

Clock Management in a Gigabit Ethernet Physical Layer Transceiver Circuit

Juan C. Diaz
Agere Systems
jdiaz@agere.com

Marta Saburit
Agere Systems
msaburit@agere.com

Abstract

This paper describes the clock management of a mixed signal, high-speed, multi-clock, fully synchronous circuit. The MA1111A13 circuit clock distribution is a complicated structure that seamlessly incorporates different well-known techniques for power reduction, asynchronous clock domains inter-operability, and compatibility with different IO timing standards and data rates. This complex clocking scheme has been successfully integrated into the standard semi-custom physical design flow. The physical implementation of the clock network with Synopsys Astro is also presented.

1. Introduction

The MA1111A13 [1] implements the gigabit physical layer (PHY) functionalities stated in the 802.3[2] standard. In order to accomplish the standard requirements, the circuit has an analog section that acts as interface to the Ethernet physical medium (CAT5 cable). The digital section of the circuit is mainly devoted to implement the DSP-intensive functions required by the gigabit standard, the specific control structures for the management of the DSP blocks sequencing, and the implementation of the controlling Finite State Machines (FSM) described in [2].

In addition to this, the digital section also implements several different interfaces to the Medium Access Control (MAC layer).

1.1. Gigabit Ethernet PHY functionality overview

A Gigabit PHY transceiver interfaces a MAC layer, where 1Gb/s Ethernet frames are transferred by an eight, four or even single bit wide digital bus (depending on the particular MAC interface technology), to a four cable pairs digital medium, using five voltage levels in each. Ethernet frames are mapped to and de-mapped from 4D PAM5 (four dimensions, five voltage levels) symbols following and standardized line code.

The figure below sketches the main blocks in the circuit. The analog block, also named Analog Front End (AFE), performs the adaptation of the TX and RX signals to the physical medium. Apart from electrical adaptation, it also implements other critical functions like analog echo cancellation (the physical medium is shared by TX and RX in the gigabit standard), analog amplification and filtering, and RX clock recovery by means of a PLL. Finally, it also implements the Analog/Digital conversion, using an oversampling architecture called IntelliRate to optimise the Signal to Noise Ratio (SNR) at the input of the digital part.

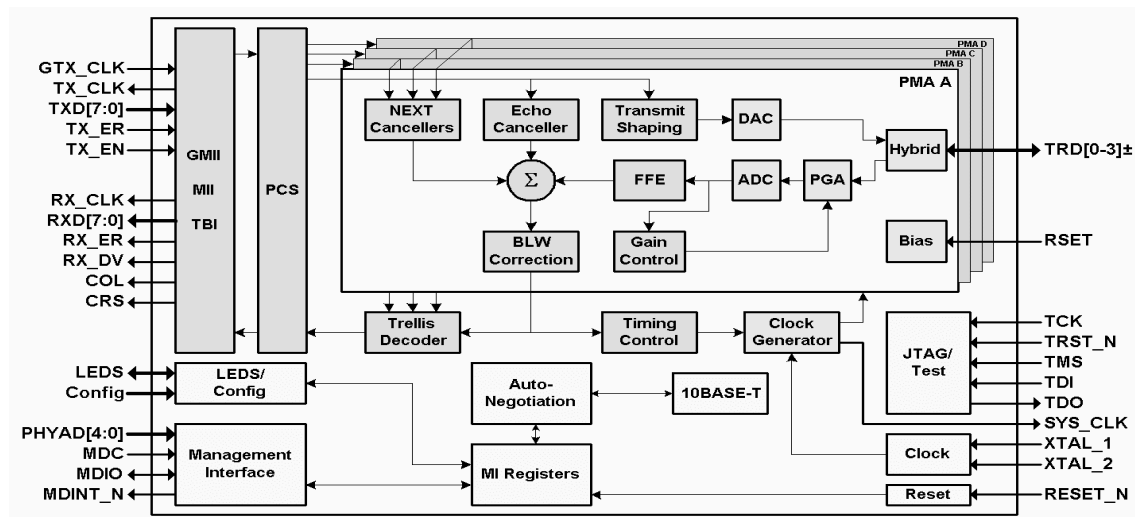


Fig. 1: MA1111A13 Integrated Circuit diagram

The digital logic implements several DSP processes over transmission and reception paths. Obviously, RX path DSP is by far more complicated. It includes different blocks to perform RX channel equalization (Feed Forward Equalization adaptation loop), line transformer high-pass response correction (Base Line Wandering), and digital echo and near crosstalk cancellation (Echo and Next adaptation loops). It also implements the digital portions of the clock recovery (Timing Recovery) and RX path adaptive amplification (Programmable Gain Amplification), complemented with the PLL and PGA blocks in the AFE.

