



DAVICOM Semiconductor, Inc.

DM9010

10/100 Mbps Single Chip Ethernet Controller
with General Processor Interface

Application Notes

Preliminary
Version: DM9010-AP-P01
Apr. 19, 2005

1 INTRODUCTION.....	6
1.1 General Description.....	6
2 GENERAL PROCESSOR BUS DESCRIPTION.....	7
2.1 ISA Bus.....	8
2.1.1 Pin Function for ISA Bus	8
2.1.2 I/O Base Address Decoding	9
2.1.3 ISA Bus 8/ 16-Bit Mode and MII Interface Setting.....	9
2.1.4 Command Type for ISA Bus.....	10
2.2 Typical Signal Connection with Processor Bus.....	11
2.2.1 Pin Function for Processor Bus	11
2.2.2 I/O Base Address Decoding Example.....	12
2.2.3 Processor Parallel Buse 8/ 16/ 32-Bit Mode and MII Interface Setting	13
2.2.4 Command Type for Processor Bus	14
3 SYSTEM HARDWARE DESIGN.....	15
3.1 How to Select Chip.....	15
3.2 HOWTO Change I/O Base Address	15
3.2.1 Strap Pins Setting	16
3.3 Serial EEPROM Operation.....	17
3.3.1 EEPROM Format.....	18
3.4 GPIO Pin Setting	19
3.4.1 GPIO 23 Pins Settings	21
3.5 Timing Analysis	22
3.5.1 Read Cycle	22
3.5.2 Write Cycle.....	23
3.6 Schematic Reference Design.....	24
4 RESET OPERATION AND PHY POWER_DOWN MODE.....	25
4.1 Hardware Reset.....	25
4.1.1 Power-On Reset	25
4.1.2 Processor Reset.....	25
4.2 Software Reset.....	25
4.3 PHY Power-Down Mode.....	26
4.3.1 MII Register Setting	26
4.3.2 GEPIO0 Setting.....	26

5 HOW TO PROGRAM DM9010.....	27
5.1 How to Read/ Write DM9010 Register.....	27
5.2 Driver Initializing Steps.....	28
5.3 How to Read/ Write EEPROM Data.....	29
5.3.1 HOWTO Read EEPROM Data.....	29
5.3.2 HOWTO Write EEPROM Data.....	30
5.4 How to Read/ Write PHY Register.....	32
5.4.1 HOWTO Read PHY Register.....	32
5.4.2 HOWTO Write PHY Register.....	33
5.5 How to Transmit Packets.....	34
5.5.1 Packet Transmission.....	34
5.5.2 Check a Completion Flag.....	35
5.6 Transmit Second Packet.....	36
5.7 Early Transmit.....	37
5.8 How to Receive Packets.....	39
5.8.1 Receive Interrupt Service Routine.....	39
5.8.2 Packet Reception.....	40
5.8.3 Check the Packet Status and Length.....	41
5.8.4 Receive the Packet's Data.....	41
5.8.5 RX Perfect-Filter with Hash Table.....	42
6 THE OTHERS.....	43
6.1 Priority of the EEPROM, the Strap Pins, and the Default Setting.....	43
6.2 Identify How Many DM9010 Chips.....	43
6.3 How to Transmit and Receive more than 2048-Byte Packets.....	44
6.4 Performance of the DM9010 LAN Chip.....	44
6.5 WOL (Wakeup on LAN).....	44
(1) Magic Packet.....	44
(2) Link Change.....	45
(3) Sample Frame.....	45
6.6 IP/TCP/UDP Checksums Offload.....	47
6.7 Auto-MDIX and Application Circuit.....	48

FIGURE 1.1 DM9010 INTERNAL BLOCK DIAGRAM	6
FIGURE 2.1 SIGNAL CONNECTION WITH A PROCESSOR INTERFACING	7
FIGURE 2.2 DM9010 CMD PIN AND ISA BUS	10
FIGURE 2.3 DM9010 CMD PIN AND PROCESSOR PARALLEL BUS	14
FIGURE 3.1 PROCESSOR REGISTER READ CYCLE	22
FIGURE 3.2 PROCESSOR REGISTER WRITE CYCLE.....	23
FIGURE 3.3 SCHEMATIC OF DEMO BOARD FOR 8/ 16/ 32-BITS	24
FIGURE 5.1 PACKET TRANSMITTING BUFFER.....	34
FIGURE 5.2 BLOCK DIAGRAM OF THE RECEIVED PACKETS	39
FIGURE 6.1 AUTO-MDIX 10BASE-T/100BASE-TX APPLICATION.....	48



TABLE 2.1 PIN FUNCTION TABLE FOR ISA BUS..... 8

TABLE 2.2 I/O BASED ADDRESS FOR ISA BUS9

TABLE 2.3 PIN FUNCTION TABLE FOR PARALLEL INTERFACE 11

TABLE 2.4 I/O BASED ADDRESS FOR PARALLEL INTERFACE.....12

TABLE 3.1 STRAP PIN CONTROL 16

TABLE 3.2 EEPROM FORMAT19

TABLE 3.3 GENERAL PURPOSE CONTROL REGISTER (GPCR)20

TABLE 3.4 GENERAL PURPOSE REGISTER (GPR).....20

TABLE 3.5 GENERAL PURPOSE (CONTROL) REGISTERS AND GPIO PINS.....21

TABLE 3.6 PARAMETERS FOR READ CYCLE22

TABLE 3.7 PARAMETERS FOR WRITE CYCLE.....27

1 Introduction

1.1 General Description

The DM9010 is a fully integrated and cost-effective Fast Ethernet MAC controller with a general processor interface, an EEPROM interface, a 10/100M PHY and 4K-dword SRAM (13K-byte for RX FIFO and 3K-byte for TX FIFO). It's designed with low power and high performance process that supports 3.3V with 5V tolerant I/O. Besides, the DM9010 supports 8/ 16/ 32-bit processor interface to internal memory accesses for many different processors. DM9010 is good integrated 10/100M transceiver with Auto-MDIX & IP/TCP/UDP-checksums.

The goal of this document is for the embedded design engineers, to implement the DM9010 LAN chip on any processor's architecture quickly and successfully, with providing the exact reference information and pertaining to many embedded systems. The software programming is very simple, so users can port the software drivers to any system easily.

