

NAND Flash Host Controller IP Integration Manual

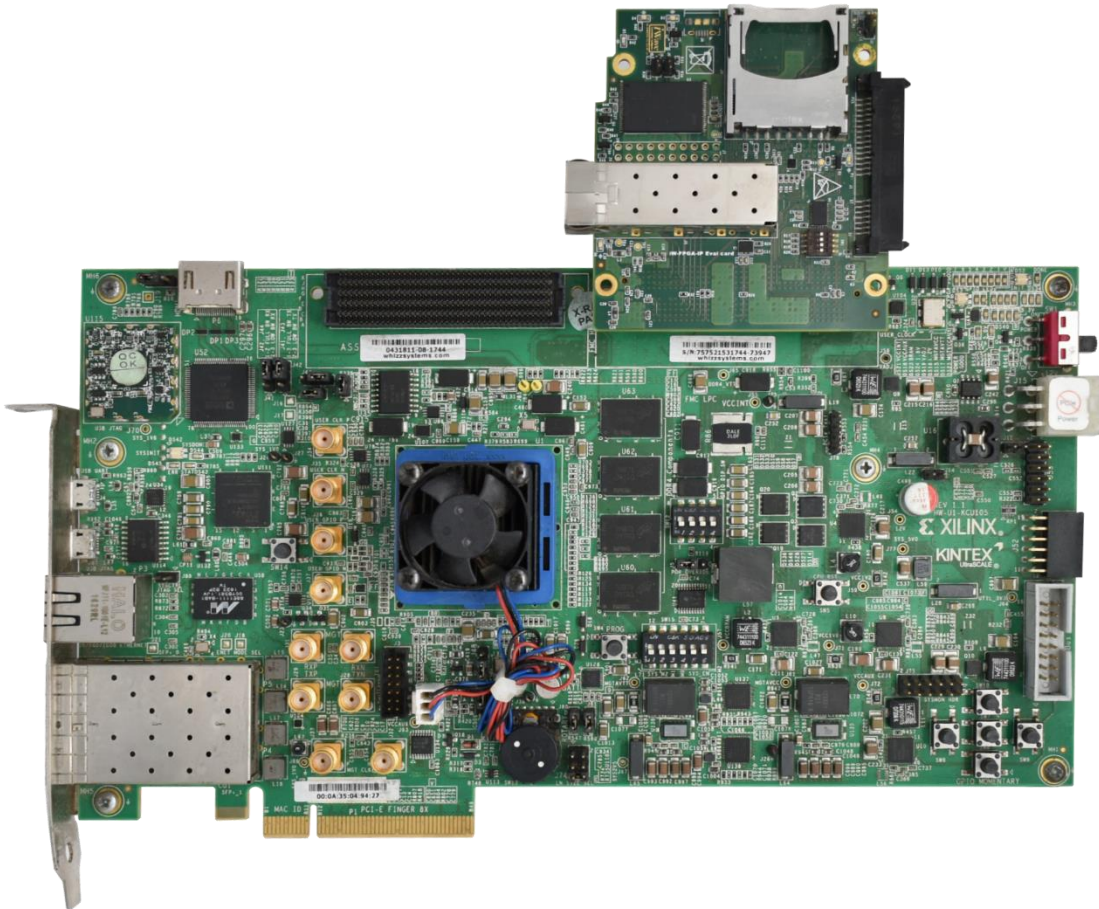


TABLE OF CONTENT

1	INTRODUCTION.....	5
1.1	PURPOSE.....	5
1.2	REFERENCE DOCUMENT	5
1.3	OVERVIEW	5
1.4	ACRONYMS AND ABBREVIATIONS.....	5
2	IP CONFIGURATION AND INSTANTIATION.....	6
2.1	EXAMPLE DESIGN.....	6
2.2	IP CONFIGURATION.....	6
2.3	STEPS TO INSTANTIATE NAND HOST IP	8
3	IMPLEMENTATION DETAILS	10
3.1	CONSTRAINTS	10
3.1.1	CLOCK CONSTRAINTS:	10
3.1.2	PIN CONSTRAINTS:.....	10
4	TEST SETUP.....	12
4.1	CONNECTIONS.....	12
4.2	PROGRAMMING FPGA AND RUNNING USING SDK.....	13
5	TEST PROCEDURE	16
5.1	TERA TERM SETUP	16
5.2	TESTING PROCEDURE	16
5.3	INITIAL PRINTS:	16
5.3.1	READ ID:	17
5.3.2	ERASE & READ BACK:	17
5.3.3	WRITE & READ BACK:	18
6	DESIGN MODIFICATION TO BE DONE FOR CUSTOM BOARD.....	20
7	RESOURCE UTILIZATION	21

List Of Figures

Figure 1: Design sources where nand host netlist is instantiated.....	7
Figure 2: Parameters for Micron MT29F64G08AFAAWP	7
Figure 3: Instantiation of microblaze and nand top modules	8
Figure 4: Nand host IP instantiation.....	9
Figure 5: Clock constraints.....	10
Figure 6: Pin Constraints in example design .xdc file.....	11
Figure 7: Selecting Serial port to adjust voltage level.....	12
Figure 8: Tera term prints in KCU105 board.....	12
Figure 9: FMC menu in KCU105.....	13
Figure 10: NAND FMC	13
Figure 11: Target setup in run configuration.....	14
Figure 12: Application project in run configuration.....	14
Figure 13: Initial print after FPGA is programmed.....	16
Figure 14: Wrong choice.....	16
Figure 15: Read ID	17
Figure 16: Erase & Read back.....	18
Figure 17: Write & Read Back.....	19
Figure 18:Read ID table snap from micron DATA sheet.....	20

List Of Tables

Table 1: Acronyms & Abbreviations	5
Table 2: Resource Utilization for device for KCU105 dev. kit	21

1 Introduction

1.1 Purpose

The purpose of this document is to describe iW- Nand Flash Host Controller IP Integration details.

