IP Core Specification

Author: Renliang Gu
Gurenliang@gmail.com

Rev. [0.1]
September 24, 2009

This page has been intentionally left blank.

Revision History

Rev.	Date	Author	Description
0.1	09/24/09	Renliang Gu	First Draft

Contents

1 Introduction

10/100M Ethernet-FIFO convertor IP core is an interface in charge of data conversion between the raw MAC frame and bit stream from one to the other. It works in full duplex mode. The bit stream's data rate depends on the input ff_clk's frequency.

This convertor is used in a project to connect the GSM modulator/demodulator with PC. So the data form is conformed to the GSM standard. However, the structure of verilog code is very clear and easy to understand so that can be easily changed to be used in other situations.

The 10/100M Ethernet-FIFO convertor IP core consists of three modules:

- The Initiation module, in charge of the power on reset (InitModule.v)
- The Transmit module (TxModule.v)
- The Receive module (RxModule.v)

The Ethernet IP Core is capable of operating at 10 or 100 Mbps for Ethernet and Fast Ethernet applications. An external PHY is needed for the complete Ethernet solution. (The PHY used to test this IP core is DM9161. The circuit schematic is attached to the appendix.)

2

Architecture

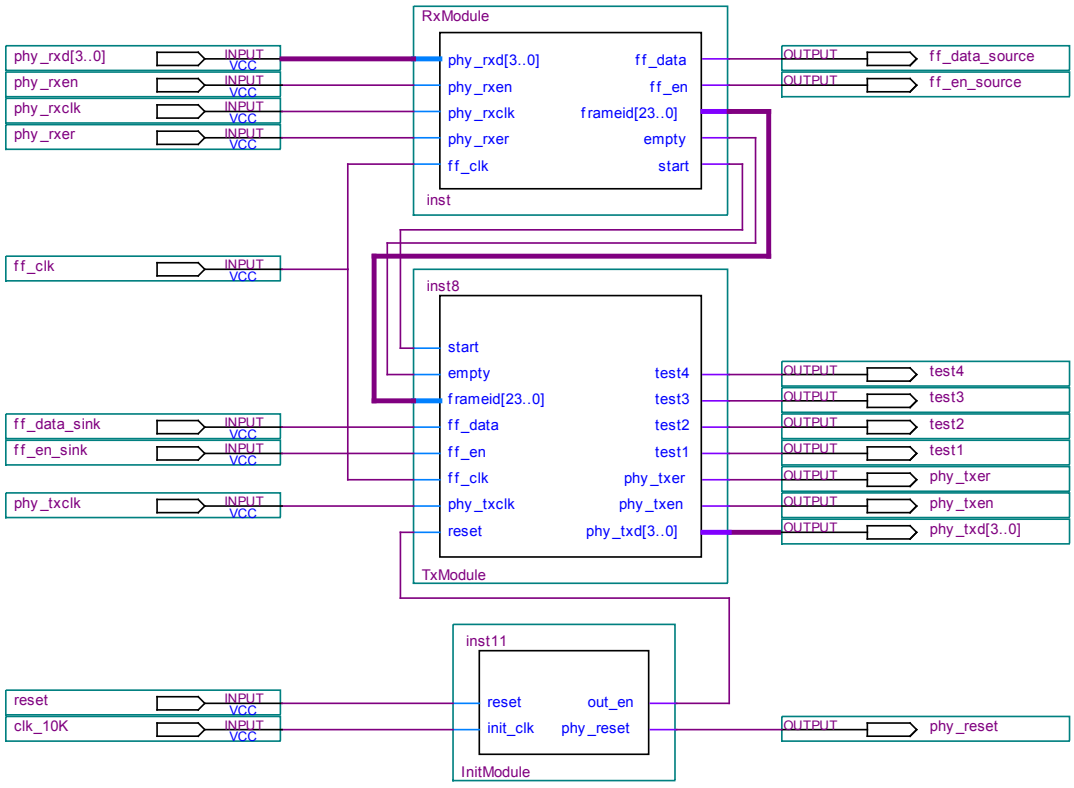

Figure 1 the Top Level Architecture

The main modules in this IP core are InitModule, RxModule and TxModule. Here, receiving and transmitting are said relatively to the Ethernet Interface. After powering up, InitMoudle would generate a reset signal to reset the PHY. And then the other two modules begin to work. The frames received from Ethernet Interface are passed to ff_data_source, while bits collected on the ff_data_sink are passed to Ethernet Interface in form of MAC frames.

