NAND Flash Controller Integration Manual

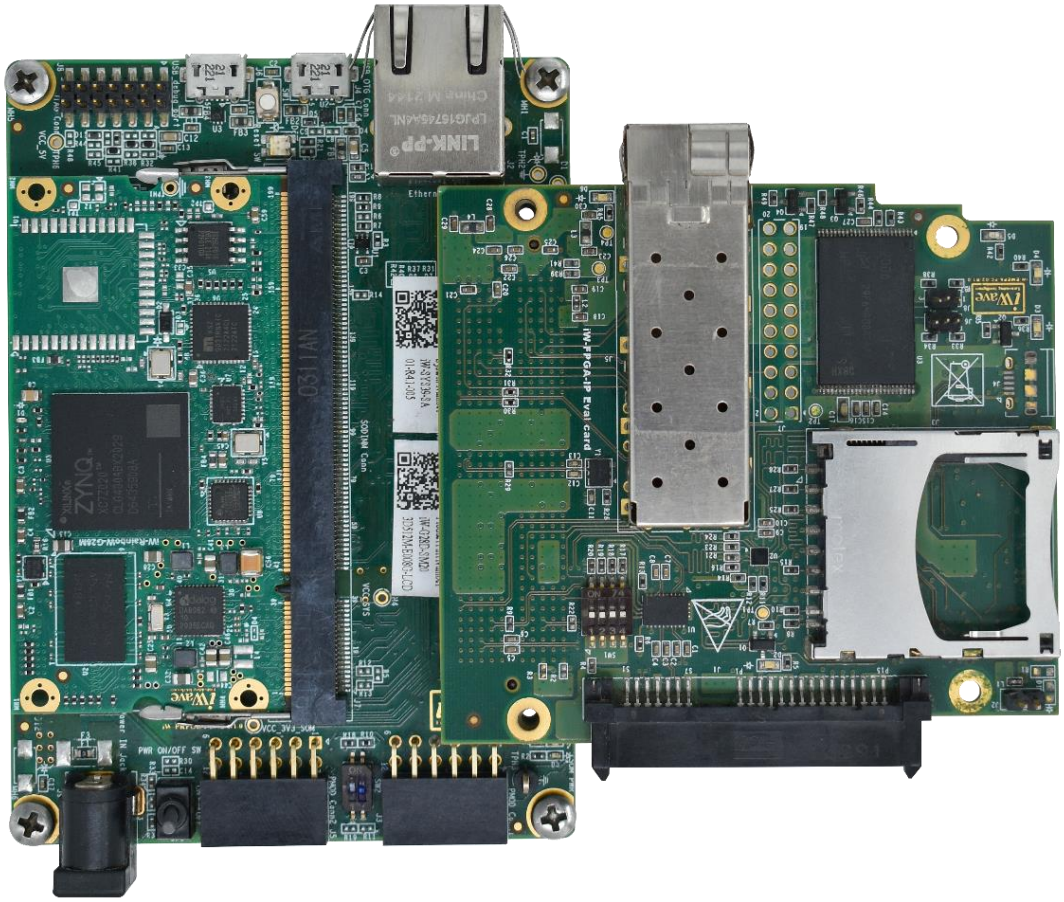

Table of Contents

List of Figures

List of Tables

1 Introduction

1.1 Purpose

The purpose of this document is to describe NAND Flash Controller IP Integration details.

1.2 Scope

This document describes the NAND Flash Controller IP and provides information about IP integration details.

1.3 Reference Document

- NAND Flash Memory Datasheet
  https://www.mouser.in/datasheet/2/671/micron_technology_micts06235-1-1759187.pdf
- Open NAND Flash Interface Specification
- AMBA® AXI™ and ACE™ Protocol Specification