1.2 Reference Document

- ONFI NAND specification.
- iW- EMEZ4-DS-01-R1.0-REL1.0

1.3 Overview

Nand Flash Host Controller interfaces the User and NAND flash via FMC using FPGA GPIO's. So this IP forms a bridge between the NAND flash and User (microblaze processor), enabling the storing, reading and erasing the data in NAND flash. This IP will issue the necessary command address and controls all the necessary actions required to program, erase and read data using NAND flash.

1.4 Acronyms and Abbreviations

Table 1: Acronyms & Abbreviations

Term	Meaning
FPGA	Field Programmable Gate Array
GPIO	General purpose input output
FMC	FPGA Mezzanine Card
LUT	Look Up Table
IO	Input and Output
FF	Flip Flop

2 IP Configuration and Instantiation

2.1 Example design

The NAND flash IP example design mainly consists of,

1. **Microblaze soft core:** User can provide the inputs to run the different tests using the bare metal code running in the Microblaze. This will pass the different control signals to NAND test driver based on the user selection.
2. **NAND Test Driver:** NAND test driver is responsible for generating the control signal for the NAND IP. The interface between the test driver and NAND IP is custom interface and these signal will be driven by the test driver based on the test cases selected by user. It is also responsible for the selection of the die's. in example design 32nd bit is used to select the die. If 32nd bit is **logic LOW** then 1st die is enabled if 32nd bit is **logic HIGH** 2nd die is enabled.
3. **NAND IP:** NAND IP is the design under test and controls the access to the external NAND chip based on the inputs from test driver.

Note:

- **Example design is compiled for Xilinx board KCU105 board with Vivado 2019.1 version.**
- **Example design currently supports double die.**
- **31st bit is used to select the die. Bits from 30 to 12 is Row (block and page) address.**
- **Column address is always bitwise and with 0. i.e. xFFF0000 for block erase and xFFFFFF00 for page program and readback.**
- **Read ID can be changed in test driver refer chapter 6.**

2.2 IP Configuration

The Nand Flash Host Controller IP netlist file i.e. nand_host.dcp file is instantiated in the design as shown in the figure below.

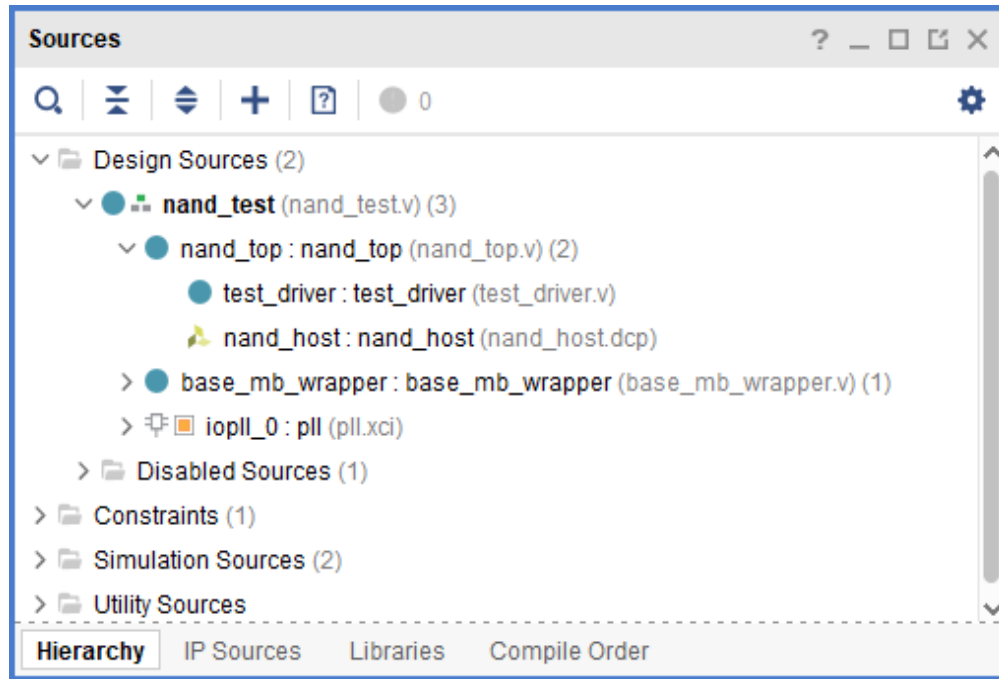


Figure 1: Design sources where nand host netlist is instantiated

Module nand_test is the top module of the project which integrates the top module of NAND IP and Microblaze. There are few timing parameters which can be modified by the user depending on which Nand flash IC to be used.

