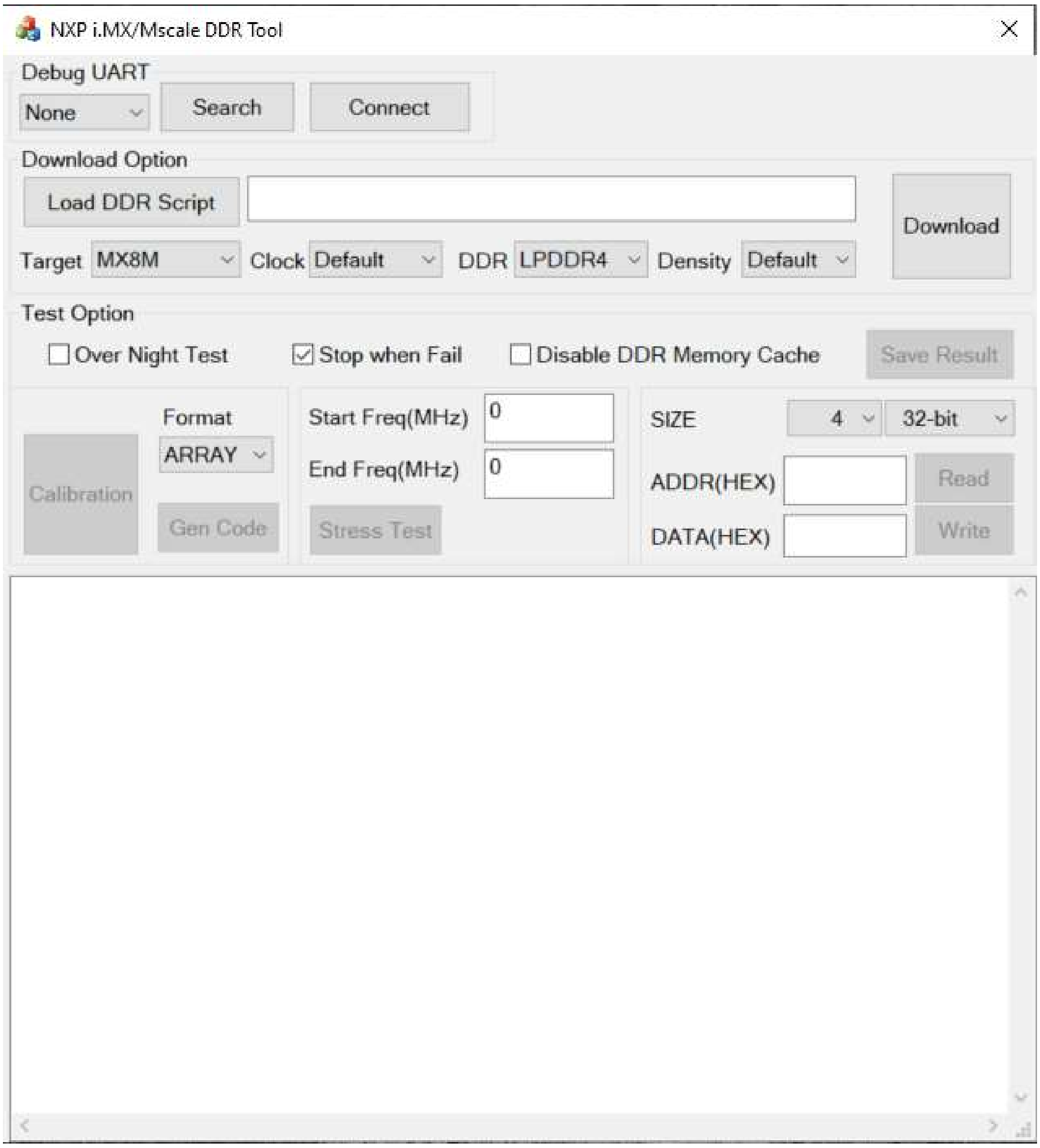

Device Manager
File
Action
View
Help
B18730-03
Batteries
Bluetooth Radios
Computer
Disk drives
Display adapters
DVD/CD-ROM drives
Human Interface Devices
Bluetooth Remote Control
HID-compliant device
USB Input Device
USB Input Device
USB Input Device
IDE ATA/ATAPI controllers
IEEE 1394 Bus host controllers
Jungo
Keyboards
Mice and other pointing devices
Monitors
Network adapters
Bluetooth Device (Personal Area Network)
Bluetooth Device (RFCOMM Protocol TDI)
Cisco AnyConnect Secure Mobility Client Virtual Miniport Adapter for Windows
Intel(R) 82577LM Gigabit Network Connection
Intel(R) Centrino(R) Advanced-N 6200 AGN
Ports (COM & LPT)
ECP Printer Port (LPT1)

**Figure 3-1 HID compliant device**

2. Launch the MSCALE_DDR_Tool.exe in **administrator mode**



Debug UART: None

Target: MX8M  Clock: Default  DDR: LPDDR4  Density: Default

☐ Over Night Test  ☑ Stop when Fail  ☐ Disable DDR Memory Cache

Format: ARRAY

Start Freq(MHz): 0

End Freq(MHz): 0

SIZE: 4  32-bit

**Figure 3-2 MSCALE DDR Tool UI**

### 1) Debug UART AREA

- **UART Drop List:** display all the UART ports in your computer.
- **Search Button**: search current UART ports in your computer and list them in the drop list.
- **Connect Button**: connect UART port displayed in the drop list.

2) **Download Option AREA**

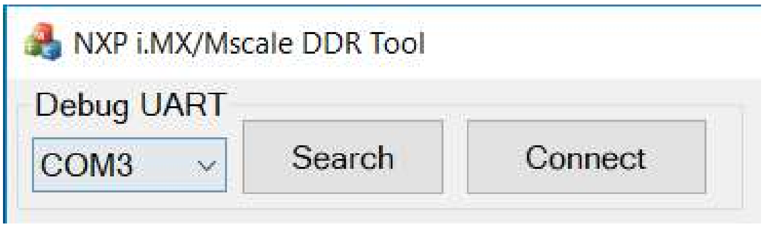
- **Load DDR Script**: you can choose a DDR initialization script to run before downloading an image to the MX8 board. If you don't want to execute a script, then leave the script text input blank.
- **TARGET Drop List**: choose MX8M test target
- **CLOCK Drop List**: choose available ARM CPU clock. 'Default' means to use initial ARM clock.
- **DDR Type Drop List**: choose board specific LPDDR4/DDR4/DDR3L type
- **DDR Density Drop List**: choose available DDR density. 'Default' means to test all DDR memory configured in the script.
- **Download Button**: If DDR script file exists, then select it first. Then download board specific image based on selected options.