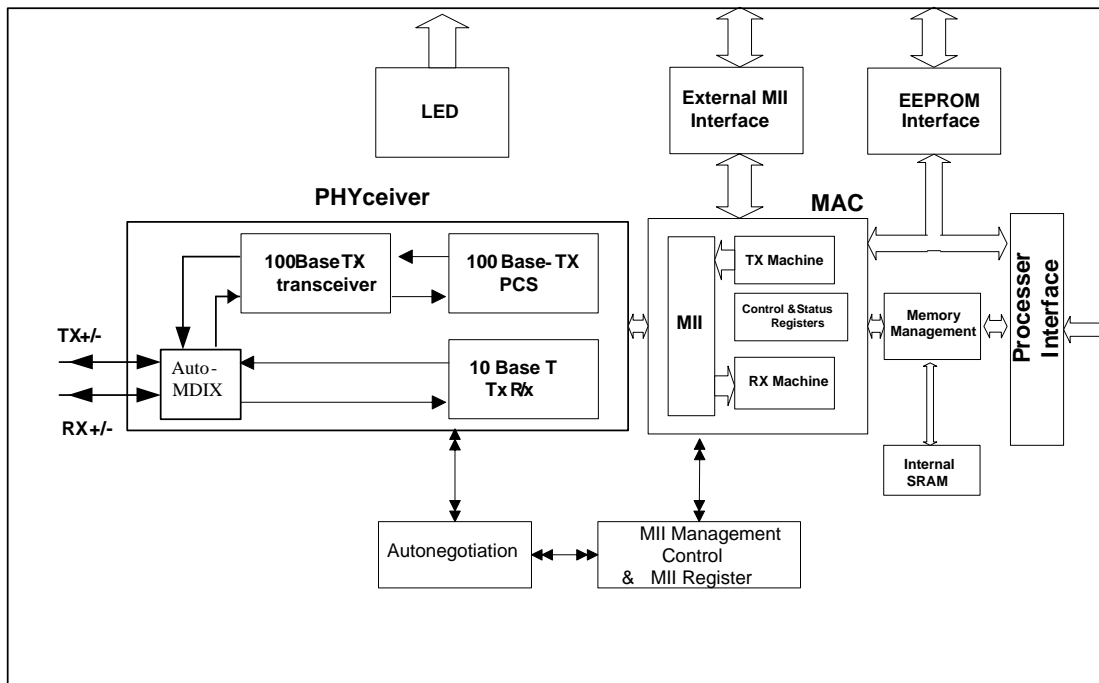


Figure 1.1 DM9010 Internal Block Diagram.

2 General Processor Bus Description

This chapter is intended to aid design engineers connecting the DM9010 device to a micro-processor or micro-controller. The discussion will include the pin functional table, and the individual control signals of the DM9010 involved in the connection between the devices and an associated micro-processor/ micro-controller in detail.

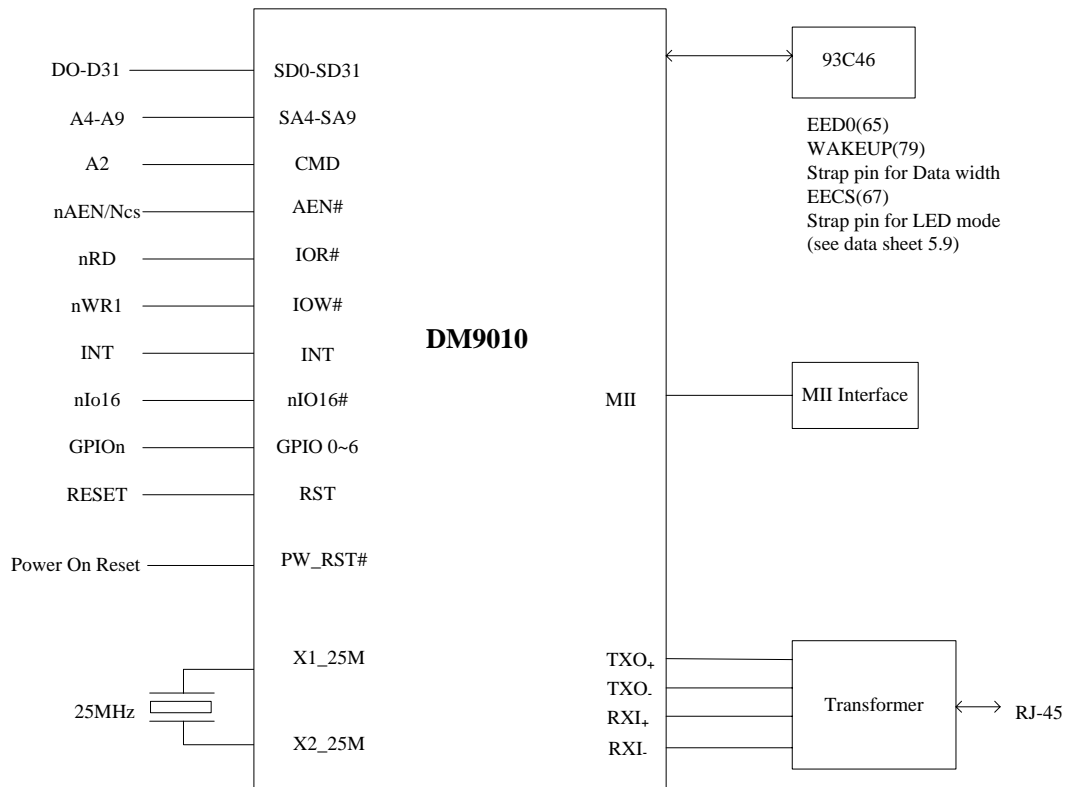


Figure 2.1 Signal Connection with a Processor Interfacing.

2.1 ISA Bus

The DM9010 supports an asynchronous bus interface. The industry standard ISA bus is one of the typical asynchronous buses.

2.1.1 Pin Function for ISA Bus

ISA Bus Signal	DM9010 Signal	Pin No.	I/O	Description
IORC#	IOR#	1	I	Processor Read Command The default is low active. The polarity can be changed by the EEPROM 93C46/ LC46 setting.
IOWC#	IOW#	2	I	Processor Write Command The default is low active. The polarity can be modified by the EEPROM 93C46/ LC46 setting.
AEN	AEN#	3	I	Address Enable A low active signal is used to select the DM9010 (chip select).
CHRDY	IOWAIT	4	O	Processor Command Ready This ISA signal is driven low to insert the wait cycles to current host read/ write command. The polarity can be changed by the EEPROM 93C46/ LC46 setting.
RESET	RST	14	I	Hardware RESET Command When this pin is asserted high, the DM9010 performs an internal system reset.
SD0 ~ 15	SD0 ~ 15	6,7,8,9, 10,11,12, 13,89, 88,87,86, 85,84, 83,82	I/O	Processor Data Bus bits 0~15
SA4 ~ 9	SA4 ~ 9	93,94,95, 96,97,98	I	Address Bus 4~9 These pins are used to select the DM9010 I/O base address 300H ~ 370H.
SA2	CMD	92	I	Command Type When an input signal is low, the access of the command cycle is ADDRESS port: The DM9010 device address port = 300H ~ 370H + 0x0 When an input signal is high, the access of the command cycle is DATA port: The DM9010 device data port = 300H ~ 370H + 0x4 (Where CMD pin is connected to Processor SA2)
IO16#	IO16	91	O	Word Command Indication This pin is used to indicate the access of internal memory is 16-bit Word mode. The default is low active and open-collected which can be programmed by the EEPROM 93C46/ LC46 setting.
IRQn	INT	100	O	Interrupt Request (INTR) The default is high active and force output. Its polarity can be modified by setting the EEPROM 93C46/ LC46 or the strap pin MDC.

Note: The pins of ISA interface, except IO16, all have a pull-low resistor about 60K Ohm

internally.

Table 2.1 Pin Function Table for ISA Bus

2.1.2 I/O Base Address Decoding

The I/O base address for ISA bus can be programmed from 300H to 370H.