Because of the common FIFO clock, the bits rates in and out of the Ethernet-FIFO convertor are the same. Thus, during every period for receiving a MAC frame, there would be two MAC frames sent out from TxModule. The signal start guarantees that the beginnings of first bit for both frames in and out are synchronized.

The following contents will introduce individual modules one by one in detail.

InitModule

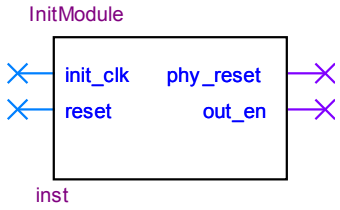

Figure 2 Block Symbol of InitModule

In some situation, the PHY doesn't work normally after powering up or the system needs to be reset for some other reasons. So we have to reset the PHY first and then make TxModule and RxModule work. And the reset signal for PHY has to hold for a certain period according to *the application note of DM9161*. [2] As the timing requirement shown in Figure 3 and Table 1, phy_reset has to keep low for at least 10mS. Thus make init_clk as a 10KHz clock would generate a qualified reset signal. And then in another 40nS, set out_en to enable TxModule.

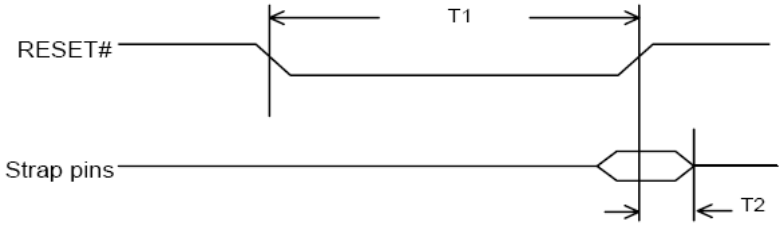

Figure 3 Waveform Shown the Timing Requirement to Reset DM9161

Table 1 Timing Requirement to Reset DM9161

Symbol	Parameter	Min.	Typ.	Max.	Unit	Conditions
T1	RESET# Low Period	10	-	-	ms	-
T2	Strap pin hold time with RESET#	40	-	-	ns	-

RxModule

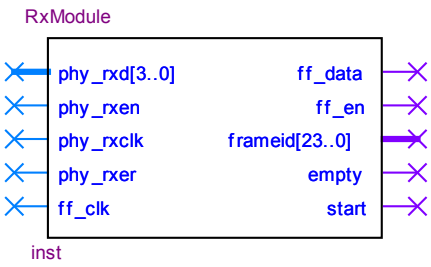

Figure 4 Block Symbol of RxModule

RxModule implements all of the functions of data receiving. Every period, during which the signal phy_rxen is high, means a frame is received. FPGA can get every MAC frame which is transmitting on the Ethernet, including the preamble, SFD, addresses and CRC. In this application, we don't check whether the value of CRC is right, but store the data directly into the FIFO.

The length of data received is 148 bytes and the data are divided into 8 groups, each 148 bits, because of GSM system specification. What's more, after every 148-bit of serial data, there must be an 8.25-bit's long gap. And the fraction 8.25 is an average value measured in four 148-bit periods which described above. The first three gaps last as long as 8 bits and the last gap lasts 9 bits. So averagely, there is an 8.25-bit's long gap after every 148-bit's data.

To get the optimal delay, which must be as short as possible but longer than the response time of PC, the size of inside buffer is 16*148 bits. Every time PC receives the frame from FPGA with the value 0x0800 in length/type field, it sends out a frame in which length of data must be 148 bytes. So one frame only fill half of the buffer, and the other half is right used to send out bit stream. The alternatively using of one half of the buffer can raise the transfer efficiency greatly.