The digital core also implements the standardized Physical Layer Coding (PCS), including scrambling and trellis convolutional encoding in the TX path, and descrambling and Trellis decoding (also known as Viterbi) in the RX path. Finally, the digital core performs the interface functions to the MAC layer (also known as Reconciliation Sub-layer, RS), including rate adaptation and clock generation for several data rates and standardized interfaces (MII, GMII, RGMII HP and 3COM, SGMII, and GBIC).

1.2. Clock issues introduction

Although the MA1111A3 circuit strictly follows the paradigm of synchronous digital design, its clock management is far from being straightforward.

First, the gigabit standard delimits plesiochronous clock domains (asynchronous, with a limited deviation from a central frequency). These plesiochronous clock domains require control signals re-synchronization and data rate adaptation. In addition to this, the circuit must implement different MAC interfaces, with different clocking characteristics, requiring ad-hoc solutions like clock multiplexing, clock division, and clock phase shifting among others. Moreover, the circuit production test is based on full-scan, which does not suit very well with asynchronous clock domains, so it has been decided to provide a single clock for all the scan-flops in the circuit in test mode. This led to the emergence of different modes for functional, scan-based test, and other test modes described below. Finally, power consumption is a major concern for high-volume productions, especially in gigabit networking, so aggressive strategies of clock gating have been put in place in order to save power. Although clock gating is a well-known and widely accepted strategy to reduce power, it has to be carefully applied because it concerns the critical and sensitive clock signals.

2. Clock domains

As it has been already mentioned, there are several asynchronous clock domains in the design. Besides, the different functional modes mix up the picture by

introducing particular subtleties for each. In addition to this, test modes create their own clock schemes, with their particular requirements. This section presents the organization of the clock management that allows an ordered control of the clock signals.

2.1. Clock sources

The clock signals that can be considered source of synchronism in the circuit are mainly three. Two clock signals are driven from the analog block: “**clk_ref_125**”, 125 MHz free running clock obtained by PLL multiplication of a 25MHz local oscillator, and “**clk_pll_125**”, 125 MHz free running clock locked to the phase recovered from the reception signals. These two clock signals are plesiochronous, because both have a frequency range of 125MHz±100ppm. The third functional clock source is the gigabit MAC interface clock input, driven by an external pin called “**GTX_CLK**”. This clock source is also plesiochronous with the same frequency range. In addition to these full-rate clocks, there are a couple of internal low-rate clocks, generated by division of the full-rate ones, named “**tx_clk_b100**” and “**tx_clk_b10**”. These two clocks can drive a given clock domain, that is considered asynchronous to the rest of clocks (irrespective of the lower frequency)

Apart from these basic clocks, there is a couple of clocks more, in preparation for the future implementation of the serial MAC interfaces SGMII and GBIC: “**s_tx_clk**”, that is supposed to be driven by the local oscillator of the SERDES block, and “**s_rx_clk**”, recovered from the serial MAC interface data. They have the same characteristics as the rest of clocks. All these functional clock sources drive asynchronous clock domains.

An external pin drives the test mode clock “**TCK**”. This clock source is routed to all the scan-flops in the design by multiplexing.

2.2. Centralized clock management

The clock management philosophy is based on a centralized clock control. All clock sources are routed to a block (“**clk_gen**”) that handles all the clock control signals and drives all the clock output branches. Each clock branch that drives circuit flops passes through the “**clk_gen**” block, where clock sources are multiplexed according to the circuit mode to drive it. Clock branches are also gated-off by means of clock gating control signals arriving to the “**clk_gen**”. The advantages of this centralized clock control structure are clear: all the sensitive clock control is confined into a single place, and clock tree balancing can be easily automated as described in a section below. In addition to this, clock branches

management can be systematic, applying the same type of multiplexing/gating structures everywhere.

2.3. Clock control structures and modes

The “clk_gen” block uses several clock control structures to deal with the different situations created by the circuit functionality and the test modes.