The default value of the I/O base address is *300H:

A9	A8	A7	A6	A5	A4	I/O based address
1	1	0	0	0	0	*300H
1	1	0	0	0	1	310H
1	1	0	0	1	0	320H
1	1	0	0	1	1	330H
1	1	0	1	0	0	340H
1	1	0	1	0	1	350H
1	1	0	1	1	0	360H
1	1	0	1	1	1	370H

Table 2.2 I/O Based Address for ISA Bus

The chart below shows the decoding of the I/O base address *300H:

A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0

2.1.3 ISA Bus 8/ 16-Bit Mode and MII Interface Setting

The DM9010 provides two I/O modes: 8/ 16-bit DATA width of memory commands for ISA bus to read/ write the data from/ to hardware DATA port. The decoder table is shown as follows,

WAKEUP (pin 79)	EEDO (pin 65)	I/O DATA width
0	0	16-bit
0	1	*32-bit
1	0	8-bit
1	1	Reserved

Where, "1" means pull-high with the 10K Ohm resistor, and "0" means floating (default**).

*Note: In ISA bus interface, the 32-bit I/O DATA mode is not supported, because ISA bus supports only 8-bit or 16-bit I/O DATA mode. And the system designer can use the pins SD16~31 as the external MII interface function.

**Note: the pins EECS, EECK, EEDO and WAKEUP are all have a pull-low resistor about 60K Ohm internally. When floating, the pin is default "0". The EEDO pin (65) is also used as a strap pin. It combines with another strap pin WAKEUP (79), and these two pins can be used to set the 8/ 16/ 32-bit DATA width of the internal memory access for the RX/ TX packets.

The interrupt status register ISR REG. FEH Bit [7:6] can indicate the I/O mode setting:

ISR Bit [7]	ISR Bit [6]	IOMODE
0	0	Word mode (16-bit)
0	1	Dword mode (*32-bit)
1	0	Byte mode (8-bit)
1	1	Reserved

2.1.4 Command Type for ISA Bus

The CMD pin (92) is an input pin and used to set the COMMAND type. When the CMD pin detected the input signal is low, the access of this command cycle is for ADDRESS port. When input is high, the access of this command cycle is for DATA port.

The following diagram (Figure 2.2) is an example for the CMD pin (92) connecting to ISA bus.

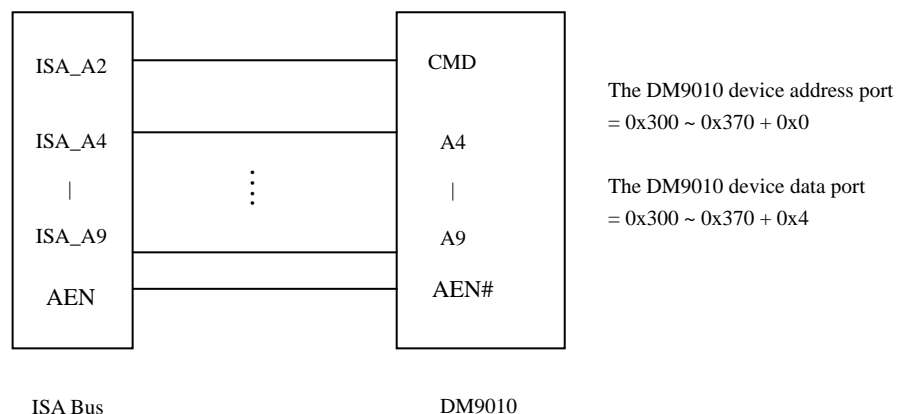


Figure 2.2 DM9010 CMD Pin and ISA Bus.

2.2 Typical Signal Connection with Processor Bus

The DM9010 can interface directly with most general processors local bus such as ARM, MIPS, Intel, TI, Motorola, NEC, and Hitachi... It's designed to suit for the implementation of Fast Ethernet connectivity solutions to many embedded systems.

2.2.1 Pin Function for Processor Bus

Processor Bus Signal	DM9010 Signal	Pin No.	I/O	Description
nRD	IOR#	1	I	Processor Read Command This pin is low active at default; its polarity can be modified by the EEPROM 93C46/ LC46 setting.
nWR	IOW#	2	I	Processor Write Command This pin is low active at default; its polarity can be changed by the EEPROM 93C46/ LC46 setting.
nAEN / nCS	AEN#	3	I	Address Enable A low active signal is used to select the DM9010 (chip select).
nIOCHRDY / nWAIT	IOWAIT	4	O	Processor Command Ready When a command is issued before last command is completed, the IOWAIT will be pulled low to indicate the current command is waited This pin is low active at default; its polarity can be modified by the EEPROM 93C46/ LC46 setting. This pin is open-collected as default, it can be programming by the EEPROM 93C46/ LC46 setting.
RESET	RST	14	I	Hardware RESET Command When this pin is asserted high, The DM9010 performs an internal system RESET.
SD0 ~ 15	SD0 ~ 15	6,7,8,9,10,11,12,13,89,88,87,86,85,84,83,82	I/O	Processor DATA Bus bits 0~15
SA4 ~ 9	SA4 ~ 9	93,94,95,96,97,98	I	Address Bus 4~9 These pins are used to select the DM9010 I/O base address 300H ~ 370H. The DM9010 device address = nCS + 300H ~ 370H These pins can be also set by hardware directly. For example, the DM9010 I/O base address is 300H: SA9 and SA8 pins are connected to VCC, and SA7, SA6, SA5, SA4 pins are connected to GND. And, the DM9010 device address = nCS + 300H.
SA2	CMD	92	I	Command Type When low, the access of this command cycle is ADDRESS port: The DM9010 device address port = nCS + 300H ~ 370H + 0x0 When high, the access of this command cycle is DATA port: The DM9010 device data port = nCS + 300H ~ 370H + 0x4 (Where CMD pin is connected to Processor SA2)
nIO16	IO16	91	O	Word Command Indication When the access of internal memory is Word or Dword width, this pin will be asserted. This pin is low active at default; its polarity can be modified by the EEPROM 93C46/ LC46 setting.

				This pin is open-collected as default; it can be programming by the EEPROM 93C46/ LC46 setting.
INT	INT	100	O	Interrupt Request (INTR) This pin is high active at default, its polarity can be modified by the EEPROM 93C46/ LC46 setting or the strap pin MDC (57). This pin is open-collected as default, it can be programming by the EEPROM 93C46/ LC46 setting or the strap pin GPIO5 (75).
SD16 ~ SD31	SD16 ~ 31 (in double word mode)	56,53,52 51,50,49 47,46,45 44,43,41 40,39,38 37	I/O	Processor DATA Bus bits 16~31 These pins are used as the DATA Bus bits 16~31. When the DM9010 is set to double word mode, the strap pin EEDO (65) is pulled high and the pin WAKEUP (79) is not pulled high.
N/A	IO32# (in double word mode)	57	O	Double Word Command Indication This pin is used as the double word command indication when the DM9010 is set to double word data mode and this pin will be asserted when the access of internal memory is double word width. This pin is low active at default.