Module nand_top.v is the top module of NAND IP which integrates the test_drive.v module and the nand_host.dcp (nand_host.vhd) file.

```
// Parameters for Micron MT29F64G08AFAAWP
parameter DW = 8,
parameter TPWR = 20'h005DC, // 30us for 50MHz clock
parameter TRST = 20'h0C350, // 500us for 50MHz clock
parameter TR_ECC = 20'h00000, // Not Used
parameter TR = 20'h006D6, // 35us for 50MHz clock
parameter TPROG = 20'h06D60, // 560us for 50MHz clock
parameter TBERS = 20'h124F8, // 1.5 ms for 50MHz clock
parameter TWB = 20'h00050, // 100ns for 50MHz clock
parameter TFEAT = 20'h00000, // Not Used
parameter TWHR = 20'h00004, // 80ns for 50MHz clock
parameter TCCS = 20'h00000 // Not Used*/
```

Figure 2: Parameters for Micron MT29F64G08AFAAWP

The figure above shows the configurable parameters in nand_test.v file. Parameters like TRST device reset time , TPROG Maximum PROGRAM PAGE time and parameters which decides the timing parameters are controlled directly from the nand_test.v module as shown in the figure 2. User need to make the required changes depending upon the NAND flash IC.

NOTE: All the above parameters should be with respect to the frequency of 50MHz and values shown is for Micron MT29F64G08AFAAWP part.

2.3 Steps to Instantiate NAND HOST IP

- 1) Instantiate the top module of IP microblaze and clocking wizard in the module nand_test.v is shown below,

```

*****
:                                     Internal Modules                               *
*****/

nand_top # (
    .DW                (DW                ),
    .TPWR              (TPWR              ),
    .TRST              (TRST              ),
    .TR_ECC            (TR_ECC            ),
    .TR                (TR                ),
    .TPROG             (TPROG             ),
    .TBERS             (TBERS             ),
    .TWB               (TWB               ),
    .TFEAT             (TFEAT             ),
    .TWHR              (TWHR              ),
    .TCCS              (TCCS              ))

nand_top (
    .clk_i              (clk_i              ),
    .rst_n_i           (rst_n_i           ),
    .addr_i            (addr              ),
    .no_of_pgs_i       (no_of_pgs         ),
    .test_case_i       (test_case         ),
    .start_i           (start              ),
    .copy_back_i       (copy_back         ),
    .dst_addr_i        (dst_addr          ),
    .status_o          (status             ),
    .nand_r_b_n        (nand_r_b_n        ),
    .nand_ce_n_o       (nand_ce_n         ),
    .nand_re_n_o       (nand_re_n         ),
    .nand_we_n         (nand_we_n         ),
    .nand_cle_o        (nand_cle_o        ),
    .nand_ale_o        (nand_ale_o        ),
    .nand_wp_n_o       (nand_wp_n_o       ),
    .nand_data_io      (nand_data_io      )
);

base_mb_wrapper (
    .addr_0_tri_o      (addr[31:0]        ),
    .addr_1_tri_o      (addr[33:32]       ),
    .cpy_bck_tri_o     (copy_back         ),
    .clk_in1           (clk_i              ),
    .dst_addr_0_tri_o  (dst_addr[31:0]    ),
    .dst_addr_1_tri_o  (dst_addr[33:32]   ),
    .no_of_pg_tri_o    (no_of_pgs         ),
    .reset             (reset_n_i         ),
    .rs232_uart_rxd    (rs232_uart_rxd    ),
    .rs232_uart_txd    (rs232_uart_txd    ),
    .start_tri_o       (start              ),
    .status_tri_i      (status             ),
    .test_case_tri_o   (test_case         )
);

pll iopll_0 (
    .clk_in1_p         (clk_p_i           ),
    .clk_in1_n         (clk_n_i           ),
    .locked            (rst_n_i           ),
    .clk_out1          (clk_i             )
);

```

Figure 3: Instantiation of microblaze and nand top modules

System clock is given to clocking wizard whose output is 50MHz given to both NAND IP and Microblaze.

nand_top.v is the top module of NAND IP which is used to integrate the test_drive.v module with nand_host.vhd (nand_host.dcp) module.

```
nand_host (
//parameters_as_signals
.TPWR          (TPWR          ),
.TRST          (TRST          ),
.TR_ECC        (TR_ECC        ),
.TR            (TR            ),
.TPROG         (TPROG         ),
.TBERS        (TBERS         ),
.TWB           (TWB           ),
.TFEAT         (TFEAT         ),
.TWHR          (TWHR          ),
.TCCS          (TCCS          ),
//interface
.clk_i         (clk_i         ),
.reset        (rst_n_i       ),
.cmd_req_i     (cmd_req      ),
.usr_cmd_i     (usr_cmd      ),
.die_select_i  (die_select   ),
.dst_die_sel_i (dst_die_sel  ),
.addr_i        (addr         ),
.dst_addr_i    (dst_addr     ),
.addr_cyc_i    (addr_cyc     ),
.data_size_i   (data_size    ),
.cmd_rdy_o     (cmd_rdy      ),
.err_o         (err          ),
.ecc_error_o   (ecc_err      ),
.ecc_complete_o (ecc_complete),
.rd_data_vld_o (rd_data_vld  ),
.rd_data_o     (rd_data      ),
.rd_ready_i    (rd_ready     ),
.wr_data_vld_i (wr_data_vld  ),
.wr_data_i     (wr_data      ),
.wr_ready_o    (wr_ready     ),
.nand_r_b_n_i  (nand_r_b_n_i),
.nand_ce_n_o   (nand_ce_n_o ),
.nand_re_n_o   (nand_re_n_o ),
.nand_we_n_o   (nand_we_n_o ),
.nand_cle_o    (nand_cle_o   ),
.nand_ale_o    (nand_ale_o   ),
.nand_wp_n_o   (nand_wp_n_o ),
.nand_data_io  (nand_data_io)
```

