## Objective

The objective of this design example is to provide users with a template for implementing the 1G/10G Ethernet or 10GBASE-KR PHY IP.

## Design Example Contents

The design example contains the following:

1. Top-level Verilog code that instantiates the PHY IP and all associated logic along with a test harness that can be used for simulation.
2. Simulation scripts written for Modelsim running on Linux and PC.

This design example was created using Quartus II v13.1 software along with Modelsim SE v10.2c.

## Overall Flow

Follow these steps to simulate and compile this design example.

1. Extract the files into a directory
2. Open design\_example\_wrapper\_nch.sv and set the parameters according to your requirements
3. Open th\_mgmt\_prog\_n.sv and include the appropriate sections based on the simulation you want to run# Load Modelsim
4. Source the sim\_n.tcl script
5. Run the compile\_de procedure to compile the libraries
6. Run the elab\_sim procedure to elaborate the design
7. Run the simulation if desired with elab.tcl
8. Open the Quartus II software, open the project and compile the design

## Usage

### Project and Library Set-Up

In order to set up libraries for both simulation and compilation, you must run the TCL scripts through Modelsim. Follow these steps to set up the project libraries and compile the design:

1. Open the design\_example\_wrapper\_nch.sv file and set the parameters according to your requirements
2. Open Modelsim
3. Source the sim\_n.tcl script within Modelsim.
4. Type “compile\_de” at the Modelsim command prompt to generate and compile the project libraries and compile the design files in Modelsim.
5. Type “elab\_sim” at the Modelsim command prompt to elaborate the design.
6. At this point you can either run a simulation and look at the nodes in the design or you can go directly to the Quartus II software and compile the design.

### Project Set-up Script (sim\_n.tcl)

The included script that sets up the project libraries and allows simulation in Modelsim is called sim\_n.tcl. By default, this script regenerates all the IP before running the simulation. If you have already generated the IP by running the script once and you want to speed future re-runs, set the FORCE\_MW\_REGEN parameter to 0 within the sim\_n.tcl script. This will skip IP regeneration if the IP files have already been generated. If the files do not exist the script will still regenerate the IP.

## Parameter Setup For Quartus II Compilation

### Standalone

The following parameters can be set in the design\_example\_wrapper\_nch.sv file.