Note: The pins of processor parallel interface, except IO16, all have a pull-low resistor about 60K Ohm internally.

Table 2.3 Pin Function Table for Parallel Interface

2.2.2 I/O Base Address Decoding Example

The I/O base address for processor bus interface can be programmed from 300H to 370H.

The default value of I/O base address is *300H:

A9	A8	A7	A6	A5	A4	I/O base address
1	1	0	0	0	0	*300H
1	1	0	0	0	1	310H
1	1	0	0	1	0	320H
1	1	0	0	1	1	330H
1	1	0	1	0	0	340H
1	1	0	1	0	1	350H
1	1	0	1	1	0	360H
1	1	0	1	1	1	370H

Table 2.4 I/O Based Address for Parallel Interface

The chart below shows the decoding of I/O based address *300H:

A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0

2.2.3 Processor Parallel Bus 8/ 16/ 32-Bit Mode and MII Interface Setting

The DM9010 provides three I/O modes: Byte/ Word/ Dword mode of memory commands for the processor parallel bus to read/ write the data from/ to hardware DATA port.

The decoder table is shown as follows:

WAKEUP (pin 79)	EEDO (pin 65)	I/O DATA width
0	0	16-bit
0	1	32-bit
1	0	8-bit
1	1	Reserved

Where, "1" means pull-high with the 10K Ohm resistor, and "0" means floating (default*).

*Note: the pins EECS, EECK, EEDO and WAKEUP are all have a pull-low resistor about 60K Ohm internally. When floating, the pin is default "0". The EEDO pin (65) is also used as a strap pin. It combines with another strap pin WAKEUP (79), and these two pins can be used to set the 8/ 16/ 32-bit DATA width of the internal memory access for the RX/ TX packets.

In the DM9010 chip, the 32-bit I/O DATA Bus and the external MII interface share the same pins SD16~31. If the designer sets the 32-bit mode, by the EEDO pin (65) pulled high and the WAKEUP pin (79) floating, the external MII interface function will be disabled automatically.

The interrupt status register (ISR REG. FEH) Bit [7:6] can indicate DATA I/O mode:

ISR Bit [7]	ISR Bit [6]	IOMODE
0	0	Word mode (16-bit)
0	1	Dword mode (32-bit)
1	0	Byte mode (8-bit)
1	1	Reserved

2.2.4 Command Type for Processor Bus

The CMD pin (92) is used to set the command type. When the CMD pin is pulled high, the access of the commands cycle is for DATA port. If pulled low, the access of the command

cycle is for ADDRESS port. The following diagram (Figure 2.3) is an example for the CMD pin (92) connected to an embedded system.

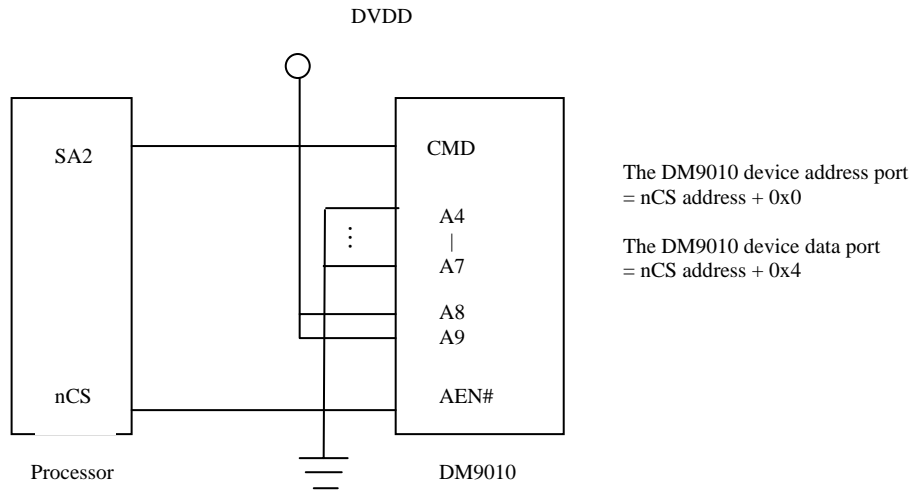


Figure 2.3 DM9010 CMD Pin and Processor Parallel Bus.

3 System Hardware Design

3.1 How to Select Chip

The AEN is an input pin. It is low active used to select the DM9010 LAN chip. The pins SA4~9 are the address bus bits 4~9. When the pins SA9, SA8 are high and the pins SA7, AEN are low, and the pins SA6~4 are matched to the strap pins TXD [2:0], the DM9010 is selected.

Both of the AEN pin and the IOR/ IOW pin should be active to read/ write the value from/ to ADDRESS port or DATA port. In the command cycle, the DM9010 chip is accessed by the AEN pin (3) cooperated with the IOR pin (1) or the IOW pin (2). These pins are all low active at default, while the pins polarity can be modified by the EEPROM 93C46/ LC46 setting to suit for the application of various processor types.

3.2 HOWTO Change I/O Base Address

The default I/O base address of the DM9010 is 300H. Once everything setup properly, the system designers can change the I/O base address to other value by writing to the EEPROM Word 6 Bit [11:9] with an available I/O base address, then write "01"b to the EEPROM Word 3 Bit [5:4] to enable the setting of Word 6 Bit [11:9]. After all these steps completed successfully, the system designer may power off and restart the device that was originally assigned to I/O base address 300H. The DM9010 will load the proper I/O base address from the serial EEPROM while powered on, power-on reset, or reload by EPCR (REG. 0BH) REEP Bit [5]= 1.

If system designers design the system without the EEPROM 93C46/ LC46, the I/O base address still can be changed. The pins TXD [2:0] are used to be the strap pins for modifying the I/O base address, $I/O \text{ base address} = (\text{strap pin value of TXD [2:0]}) * 10H + 300H$.

Besides, it's worthy of note that if the system designer designs the system with the 32-bit mode but without EEPROM, the TXD [2:0] will be used as the data port connected to the host; but according to description above, the TXD [2:0] is also the strap pins for modifying the I/O base address. Therefore, the DM9010 cannot be selected with correct I/O base address. We recommend the system designer uses the EEPROM to load on the design device instead of the strap pins TXD [2:0] for the I/O base address setting in the 32-bit mode. The I/O base

address setting by EEPROM will not be changed by the strap pins TXD [2:0]. It is because the priority of the EEPROM is higher than that of the strap pins TXD [2:0] for setting the I/O base address. The priority for setting the pins polarity is EEPROM > strap pins > default setting.

3.2.1 Strap Pins Setting

The DM9010 provides 13 strap pins for the system designs. Note: the I/O base address would not be changed by the strap pins TXD [2:0] in the 32-bit mode. And, the INT polarity also would not be modified by the strap pin MDC (57) in the 32-bit mode. Because of the pin configuration, from the pin 37 LINK_I to the pin 57 MDC originally, for the 32-bit Data bus now.