3) **Test Option AREA**

- **Over Night Test Check Box:** if this option is selected, the stress test loops again and again and never stops, otherwise the stress test would run only once.
- **Stop when Fail Check Box**: if this option is selected, the stress test would stop if there is an error, otherwise the stress test would continue to run.
- **Disable DDR Memory Cache**: if this option is selected, DDR memory is mapped as none-cacheable/none-buffable memory when you run stress test, throughput test and memory read/write test, otherwise DDR memory is mapped as cacheable/buffable memory.
- **Format Drop List:** You can choose 'ARRAY' format for the u-boot SPL driver style or 'CODE' format for a complete DDR initialization code when use '**Gen Code**' button to generate DDR initial file.
- **Calibration and Gen Code buttons**: You must do calibration before code generation and DDR stress test for MX8M. After calibration is done, you can press '**Gen Code**' button to generate MX8M u-boot SPL DDR initialization code.

- **Start/End Freq Text Input**: You can input start and end frequency you want to test in stress or throughput test. You can leave them to 0, which means DDR works at the initial frequency set in the script.  If selecting a frequency range to test, ensure that the start frequency test is within ±50MHz of the frequency that the DDR initialization script is tuned for and make sure end frequency is no less than start frequency and within 100Mhz of the start frequency.
- **Stress Test Button**: perform stress test function.
- **ADDR/DATA HEX Text Input:** You can read/write memory in 8bit/16bit/32bit/64bit mode
- **Read/Write Button:** perform memory read/write function

3. Press **Search** button in the **Debug UART area**, then choose the correct UART port connected to the MX8 Cortex A-Core Debug UART and press **Connect** button.