Figure 4: Nand host IP instantiation

- 2) Include nand_host.dcp netlist source file provided under release
- 3) The Constraint file (.xdc) provided in the design can be changed by user depending on requirements.
- 4) Give the required clock, UART, Pin/IO constraints for NAND Interface in the .xdc file and compile the custom design with NAND host IP.

NOTE: IP configuration of iopll need to be done according to the design. i.e. the clocking wizard need to be generated in accordance to the input clock. And its output should be 50MHz always.

3 Implementation Details

Nand Flash Host Controller IP works at clock speed 50Mhz. System differential clock input whose frequency is 300Mhz is converted to 50Mhz using clocking wizard and given to microblaze and Nand Flash Host Controller IP.

3.1 Constraints

3.1.1 Clock constraints:

The following figure shows the clock constraints used in this design. The constraints given here the system clock whose frequency is 300Mhz.

These constraints need to be added in the .xdc file along with the NAND Interface pin constraints.

```
create_clock -period 3.333 -name clk_p_i -waveform {0.000 1.667} [get_ports clk_p_i]
```

Figure 5: Clock constraints

NOTE: Please change the clock constraints as required for custom design. As also chose the required IO standards

3.1.2 Pin Constraints:

Following figure 6 shows the example pin and IO standard constraints.

```

set_property PACKAGE_PIN AK17 [get_ports clk_p_i]
set_property IOSTANDARD LVDS [get_ports clk_p_i]

set_property PACKAGE_PIN AN8 [get_ports reset_n_i]
set_property IOSTANDARD LVCMOS18 [get_ports reset_n_i]

set_property PACKAGE_PIN K26 [get_ports rs232_uart_txd]
set_property PACKAGE_PIN G25 [get_ports rs232_uart_rxd]
set_property IOSTANDARD LVCMOS18 [get_ports rs232_uart_txd]
set_property IOSTANDARD LVCMOS18 [get_ports rs232_uart_rxd]

set_property PACKAGE_PIN H21 [get_ports nand_r_b_n_i]
set_property PACKAGE_PIN B21 [get_ports nand_r_b_n_l_i]
set_property PACKAGE_PIN D20 [get_ports nand_ce_n_o]
set_property PACKAGE_PIN G20 [get_ports nand_ce_n_l_o]
set_property PACKAGE_PIN G24 [get_ports nand_re_n_o]
set_property PACKAGE_PIN G22 [get_ports nand_we_n_o]
set_property PACKAGE_PIN E20 [get_ports nand_cle_o]
set_property PACKAGE_PIN C26 [get_ports nand_ale_o]
set_property PACKAGE_PIN B20 [get_ports nand_wp_n_o]
set_property PACKAGE_PIN F23 [get_ports nand_data_dir]
set_property PACKAGE_PIN E22 [get_ports {nand_data_io[0]}]
set_property PACKAGE_PIN C21 [get_ports {nand_data_io[1]}]
set_property PACKAGE_PIN D24 [get_ports {nand_data_io[2]}]
set_property PACKAGE_PIN B24 [get_ports {nand_data_io[3]}]
set_property PACKAGE_PIN D8 [get_ports {nand_data_io[4]}]
set_property PACKAGE_PIN B10 [get_ports {nand_data_io[5]}]
set_property PACKAGE_PIN B9 [get_ports {nand_data_io[6]}]
set_property PACKAGE_PIN D9 [get_ports {nand_data_io[7]}]

set_property IOSTANDARD LVCMOS18 [get_ports nand_r_b_n_i]
set_property IOSTANDARD LVCMOS18 [get_ports nand_r_b_n_l_i]
set_property IOSTANDARD LVCMOS18 [get_ports nand_ce_n_o]
set_property IOSTANDARD LVCMOS18 [get_ports nand_ce_n_l_o]
set_property IOSTANDARD LVCMOS18 [get_ports nand_re_n_o]
set_property IOSTANDARD LVCMOS18 [get_ports nand_we_n_o]
set_property IOSTANDARD LVCMOS18 [get_ports nand_cle_o]
set_property IOSTANDARD LVCMOS18 [get_ports nand_ale_o]
set_property IOSTANDARD LVCMOS18 [get_ports nand_wp_n_o]
set_property IOSTANDARD LVCMOS18 [get_ports nand_data_dir]
set_property IOSTANDARD LVCMOS18 [get_ports {nand_data_io[0]}]
set_property IOSTANDARD LVCMOS18 [get_ports {nand_data_io[1]}]