DM9010 Strap pins control list:

Pin	Name	Strap	Description															
50~52	TXD [2:0]	I/O base address 8/16-bit available	External MII Transmit Data, TXD [2:0] can modify the I/O base address. I/O base address = (strap pin value of TXD [2:0]) * 10H + 300H. Note: When SA9 and SA8 are high, SA7 and AEN are low, and SA6, SA5, SA4 are matched to the TXD [2:0] 3 pins, and then the DM9010 is selected in the processor mode.															
53	TXD [3]	Int./ Ext. MII 8/16-bit available	0: Internal MII mode, mapping to Bit [5] of REG. 2EH 1: External MII mode (not available in 32-bit mode)															
54	TXEN	EEPROM Load 8/16-bit available	This pin is External MII Transmit Enable. 0: Enable to load EEPROM after power-ON reset 1: Disable to load EEPROM after power-ON reset															
57	MDC	INT polarity 8/16-bit available	MII Serial Management Data Clock, MDC can modify INT pin 100 polarity at default active high. When the MDC pin is pulled high, the INT pin is low active; otherwise the INT pin is high active.															
65	EEDO	32-bit DATA I/O	Data to EEPROM, EEDO combines with the strap pin WAKEUP, and it can set the DATA width of the internal memory access for the RX/ TX packets. The decoder table is the following, where the logic "1" means the strap pin is pulled high. <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>WAKEUP</th> <th>EEDO</th> <th>DATA width</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>16-bit</td> </tr> <tr> <td>0</td> <td>1</td> <td>32-bit</td> </tr> <tr> <td>1</td> <td>0</td> <td>8-bit</td> </tr> <tr> <td>1</td> <td>1</td> <td>Reserved.</td> </tr> </tbody> </table>	WAKEUP	EEDO	DATA width	0	0	16-bit	0	1	32-bit	1	0	8-bit	1	1	Reserved.
WAKEUP	EEDO	DATA width																
0	0	16-bit																
0	1	32-bit																
1	0	8-bit																
1	1	Reserved.																

67	EECS	LED mode 0/ 1	This pin is Chip Select to EEPROM and is also used as a strap pin to define the LED modes. When it's pulled high, the LED mode is mode 1; otherwise it's mode 0.
74	GPIO4	PHY Power-Up	0: PHY is power-down after power-ON 1: PHY is power-up after power-ON
75	GPIO5	INT Output Type	0: INT Force-Output mode 1: INT Open-Collect mode
77	GPIO6	Auto-MDIX	0: Auto-MDIX turn OFF 1: Auto-MDIX turn ON
78	LINK_O	MII/ RMII	This pin is Cable Link Status Output and is also used as a strap pin to define the MII interface modes. Pulled high is RMII, or otherwise it's a normal MII. With a pull-low resistor about 60K Ohm internally.
79	WAKEUP	8-bit DATA I/O	WOL signal pin to issue a wake-up event happened. As a strap pin, it combines with EEDO pin 65 above.

Note: "1" means pull-high with the 10K Ohm resistor, and "0" means floating (default).

Table 3.1 DM9010 Strap Pin Control

3.3 Serial EEPROM Operation

The DM9010 supports a serial EEPROM interface. The EEPROM of the DM9010 holds the following parameters:

1. Ethernet node address.
2. Vendor ID and Product ID auto load.
3. Processor control bus pins polarity setting.
4. Wake-up mode control.
5. PHY-on or PHY-off while powered on.
6. Auto-MDIX enable/ disable setting.
7. LED1 & LED2 pins act as IO16 pin and IOWAIT/ WAKE pin in the 16-bit mode.
8. LED mode 0 or mode 1 setting.

All of the above mentioned values are read from the EEPROM upon hardware power-on reset.

For example, the specific target device is IC 93C46, the 1024-bit serial EEPROM. And the EEPROM of the DM9010 is 93C46/ LC46 such as support ATmel, Micro-chip, ATC, and CSI. All of the EEPROM accesses are done in words for the DM9010-to-EEPROM word address.

3.3.1 DM9010 EEPROM Format

Name	Word	Offset	Programming Value (Hex)	Description
MAC Address	0	0 ~ 5	XX: XX: XX: XX: XX: XX	Auto download 6-byte Ethernet node address to PAR REG10~15H
Auto Load Control	3	6 ~ 7	5455	Bit [1:0] = 00: Disable vendor ID and product ID programming. *01: Accept vendor ID and product ID programming. Bit [3:2] = 00: Disable setting of Word 6 Bit [8:0]. *01: Accept setting of Word 6 Bit [8:0]. Bit [5:4] = 00: Disable setting of Word 6 Bit [11:9]. *01: Accept setting of Word 6 Bit [11:9]. Bit [7:6] = 00: Disable setting of Word 7 Bit [3:0]. *01: Accept setting of Word 7 Bit [3:0]. Bit [9:8]: Reserved = 0. Bit [11:10] = 00: Disable setting of Word 7 Bit [7]. *01: Accept setting of Word 7 Bit [7]. Bit [13:12] = 00: Disable setting of Word 7 Bit [8]. *01: Accept setting of Word 7 Bit [8]. Bit [15:14] = 00: Disable setting of Word 7 Bit [14]. *01: Accept setting of Word 7 Bit [14]. Note: The remark * is now programming value.
Vendor ID	4	8 ~ 9	0A46	2-byte vendor ID.
Product ID	5	10~11	9010	2-byte product ID.
Pin Control	6	12~13	01E7	When Word 3 Bit [3:2] = 01, these bits can control the CS, IOR, IOW, INT, IOWAIT and IO16 pins polarity: Bit [0] = 0: CS pin is active high. *1: CS pin is active low. Bit [1] = 0: IOR pin is active high. *1: IOR pin is active low. Bit [2] = 0: IOW pin is active high. *1: IOW pin is active low. Bit [3] = *0: INT pin is active high. 1: INT pin is active low. Bit [4] = *0: INT pin is force output. 1: INT pin is force open-collected. Bit [5] = 0: IOWAIT pin is active high. *1: IOWAIT pin is active low. Bit [6] = 0: IOWAIT pin is force output. *1: IOWAIT pin is force open-collected. Bit [7] = 0: IO16 pin is active high. *1: IO16 pin is active low. Bit [8] = 0: IO16 pin is force output. *1: IO16 pin is force open-collected. When Word 3 Bit [5:4] = 01, the I/O base can be re-configured: Bit [11:9] = 000: Default I/O base address = 300H,

				*000 : 300H 001 : 310H 010 : 320H 011 : 330H 100 : 340H 101 : 350H 110 : 360H 111 : 370H Bit [15:12]: Reserved = 0.
Wake-up Mode Control	7	14~15	4180	Depend on the setting of Word 3 Bit [15:6] to accept auto load control: Bit [0] = *0: WAKEUP pin is active high. 1: WAKEUP pin is active low. Bit [1] = *0: WAKEUP pin is in level mode. 1: WAKEUP pin is in pulse mode. Bit [2] = *0: Magic packet wakeup event is disabled. 1: Magic packet wakeup event is enabled. Bit [3] = *0: Link change wakeup event is disabled. 1: Link change wakeup event is enabled. Bit [6:4]: Reserved = 0. Bit [7] = 0: LED mode 0. *1: LED mode 1. Bit [8] = 0: The internal PHY is disabled after power-on. *1: The internal PHY is enabled after power-on. The GPR REG. 1FH Bit [0] value and the GPIO0 (68) pin value are modified from this Bit [8] respectively. Bit [13:9]: Reserved = 0. Bit [14] = *1: AUTO-MDIX ON. 0: AUTO-MDIX OFF. Bit [15]: Reserved = 0.
Useable	8~63	16~127	-	Programmable by user.