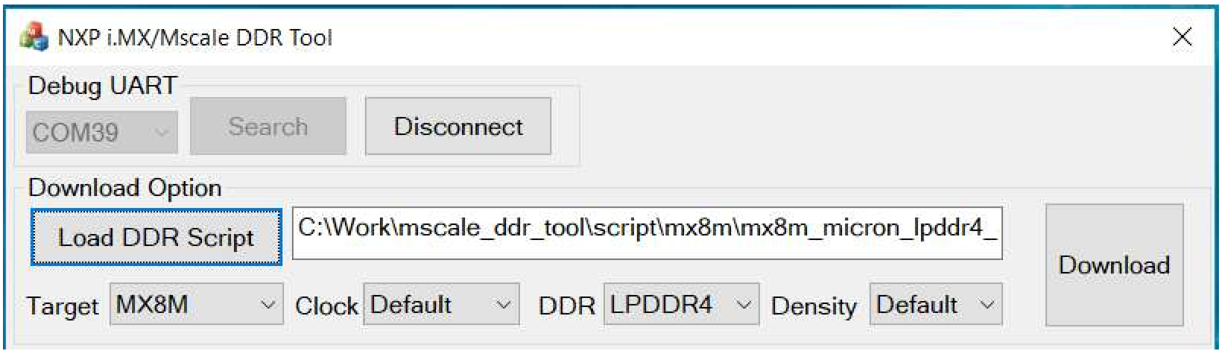


NXP i.MX/Mscale DDR Tool

Debug UART: COM3

**Figure 3-3 Connect to MX8 A-Core Debug UART**

4. Load DDR initialization script and choose correct downloading options



NXP i.MX/Mscale DDR Tool

Debug UART: COM39

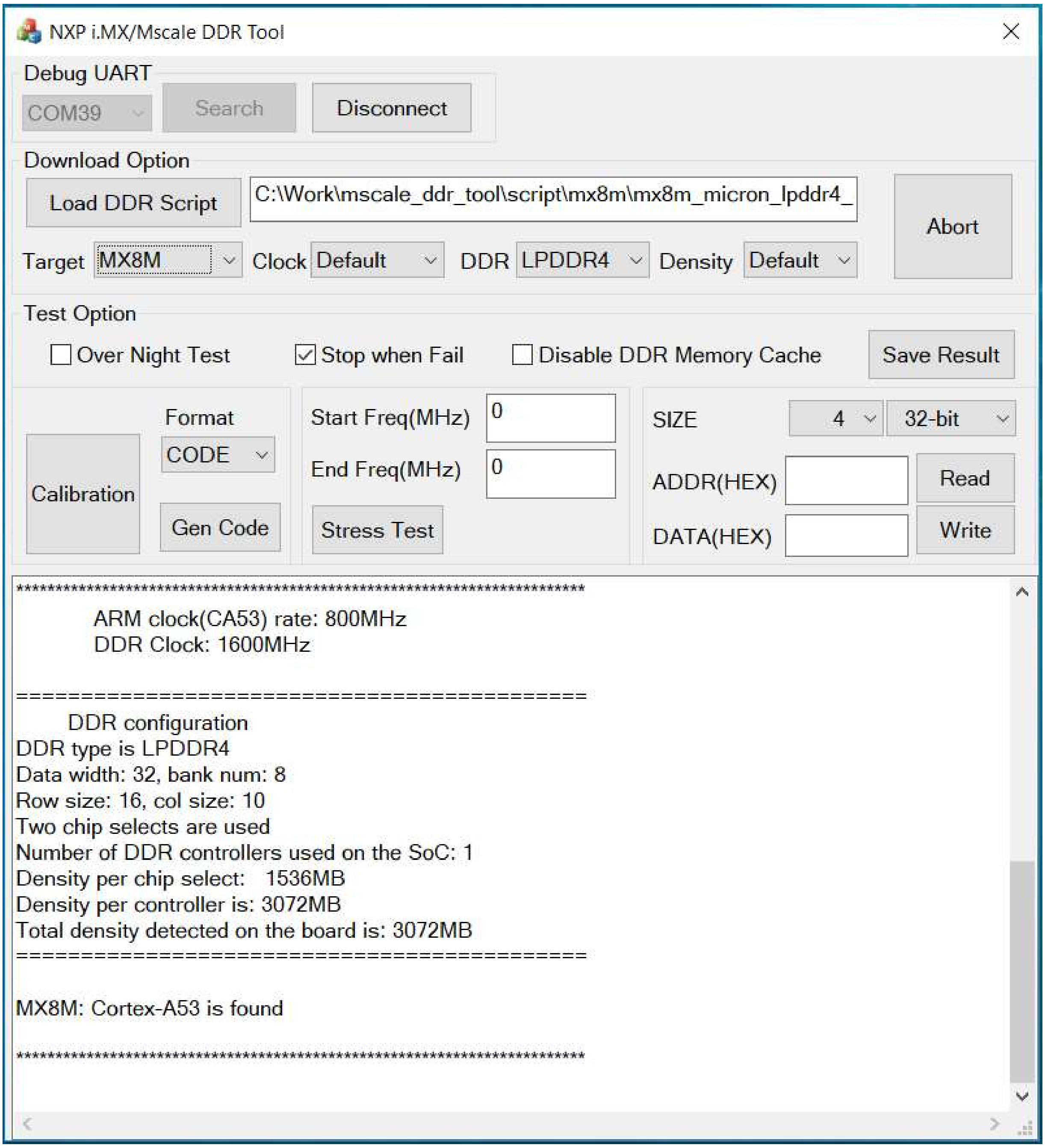
Download Option

C:\Work\mscale_ddr_tool\script\mx8m\mx8m_micron_lpddr4_

Target MX8M Clock Default DDR LPDDR4 Density Default

**Figure 3-4 Choose all download options**

5. Press **Download** button and wait for target board ready



COM39

C:\Work\mscale_ddr_tool\script\mx8m\mx8m_micron_lpddr4_

Target MX8M  Clock Default  DDR LPDDR4  Density Default

□ Over Night Test  ☑ Stop when Fail  □ Disable DDR Memory Cache

Format CODE

Start Freq(MHz) 0

End Freq(MHz) 0

SIZE 4  32-bit

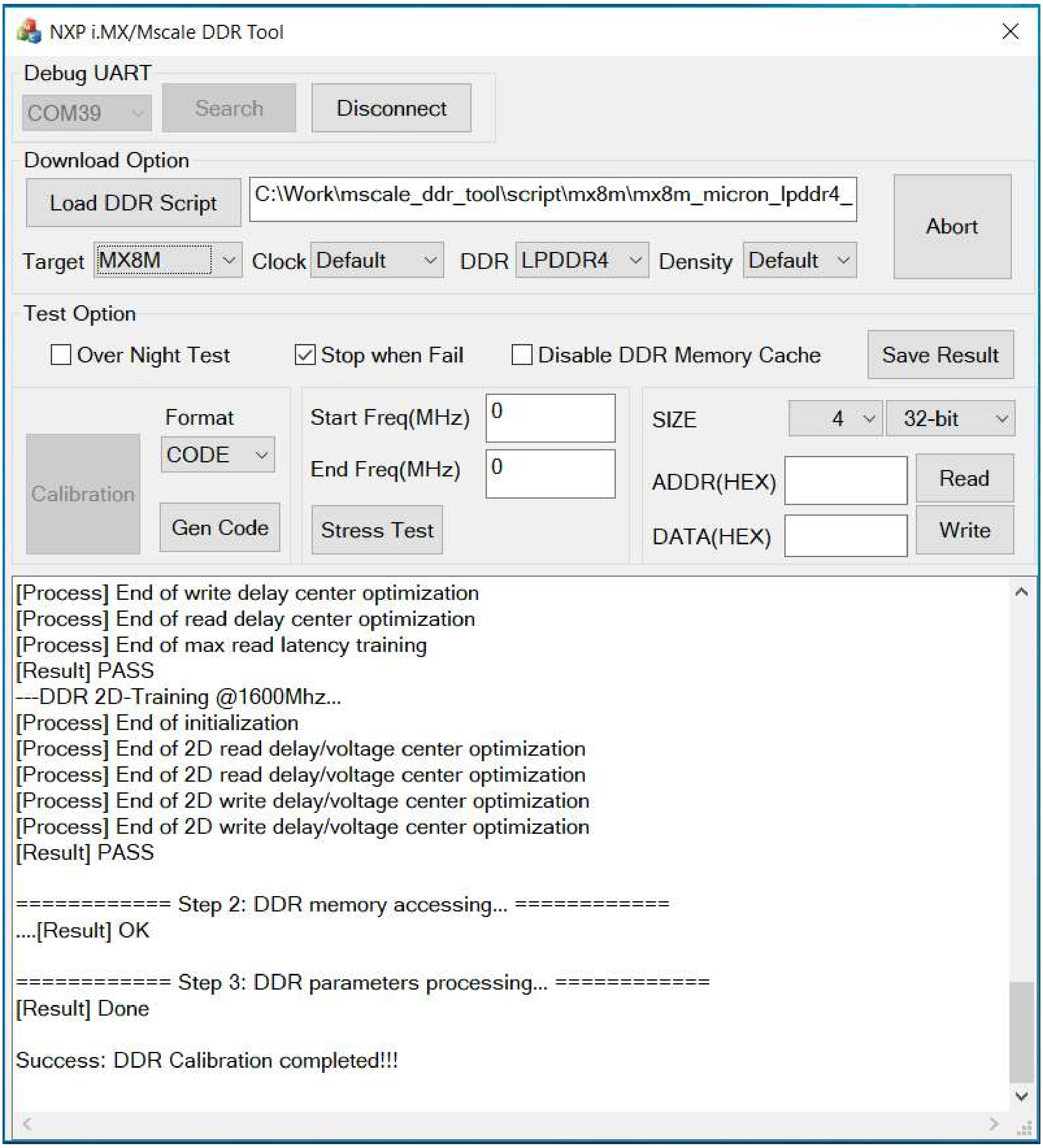
ADDR(HEX)