```

Figure 6: Pin Constraints in example design .xdc file

NOTE: These constraints are in accordance to example design . Please change the pin constraints for NAND interface as required for custom design. And also define the constraints for clock, reset and UART pins accordingly

4 Test setup

4.1 Connections

Make the necessary connection to program the target FPGA board. Connect UART there will be two serial port available i.e. For system control and other one for FPGA design. Select the port for system control and observe the prints as shown in figure 8. For example in example design serial port 10 is for system control and serial port 9 is for FPGA design. System control is used to set the FMC voltage in 1.8V

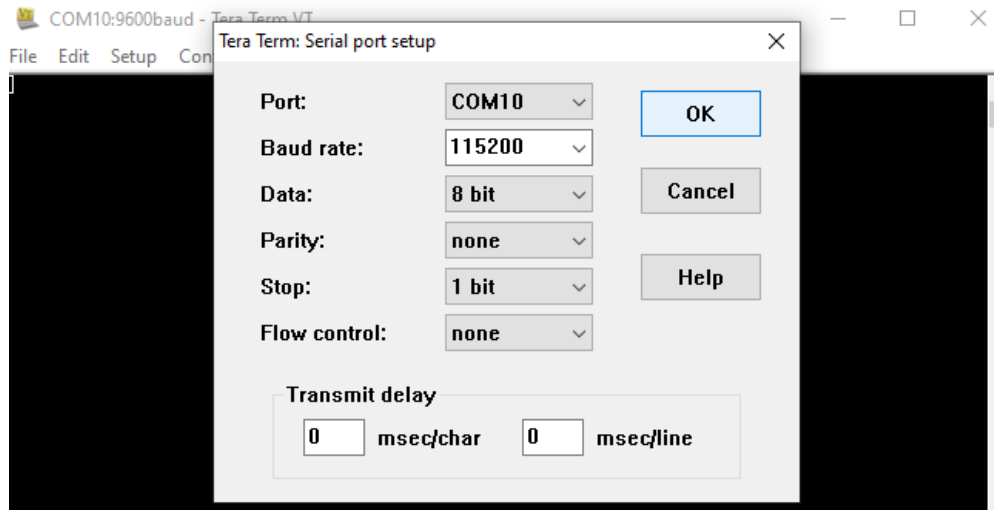


Figure 7: Selecting Serial port to adjust voltage level

After Selecting the serial port Check for tera term prints of KCU105 board main menu as shown in figure below.

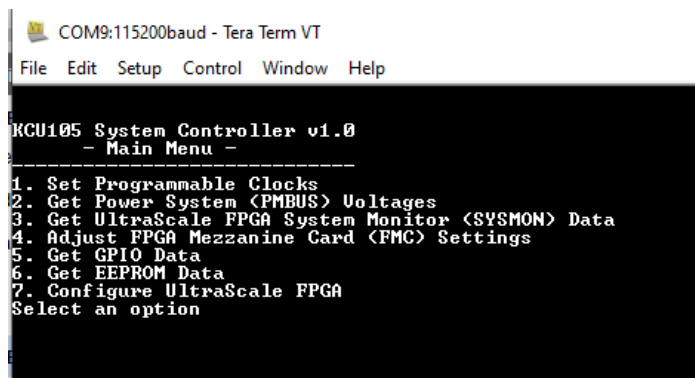


Figure 8: Tera term prints in KCU105 board.

In main menu select option 4 i.e. adjust FPGA FMC settings. Then in FMC menu select Option 4 i.e. Set FMC VADJ to 1.8V. as shown in the figure 9.

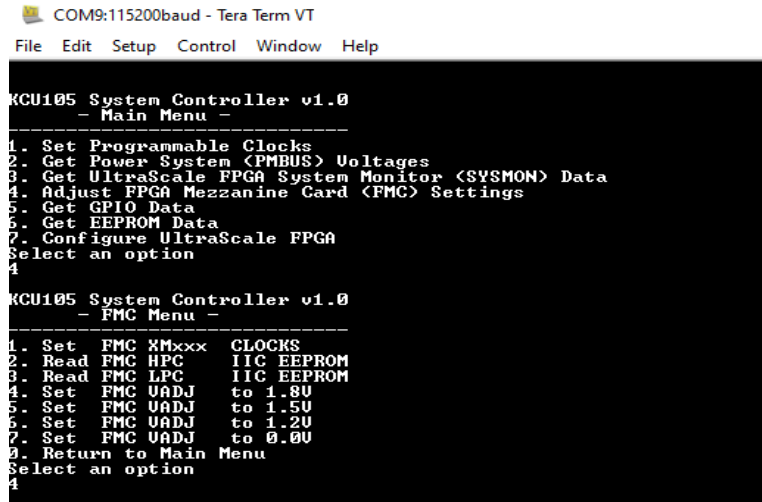


Figure 9: FMC menu in KCU105

Please make sure that FMC card of NAND flash is connected to FPGA board FMC is set to 1.8V and 2 Blue LED in FMC card is glowing. As shown in the figure below



Figure 10: NAND FMC

4.2 Programming FPGA and running using SDK.

- After generating bitstream in Vivado go to files, export hardware in appeared window click to include bitstream keep directory as local to project and click export.
- After exporting the project go to files and click on launch SDK in appeared window keep directory as local to project and click ok.
- Once the SDK is opened in SDK go to file create new application project and click next.
- In Available templets click on helloworld.c and click on finish.
- Once the project is created double-click the helloworld.c file in the Project Navigator to open the file.
- Clear the code and copy paste the given source c code (helloworld available at `..\iW-EMEZ4-PF-01-R1.0-REL1.1\iW-EMEZ4-SF-01-R1.0-REL1.1\iW-EMEZ4-SC-01-R1.0-REL1.3`).