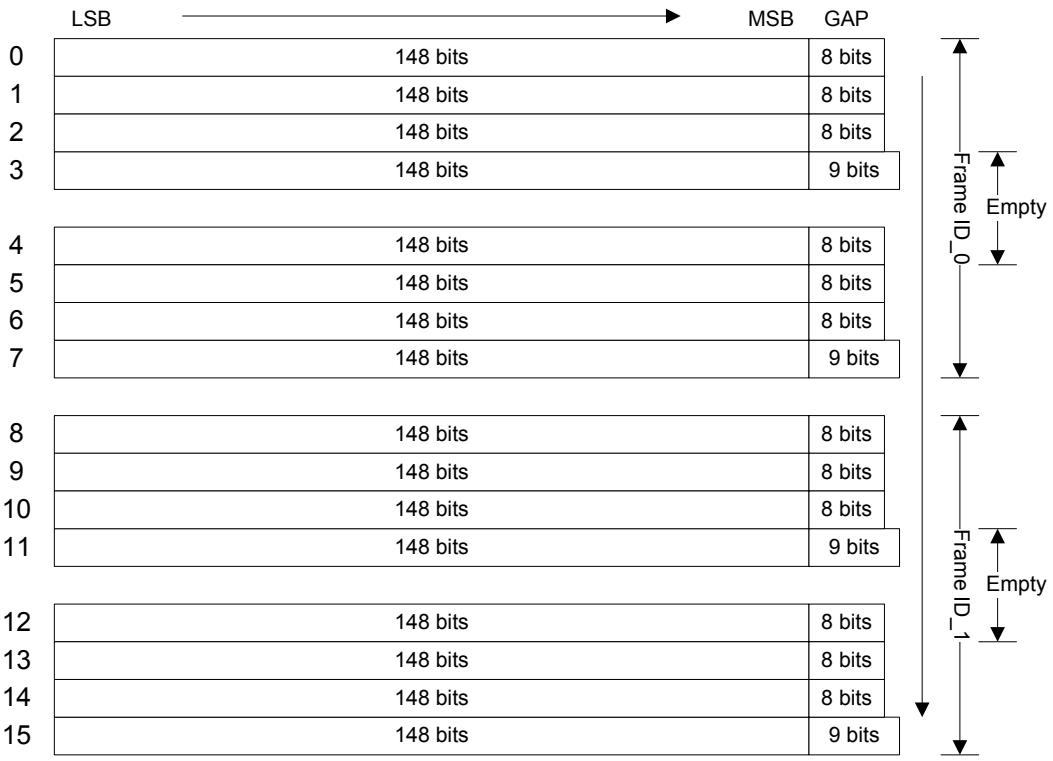

Figure 5 Internal Register Structure & How it works

As shown in Figure 5, inside of the RxModule, the 148-bit buffers are numbered from 0 to 15. If the first MAC frame received fills the buffers from 0 to 7, the next frame would fill the remained 8 buffers. So how to guarantee there is always data in buffer for FIFO to send out serial bit stream? When the buffers from 0 to 3 are empty, TxModule also collects enough bits to send out a MAC frame. And in this frame, the length/type field's value is set to 0x0800 to indicate that PC should send another 8 148-bit data. And in the next 4 148-bit's time, PC should respond the request and send out a MAC frame as quickly as possible. Similarly, while sending the other half part of buffer, RxModule informs TxModule to send a frame request MAC frame to PC when buffers from 8 to 11 are empty.

Only after RxModule receives the first frame, the signal *start* would be set to inform TxModule to sample on ff_data_sink. This mechanism makes the beginnings of MAC frames, both in and out, snapped to the same time grid.

TxModule

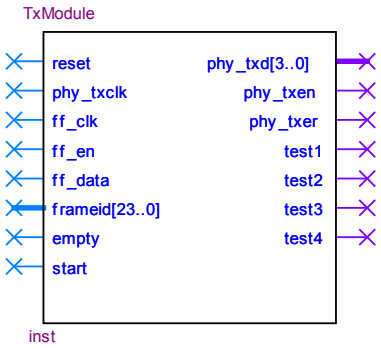

Figure 6 Block Symbol of TxModule

To transmit data through Ethernet, the frame sent to PHY must conform to the MAC frame structure. So before the data being transmitted, the TxModule adds the preamble, SFD, destination address, source address and length/type field. To be synchronized with the current data received from PC, another 24-bit frame ID is added to the head of Data Field. What's more, following the last bit of data, TxModule adds the 4-byte frame check sequence, know as CRC. And the value of 4-byte CRC only depends on the data in the period from the first bit of destination address until the last bit of data.