**Table 1: design\_example\_wrapper\_nch.sv parameters.**

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Valid Settings** | **Valid For** | **Description** |
| NO\_CHANNEL | 1-12 | Both | Number of channels in design. |
| SYNTH\_AN\_DE | 0,1 | KR Only | 1= Include Auto-negotiation logic for KR 0= Don’t include Auto-negotiation logic for KR This must be set to an opposite value from SYNTH\_CL37ANEG\_DE |
| SYNTH\_LT\_DE | 0,1 | kr only | 1= include link training logic for kr 0= don’t include link training logic for kr |
| OPTIONAL\_RXEQ | 0,1 | KR Only | 1= Include RX Equalization during Link Training 0= Don’t include RX Equalization during Link Training. Use in conjunction with RECONFIG\_CONTROLLER\_DFE and RECONFIG\_CONTROLLER\_CTLE parameters below. |
| RECONFIG\_CONTROLLER\_DFE | 0,1 | KR Only | 1= Enables the DFE feature in the reconfiguration controller 0= Disables the DFE feature in the reconfiguration controller Must be set to ‘1’ if enabling OPTIONAL\_RXEQ parameter. |
| RECONFIG\_CONTROLLER\_CTLE | 0,1 | KR Only | 1= Enables the CTLE feature in the reconfiguration controller 0= Disables the CTLE feature in the reconfiguration controller Must be set to ‘1’ if enabling OPTIONAL\_RXEQ parameter. |
| USER\_RECONFIG\_CONTROL | 0,1 | Both | 1= Expose the reconfiguration interface for user access 0= Do not expose the reconfiguration interface |
| SYNTH\_FEC\_DE | 0,1 | KR Only | 1= Include FEC logic in the PHY (uses soft 10G PCS and FEC) 0= Don’t include FEC logic in the PHY |
| CAPABLE\_FEC | 0,1 | KR Only | Sets power-up/reset value of “FEC Ability” parameter (0xB0 bit 16) |
| ENABLE\_FEC | 0,1 | KR Only | Sets power-up/reset value of “FEC Request” parameter (0xB0 bit 18) |
| SYNTH\_SEQ\_DE | 0,1 | Both | 1= Include sequencer logic in the PHY 0= Don’t include sequencer logic in the PHY |
| SYNTH\_GIGE\_DE | 0,1 | Both | 1= Include GIGE logic in the PHY 0= Don’t include GIGE logic in the PHY |
| SYNTH\_CL37ANEG\_DE | 0,1 | Both | 1= Include Clause 37 logic in the PHY 0= Don’t include Clause 37 logic in the PHY. This must be set to an opposite value from SYNTH\_AN\_DE |
| AN\_TECH\_DE | 6'd4, 6'd5 | Both | 6’d4= 10G mode supported only. 6’d5= 1G and 10G modes supported |
| SYNTH\_1588\_DE | 0,1 | Both | 1= Include IEEE 1588 logic in the PHY 0= Don’t include IEEE 1588 logic in the PHY |
| TENG\_REFCLK | “322.265625 MHz”, “644.53125 MHz” | Both | Reference clock frequency for 10G TX PLL |
| PLL\_TYPE\_10G | "ATX","CMU" | Both | PLL type to use for 10G link. ATX is recommended for best jitter performance. |
| PLL\_TYPE\_1G | “ATX”, “CMU”, “FPLL” | Both | PLL type to use for 1G link. Use ATX or FPLL if channel limited. Setting it to “CMU” will use one transceiver channel (channel 1 or 4 in a transceiver bank).  *Notes: If using ATX you must update the provided MIF file (see*[*MIF Files*](https://fpgawiki.intel.com/wiki/Design_Example_-_10GBase-KR_-_Stratix_V#MIF_Files)*section below) If using fPLL, you must add the following assignment to your QSF file:* set\_instance\_assignment –name PLL\_COMPENSATION\_MODE DIRECT –to “sv\_rcn\_wrapper:LOCAL\_sv\_rc\_wrapper|altera\_xcvr\_10gba se\_kr:INST\_PHY\*.kr\_phy\_ch\_inst|sv\_xcvr\_plls:GIGE\_ENA BLE.PLL\_1G.tx\_pll\_1g|pll[0].generic\_pll.tx\_pll” |

### Integrated

When integrating the design example into your larger design, you no longer need the test harness or the design\_example\_wrapper\_nch.sv file. The module you instantiate into your own design is the sv\_rcn\_wrapper.sv file. This module contains the PHY and all necessary supporting logic to reconfigure the PHY. Set the parameters in the table below based on your requirements.