DATA(HEX)

```
*****************************************************************************
          ARM clock(CA53) rate: 800MHz
          DDR Clock: 1600MHz

============================================
       DDR configuration
DDR type is LPDDR4
Data width: 32, bank num: 8
Row size: 16, col size: 10
Two chip selects are used
Number of DDR controllers used on the SoC: 1
Density per chip select:   1536MB
Density per controller is: 3072MB
Total density detected on the board is: 3072MB
============================================

MX8M: Cortex-A53 is found

*****************************************************************************
```

Figure 3-5 Target board is ready

6. Press **Calibration** button to do DDR training



COM39

C:\Work\mscale_ddr_tool\script\mx8m\mx8m_micron_lpddr4_

Target MX8M Clock Default DDR LPDDR4 Density Default

□ Over Night Test ☑ Stop when Fail □ Disable DDR Memory Cache

Format CODE

Start Freq(MHz) 0

End Freq(MHz) 0

SIZE 4 32-bit

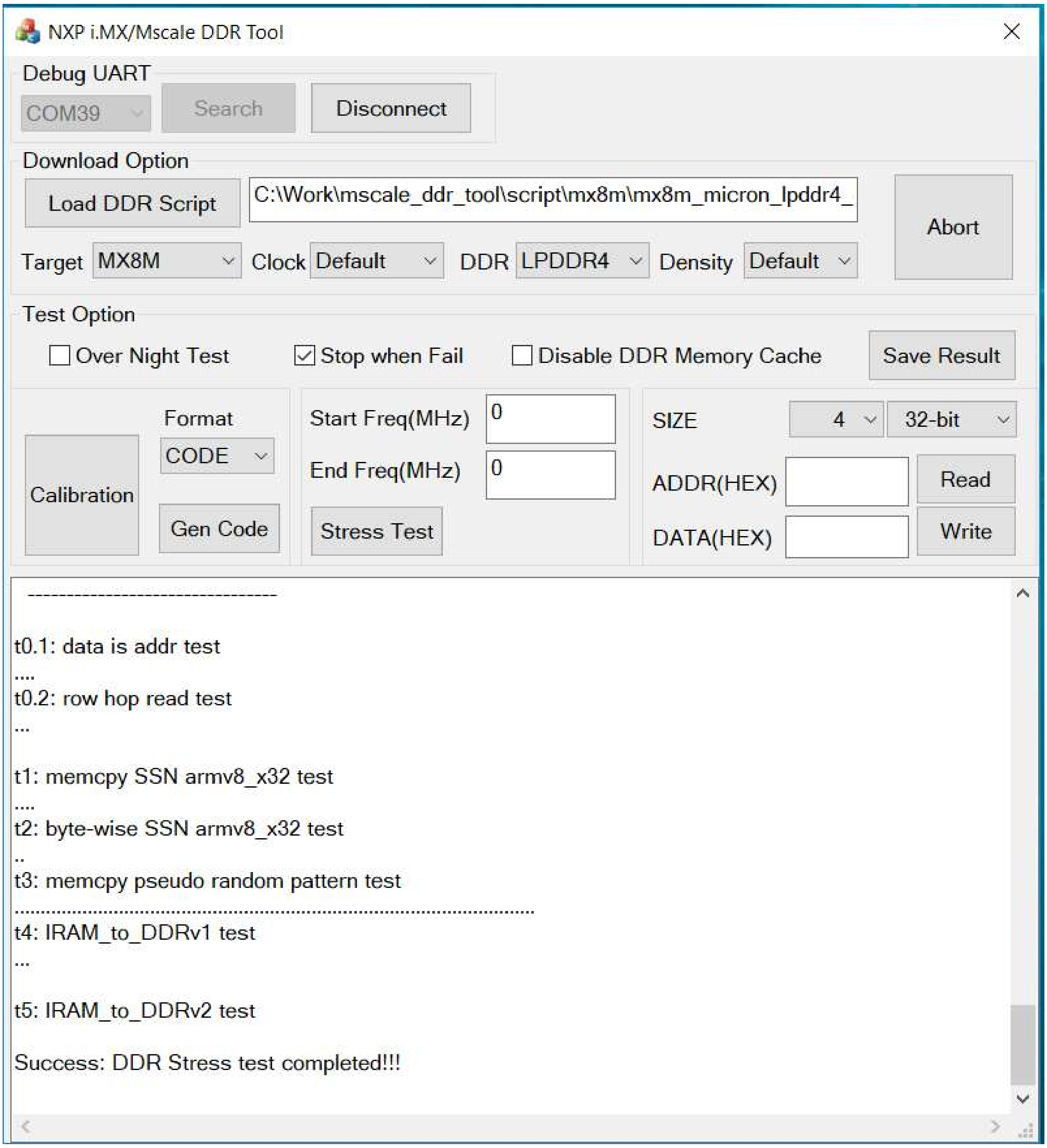
ADDR(HEX)

DATA(HEX)

```
[Process] End of write delay center optimization
[Process] End of read delay center optimization
[Process] End of max read latency training
[Result] PASS
---DDR 2D-Training @1600Mhz...
[Process] End of initialization
[Process] End of 2D read delay/voltage center optimization
[Process] End of 2D read delay/voltage center optimization
[Process] End of 2D write delay/voltage center optimization
[Process] End of 2D write delay/voltage center optimization
[Result] PASS

============ Step 2: DDR memory accessing... ============
....[Result] OK

============ Step 3: DDR parameters processing... ============
[Result] Done

Success: DDR Calibration completed!!!
```

**Figure 3-6 DDR Calibration**

7. Press **Stress Test** button (Use all default settings: default DDR frequency, cache enabled, one loop DDR stress test, stop when encounter error)



Debug UART: COM39

C:\Work\mscale_ddr_tool\script\mx8m\mx8m_micron_lpddr4_

Target MX8M   Clock Default   DDR LPDDR4   Density Default

□ Over Night Test   ☑ Stop when Fail   □ Disable DDR Memory Cache

Format CODE

Start Freq(MHz) 0

End Freq(MHz) 0

SIZE 4   32-bit

ADDR(HEX)

DATA(HEX)

```
 --------------------------------

t0.1: data is addr test
....
t0.2: row hop read test
...

t1: memcpy SSN armv8_x32 test
....
t2: byte-wise SSN armv8_x32 test
..
t3: memcpy pseudo random pattern test
..........................................................................................................
t4: IRAM_to_DDRv1 test
...

t5: IRAM_to_DDRv2 test

Success: DDR Stress test completed!!!
```

**Figure 3-7 DDR Stress Test**

8. Choose ‘**CODE**’ format or ‘**ARRAY**’ format from the dropdown list. Please choose ‘**ARRAY**’ format for the latest u-boot SPL driver. If you are using old u-boot code or study the complete DDR initialization flow, please choose ‘**CODE**’ format.