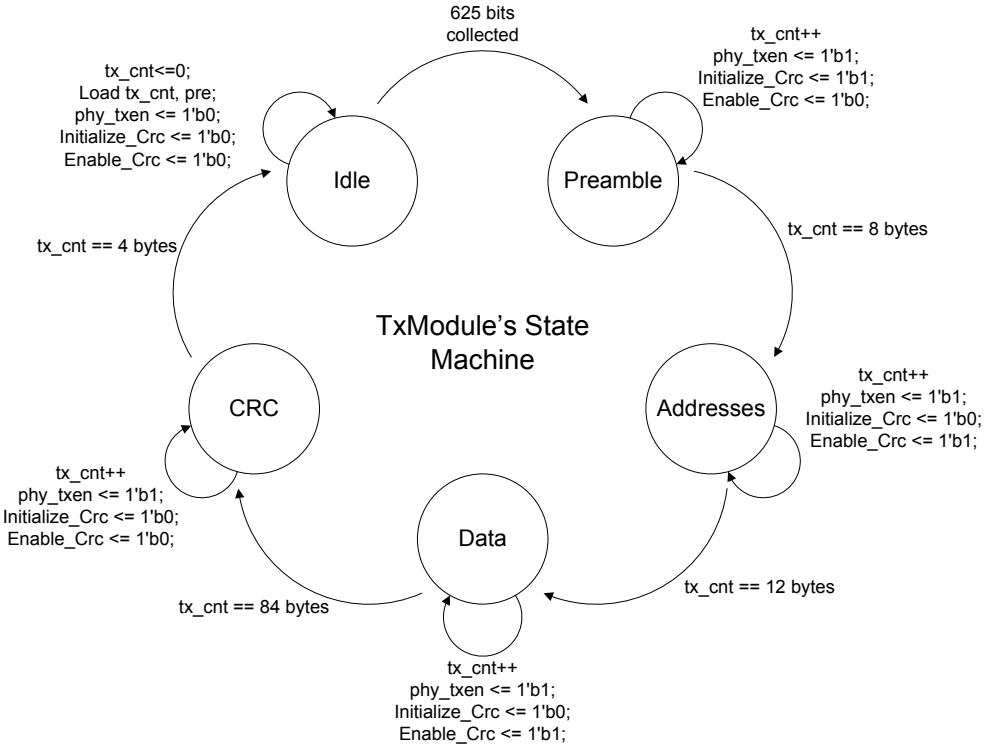

Figure 7 the State Machine of TxModule

In order to receive data from FIFO, latch the data from ff_data_sink at the rising edge of ff_clk when ff_en_sink is high. The bit stream is pushed into a FIFO, whose length is 4*156.25 bits, 625 bits. As long as the FIFO is full, there is a MAC frame transmitted out to the PHY. As we know, 625 is not integral multiple of 8, while the data transmitted on the Ethernet is measured in bytes. So another 7 bits of zero are appended to the 625-bit data. By doing so, we get a MAC Client Data of 79 bytes. With another 12-byte MAC address, 2-byte length/type field and 3-byte frame ID, the length of MAC frame the terminal on the other end of Ethernet received is 96.

When the raw MAC frame has been completed, TxModule sets the phy_txen and puts the data on the wire phy_txd[3..0] nibble by nibble. Because PHY latch the phy_d[3..0] at the rising edge of phy_txclk, TxModule changes the values on phy_txd[3..0] at the falling edge of phy_txclk.