- Save (CTRL+S) and build (CTRL+B) the project
- Then right click on project go to run and select run configuration select single applicationdebug
- In the appeared window under target setup click on reset entire system and programFPGA as shown in the figure 11.
- In application select the appropriate processor and click on reset processor.

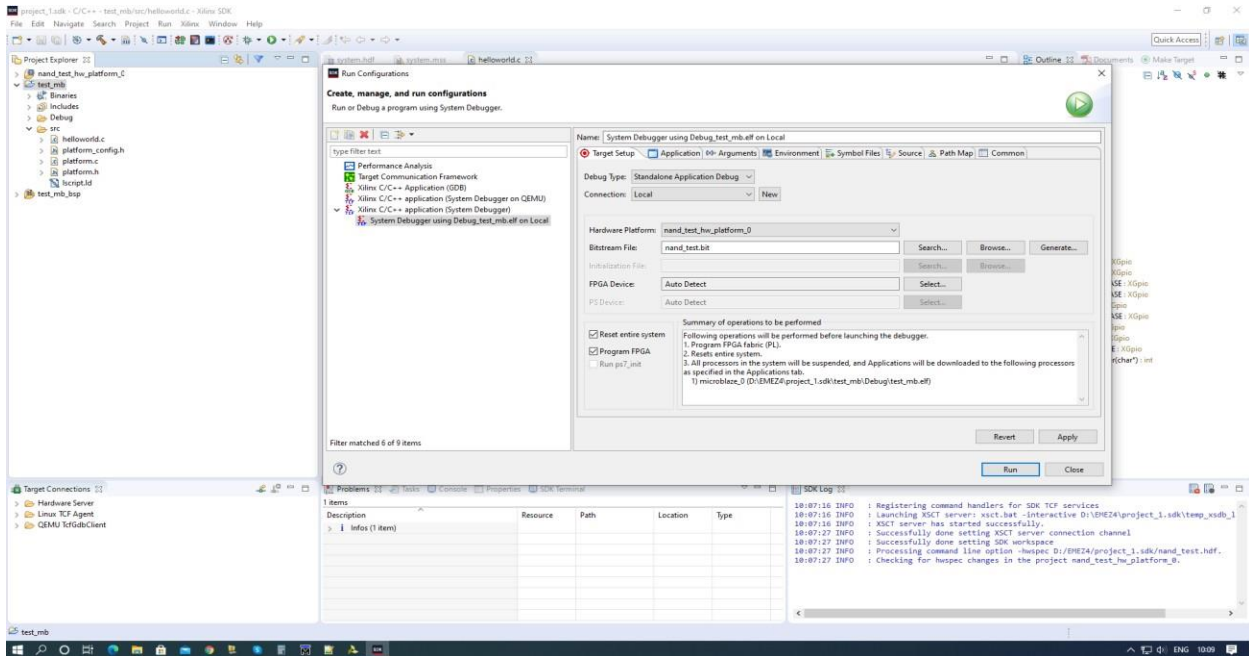


Figure 11: Target setup in run configuration.

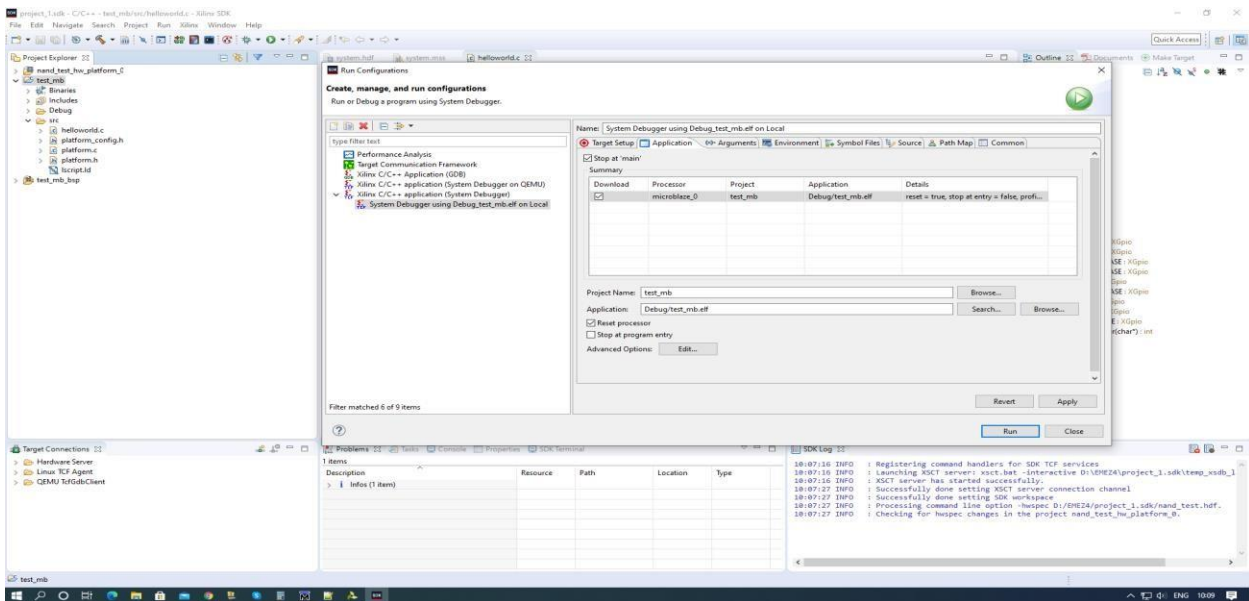


Figure 12: Application project in run configuration.