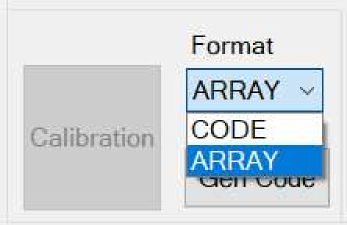
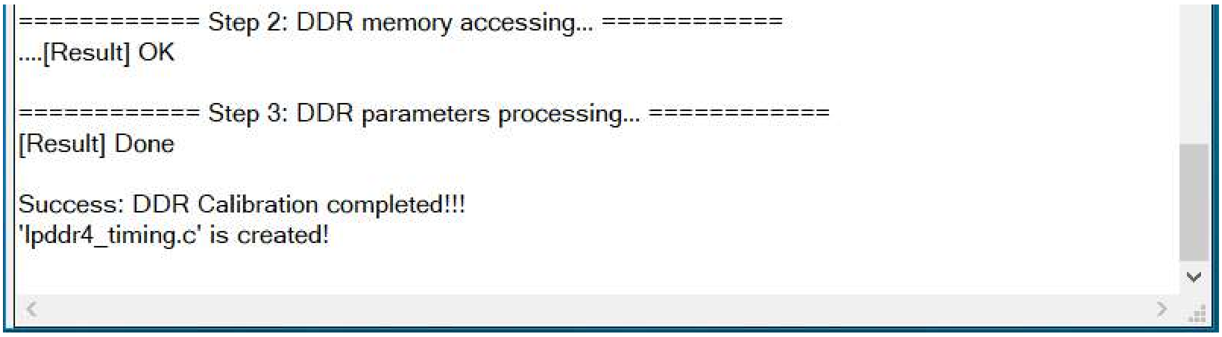


**Figure 3-8 Generated Code Format**

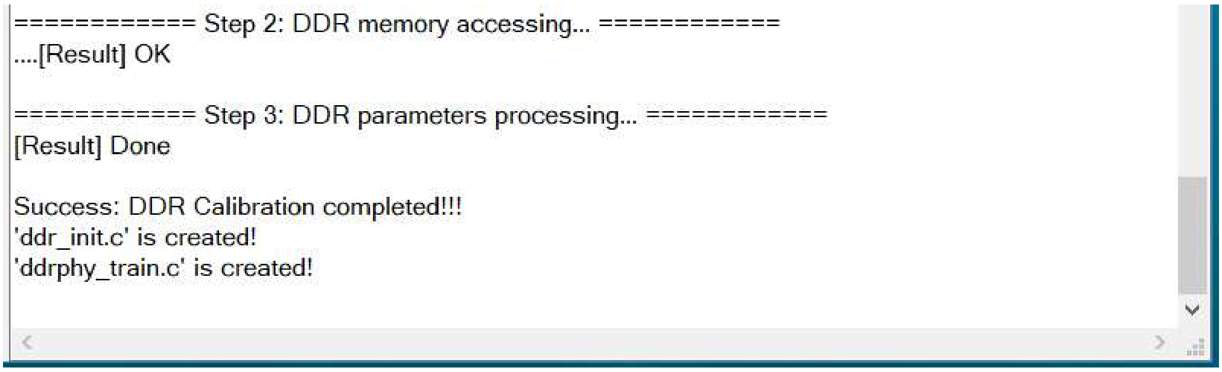
9. Press **Gen Code** button to generate u-boot SPL DDR initialization code. For ‘ARRAY’ format, you can get ‘xxx_timing.c’ in the tool’s directory.



```
============ Step 2: DDR memory accessing... ============
....[Result] OK

============ Step 3: DDR parameters processing... ============
[Result] Done

Success: DDR Calibration completed!!!
'lpddr4_timing.c' is created!
```

**Figure 3-9a ARRAY format file**

For ‘CODE’ format, you can get ‘ddrc_init.c’ and ‘ddrphy_train.c’ in the tool’s directory.



```
============ Step 2: DDR memory accessing... ============
....[Result] OK

============ Step 3: DDR parameters processing... ============
[Result] Done

Success: DDR Calibration completed!!!
'ddr_init.c' is created!
'ddrphy_train.c' is created!
```

**Figure 3-9b CODE format file**

# Chapter 4
# How to bring up a new MX8MSCALE board

When you design a new MX8MSCALE board, please make sure to follow the rules to use MX8MSCALE DDR tool. With the tool help, you can easily bring up DDR devices on the MX8MSCALE board, otherwise, you may pay much more effort to bring up the board.

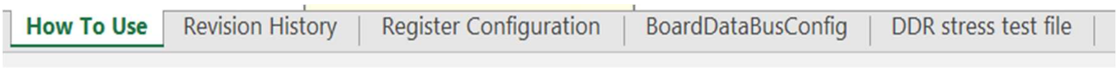
- Reserve MX8MSCALE serial download mode and USB OTG port.
- Reserve the debug UART as MX8MSCALE EVK board does. If you have a different debug UART, please follow chapter 4.2 to configure your own debug UART in DDR script.
- DDR Tool configures DDR power supply as NXP EVK board does. If you have a different power solution, please follow chapter 4.3 to customize your DDR power supply in DDR script.

This chapter introduces how to customize your DDR script and bring up DDR devices on your MX8MSCALE board.