**Table 2: sv\_rcn\_wrapper.sv parameters.**

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter** | **Valid Settings** | **Valid For** | '*Description* |
| CHANNELS | 1-12 | Both | Number of channels in design |
| PMA\_RD\_AFTER\_WRITE | 0,1 | KR Only | Set to 0 unless debugging |
| SYS\_CLK\_IN\_MHZ | 100-125 | Both | No description. |
| PLL\_REF\_CLK\_IN\_MHZ | “322.265625 MHz”, “644.53125 MHz” | Both | Reference clock frequency for 10G TX PLL |
| KR\_PHY\_SYNTH\_AN | 0, 1 | KR Only | 1= Include Auto-negotiation logic for KR  0= Don’t include Auto-negotiation logic for KR  This must be set to an opposite value from SYNTH\_CL37ANEG |
| KR\_PHY\_SYNTH\_LT | 0, 1 | KR Only | 1= Include Link Training logic for KR  0= Don’t include Link Training logic for KR |
| KR\_PHY\_SYNTH\_FEC | 0,1 | KR Only | 1= Include FEC logic in the PHY (uses soft 10G PCS and FEC)  0= Don’t include FEC logic in the PHY |
| SYNTH\_SEQ | 0,1 | Both | 1= Include sequencer logic in the PHY  0= Don’t include sequencer logic in the PHY |
| KR\_PHY\_SYNTH\_GIGE | 0,1 | Both | 1= Include GIGE logic in the PHY  0= Don’t include GIGE logic in the PHY |
| SYNTH\_GMII | 0,1 | Both | This must be set to 1 when GIGE mode is enabled |
| SYNTH\_CL37ANEG | 0,1 | Both | 1= Include Clause 37 logic in the PHY  0= Don’t include Clause 37 logic in the PHY  This must be set to an opposite value from KR\_PHY\_SYNTH\_AN |
| SYNTH\_1588\_1G | 0,1 | Both | 1= Include IEEE 1588 logic for GIGE in the PHY  0= Don’t include IEEE 1588 logic for GIGE in the PHY |
| SYNTH\_1588\_10G | 0,1 | Both | 1= Include IEEE 1588 logic for 10G in the PHY  0= Don’t include IEEE 1588 logic for 10G in the PHY |
| OPTIONAL\_RXEQ | 0,1 | KR Only | 1= Include RX Equalization during Link Training  0= Don’t include RX Equalization during Link Training |
| RECONFIG\_CONTROLLER\_DFE | 0,1 | KR Only | 1= Enables the DFE feature in the reconfiguration controller  0= Disables the DFE feature in the reconfiguration controller |
| RECONFIG\_CONTROLLER\_CTLE | 0,1 | KR Only | 1= Enables the CTLE feature in the reconfiguration controller  0= Disables the CTLE feature in the reconfiguration controller |
| USER\_RECONFIG\_CONTROL | 0,1 | Both | 1= Expose the reconfiguration interface for user access  0= Do not expose the reconfiguration interface |
| CAPABLE\_FEC | 0,1 | KR Only | Sets power-up/reset value of “FEC Ability” parameter (0xB0 bit 16) |
| ENABLE\_FEC | 0,1 | KR Only | Sets power-up/reset value of “FEC Request” parameter (0xB0 bit 18) |
| KR\_PHY\_BERWIDTH | 4-8 | KR Only | Width of link training BER counter. Use higher value to get better results during LT. |
| AN\_TECH | 6’d4, 6’d5 | Both | 6’d4= 10G mode supported only  6’d5= 1G and 10G modes supported |
| PLL\_TYPE\_1G | ATX,CMU,FPLL | Both | PLL type to use for 1G link. Use ATX or FPLL if channel limited. Use ATX or FPLL if channel limited. Setting it to CMU will use one transceiver channel (channel 1 or 4 in a transceiver bank).  *Notes: If using ATX you must update the provided MIF file (see*[*MIF Files*](https://fpgawiki.intel.com/wiki/Design_Example_-_10GBase-KR_-_Stratix_V#MIF_Files)*section below). If using FPLL you must add the following assignment to your QSF file:* |

### Sample Parameter Settings for Common Configurations

*Note: Parameters not shown do not affect the shown configuration and can be set to your desired value.*

### 1G/10G Line-side Ethernet with Sequencer

SYNTH\_AN\_DE = 0;   
SYNTH\_LT\_DE = 0;   
OPTIONAL\_RXEQ = 0;   
SYNTH\_FEC\_DE = 0;   
CAPABLE\_FEC = 0;   
ENABLE\_FEC = 0;   
SYNTH\_SEQ\_DE = 1;   
SYNTH\_GIGE\_DE = 1;   
AN\_TECH\_DE = 6'd5;