Note: list 16-byte value reference, "XX:XX:XX:XX:XX:XX, 5455, 0A46, 9010, 01E7, 4180".

Table 3.2 DM9010 EEPROM Format

3.4 GPIO Pin Setting

The DM9010 provides GPIO 7 pins for the system design. The default value of GPCR Bit [0] is "1" (GPC0 = 1) for GPIO0 as the output only, and outputting a high signal "1" will power down the internal PHY by setting GPCR Bit[0] = 1 (GEPIO0 = 1). Set GEPIO0 = 0 to turn it on. The default values of GPCR Bit [3:1] are all "0"s for GPIO3~1 as the input pins respectively. The two registers, GPCR REG. 1EH and GPR REG. 1FH, are used to program high/ low values of the output/ input ports for the GPIO0 ~ GPIO6 pins at pin 68, 69, 70, 71, 74, 75, 77.

GPCR (REG. 1EH) GPIO interface control:

Bit	Name	Default	Description
7	RESERVED	0,RO	Reserved
6:4	GPC6~4	1,RO	All forced to "1"s, respectively GPIO6~4 pins 77, 75, 74 are output only.
3:1	GPC3~1	0,RW	Setting the input/ output port of the pins 71, 70, 69 GPIO3~1 respectively. "1" is represented the output port, and "0" is represented the input port.
0	GPC0	1,RO	Forced to "1", so the pin 68 GPIO0 is only output for the power management of the internal PHY.

Table 3.3 General Purpose Control Register (GPCR)
GPR (REG. 1FH) GPIO interface control value:

Bit	Name	Default	Description
7	RESERVED	0,RO	Reserved
6:4	GEPIO6~4	0,RW	GPR Bit [6:4] sets to "1" to enable the pin 77/ 75/ 74 GPIO6~4 respectively outputting high "1", or sets to "0" to enable this pin respectively outputting low.
3:1	GEPIO3~1	0,RW	If the GPIO pin is output port, GPR Bit [3:1] sets to "1" to enable the pin 71/ 70/ 69 GPIO3~1 respectively outputting high "1", or sets to "0" to enable this pin respectively outputting low "0". If the GPIO pin is input port, the value of GPR Bit [3:1] is "1" to represent a high signal is received. In contract, the value of GPR Bit [3:1] is "0" to represent a low signal is received.
0	GEPIO0	1,RW	Default GPR Bit [0] is "1" to enable the pin 68 GPIO0 outputting high "1" and power down the internal PHY, or sets to "0" to enable the pin 68 GPIO0 outputting low "0" and power it up. This default value can be programmed by the strap pin GPIO4 pin 74, or modified from the EEPROM 93C46 Word 7 Bit [8].

Table 3.4 General Purpose Register (GPR)

3.4.1 GPIO 23 Pins Settings

The DM9010 chip supports GPIO 7 pins, or up to GPIO 23 pins in 8/16-bit mode without MII:

The two registers, GPCR REG. 1EH and GPR REG. 1FH, are used to program high/ low values of the output/ input ports for the GPIO0 ~ GPIO6 pins at pin 68, 69, 70, 71, 74, 75, 77.

And, to list the GPIO 23 pins controlled by GPCRs and GPRs registers:

Reg.	Name	Port No.	Description
GPCR	General Purpose Control Register	REG. 1EH	GPIO1~3 pins 69, 70, 71, for Input/ Output direction, but the GPIO0 & GPIO4~6 pins 68, 74, 75, 77, are forced output only.
GPR	General Purpose Register	REG. 1FH	GPIO0~6 pins for Data bit value: low "0" or high "1" outputting if the GPIO pin respectively is output port, or low/ high signal received if the pin is input.
GPCR2	General Purpose Control Register 2	REG. 34H	SD23~16 pins 56, 53, 52, 51, 50, 49, 47, 46, for Input/ Output direction.
GPR2	General Purpose Register 2	REG. 35H	SD23~16 pins for Data bit value: low "0" or high "1" outputting if the GPIO pin respectively is output port, or low/ high signal received if the pin is input.
GPCR3	General Purpose Control Register 3	REG. 36H	SD31~24 pins 45, 44, 43, 41, 40, 39, 38, 37, for Input/ Output direction.
GPR3	General Purpose Register 3	REG. 37H	SD31~24 pins for Data bit value: low "0" or high "1" outputting if the GPIO pin respectively is output port, or low/ high signal received if the pin is input.

Table 3.5 General Purpose (Control) Registers and GPIO Pins

When the correspondent bit of General Purpose Control Register 2 is set, the value of the bit is reflected to SD23~16 pins; and GPCR3 reflected to SD31~24 pins. GPR2 & GPR3 keep SD23~16 & SD31~24 data, as the GPIO pins. Ex. to output SD24=1 by set REG. 36&37=0x1. For example, setting GPCR3 REG. 36H value 0x1 and GPR3 REG. 37H value 0x1 for outputting the GPIO pin 45 high "1" (i.e. to set REG. 36H= 1 and REG. 37H= 1 for SD24= 1).

3.5 Timing Analysis

The timing diagrams and parameter tables are shown below for the Read and Write cycle presented to the DM9010 device.

3.5.1 Read Cycle

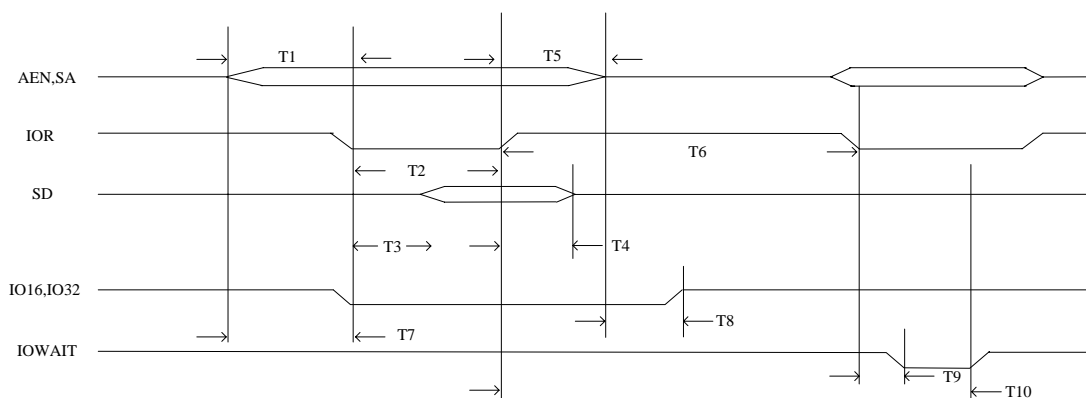


Figure 3.1 Processor Register Read Cycle

Symbol	Parameter	Min.	Typ.	Max.	Unit
T1	System address valid to IOR valid	0			ns
T2	IOR width	10			ns
T3	SD valid after IOR ready			3	ns
T4	IOR invalid to SD invalid			3	ns
T5	IOR invalid before system address invalid	0			ns
T6	IOR invalid to next IOR valid (read DM9010 register)	2			clk*
T6	IOR invalid to next IOR valid (read DM9010 memory, F0)	4			clk*
T2+T6	IOR invalid to next IOR valid (read DM9010 memory, F2)	1			clk*
T7	System address valid to IO16, IO32 valid			3	ns
T8	System address invalid to IO16, IO32 invalid			3	ns
T9	IOWAIT asserted if T6 < IOR valid clock		5		ns
T10	Last IOR de-asserted to IOWAIT de-asserted			T6	clk*

Note: the default clock period is 20 ns, clk*.