3

Operation

When the system powers up, the PHY can not work normally because of the uncertain level of some pins. So we need the InitModule to reconfigure the PHY by pulling down the signal of phy_reset. And at the same time, pins such as phy_rxen, phy_txclk, phy_col, phy_linksts and phy_crs are configured with proper value. This initiate process should also conform to the timing requirement of the PHY. For more information, please refer to *DM1961A/DM9161E Power on reset Application Notes*. This function is implemented in the module of InitModule.v

The operation of the convertor including two main part, one is MAC frame transmitting and the other is frame receiving. Data transmitting means that bits received from the FIFO are put into the MAC frame according to the standards of IEEE802.3 and then send out through MII interface. Data receiving is totally a reverse process. The following passages would introduce the detail of them one by one.

After the configuration of the FPGA, the operation should follow the below steps:

- Send a MAC frame to the board to start the convertor. The length should be as long as 165 bytes with preamble, SFD and CRC excluded. The destination address should be 0x000000_000001.
- Keep listening to the Ethernet. Whenever there is a frame received and the value of its length/type field is 0x0800, send out the next 165-byte frame as soon as possible. The upper bound of the response time is 625/ff_clk.

By maintaining the second step listed above, the data can be transmitted in a full duplex mode.

To better understand the operation of the IP core, the reader must grasp the structure of MAC frame specified by IEEE802.3.

Media Access Control Frame Structure

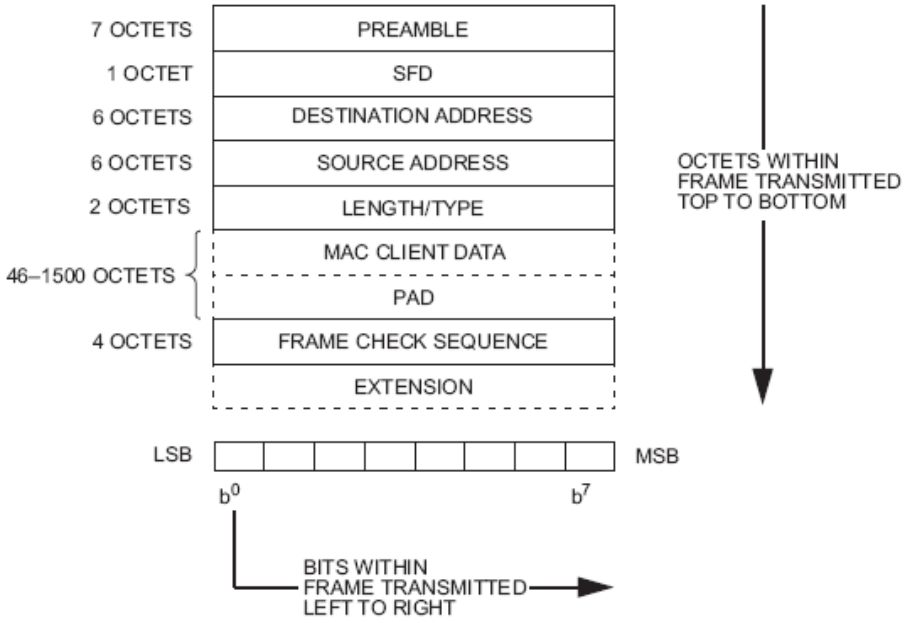

Figure 8 MAC frame format

Figure 8 shows the nine fields of a frame: the preamble, Start Frame Delimiter (SFD), the addresses of the frame's source and destination, a length or type field to indicate the length or protocol type of the following field that contains the MAC Client data, a field that contains padding if required, the frame check sequence field containing a cyclic redundancy check value to detect errors in a received frame, and an extension field if required (for 1000 Mb/s half duplex operation only).

In this IP core, the lengths of most fields are fixed except for those of the data and pad fields. Moreover extension fields are not used. [1]

The minimum and maximum frame size limits are respectively 72 bytes and 1526 bytes, inclusive. On Ethernet, the octets of a frame are transmitted from top to bottom, and the bits of each octet are transmitted from left to right, that is from the least significant bit to the most significant bit.

Figure 9 is the MAC frame structure in registers of TxModule before they are shifted out.