2.3.1 Functional clock multiplexing and gating. There are two types of functional clock gating in the design: **long-term clock gating**, when clocks are stopped because the data rate does not need the part of the circuit clocked by a given clock branch, and **decimation**, when clock frequency is reduced by gating-off clock cycles, for low rate adaptive filtering. The first case can be easily solved by keeping the logic at reset state when clock is gated, so clock glitches are harmless. In the second case, glitch free gating is key. The following clock gating structure has been used to avoid clock glitches:

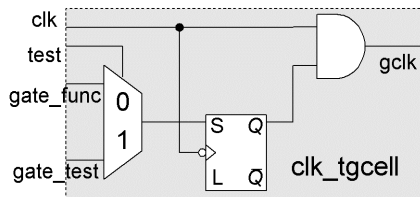


Fig. 2: Glitch-free clock gating structure

Some clock branches require glitch-free source multiplexing, because they can change when the circuit is up and running. For these branches, the following glitch-free multiplexing structure has been used:

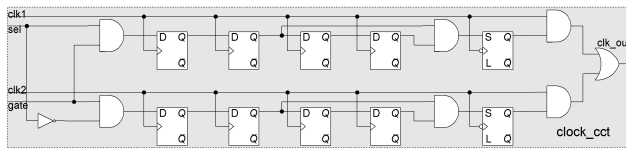


Fig. 3: Glitch-free multiplexing structure

This logic structure works when clock frequencies are similar, by stopping both clock sources during at least one guard cycle, regardless of the clock phases. Finally, there are some branches that can be driven by different sources, but clock sources can only be changed when the branch is at reset state. In these cases, no special attention has been taken but confining the multiplexing logic into the “clk_gen” block.

2.3.2 Test Modes. The most important test mode is the **scan test mode**. Clock management block contains the logic to drive all the circuit clock branches that contain

scan flops from the test clock “TCK”, when this test mode is selected by setting external pins. As the selection of the mode is static, there is no need to care about clock glitches during the clock multiplexing. Another additional test mode that has been incorporated is the **pseudo-functional test mode**. This mode programs the circuit to work in functional mode independently from the AFE clocks: the test clock “TCK” drives the branches that are normally driven by the AFE clocks “clk_pll_125” and “clk_ref_125”. When the circuit is programmed in this test mode by setting external pins, circuit functionality can be tested (in loop-back mode), irrespectively of the AFE state.

3. Special clocking structures

3.1. Asynchronous clock domains

The existence of different asynchronous clock domains requires the application of well-known methods to adapt data and control signals crossing clock domains boundaries.

3.1.1 Full-rate data. Using asynchronous FIFO structures accommodates full-rate data between clock domains. A FIFO is a data storage structure where data is written using the source clock, and it is read using the target clock. The clock accommodation is provided by reading data some time after it is actually written, and usually the number of data positions required in the FIFO is calculated by guaranteeing a number of positions to store data values prior to their reading, and some more positions to leave room for a writing clock faster than the reading clock. Usually these two margins are symmetrical: the number of clock position in front of the reading starting point (also referred as reading pointer) is the same as the number of positions after it. Given the characteristics of the 802.3 standard clocks (125MHz±100ppm), the maximum difference between two standard clocks is 200ppm. In other words, the fastest clock will swing 5001 times when the slowest one swings 5000 times. The maximum Ethernet frame size is 1525 bytes, so a single FIFO position would be enough to fit the maximum clock difference. However, it has to be taken into account the phase uncertainty between clocks, which needs an extra FIFO position. Then, the minimum Ethernet data FIFO size is 3 positions. There are additional issues related to Ethernet data FIFO dimensioning and control, which are out of the scope of the present paper [3]

3.1.2 Re-synchronization registers. As control signals are supposed to change at a lower rate than data signals, they are synchronized by the typical two stages synchronizer block:

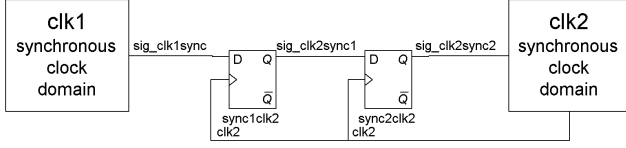


Fig. 4: Two-stages synchronizer structure

The Mean Time Between Failures (MTBF) probability due to a metastable sig_clk2sync2 signal can be calculated as follows [4]:

$$\text{MTBF} = \frac{e^{(t_r/\tau)}}{2 \cdot f_{\text{clock}} \cdot f_{\text{data}} \cdot T_0}$$

where:

$$T_0 = 0.8 \cdot (t_{\text{setup}} + t_{\text{hold}}) \approx 0.4\text{ns}$$

$$\tau = \text{library_parameter} \approx 0.15\text{ns}$$

t_r is the “time allowed for metastability to resolve itself”, and it can be calculated as clk2 clock period minus the path delay from sync1clk2(CK) to sync2clk2(D) . It is important to note that the MTBF depends on the path delay; so minimizing this delay is key for reliability. Multibit control signals use a different re-synchronization method. A signal indicating control signal change is generated at the “transmitting” clock domain, and this one is re-synchronized and used to enable the flops that latch the control signal in the “receiving” clock side, as indicated in the figure.

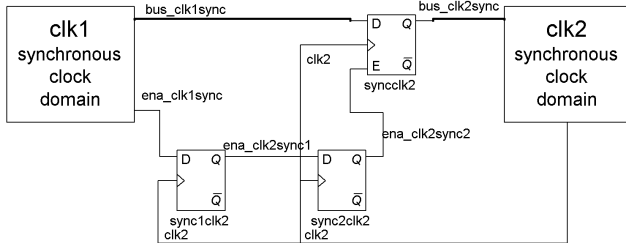


Fig. 5: Multibit synchronizer structure

3.2. IO interfaces synchronization: front-ends

The circuit interface with the MAC layer has to meet different timing requirements, sometimes incompatible. In addition to this, the timing requirements are so tight that they do not allow the possible insertion delay that would be introduced by a clock tree connected to the external pin. The “front-end” logic scheme has been implemented as solution for these problems. This front-end logic seamlessly synchronizes the incoming data with the incoming clock immediately after input pins.

3.2.1 Input clock signals out of the centralized distribution. Front-end flops have been intentionally left

out of the scan-chains in order to avoid the need for clock multiplexing in these sensitive timing interfaces. Then, input clock signals are directly connected to front-end flops, as an exception to the centralized clock management. Overriding front-end outputs connected to the rest of the logic during scan-test mode minimizes the impact in the scan-based testability.

3.2.2 Different front-end slices for different timing constraints. As timing requirements are sometimes incompatible, there is not a single solution, so a different front-end module has been implemented fulfilling a different set of timing constraints. Front-end outputs are multiplexed according to the interface that is running (statically, no “on the fly” MAC interfaces changes are allowed), while the rest of interfaces are kept at reset state.

3.2.3 Transparent-pipe. The timing interface between the front-end flops that synchronize the incoming signals and the internal logic is based on a transparent pipeline scheme. After latching the front-end flops, the incoming clock is routed up to the central clock management module, where it is multiplexed to drive the appropriate clock branches. There, it suffers the insertion delay of the clock tree, so its phase is shifted with respect to the front-end original clock. In order to ensure that no setup/hold violations happen in the paths from the front-end flops to the internal logic, the clock path up to the clock management block is artificially delayed to be always longer than the data path, so clock edge always arrives later than data changes. This gives a timing clean interface without the need for balancing the insertion delay of the internal logic clock branches with the front-end clocks.

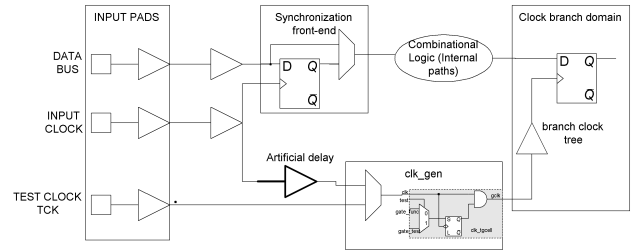


Fig. 6: Front-end and transparent pipeline

3.3. Analog/digital interface

The interoperability of the analog and digital parts of the circuit has been made possible by synchronizing the signals at the A/D interface. As the AFE drives the functional logic sources, they are used to synchronize the clock data signals in the analog-to-digital and digital-to-analog directions. The scheme is valid for both AFE clock sources “ clk_ref_125 ” and “ clk_pll_125 ”.