Table 3.6 Parameters for Read Cycle

3.5.2 Write Cycle

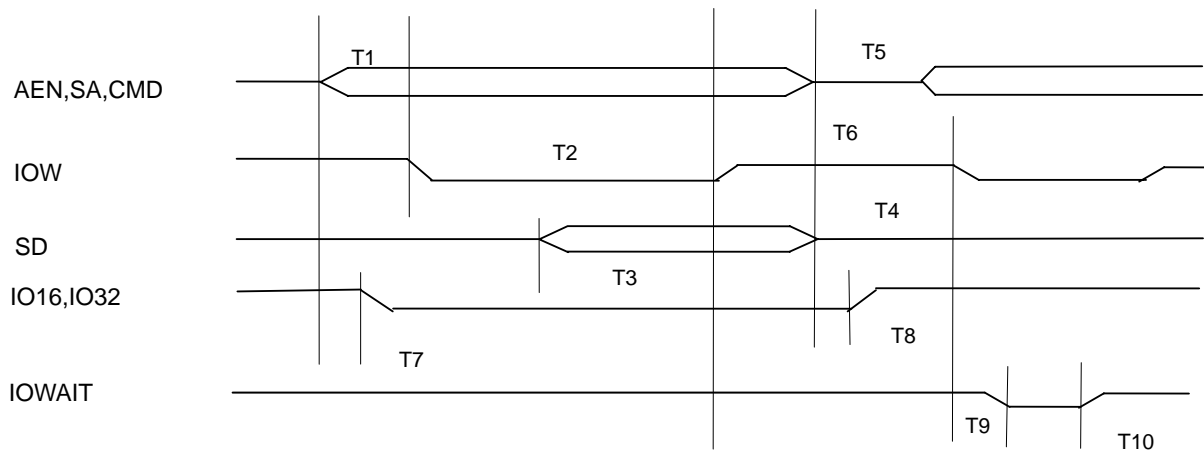


Figure 3.2 Processor Register Write Cycle

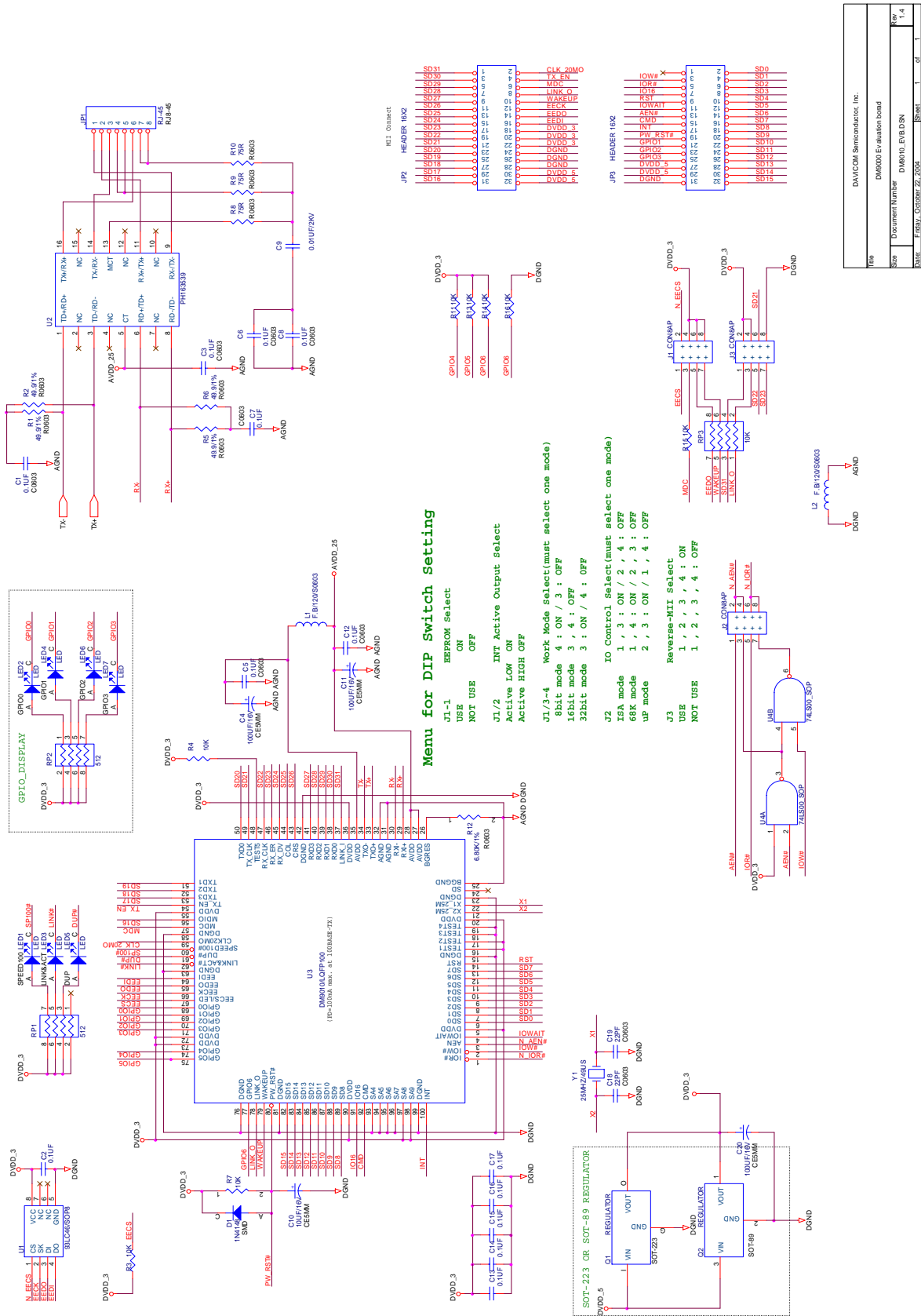
Symbol	Parameter	Min.	Typ.	Max.	Unit
T1	System address valid to IOW valid	0			ns
T2	IOW width	10			ns
T3	SD setup time	3			ns
T4	SD hold time	3			ns
T5	IOW invalid before system address invalid	0			ns
T6	IOW invalid to next IOW valid (write 9010 Address port)	1			clk*
T6	IOW invalid to next IOW valid (write DM9010 Data port)	2			clk*
T2+T6	IOW invalid to next IOW valid (write DM9010 memory)	1			clk*
T7	System address valid to IO16, IO32 valid			3	ns
T8	System address invalid to IO16, IO32 invalid			3	ns
T9	IOWAIT asserted if T6 < IOW valid clock		5		ns
T10	Last IOW de-asserted to IOWAIT de-asserted			T6	clk*

Note: the default clock period is 20 ns, clk*.