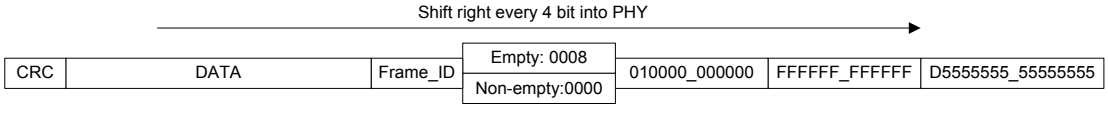

Figure 9 Frame Structure Sent out from TxModule

Preamble field

The preamble field is a 7-octet field that is used to allow the PLS circuitry to reach its steady-state synchronization with the received frame's timing. The content of the field is 0x55 iterated for 7 times.

Start Frame Delimiter (SFD) field

The SFD field is the sequence 10101011. It immediately follows the preamble pattern and indicates the start of a frame. Because of the different order of the bits between in Ethernet and in Computer, the byte seen from PC is 0xD5.

Address fields

Each MAC frame shall contain two address fields: the Destination Address field and the Source Address field, in that order. The Destination Address field shall specify the destination addressee for which the frame is intended. The Source Address field shall identify the station from which the frame was initiated.

The representation of each address field shall be as follows:

1. Each address field shall be 48 bits in length.
2. Each octet of each address field shall be transmitted least significant bit first.

In the 10/100M Ethernet-FIFO convertor, the MAC address of this convertor is assigned to be 0x000000_000001 by defining the macro variable MAC_ADD. All the frames sent by the convertor are broadcast frame, whose destination address is assigned to be 0xFFFFFF_FFFFFF. On the other hand, the convertor would check the received MAC frames' destination addresses and their length. The MAC frame with wrong Destination Address and wrong length would be ignored.

Length/Type field

This two-octet field takes one of two meanings, depending on its numeric value. In this IP core, the 10/100M Ethernet-FIFO convertor does not care the value of this field when it receives frames. However, when sends MAC frame out, there are two kinds of value in the length/type field. The frames with a value of 0x0800 can not only transmit data but request the other end of Ethernet for data. Otherwise, MAC frames sent out are only for data transfer.

Data and PAD fields

Among the frames received, the valid data length is required to be 151 bytes exactly. The first 3 bytes are frame ID and the following 148 bytes are the data that would be sent to FIFO and get out of the convertor as a form of bit stream with least significant bit first every byte. The frame ID is sent back to computer in the MAC frame for synchronization

consideration. The input and output bit streams at the same time have a same frame ID. Following the specification of GSM, the 148 bytes are divided into 8 groups, each with 148 bits. When sent out, averagely 8.25 bits are inserted between every 148 bits.

On the contrary, convertor collects bits from FIFO. Whenever the number of bits gets 625, that is 4 times (148+8.25), the convertor makes up a frame with the bits received and puts the frame ID to the head of the data field. The frame ID is sampled when the first bit gets into FIFO. In the tail of data field, TxModule appends another 7 bits' pad to round the 625 bits' data up to 79 bytes.

Frame Check Sequence (FCS) field

A cyclic redundancy check (CRC) is used by the transmit and receive algorithms to generate a CRC value for the FCS field. The frame check sequence (FCS) field contains a 4-octet (32-bit) cyclic redundancy check (CRC) value. This value is computed as a function of the contents of the source address, destination address, length, data and pad (that is, all fields except the preamble, SFD, FCS, and extension). The encoding is defined by the following generating polynomial:

$$G(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$$

4

IO Ports

The 10/100M Ethernet-FIFO convertor uses four types of signals:

- Reset and clock ports
- FIFO interface with both input and output ports
- MII ports interface with PHY (DM9161), MII serial management ports excluded.
- Test ports

Reset and Clock Ports

The table below contains the reset port (active high) used to reset the whole convertor module and the clock should be exactly 10 KHz to conform to the time request of the PHY chip when it is power-on reset.

Table 2 Reset & Clock Ports

Port	Width	Direction	Description
reset	1	Input	Reset the whole IP core, active high
clk_10K	1	Input	Clock input, 10KHz, mainly used for timing

FIFO Ports

The table below is a list of ports that is use to transmit and receive serial data. The serial data is valid only when the corresponding enable signal is high.