## 4.1 Customize board specific DDR script

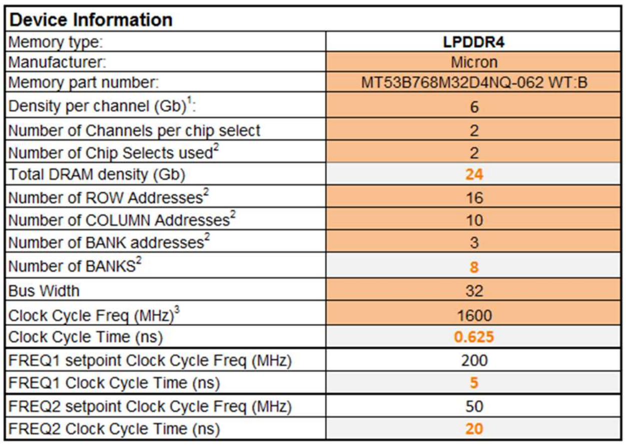
1. **Generate DDR script from Register Programming Aid tool (RPA)**

   You can get latest RPA tool from community or NXP official website and you also need to read DDR datasheet and board schematic to get enough information required by RPA tool. You can open RPA tool and switch to worksheet tab “How To Use” for more details.



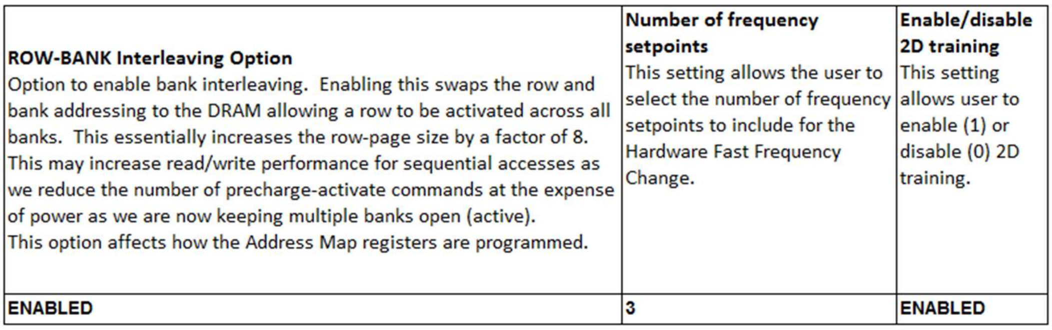
   Step 1. Obtain the desired DRAM data sheet from the DRAM vendor and switch to worksheet tab “Register Configuration”

Step 2. Update the Device Information table to include the DRAM information and system usage.



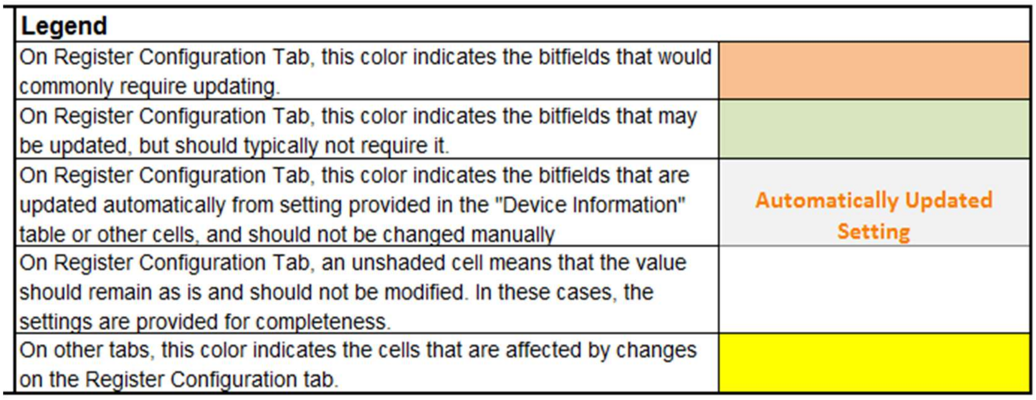
| Device Information | |
|---|---|
| Memory type: | LPDDR4 |
| Manufacturer: | Micron |
| Memory part number: | MT53B768M32D4NQ-062 WT:B |
| Density per channel (Gb)[1]: | 6 |
| Number of Channels per chip select | 2 |
| Number of Chip Selects used[2] | 2 |
| Total DRAM density (Gb) | 24 |
| Number of ROW Addresses[2] | 16 |
| Number of COLUMN Addresses[2] | 10 |
| Number of BANK addresses[2] | 3 |
| Number of BANKS[2] | 8 |
| Bus Width | 32 |
| Clock Cycle Freq (MHz)[3] | 1600 |
| Clock Cycle Time (ns) | 0.625 |
| FREQ1 setpoint Clock Cycle Freq (MHz) | 200 |
| FREQ1 Clock Cycle Time (ns) | 5 |
| FREQ2 setpoint Clock Cycle Freq (MHz) | 50 |
| FREQ2 Clock Cycle Time (ns) | 20 |

Step 3. Select DDR features as you want



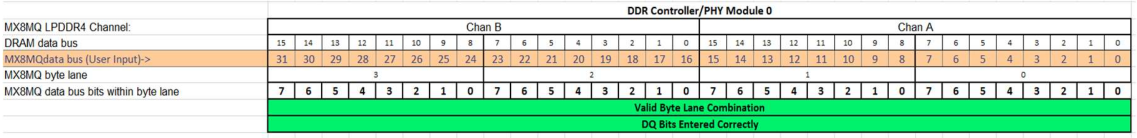
| **ROW-BANK Interleaving Option**<br>Option to enable bank interleaving. Enabling this swaps the row and bank addressing to the DRAM allowing a row to be activated across all banks. This essentially increases the row-page size by a factor of 8. This may increase read/write performance for sequential accesses as we reduce the number of precharge-activate commands at the expense of power as we are now keeping multiple banks open (active).<br>This option affects how the Address Map registers are programmed. | **Number of frequency setpoints**<br>This setting allows the user to select the number of frequency setpoints to include for the Hardware Fast Frequency Change. | **Enable/disable 2D training**<br>This setting allows user to enable (1) or disable (0) 2D training. |
|---|---|---|
| **ENABLED** | **3** | **ENABLED** |

Step 4. Go through the various shaded cells in the spread sheet to update with data from the DRAM sheet (take special note of the “Legend” table to ascertain the meaning of different shaded cells; in many cases, the cells may not need to be updated).