### 1G/10G Line-side Ethernet without Sequencer

SYNTH\_AN\_DE = 0;   
SYNTH\_LT\_DE = 0;   
OPTIONAL\_RXEQ = 0;   
SYNTH\_FEC\_DE = 0;   
CAPABLE\_FEC = 0;   
ENABLE\_FEC = 0;   
SYNTH\_SEQ\_DE = 0;   
SYNTH\_GIGE\_DE= 1;   
AN\_TECH\_DE = 6'd5;

### 10GBASE-KR Backplane Ethernet with Auto-Negotiation and RX Equalization (10G Only)

SYNTH\_AN\_DE = 1;   
SYNTH\_LT\_DE = 1;   
OPTIONAL\_RXEQ = 1;   
RECONFIG\_CONTROLLER\_DFE = 1;  
RECONFIG\_CONTROLLER\_CTLE = 1;   
SYNTH\_FEC\_DE = 0;   
CAPABLE\_FEC = 0;   
ENABLE\_FEC = 0;   
SYNTH\_SEQ\_DE = 1;   
SYNTH\_GIGE\_DE = 0;   
SYNTH\_CL37ANEG\_DE = 0;   
AN\_TECH\_DE = 6'd4;   
SYNTH\_1588\_DE = 0;

### 10GBASE-KR Backplane Ethernet with Auto-Negotiation and FEC, No RX Equalization (10G Only)

SYNTH\_AN\_DE = 1;  
SYNTH\_LT\_DE = 1;  
OPTIONAL\_RXEQ = 0;  
RECONFIG\_CONTROLLER\_DFE = 0;  
RECONFIG\_CONTROLLER\_CTLE = 0;  
SYNTH\_FEC\_DE = 1;  
CAPABLE\_FEC = 1;  
ENABLE\_FEC = 1;  
SYNTH\_SEQ\_DE = 1;  
SYNTH\_GIGE\_DE = 0;  
SYNTH\_CL37ANEG\_DE = 0;  
AN\_TECH\_DE = 6'd4;  
SYNTH\_1588\_DE = 0;

## "Management Master" Micro-Controller

### th\_mgmt\_master

The Test Harness Management Master (“th\_mgmt\_master”) block is an HDL-based, synthesizable micro-controller that is used in the test harness to control the links and pass data to/from the data generators and checkers. The code that runs on this management master is contained in the th\_mgmt\_program\_n.sv file. This file is set up for a two channel design by default. If you change the number of channels in the design and you want to simulate it, you must modify this file to control the links.

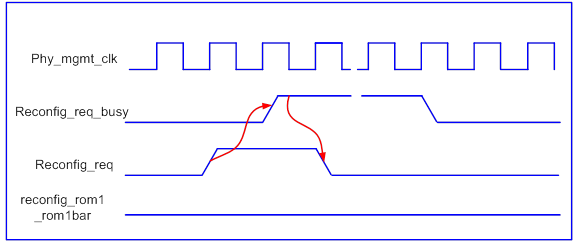
### Management Program Communication with Design

The management program communicates with the design via programmable IOs (PIOs). The PIOs for control of the simulation are defined in the test\_harness.sv file and consist of handshaking signals between the PHY and the test harness. The management master micro-controller reads and writes these PIOs to control the simulation.

### Using Non-Sequencer Mode

The sequencer logic in the PHY allows the IP to automatically react to changes to incoming data and reconfigure itself into another mode (either 1G or 10G). If you choose to disable the sequencer, here are some things to keep in mind:

1. Rate change is controlled manually using the reconfig\_req and reconfig\_rom1\_rom0bar ports. These ports are exposed when you set SYNTH\_SEQ\_DE = 0.
2. The reconfig\_req port is the same width as the number of channels you are instantiating. Each bit of the port corresponds to one channel. This port initiates a request for reconfiguration.
3. The reconfig\_rom1\_rom0bar port is also the same width as the number of channels. This port indicates which mode you would like the channel reconfigured into: 1 = GigE, 0 = 10G.
4. The timing relationship and handshaking for these signals is as shown below. The figure below shows a reconfiguration request for the channel into 10G mode. When using the supplied test harness, this handshaking is automatically taken care of when the instructions in the th\_mgmt\_program are followed.