1.4 Overview

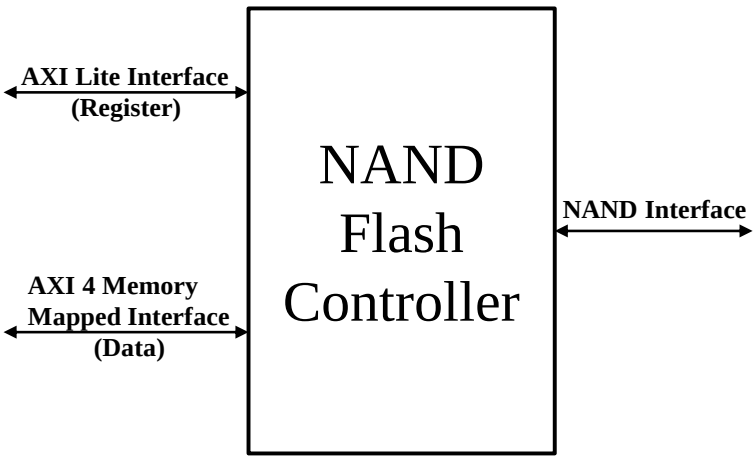

Figure 1: NAND Flash Controller Block Diagram

NAND Flash Controller defines a standard register set for control and status of the NAND device. iWave Supplies NAND Flash Controller as standalone component in addition to Verilog modules

1.5 Acronyms and Abbreviations

Table 1: Acronyms & Abbreviations

Term	Meaning
AXI	Advanced eXtensible Interface
FPGA	Field Programmable Gate Array
DDR	Double Data Rate
HDL	Hardware Description Language
IP	Intellectual Property
irq	Interrupt
ns	Nano seconds
PL	Programmable Logic
PS	Processor System
RTL	Register Transfer Logic
SoC	System On Chip
us	Micro Seconds

2 IP Configuration and Instantiation

2.1 Example design

The NAND flash IP example design mainly consists of,

1. **NAND HOST IP:** NAND HOST IP is the design under test and controls the access to the external NAND chip based on the inputs from register set module.
2. **NAND Wrapper:** This module accepts the commands and data over AXI 4 memory mapped interface and translates them to the user interface of NAND Host
3. **NAND Register Set:** This module implements register set required for NAND Device control and status information

Note:

- **Example design is compiled for XC7Z020CLG400-1 device with Vivado 2020.1 version.**
- **Example design currently supports dual die.**

An example design project is created using Vivado 2020.1, at location ...\iW-EMFEW-Release-1.0\iW- EMFEW -Release-1.0\iW- EMFEW -PF-01-R1.0-REL1.0\iW-EMFEW -FF-01-R1.0\iW- EMFEW -SY-01-R1.0-REL1.0\iW_ EMFEW _ED.
The design units that make the Block Design are shown in the figure below:

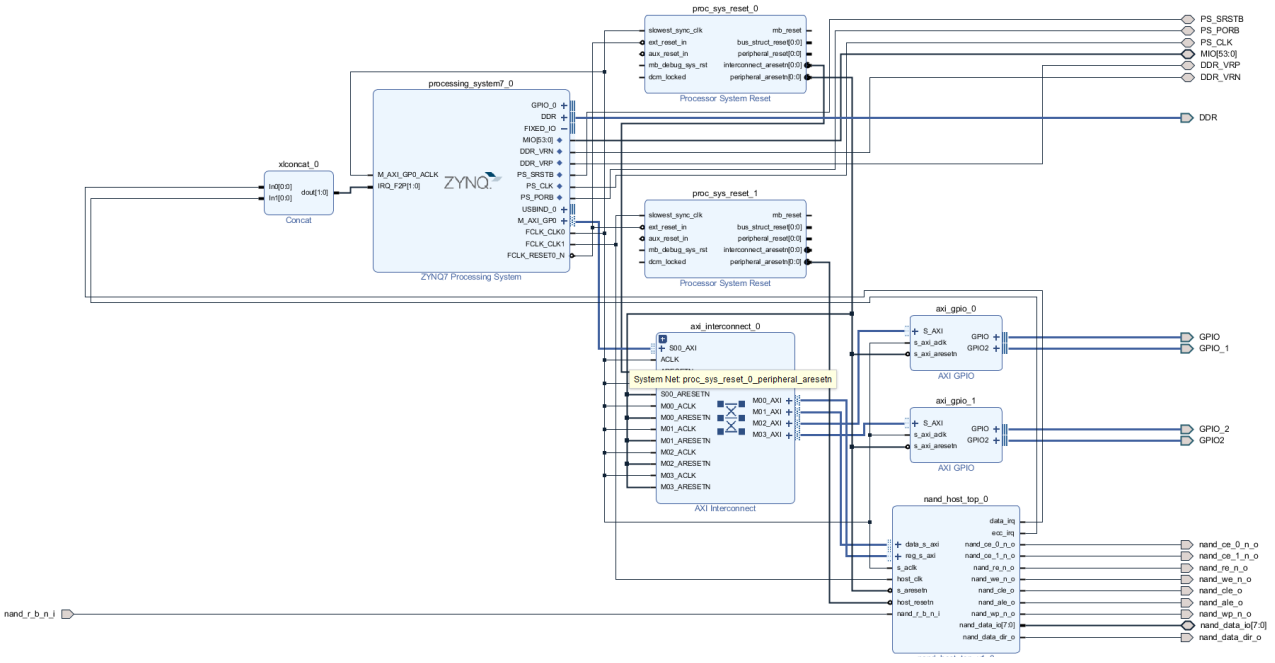

Figure 2: Block Design Modules of the example design project

Note: Change the assigned addresses in Address Editor tab present along with the block design/Diagram as below

1) For the register access: reg_axi port assign 0x6800_000 as address and range to 8M

2) For the data transfer: data_axi port assign 0x5000_0000 as address and range to 128M

/nand_host_top_0	data_s_axi	reg0	0x5000_0000	128M	0x57FF_FFFF
/nand_host_top_0	reg_s_axi	reg0	0x6800_0000	8M	0x687F_FFFF

Figure 3: Assign Specific Addresses & Range to ports

2.2 IP Configuration

The Nand Flash Host Controller IP includes the netlist file i.e. nand_host.dcp along with nand_host.v(stub file),register_set and nand_wrapper in the design as shown in the figure below

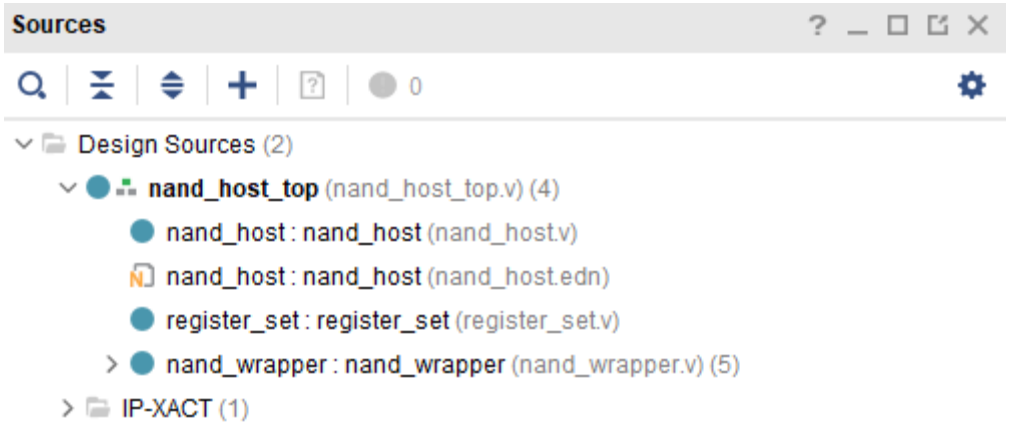

Figure 4: nand_host.edn(netlist file)is instantiated in Nand Flash Host Controller IP

Top Module of the Example design project is named as the top_test_module

Create a top module as shown in Figure below

```
module top_test_module
  (
  // system clocks
    input  wire          sys_clk_p,    //50mhz clock

  //PS DDR Signals
    inout  wire [14:0]   DDR_addr,
    inout  wire [2:0]    DDR_ba,
    inout  wire          DDR_cas_n,
    inout  wire          DDR_ck_n,
    inout  wire          DDR_ck_p,
    inout  wire          DDR_cke,
    inout  wire          DDR_cs_n,
    inout  wire  [3:0]   DDR_dm,
    inout  wire  [31:0]  DDR_dq,
    inout  wire  [3:0]   DDR_dqs_n,
    inout  wire  [3:0]   DDR_dqs_p,
    inout  wire          DDR_odt,
    inout  wire          DDR_ras_n,
    inout  wire          DDR_reset_n,
    inout  wire          DDR_we_n,
    inout  wire          DDR_VRN,
    inout  wire          DDR_VRP,
  // PS MIO Signals
    inout  wire  [53:0]  MIO,
    inout  wire          PS_CLK,
    inout  wire          PS_PORB,
    inout  wire          PS_SRSTB,

    //FMC loopback
    inout  wire   gpio_tri_io,
    inout  wire   gpio_1_tri_io,
    inout  wire   gpio_2_tri_io,
    inout  wire   gpio2_tri_io,

   //NAND Flash Interface
    input                nand_r_b_0_n_i     , // RDY/BSY#
    input                nand_r_b_1_n_i     , // RDY/BSY#
    output               nand_ce_0_n_o      , // Chip Enable
    output               nand_ce_1_n_o      , // Chip Enable
    output               nand_re_n_o        , // Read Enable
    output               nand_we_n_o        , // Write Enable
    output               nand_cle_o         , // Command Latch Enable
    output               nand_ale_o         , // Address Latch Enable
    output               nand_wp_n_o        , // Write Protect
    output               nand_data_dir_o    , // NAND data direction
    inout    [7:0]       nand_data_io         // NAND data
  );
```