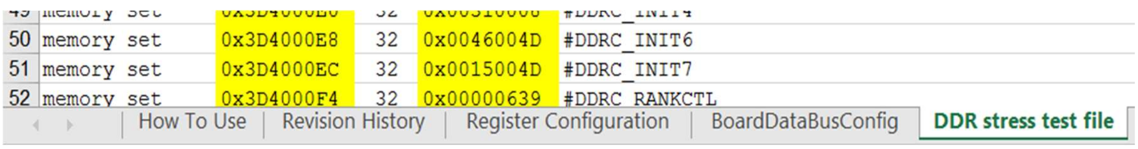
| **Legend** | |
|---|---|
| On Register Configuration Tab, this color indicates the bitfields that would commonly require updating. | 🟧 |
| On Register Configuration Tab, this color indicates the bitfields that may be updated, but should typically not require it. | 🟩 |
| On Register Configuration Tab, this color indicates the bitfields that are updated automatically from setting provided in the "Device Information" table or other cells, and should not be changed manually | Automatically Updated Setting |
| On Register Configuration Tab, an unshaded cell means that the value should remain as is and should not be modified. In these cases, the settings are provided for completeness. | |
| On other tabs, this color indicates the cells that are affected by changes on the Register Configuration tab. | 🟨 |

Step5. Switch to worksheet tab “BoardDataBusConfig” to check data bus assignment.



| | DDR Controller/PHY Module 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MX8MQ LPDDR4 Channel: | Chan B | | | | | | | | | | | | | | | | Chan A | | | | | | | | | | | | | | | |
| DRAM data bus | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| MX8MQdata bus (User Input)-> | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| MX8MQ byte lane | 3 | | | | | | | | 2 | | | | | | | | 1 | | | | | | | | 0 | | | | | | | |
| MX8MQ data bus bits within byte lane | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | Valid Byte Lane Combination | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | DQ Bits Entered Correctly | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

Step6. Switch to worksheet tab “DDR Stress test file” and copy and paste this into a text document. Make sure to rename to .ds file type.



```
memory set    0x3D4000E0    32    0x00310008    #DDRC_INIT4
memory set    0x3D4000E8    32    0x0046004D    #DDRC_INIT6
memory set    0x3D4000EC    32    0x0015004D    #DDRC_INIT7
memory set    0x3D4000F4    32    0x00000639    #DDRC_RANKCTL
```

2. **Generate DDR script based on current DDR script**

   If you are very familiar with DDR and MSCALE DDR controller or your board design is nearly the same as NXP MX8MSCALE reference design, you can simply modify some registers in the existing DDR script.

## 4.2 Customize debug UART in DDR script.

If you have the same debug UART port as NXP EVK does, please skip this chapter.

You must configure debug UART IOMUX and debug port in the beginning of the DDR script according to your schematics. Here is the sample commands used by i.MX8MP LPDDR4 EVK board.

```
################step 0: configure debug uart port. Assumes use of UART IO Pads.   #####

##### If using non-UART pads (i.e. using other pads to mux out the UART signals), #####

##### then it is up to the user to overwrite the following IO register settings   #####


memory set      0x30330228     32     0x00000000     #IOMUXC_SW_MUX_UART2_RXD

memory set      0x3033022C     32     0x00000000     #IOMUXC_SW_MUX_UART2_TXD

memory set      0x30330488     32     0x00000016     #IOMUXC_SW_PAD_UART2_RXD

memory set      0x3033048C     32     0x00000016     #IOMUXC_SW_PAD_UART2_TXD

memory set      0x303305F0     32     00000006       #IOMUXC_SW_MUX_UART2_SEL_RXD

sysparam set    debug_uart    1  #UART index from 0 ('0' = UART1, '1' = UART2, '2' = UART3, '3' = UART4)
```

## 4.3 Customize PMIC I2C and DDR power supply

If you have the same PMIC and power supply as NXP EVK does or you have simple DC/DC power supply for DDR part, please skip this chapter.

- you can use **pmic_cfg** and **pmic_set** script commands to do PMIC initialization.

  **pmic_cfg** : bit[7:0] = I2C slave address, bit[15:8]=I2C Bus (0 for I2C1, 1 for I2C2 …)

  **pmic_set :** bit[7:0] = PMIC register value, bit[15:8] = PMIC register address

- If you have **pmic_cfg/pmic_set** commands in the script, DDR tool will skip all default PMIC initialization process. Please handle the full of I2C IOMUX and PMIC initialization in the DDR script.

Please add I2C IOMUX and PMIC initialization in the beginning of the DDR script. It is OK to place the commands before or after debug UART initialization. Here is the sample commands for i.MX8MP PMIC initialization.

```
memory set     0x30330214    32     0x00000010     #IOMUXC_SW_MUX_I2C1_SCL
memory set     0x30330218    32     0x00000010     #IOMUXC_SW_MUX_I2C1_SDA
memory set     0x3033047C    32     0x000000C6     #IOMUXC_SW_PAD_I2C1_SCL
memory set     0x30330480    32     0x000000C6     #IOMUXC_SW_PAD_I2C1_SDA
sysparam set   pmic_cfg    0x0008             #I2C slave address=8, I2C_BUS=I2C1
sysparam set   pmic_set    0x3C18              # Write 0x18 to REG_3C
sysparam set   pmic_set    0x230C              # Write 0x0C to REG_23
sysparam set   pmic_set    0x310C              # Write 0x0C to REG_31
sysparam set   pmic_set    0x380C              # Write 0x0C to REG_38
sysparam set   pmic_set    0x3F0C              # Write 0x0C to REG_3F
sysparam set   pmic_set    0x460C              # Write 0x0C to REG_46
sysparam set   pmic_set    0x4D0C              # Write 0x0C to REG_4D
sysparam set   pmic_set    0x540C              # Write 0x0C to REG_54
```

## 4.4 Run DDR Calibration and generate DDR initial code

Please follow chapter 3 to run DDR calibration and stress test with your board specific script. If there is no problem, Congratulations, you can generate DDR initial code now. In initial DDR script, RPA tool always use NXP reference board related parameters. Your board design and manufacturing technology are different from NXP reference board, and board related parameters may differ from initial DDR script. If DDR calibration failed, you can try to modify following DDR parameters in script.

- TrainInfo

  This parameter controls DDR training debug message. The default value is 0xc8, which means only display stage completion message. You can change to 0x05 to get detailed debug message when DDR training failed.