[](https://fpgawiki.intel.com/wiki/File:10Gb_Ethernet_timing.png)

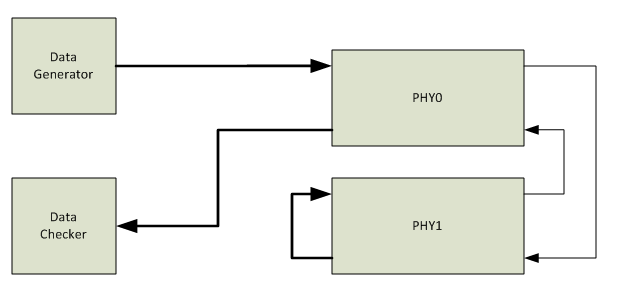
### Modifying the Test Harness Management Program (th\_mgmt\_program\_n.sv)

This program must only be modified if you are changing the number of channels in the design. The modifications in this file are mainly related to controlling the mode of the PHY and the start/end of the data generators and checkers. The comments in the file explain what is being done so you can easily modify it as required. Modify the following items when changing the number of channels:

1. Define address locations for each new channel at the top of the file.
2. Add/remove lines to wait for all channels’ block\_lock (for 10G) or rx\_syncstatus (for 1G) signals to go high before beginning traffic.
3. Add/remove lines to change modes of the PHYs.

### Modifying Number of Channels Used

By default, the design example is set up for two channels. The channels are connected as shown below and can be used to show 1G/10G line-side as well as 10GBASE-KR backplane Ethernet functionality.

[](https://fpgawiki.intel.com/wiki/File:Channel_connection.PNG)

There are two data generators and checkers connected to PHY0: an XGMII generator/checker and a GMII generator/checker.

### For simulation & compilation in Quartus

In order to simulate and compile the design example with a different number of channels, you must modify the connections of the data generators/checkers and the PHYs to reflect your desired implementation. Follow these general steps to modify the design:

1. Set the “NO\_CHANNEL” parameter in the design\_example\_wrapper\_nch.sv file.
2. Instantiate more data generators and checkers in the test\_harness.sv file as required.
3. Modify the data connections to/from the data generators and checkers as appropriate. These connections start around line 219 of the design\_example\_wrapper\_nch.sv file.
4. Modify the th\_mgmt\_program\_n.sv file to account for the change in number of channels. See comments in the file for more details.
5. Re-run the sim\_n.tcl script and set the appropriate parameters.
6. Run the compile\_de and elab\_sim procedures as shown previously.
7. After simulation is complete, compile the design using the Quartus II software.

### For compilation in Quartus only (no simulation)

In this case you do not need the test\_harness since no simulation will be performed. You will simply be integrating the sv\_rcn\_wrapper.sv file into your design. Ensure the libraries have been properly compiled as described in the “Project and Library Set-Up” section above so that the Quartus II software can find all the lower-level modules required for the design.

1. Set the “CHANNELS” parameter to the desired number of channels in the sv\_rcn\_wrapper.sv file.
2. Set other parameters for the PHY as desired.
3. Instantiate the sv\_rcn\_wrapper.sv file in your design.
4. Copy the /base\_kr\_top, /rst\_ctrl and /reconfig folders and all source files to your project directory.
5. Add the reconfig.qip, base\_kr\_top.qip and rst\_ctrl.qip files to your project using Quartus and compile.

### How to Access the Reconfiguration Controller

The design example provides user access to the reconfiguration controller to allow you to change any of the PHY analog parameters or settings. For example, you can use the reconfiguration interface to change the CTLE setting in real-time. To expose the reconfiguration interface, set the USER\_RECONFIG\_CONTROL parameter to ‘1’. This will expose the avmm\_\* ports and allow you to access the reconfiguration controller.