- Now click on Run
- After clicking on Run FPGA will be programmed and board is test ready.

NOTE: Before clicking on Run please make sure board is turned on with JTAG connect and proper tera term setup and if program FPGA is done using Vivado please don't select program FPGA option in target setup.

5 Test Procedure

5.1 Tera Term Setup

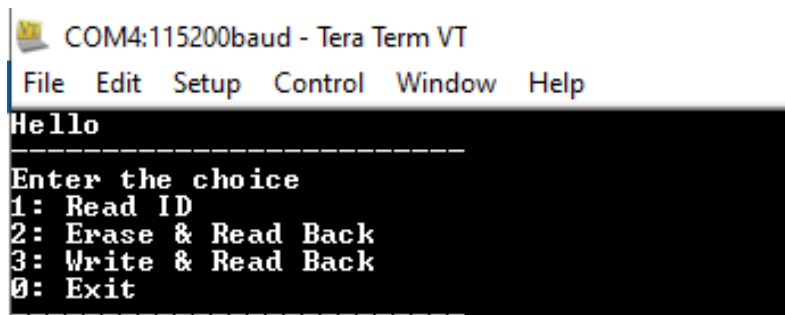
- Open tera term in PC and select the com port.
- Select baud rate as 115200.

5.2 Testing Procedure

After set up follow the below procedure to test the NAND flash controller IP.

5.3 Initial prints:

After clicking on Run in Run configuration software will program the FPGA. As soon as the programming is done below print will be displayed in Tera term.



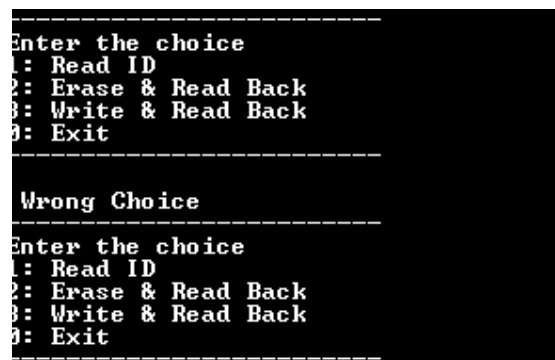
```
COM4:115200baud - Tera Term VT
File Edit Setup Control Window Help
Hello
-----
Enter the choice
1: Read ID
2: Erase & Read Back
3: Write & Read Back
0: Exit
-----
```

Figure 13: Initial print after FPGA is programmed

As shown in the figure 3 test case is available i.e.

1. Read ID
 2. Erase & Read Back
 3. Write & Read Back
- And 0 is to exit the test.

Any other test case than the mentioned above will result in wrong choice. As shown in figure below.

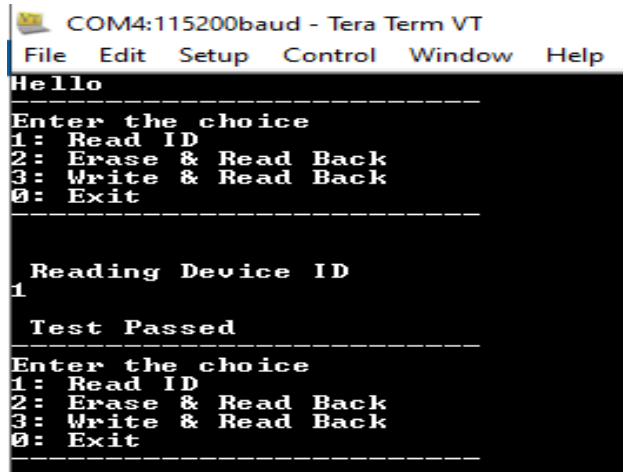


```
-----
Enter the choice
1: Read ID
2: Erase & Read Back
3: Write & Read Back
0: Exit
-----
Wrong Choice
-----
Enter the choice
1: Read ID
2: Erase & Read Back
3: Write & Read Back
0: Exit
-----
```

Figure 14: Wrong choice

5.3.1 Read ID:

Read ID is the 1st testcase in the current design. After initial prints, press “1” and then press “ Cntrl + Enter” together to run the test. After the Read ID operation is completed if the test is successful it indicates test passed. If any error occurs during the operation it indicates the reason for error. The prints of Read ID is shown below.



```
COM4:115200baud - Tera Term VT
File Edit Setup Control Window Help
Hello
-----
Enter the choice
1: Read ID
2: Erase & Read Back
3: Write & Read Back
0: Exit
-----

Reading Device ID
1

Test Passed
-----
Enter the choice
1: Read ID
2: Erase & Read Back
3: Write & Read Back
0: Exit
-----
```

Figure 15: Read ID

5.3.2 Erase & Read Back:

After initial prints, press “2” and then press “ Cntrl + Enter” together to run the erase & read back test. The prints regarding Erase and Read Back is shown in figure 14.