Figure 5: Top Module for the example design

2.3 Steps to Instantiate NAND HOST IP

- Install the required Vivado Design Suite 2020.1 for the host PC adding the license path. https://www.xilinx.com/support/download/index.html/content/xil%09%09inx/en/downloadNav/vivado-design-tools/archive.html

- Copy the project from ...\iW-EMFEW-Release-1.0\iW- EMFEW -Release-1.0\iW-EMFEW -PF-01-R1.0-REL1.0\iW- EMFEW -FF-01-R1.0\iW- EMFEW -SY-01-R1.0-REL1.0\iW_ EMFEW _ED" to your project directory in the host PC
- Open the Vivado 2020.1 and click on "Open Project" as shown in the below figure and select the required project from the folder by clicking ok

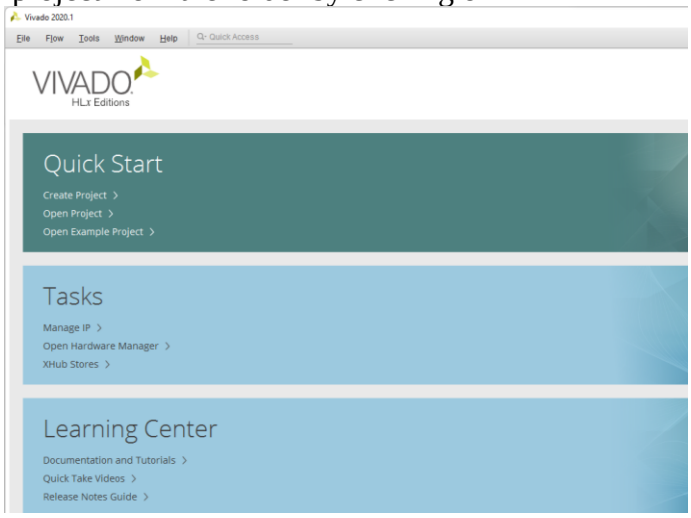

Figure 6: Click on "Open Project"

- Now the project selected will be opened in Vivado Design Suite 2020.1
- Go to settings of Vivado 2020.1 and add the IP to the Project Settings → IP → Repository→then add the IP path.