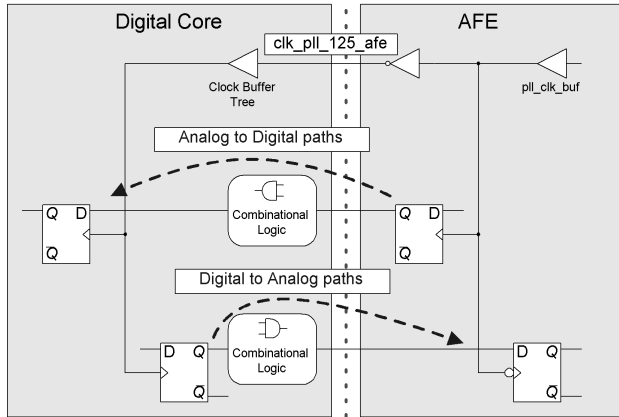


Fig. 7: Analog/Digital interface simplified representation

3.3.1 Analog-to-digital clock scheme. The AFE clock that latches the A-to-D flops is inverted prior to the digital part, performing a virtual half a cycle (4ns) clock shift. Therefore, there is no possibility of having hold violations at the digital side. The clock insertion delay in the digital side, slightly smaller than 4ns in worst case, also shifts the clock phase, resulting in a net effect of less than a clock cycle, large enough to accommodate the logic in the A-to-D path.

3.3.2 Digital-to-analog clock scheme. Flops in the digital part are latched using the clock(s) provided by the AFE after being routed, so it is delayed by the digital clock tree buffers around 3.5ns (worst case). Inside the AFE, data coming from digital is latched with the same clock that is provided to the digital (inverted with respect to the internal AFE timing reference). Thanks to the digital insertion delay, hold time violations in the analog flops are prevented (the best case insertion delay is around 1.5ns); the insertion delay worst case, 3.5ns, is not an issue as far as digital-to-analog combinational logic delay is small.

4. Clock tree generation

This complex clocking scheme requires a carefully planned clock tree generation in layout. In order to get all the clock branches in the circuit well balanced in all the possible modes, clock trees must be synthesized following a certain sequence. This method relies on having an automated Clock Tree Synthesis (CTS) software tool able to deal with gated clocks (GCTS).

4.1. Automated GCTS on test clock “TCK”

As most of the circuit flops (with relevant exceptions like the MAC interface front-ends) are driven by the test clock source “TCK” in scan test mode, this clock is the first one to be synthesized. The GCTS tool is able to balance the insertion delay of the clock arriving to all the scan flops, considering the multiplexing and gating cells in the clock distribution as part of the clock network. Once this clock source has been synthesized, all the buffers in the clock branches at the output of the central clock management block are “frozen” (they are not going to be changed anymore), and scan test clock balancing is solved.

4.2. Functional clock balancing

The balancing of the insertion delay from the test clock source up to the flops in the circuit clock branches does not guarantee a correct balancing when these branches are driven by their functional clock sources. Now the balancing of the insertion delay from the functional sources up to the clock branches that they drive in any of the functional modes is required.

	Clock source name			
	clk_ref_125	clk_pll_125	GTX_CLK	TCK
clk_dac_b1100	✗	✗		✗
gclk_b100_tx	✗	✗		✗
gclk_b100_rx		✗		✗
gclk_b10		✗		✗
gclk_b10_dbg		✗		✗
gclk_b100_b1000		✗		✗
gclk_b100_b1000_au		✗		✗
gclk_b1000		✗		✗
gclk_b1000_au		✗		✗
clk_pll_125				✗
gclk_tbi_rx			✗	✗
ebufclk		✗	✗	
jtag_clk				✗
jtag_clk_n				✗
clk_rx_fe				✗

Table 1: Simplified clock balancing matrix

4.2.1 Clock balancing matrix. The information required for the functional clock balancing is organized as a connectivity matrix, where a cell is ticked when the clock branch in the cell line can be driven by the clock source in the cell column. When a particular source drives a set of branches, they need balancing irrespectively of the clock multiplexing/gating signals. This adds extra (maybe useless) requirements to the clock balancing system, but it allows systematisation. Then, the functional clock

balancing can be achieved by synthesizing the functional clock sources from their source point up to the central clock management block in a way that the total delay from the clock source to the flops is balanced.