After selecting test case, user need to specify from which address data need to be erased and read back. So user should enter the block address to be erased and then the number of blocks to be erased. This test case will erase the blocks and read it back .If the erase and read back is successful, user gets the print as test passed.

Note: User need to select the number of blocks between 1 to 4096 and press “Cntrl +Enter” together to enter any inputs.

```
-----  
Enter the choice  
1: Read ID  
2: Erase & Read Back  
3: Write & Read Back  
0: Exit  
-----  
  
Erase Command Test  
  
Enter the block address to be erased  
  
Input address string is 11  
Address after conversion is 11  
Address after masking is 0  
Address of Block : 0  
  
Enter Number of Blocks to Erase  
Number of block 10 ::  
  
Erasing 10 block  
  
Status :: 0  
Status :: 0  
  
Status :: 1  
  
9 block has to be erased  
Status :: 1  
8 block has to be erased  
Status :: 1  
7 block has to be erased  
Status :: 1  
6 block has to be erased  
Status :: 1  
5 block has to be erased  
Status :: 1  
4 block has to be erased  
Status :: 1  
3 block has to be erased  
Status :: 1  
2 block has to be erased  
Status :: 1  
1 block has to be erased  
Status :: 1  
  
All Block Erased Successfully. Test Passed  
-----  
Enter the choice  
1: Read ID  
2: Erase & Read Back  
3: Write & Read Back  
0: Exit
```

Figure 16: Erase & Read back

5.3.3 Write & Read Back:

The 3rd test case is write & Read Back. After initial prints, press “3” and then press “Cntrl + Enter” together to run the write & read back test. The prints for 3rd test case is shown in the figure 17.

After selecting test case, user need to specify from which page address to be programmed and read back. So user should enter the page address to be programmed and then the number of pages to program and read back. This test case will program the number of pages specified by user with incremental data pattern and reads it back. If the write/program and read back is successful, user gets the print as test passed.

Note: User need to select the number of pages to be programmed between 1 to 128 and press “Ctrl + Enter” together to enter any inputs.

```
-----  
Enter the choice  
1: Read ID  
2: Erase & Read Back  
3: Write & Read Back  
0: Exit  
-----  
  
Page Program and Read Back Test  
Enter the page address to be programmed  
Address of Page : 0  
Enter Number of Pages to Program  
Programming 10 Pages  
All Pages Programmed and Successfully Read Back. Test Passed  
-----  
Enter the choice  
1: Read ID  
2: Erase & Read Back  
3: Write & Read Back  
0: Exit  
-----
```

Figure 17: Write & Read Back

NOTE: Due to evaluation time limit of 4 hours after programming the FPGA, all the abovetestcases fails. User can reprogram the board again to continue the testing.

6 Design modification to be done for Custom Board

- Update the FPGA part number according to the FPGA device available in the custom board
- Update the complete design for the selected FPGA device
- Update the NAND flash timing parameters according to the device selected.
 - Example design parameters are given for MT29F64G08AFAAWP .
- Make sure that 50Mhz input is provided properly to the design. If the clock available is other than 50Mhz, use the clocking wizard to generate the 50Mhz clock as done in example design.
- Update the pin constraints for NAND interface, clock, reset and UART pins
- Update the clock constraint according to the input clock frequency
- Recompile the design to generate the new binaries and use the XSA file to create the new application project
- Read ID parameters in Example design for MT29F64G08AFAAWP is given in figure below.

Table 6: Read ID Parameters for Address 00h

Device	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7
MT29F32G08ABAAA	2Ch	68h	00h	27h	A9h	00h	00h	00h
MT29F32G08ABCAB	2Ch	68h	00h	27h	A9h	00h	00h	00h
MT29F64G08AFAAA	2Ch	68h	00h	27h	A9h	00h	00h	00h
MT29F64G08AECAB	2Ch	68h	00h	27h	A9h	00h	00h	00h
MT29F128G08AJAAA	2Ch	88h	01h	A7h	A9h	00h	00h	00h
MT29F128G08AKAAA	2Ch	88h	01h	A7h	A9h	00h	00h	00h
MT29F128G08AKCAB	2Ch	88h	01h	A7h	A9h	00h	00h	00h
MT29F128G08AMAAA	2Ch	68h	00h	27h	A9h	00h	00h	00h
MT29F128G08AMCAB	2Ch	68h	00h	27h	A9h	00h	00h	00h
MT29F256G08AUAAA	2Ch	88h	01h	A7h	A9h	00h	00h	00h
MT29F256G08AUCAB	2Ch	88h	01h	A7h	A9h	00h	00h	00h

Figure 18:Read ID table snap from micron DATA sheet

- User can change this parameters in accordance to the Device ID in test driver.

7 Resource Utilization

The table below shows the resource utilization summary for KCU105 dev.kit. for NAND flashController IP.

Table 2: Resource Utilization for device for KCU105 dev. kit.

Resource	Utilization	Available
LUT	1497	242400
FF	916	484800
BRAM	6	600