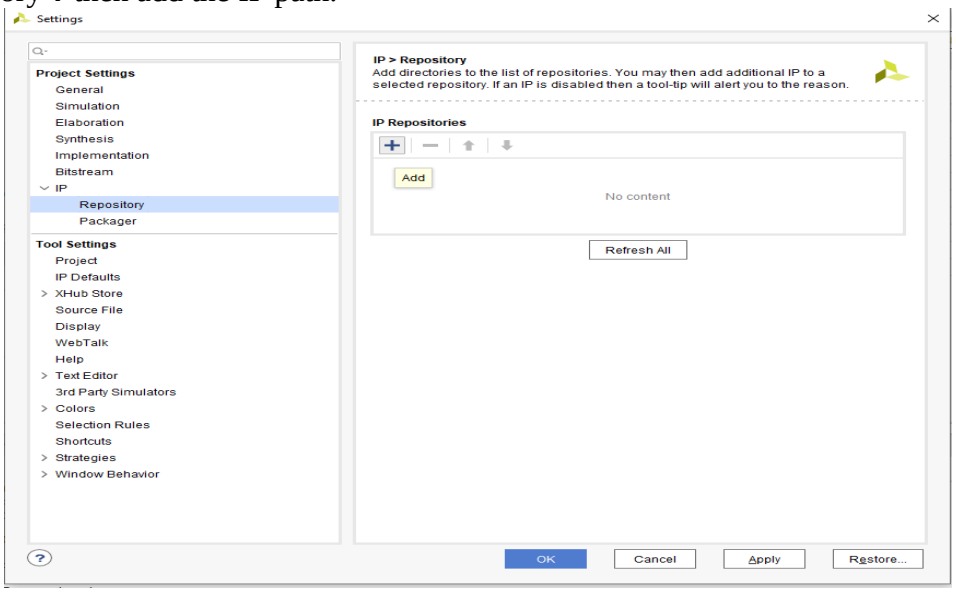

Figure 7: Adding NAND Flash Controller IP to the IP repository

IP location is ...\iW-EMFEW-Release-1.0\iW- EMFEW -Release-1.0\iW- EMFEW -PF-01-R1.0-REL1.0\iW- EMFEW -FF-01-R1.0\iW- EMFEW -SY-01-R1.0-REL1.0\iW_

EMFEW _ED \iW_nand_host. Add IP by clicking on the + sign as shown in above Figure. After the IP is added click on Apply,OK and close the dialogue box
Once the IP is added in the project repository, click on open block design present at the left most side of the Vivado 2020.1 tool.

1. After adding the other IP's as shown in the block design of the example project, add the NAND Flash Controller IP to the block design by clicking on the + sign (Ctrl +I) and make the necessary connections as shown in below Figure

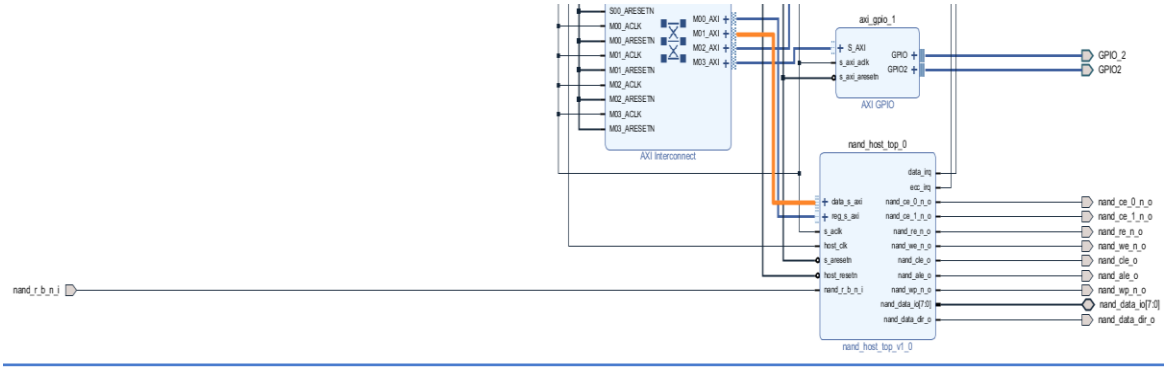

Figure 8: Adding the NAND Flash Controller IP to the block design

2. Upgrade the IP by clicking on Reports→Report IP Status present on the toolbar of the Vivado tool if required
3. Save the Block design, validate it (Press F6) and generate the block design by right clicking on the design_1(block design)→generate output products in the sources tab of the Vivado tool

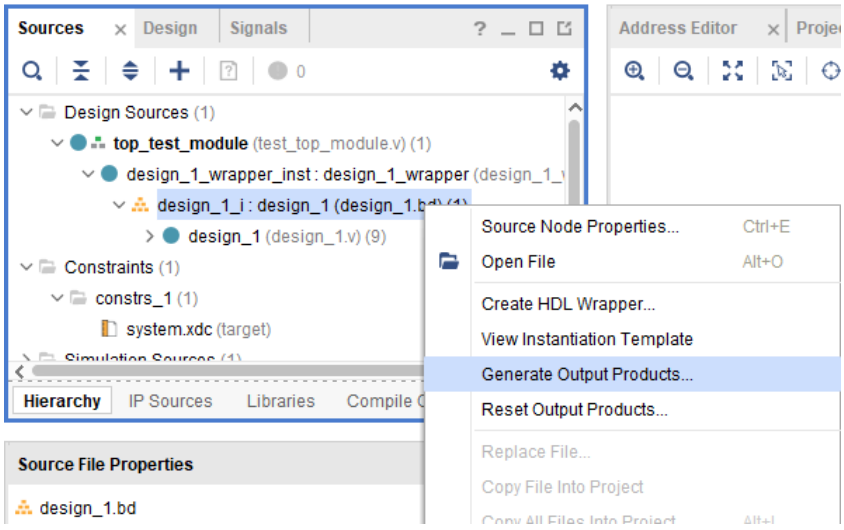

Figure 9: Generate output products

4. After generating output products, right click on the design_1→Create HDL wrapper and click on the 2nd option as shown in below Figure and click ok