Table 3 FIFO Ports

Port	Width	Direction	Description
ff_clk	1	Input	FIFO common clock for both input and output
ff_en_source	1	Output	Data valid signal, active high
ff_en_sink	1	Input	Data valid signal, active high
ff_data_source	1	Output	Change on the falling edge of ff_clk
ff_data_sink	1	Input	Sample on the rising edge of ff_clk

Note: data are snapped to the time grids of ff_clk

MII Interface Ports

The following list of ports does not contain the MII Serial Management Ports, such as MDIO and MDC. By doing this, the project can be as simple as possible. And because of the power up/reset latch feature of the PHY, the tasks, which should have been done by the Serial Management Interface, can be complete by set the level of specific pins of PHY when power up or reset. So, some of the following ports are declared as bidirectional ports to configure the PHY when power up/reset.

Table 4 MII Interface Ports

Port	Width	Direction	Description
phy_rxd	4	Input	Get data from PHY in unit of nibble
phy_rxen	1	IO	Active high, when set, the data is valid
phy_rxclk	1	Input	25MHz in 100Mbps mode and 2.5M in 10Mbps mode.
phy_rxer	1	Input	When high, means error occurred
phy_txd	4	Output	Send data to PHY every time 4 bits
phy_txen	1	Output	Validate the data on phy_txen
phy_txclk	1	IO	25MHz in 100Mbps mode and 2.5M in 10Mbps mode
phy_txer	1	Output	When high, means error occurred
phy_reset	1	Output	Reset the PHY
phy_col	1	IO	
phy_linksts	1	IO	
phy_crs	1	IO	

Test Ports

The ports are remained for test. By observing it from the oscillograph, we can see what the state of TxModule is in. They can be removed from the project or remained unconnected before submitting the final version.

Appendix A

Suggested Circuit

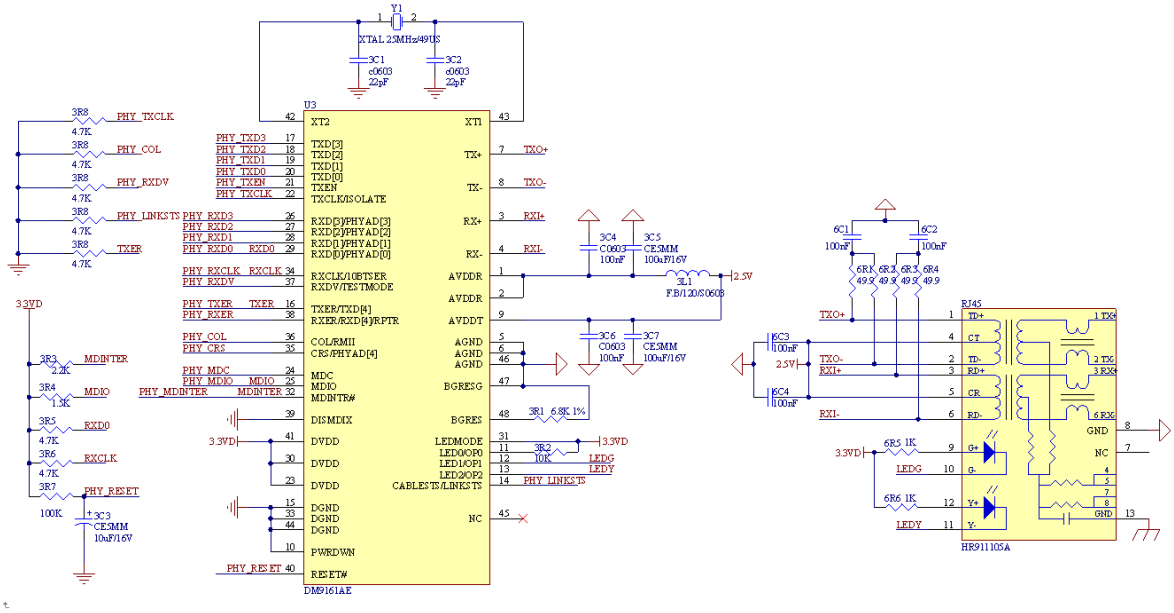

Figure 10 Reference Circuit Schematic for the Portion of Ethernet