When in 10GBASE-KR backplane mode, the 10GBASE-KR PHY IP bring-up process (AN, LT) requires access to the reconfiguration controller. This interface must be arbitrated between the PHY and the user. This arbitration is done through register settings.

To request access to the reconfiguration controller interface, write a ‘1’ to offset 0x80, bit 0 and wait for the grant register 0x80, bit 8 to go high. Once the grant bit is asserted, you have access to the reconfiguration controller’s AVMM interface until you de-assert the request. To de-assert the request, write a ‘0’ to 0x80, bit 0. Note that while the user has access to the reconfiguration controller’s AVMM interface, none of the KR PHYs can perform AN or LT if something happens to the link.

**Table 3: Register map for user access to the reconfiguration controller.**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Offset** | **Access** | **Register** | **Data Field** | **Data Bit(s)** | **Description** |
| 0x00-0x7F | R/W | Reconfig controller AVMM address |  |  | Direct user access to the PHY reconfig controller IP. Use if status register [0] = 1 and [8] = 1 |
| 0x80 | R/W | Status Register | User reconfig controller AVMM access request | 0 | Write 1 to request reconfig controller AVMM access. AVMM available when grant bit-8 is asserted. After grant, interface is the same as the reconfig controller. |
| 0x80 | NA | Status Register | Reserved | 7:1 | Not Used |
| 0x80 | RO | Status Register | Grant for user reconfig controller AVMM access | 8 | Grant will indicate when the user can access the reconfig controller AVMM interface |
| 0x80 | NA | Status Register | Reserved | 15:9 | Not Used |
| 0x80 | RO | Status Register | Channel in LT/AN mode | 31:16 | Indicate when the channel is in LT/AN mode. Channel 0 = bit-16, channel 1 = bit-17, channel 2 = bit-18. If number of channels is less than 16, the remaining non associated bits are zeros. |
| 0x81 | R/W | Scratch Register | User Data | 31:0 | Scratch register for user data or AVMM read/write tests. |
| 0x82 | R/W | Disable\_tsle\_dfe\_before\_an | Disable DFE before AN | 0 | 1=Disables PHY DFE circuit before starting AN. (Normal Operations). 0= do not disable DFE and proceed to AN with existing settings. |
| 0x82 | R/W | Disable\_tsle\_dfe\_before\_an | Disables CTLE before AN | 1 | 1=Disables PHY CTLE circuit before starting AN. (Normal Operations). 0= do not disable CTLE and proceed to AN with existing settings. |
| 0x82 | NA | Disable\_tsle\_dfe\_before\_an | Reserved | 31:2 | Not used. |
| 0x83 | R/W | Bypass\_ctle\_reconfig | Bypass CTLE reconfig | 0 | 1=Bypass PHY request and data to reconfig the CTLE settings (useful for keeping user specified settings). 0= Honor PHY request to reconfig the CTLE per settings/mode provided by PHY (Normal Operations). |
| 0x83 | NA | Bypass\_ctle\_reconfig | Reserved | 31:1 | Not Used. |

### MIF Files

Creating and Modifying MIF files for Mode Reconfiguration The 1G/10G Line-side and 10GBASE-KR Backplane Ethernet PHY IP uses PCS transceiver reconfiguration to switch between different operating modes (this is separate from the PMA reconfiguration process that happens during Link Training). There are three main modes the PHY operates in:

1. GigE mode: employs the transceiver’s Standard PCS data path
2. 10GBASE-R mode: employs the transceiver’s 10G PCS data path in 10GBASE-R mode
3. Auto-negotiation/Link Training (AN/LT) mode: employs the transceiver’s 10G PCS data path configured in Low Latency mode
4. 10GBASE-R with FEC: employs the transceiver’s 10G PCS data path configured in Low Latency mode, which feeds into the FEC logic and then a soft 10G PCS.