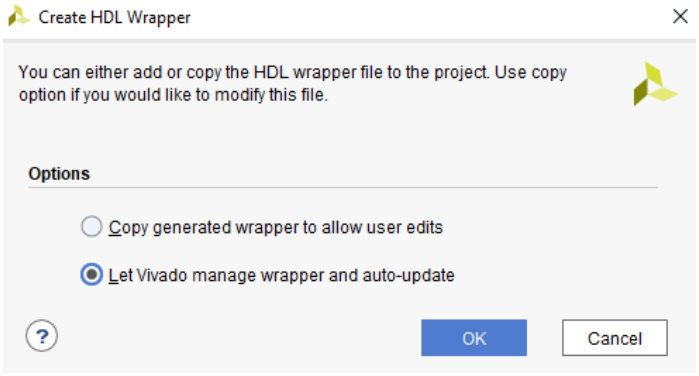

Figure 10: Create HDL wrapper

5. Make Signal connections between the design_1_wrapper that contains NAND flash signals and the top_test_module in the top_test_module. v of the example design as shown in below Figure

```
 // IP Integrator block Instantiation
design_1_wrapper  design_1_wrapper_inst
 (.DDR_addr                (DDR_addr),
  .DDR_ba                  (DDR_ba),
  .DDR_cas_n               (DDR_cas_n),
  .DDR_ck_n                (DDR_ck_n),
  .DDR_ck_p                (DDR_ck_p),
  .DDR_cke                 (DDR_cke),
  .DDR_cs_n                (DDR_cs_n),
  .DDR_dm                  (DDR_dm),
  .DDR_dq                  (DDR_dq),
  .DDR_dqs_n               (DDR_dqs_n),
  .DDR_dqs_p               (DDR_dqs_p),
  .DDR_odt                 (DDR_odt),
  .DDR_ras_n               (DDR_ras_n),
  .DDR_reset_n             (DDR_reset_n),
  .DDR_we_n                (DDR_we_n),
  .DDR_VRN                 (DDR_VRN),
  .DDR_VRP                 (DDR_VRP),
  .MIO                     (MIO),
  .PS_CLK                  (PS_CLK),
  .PS_PORB                 (PS_PORB),
  .PS_SRSTB                (PS_SRSTB),
  .GPIO_tri_o              (gpio_tri_io),
  .GPIO_1_tri_o            (gpio_1_tri_io),
  .GPIO_2_tri_o            (gpio_2_tri_io),
  .GPIO2_tri_o             (gpio2_tri_io),

  // NAND Flash Interface
  .nand_r_b_n_i            ( nand_r_b_n       ),
  .nand_ce_0_n_o           ( nand_ce_0_n_o    ),
  .nand_ce_1_n_o           ( nand_ce_1_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     ),
  .nand_data_dir_o         ( nand_data_dir_o  )
);
```

Figure 11: Instantiation of NAND Host IP in NAND Flash Controller module

6. Give the pin constraint for NAND interface using .xdc file as shown in below figure
7. Save the project and Generate the Bit stream file

3 Implementation Details

3.1 Parameter Settings

NAND IP consists of several parameters. These are to be set before compiling the project. The values of the parameters can be found from the NAND Datasheet. Following are the parameters required by the NAND Host Controller.

1. TPWR : Power-On Recovery Time
2. TRST : Device Resetting Time
3. TR_ECC : Data Transfer Time Flash Array to Data Register, Internal ECC Enabled
4. TR : Data Transfer Time Flash Array to Data Register, Internal ECC Disabled
5. TPROG : PROGRAM PAGE Operation Time
6. TBERS : BLOCK ERASE Operation Time
7. TWB :WE# HIGH to Busy
8. TFEAT :Busy Time for FEATURES command operations
9. TWHR :WE# HIGH to RE# LOW
10. TCCS :Column Address Set Time
11. DEV_PG_SIZE :Page size of the NAND device
12. DEV_OOB_SIZE:OOB size of the NAND device

Most of the parameters are given as the maximum time; these have to be considered as it is. Parameters given with minimum time, additionally consider a buffer time of 50-100ns. Get the values of the timing parameters (ns or us) from the datasheet and using the following example apply all the values to the above parameters. Consider the TRST time, in datasheet this time is defined as 500us (MAX). Based on the clock configured say, it is 100MHz (10ns). Then set the TRST parameter with value = 500us/10ns = 50000d If clock configured is 50MHz(20ns), set the TRST as 500us/20ns = 25000d, Similarly other parameters are to be added before compilation, in the file top_test_module.v.