4.2.2 Manual balancing. As the test clock has already been synthesized, each clock branch that can be driven by a particular source has a real insertion delay from the multiplexing cell in the central clock management block up to the flops in the branch. The test clock drives one input of the multiplexing gate, while the functional clock drives the other one. Supposing that the gate delay is similar for both inputs, the path from the clock source up to the multiplexers inputs in the clock management block can be carefully buffered in order to compensate for the different insertion delays from the multiplexers up to the flops, provided by the test clock synthesis run. This can be done either manually or by setting synchronization points with an annotated insertion delay at the multiplexer functional inputs and then run automated CTS.

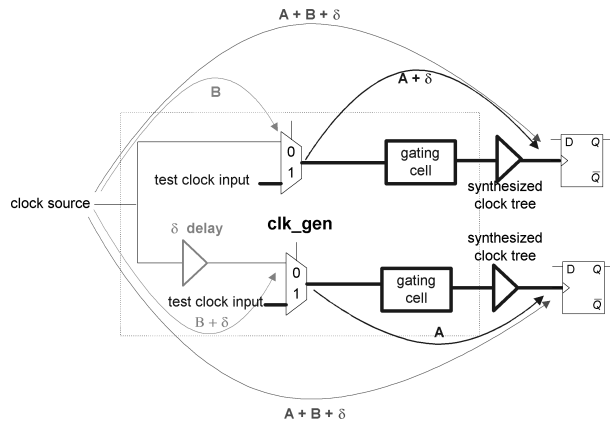


Fig. 8: Balancing of pre-synthesized clock branches

4.2.3 Automated balancing with Astro CTS. Astro CTS can take into consideration the insertion delay of the already deployed clock tree when synthesizing a clock branch. This is particularly useful for the present case, because test clock is already synthesized up to all the clock branches in the circuit, so each functional clock branch can simply be synthesized with Astro CTS to get a fully balanced functional clock tree.

5. Timing analysis methodology

The timing analysis is critical for the verification of the clock tree synthesis. If the clock balancing process failed, it would end up producing setup/hold violations in timing paths between clock branches driven by the same clock source. In the case of the MA1111A13 circuit, the timing analysis relies on Synopsys Primetime Static Timing Analysis (STA). Different scripts have been developed to

test the circuit timing in different modes (functional, scan-test mode, and pseudo-digital test mode), setting different case analysis at the external pins that control the circuit mode. In order to integrate the analog/digital interface timing into the STA flow, a timing model for the AFE has been developed. The functional mode script has separate sections to check each MAC interface front-end timing separately, paying attention to test the tight IO constraints and the transparent pipeline. The analysis is run for fast and slow corners, using back-annotated parasitics from post-layout extraction [5].

6. Conclusions

Nowadays, the integrated circuits for broadband communications require the management of different high-speed asynchronous clock domains. Moreover, reducing power consumption demands the application of clock gating techniques that complicate the clock networks. Furthermore, Design For Testability (DFT) must be incorporated into the design flow, usually adding extra complication to the clock control. Finally, tight IO timing constraints lead to the adoption of novel strategies to deal with the synchronization of external interfaces.

The present work has described the methods applied to solve these particular problems in the practical case of a Gigabit PHY IC. The manageability of the clock distribution complexity is obtained by using a centralized clock management unit. Several particular structures have been proposed for glitch-free clock gating and multiplexing. Clock domains border signals are synchronized by means of well-known techniques. External interfaces timing constraints are met by applying the concept of the front-end clocking. The analog logic has been incorporated into the digital clock management, paying special attention to the analog to digital interface. The clock management method has been completed with the description of the physical synthesis of the clock tree during layout, and the verification of the process correctness by means of STA.

7. References

- [1] Massana Ltd., “Everest MA1110 Gigabit Ethernet Transceiver Product Brief”, February 2002. <http://www.massana.com/>.
- [2] The Institute of Electrical and Electronics Engineers, “IEEE 802.3 Standard, 2000 Edition”, October 2000.
- [3] M. Arora, P. Bhargava, S. Gupta, “Handling Multiple Clocks”, *SNUG India*, 2002, <http://www.snug-universal.org>.
- [4] David Shear, “Exorcise Metastability from your Design”, *EDN*, December 10, 1992, pp. 58-64.
- [5] J.C. Diaz, G. Parodi, “Static Timing Analysis of the Everest MA1110 Gigabit PHY transceiver”. *Proceedings XVII DCIS Conference*, 2002, pp. 513-516