- ODTImpedance

  Desired ODT impedance in Ohm. Valid values for DDR4=240,120,80,60,40. Valid values for DDR3L=high-impedance,120,60,40. Valid values for LPDDR4=240,120,80,60,40

- TxImpedance

  Write Driver Impedance for DQ/DQS in ohm (Valid values for all DDR type= 240, 120, 80, 60, 48, 40, 34)

- ATxImpedance

  Write Driver Impedance for Address/Command (AC) bus in ohm (Valid values for all DDR type = 120, 60, 40, 30, 24, 20)

- PhyVref

  This parameter is used for 1D training process. You can refer to DDR datasheet for detailed meaning.

- Mode Registers (MR0~MR22)

  There are different meanings for different DDR types. Please refer to DDR datasheet for detailed information. Remember don’t manually modify the Mode Registers. Instead, please modify Mode Registers in RPA tool. Because there may be other parameters related to the Mode Registers.

## 4.5 Building u-boot image

MX8MSCALE integrates a MCU based DDR PHY, which needs to load DDR firmware before DDR initialization. The version of the DDR firmware used in the BSP may differ from the version used by the MSCALE DDR Tool. The MSCALE DDR tool always uses the latest firmware. When you use the DDR tool generated SPL codes instead of the original ones, please make sure to replace all firmware binaries with DDR tool provided in the bin directory. Here is the introduction for two different u-boot building environments.

### 4.5.1 Building u-boot image by toolchain command

In this building environment, you need download imx_mkimage source code for MX8MSCALE first.

**Step1a. If you are using ddr_init.c and ddrphy_train.c in old version of u-boot:**

```
$ cd uboot-imx
$ cp directory_to generated_code/ddr_init.c board/freescale/imx8mq_evk/ddr/
$ cp directory_to generated_code /ddrphy_train.c board/freescale/imx8mq_evk/ddr/
```

**Step1b. If you are using xxx_timing.c in new version of u-boot**

```
$ cd uboot-imx
$ cp directory_to generated_code/lpddr4_timing.c board/freescale/imx8mq_evk/
```

**Step2. Build u-boot image**

```
$ export SYSROOT=~/toolchain/sysroots/aarch64-poky-linux/
$ export PATH=~/toolchain/sysroots/x86_64-pokysdk-linux/usr/bin/aarch64-poky-linux/:$PATH
$ export CC="aarch64-poky-linux-gcc"
$ export ARCH=arm64
$ export CROSS_COMPILE=aarch64-poky-linux-
$ make CC="$CC" imx8mq_evk_config
$ make CC="$CC"
```

**Step3a. Copy corresponding LPDDR4 firmware and u-boot to imx_mkimage directory (optional)**

```
$ cd imx_mkimage
$ cp directory_to_ddr_tool/bin/ lpddr4_imem_1d_v????.bin iMX8M/
$ cp directory_to_ddr_tool/bin/ lpddr4_imem_2d_v????.bin iMX8M/
$ cp directory_to_ddr_tool/bin/ lpddr4_dmem_1d_v????.bin iMX8M/
$ cp directory_to_ddr_tool/bin/ lpddr4_dmem_2d_v????.bin iMX8M/
$ cp directory_to_uboot/spl/u-boot-spl.bin iMX8M/
$ cp directory_to_uboot/u-boot-nodtb.bin iMX8M/
$ make SOC=iMX8M flash_hdmi_spl_uboot
```

**Step3b. Copy corresponding DDR4 firmware and u-boot imx_mkimage directory (optional)**

```
$ cd imx_mkimage
$ cp directory_to_ddr_tool/bin/ ddr4_imem_1d_v????.bin iMX8M/
$ cp directory_to_ddr_tool/bin/ ddr4_imem_2d_v????.bin iMX8M/
$ cp directory_to_ddr_tool/bin/ ddr4_dmem_1d_v????.bin iMX8M/
$ cp directory_to_ddr_tool/bin/ ddr4_dmem_2d_v????.bin iMX8M/
$ cp directory_to_uboot/spl/u-boot-spl.bin iMX8M/
$ cp directory_to_uboot/u-boot-nodtb.bin iMX8M/
$ make SOC=iMX8M flash_hdmi_spl_uboot
```

**Step3c. Copy corresponding DDR3 firmware and u-boot to imx_mkimage directory (optional)**

```
$ cd imx_mkimage
$ cp directory_to_ddr_tool/bin/ ddr3_imem_1d_v????.bin iMX8M/
$ cp directory_to_ddr_tool/bin/ ddr3_dmem_1d_v????.bin iMX8M/
$ cp directory_to_uboot/spl/u-boot-spl.bin iMX8M/
$ cp directory_to_uboot/u-boot-nodtb.bin iMX8M/
$ make SOC=iMX8M flash_hdmi_spl_uboot
```

**Step4. Burn u-boot image into SD card**

```
$ dd if=iMX8M/flash.bin of=/dev/sdx bs=1k seek=33
```

### 4.5.2 Building u-boot image under Yocto environment

Suppose you are familiar with building image under Yocto environment and you have one building directory for MX8M target board. Otherwise, please refer to NXP Linux BSP user guide for detailed information.

**Step1. Extract u-boot source code in building directory**

```
$ bitbake u-boot-imx -c compile -f
```

**Step2. Copy generated files to u-boot directory**

```
$ cd tmp/work/imx8mqevk-poky-linux/u-boot-imx/2017.03-r0/git/
$ cp directory_to generated_code/ddr_init.c board/freescale/imx8mq_evk/ddr/
$ cp directory_to generated_code /ddrphy_train.c board/freescale/imx8mq_evk/ddr/
```

**Step3. Copy DDR firmware binaries to** *tmp/deploy/images/imx8mqevk/***. (optional)**

**Step4. Build u-boot**

```
$bitbake u-boot-imx -c compile -f
```

# Chapter 5
# Revision History

Table 1 provides a revision history for this user guide.

Table 1. Document Revision History

| Rev. Number | Date | Substantive Change(s) |
|---|---|---|
| Rev. 1.0 | 01/2018 | Initial public release. |
| Rev. 1.1 | 12/12/2018 | Add new format in code generation |
| Rev. 2.0 | 2/23/2021 | Add UART and PMIC customization information |
|  |  |  |
|  |  |  |