NOTE: All the above parameters calculations is done with respect to the frequency of 100MHz and values shown is for Micron MT29F64G08AFAAA part. This is just for reference.

3.2 Constraints

Following figure shows the example pin and IO standard constraints.

```
set_property PACKAGE_PIN C20 [get_ports nand_r_b_0_n_i]
set_property PACKAGE_PIN B19 [get_ports nand_r_b_1_n_i]
set_property PACKAGE_PIN J20 [get_ports nand_ce_0_n_o]
set_property PACKAGE_PIN M17 [get_ports nand_ce_1_n_o]

set_property PACKAGE_PIN L19 [get_ports nand_re_n_o]
set_property PACKAGE_PIN D19 [get_ports nand_we_n_o]
set_property PACKAGE_PIN F19 [get_ports nand_cle_o]
set_property PACKAGE_PIN K14 [get_ports nand_ale_o]
set_property PACKAGE_PIN E17 [get_ports nand_wp_n_o]
set_property PACKAGE_PIN N15 [get_ports nand_data_dir_o]
set_property PACKAGE_PIN G19 [get_ports {nand_data_io[0]}]
set_property PACKAGE_PIN L14 [get_ports {nand_data_io[1]}]
set_property PACKAGE_PIN N20 [get_ports {nand_data_io[2]}]
set_property PACKAGE_PIN V17 [get_ports {nand_data_io[3]}]
set_property PACKAGE_PIN T16 [get_ports {nand_data_io[4]}]
set_property PACKAGE_PIN N17 [get_ports {nand_data_io[5]}]
set_property PACKAGE_PIN T20 [get_ports {nand_data_io[6]}]
set_property PACKAGE_PIN W18 [get_ports {nand_data_io[7]}]

set_property IOSTANDARD LVCMOS33 [get_ports nand_r_b_0_n_i]
set_property IOSTANDARD LVCMOS33 [get_ports nand_r_b_1_n_i]
set_property IOSTANDARD LVCMOS33 [get_ports nand_ce_0_n_o]
set_property IOSTANDARD LVCMOS33 [get_ports nand_ce_1_n_o]

set_property IOSTANDARD LVCMOS33 [get_ports nand_re_n_o]
set_property IOSTANDARD LVCMOS33 [get_ports nand_we_n_o]
set_property IOSTANDARD LVCMOS33 [get_ports nand_cle_o]
set_property IOSTANDARD LVCMOS33 [get_ports nand_ale_o]
set_property IOSTANDARD LVCMOS33 [get_ports nand_wp_n_o]
set_property IOSTANDARD LVCMOS33 [get_ports nand_data_dir_o]
set_property IOSTANDARD LVCMOS33 [get_ports {nand_data_io[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {nand_data_io[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {nand_data_io[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {nand_data_io[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {nand_data_io[4]}]
set_property IOSTANDARD LVCMOS33 [get_ports {nand_data_io[5]}]
set_property IOSTANDARD LVCMOS33 [get_ports {nand_data_io[6]}]
set_property IOSTANDARD LVCMOS33 [get_ports {nand_data_io[7]}]
```

Figure 12: Pin Constraints in example design .xdc file

NOTE:
These constraints are in accordance to example design for XC7Z020CLG400-1 FPGA device. Please change the pin constraints for NAND interface for other custom board. Also define the constraints for clock, reset and UART pins accordingly if required for custom board.
GPIO Pins can be used in .xdc file if required.

4 FPGA Implementation

4.1 Resource Utilization

The table below shows the utilization summary from the implementation of NAND Flash Host Controller for Zynq-7000 SoC FPGA device with part number XC7Z020CLG400-1.

Table 2: Device Utilization Summary for NAND Flash Host Controller IP

Resource	Utilization	Available
LUT	2137	53200
FF	1242	106400
BRAM	8	140