Each mode above requires a MIF file that is used for reconfiguration. This design example is set up to have a single, combined MIF file that contains the mode/setting combinations shown in Table 4 below. This MIF file is named all\_modes.mif and is contained in the project directory.

**Table 4: all\_modes.mif file default contents**

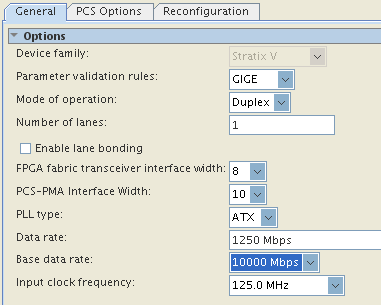
|  |  |  |  |
| --- | --- | --- | --- |
| **Mode** | **Reference Clock** | **IEEE 1588** | **IP File to use to create new variation** |
| 10G w/o FEC | 322MHz | Disabled | baser.v |
| GigE (CMU or fPLL only) | 125MHz | Disabled | oneg.v |
| AN/LT | 322MHz | N/A | ll.v |
| 10G | 322MHz | Enabled | baser.v |
| GigE | 125MHz | Enabled | oneg.v |
| 10G w/ FEC | 322MHz | Disabled | fec\_ll.v |

**If your settings or reference clocks differ from these you will need to create new MIFs for those modes and replace the appropriate sections in the all\_modes.mif file.** For example, if your design uses 10G mode with a 644 MHz reference clock, you must create a new MIF file for this mode and replace the existing 10G section in the all\_modes.mif file with the new content. To help in creating new MIF files, there are four base Megawizard IP files provided in the /mif\_generation directory: baser.v, ll.v, fec\_ll.v and oneg.v. To create new variations specific to your requirements, do the following:

1. Copy the file to a new directory outside of the current project directory.
2. Open the file using the MegaWizard Plug-In Manager.
3. Modify the settings as desired and click Finish.
4. Create a new project with this file as the top-level file. Make sure to add the .qip file to the project to set up the libraries correctly.
5. Choose a Stratix V device as the target device.
6. Compile the project. Two MIF files will be generated during the Assembler stage – one for the PLL and one for the channel. Open the \*channel\* MIF file from the /reconfig\_mif folder of that project directory.
7. Modify the MIF to remove the PMA settings as shown in the section below.
8. Copy and paste the newly-created MIF file over the appropriate section of the all\_modes.mif file. Make sure the number of lines match – there should be 163 lines after the PMA section is removed.

Note: if you use an ATX PLL for GigE mode (by setting PLL\_TYPE\_1G = ”ATX” in design\_example\_wrapper\_nch.sv) you must update the MIF file. Open the oneg.v IP variant file and use the following settings:

PLL Type = ATX Base data rate = 10000 Mbps

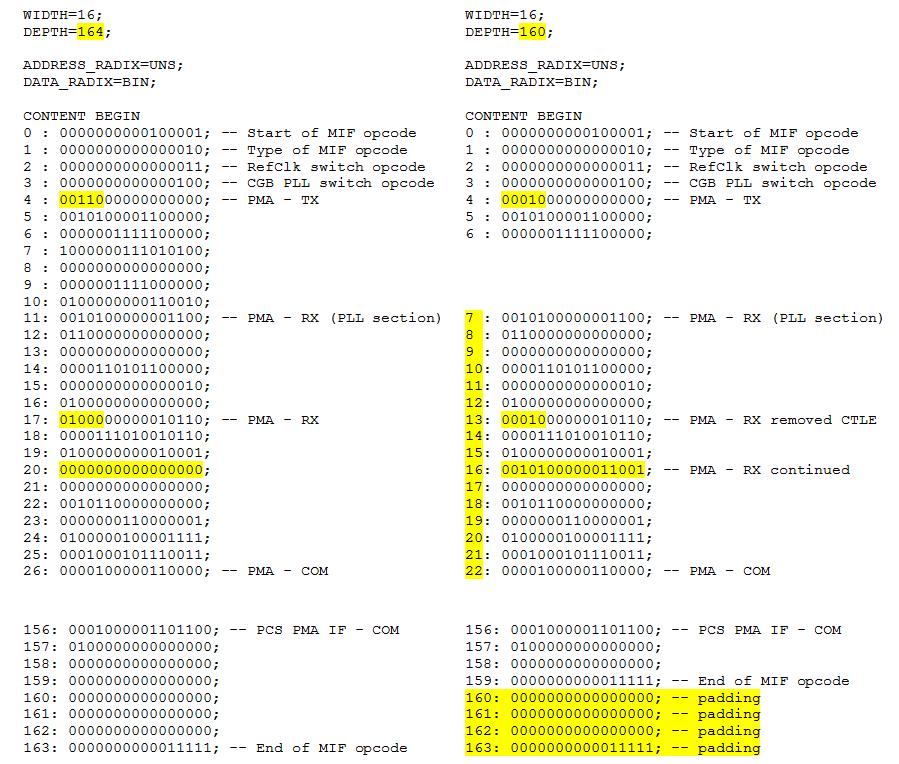
[](https://fpgawiki.intel.com/wiki/File:PLL_config.png)

### Removing PMA Settings from the MIF File

As noted in Chapter 4 of the [Altera Transceiver PHY IP Core User Guide](http://www.altera.com/literature/ug/xcvr_user_guide.pdf), special considerations must be taken for MIF files when using 10GBASE-KR mode. In this design example we have modified the MIF file for all modes in this manner. The MIF format contains all bit settings for the transceiver PMA and PCS. Because the 10GBASE-KR PHY IP Core only requires PCS reconfiguration for a rate change, the PMA settings should not be changed. Removing the PMA settings from the MIF file prevents an unintended overwrite of the PMA parameters set during Link Training. In this design example, all of the sub-MIFs in the all\_modes.mif file have their PMA sections removed. A standard reconfiguration MIF file generated by Quartus contains 164 words of data. The PMA settings take up four words in the MIF file, so you will end up with a MIF file that is 160 words deep. The steps below outline how to remove PMA settings from a MIF file.

1. Replace line 17 with “13 : 0001000000010110; -- PMA - RX removed CTLE"
2. Replace line 20 with “16 : 0010100000011001; -- PMA - RX continued”
3. Replace line 4 with “4  : 0001000000000000; -- PMA – TX”
4. Remove lines 7-10. These lines contain the TX settings (VOD, post-tap, pre-tap)
5. Re-number the lines starting with the old line 11

Below is an example to illustrate how this is done. Only a portion of the file is shown for brevity. All changes are highlighted.

[](https://fpgawiki.intel.com/wiki/File:Figure.PNG)

Add padded zeros to the last four lines of the MIF (addresses 160-163) so it fits into the space defined in all\_modes.mif. Once the new MIF file is modified as shown above, copy the contents of it and paste it over the section in the all\_modes.mif file you wish to replace.

### Timing Constraints

A parameterized SDC file is included with the design example and should be used as a base SDC and modified as appropriate depending on your implementation. This SDC file is located in the project directory and is named design\_example\_wrapper\_nch.sdc. Set the following parameters according to your design implementation:

* num\_channels: specify the total number of 1G/10G or KR channels you are implementing in the design. Default is 2.
* period\_10g: specify the period (in ns) of the 10G reference clock. Default is 3.103 (322 MHz).
* period\_8g: specify the period (in ns) of the 1G reference clock. Default is 8 (125 MHz).
* period\_mgmt: specify the period (in ns) of the phy\_mgmt\_clk. Default is 10 (100 MHz).
* path\_project: specify the hierarchical path to the instance name for sv\_rcn\_wrapper. Modify this when integrating the sv\_rcn\_wrapper.sv file into your design.