Table 3.7 Parameters for Write Cycle

3.6 Schematic Reference Design

Figure 3.3 Schematic of Demo Board for 8/ 16/ 32-Bit



Rev	DAVICOM Semiconductor, Inc.
Size	DM9010 Evaluation board
Doc Number	DM9010_EVALSN
Date	Friday, October 28, 2005
Sheet	1 of 1
Rev	1.4

4 Reset Operation and PHY Power-down Mode

The DM9010 can be reset by either hardware reset or software reset. A hardware reset can be accomplished by power-on active low or asserting high the RST pin 14. A software reset can be accomplished by setting the network control register (NCR REG. 00) RST Bit [0] = 1.

The internal PHY of the DM9010 can be powered down by writing "1" to GEPIO0 Bit [0] of the GPR register, or by setting the Power-down Bit [11] = 1 in the basic mode control register (BMCR) of the MII registers. In the power-down state, the power consumption is reduced to a minimum under 23 mA.

4.1 Hardware Reset

Both the MAC and the PHY are reset when the hardware reset operation is accomplished.

4.1.1 Power-On Reset

An active low signal is used to reset the DM9010 LAN chip. The PW_RST# pin 80 is asserted low for 20 ms at least. All of the MAC and PHY registers will be reset to the default values and the strap pins will also be latched for the hardware pins polarity. The DM9010 chip is ready after 5 us when this pin is de-asserted and then the data are downloaded from the EEPROM.

4.1.2 Processor Reset

The DM9010 can be reset by the GPIO pin of the host processor. The RST pin 14 is an input pin and asserted high for 2 us at least to reset the DM9010 LAN chip. All of the MAC registers will be reset to the default values.

4.2 Software Reset

A software reset can be accomplished by setting RST Bit [0] = 1 in the network control register (NCR REG. 00). And, this RST bit will auto clear after 10 us. The DM9010 needs only 10 us for the reset. After the reset, some registers will be reset to their default values.

4.3 PHY Power-Down Mode

In the power-down mode, the DM9010 will disable all transmit, receive and MII interface function except the MDC/ MDIO management interface. There are two ways to set the power-down mode.

4.3.1 MII Register Setting

In the MII registers, the basic mode control register (BMCR) Power-down Bit [11] can be set high "1" to enable the PHY power-down mode.

4.3.2 GPIO0 Setting

GPIO0 is used for powering down the internal PHY, and it's the pin 68 default output high. Once if the PHY is desired to be activated, the system driver needs to clear this power-down bit, GEPIO0 Bit [0] of the GPR register, by writing "0". Or, set the EEPROM Word 7 Bit [8] = 1 for power-on auto downloaded GEPIO0 = 0 to power up PHY.

5 How to Program DM9010

5.1 How to Read/ Write DM9010 Register

The DM9010 supports 8 different sets of I/O base address: 0x300/ 0x304, 0x310/ 0x314, 0x320/ 0x324, 0x330/ 0x334, 0x340/ 0x344, 0x350/ 0x354, 0x360/ 0x364 and 0x370/ 0x374. The default set of I/O base address is 0x300/ 0x304. It also can be re-configured to one of the sets for another I/O base address, by power-on downloaded the Word 6 Bit [11:9] pin control from the serial EEPROM 93C46/ LC46, or by latched from the strap pins TXD [2:0].

There are only two addressing ports through the access of the host interface. One port is ADDRESS port and the other is DATA port. The ADDRESS port is decoded by the pin CMD=0 and the DATA port is decoded by the pin CMD = 1. The content of the ADDRESS port is the address of the register to access the DATA port later. Before accessing the 8-bit data in any register or the 8/ 16/ 32-bit data in the RX/ TX FIFO SRAM (total 16K bytes), the address of the register must keep at the ADDRESS port. And, the address of the register is also named "port number". Please refer to the DM9010 datasheet chapter 9.1 about host interface.

For example, to read and write the DM9010 register:

```
UINT16 I0addr; /* UINT32 I0addr=0x19000000; for example defined in ARM-base HPI BANK3*/

UINT8 ior ( UINT16 reg )
{
    outb ( reg, I0addr ); /* I/O output a byte to ADDRESS port, select the register*/
    return inb(I0addr + 4); /*I/O input a byte from DATA port, READ the register data*/
}

void iow ( UINT16 reg, UINT8 dataB )
{
    outb ( reg, I0addr ); /* I/O output a byte to ADDRESS port, select the register*/
    outb (dataB, I0addr+4); /* I/O output a byte to DATA port, WRITE the register data*/
}
```

5.2 Driver Initializing Steps

1. Power on the internal PHY:

```
low ( 0x1F, 0x00 ); /* set GPR REG. 1FH GEPI00 Bit [0] = 0 (GPI00 = 0) */
```

The PHY is powered down at default. The power-up procedure will be needed to enable it by writing low "0" to GEPIO0. Please refer to the chapter 3.4 about setting the GPIOs settings.

2. To do a software reset for the DM9010 initial (see chapter 4.2):

```
low ( 0x00, 0x01 ); /* set NCR REG. 00 RST Bit [0] = 1, for a period time 10 us */
udelay (10); low ( 0x00, 0x00 ); /* wait 10us then clear it, or let it auto-clear */
```

3. Program the NCR register. Choose normal mode by setting NCR (REG. 00) LBK Bit [2:1] = 00b. The system designer can choose the network operations such as the internal MAC/ PHY loop-back mode, forced the external PHY Full/ Half duplex mode or to force collisions, and the wake-up events enable. Please refer to the datasheet chapter 6.1 about the NCR setting.

4. To set the IMR register (REG. FFH) Bit [7] = 1 to enable the Pointer Auto Return function, which is the memory read/ write address pointer of the RX/ TX FIFO SRAM.

5. Read the EEPROM data 3 words, for the individual Ethernet node address (if necessary).

6. Write 6-byte Ethernet node address into the Physical Address Registers (REG. 10H ~ 15H).

7. Write Hash Table 8 bytes into the Multicast Address Registers (REG. 16H ~ REG. 1DH).

8. To clear TX & INTR status by R/W1 the NSR REG. 01 and ISR REG. FEH registers. The Bit [2] TX1END, Bit [3] TX2END, and Bit [5] WAKEST will be cleared automatically by reading it or writing "1" back. Please refer to the datasheet chapter 6.2 & 6.33 about setting NSR & ISR.

9. Handle the NIC interrupts or polling service routines.

10. Program the IMR register (REG. FFH) Bit [0]/ Bit [1] to enable the RX/ TX interrupt.

11. Program the RCR register to enable RX. The RX function is enabled by setting the RXEN Bit [0] = 1 in the RX control register, RCR REG. 05. Please refer to the datasheet ch.6.6 RCR.

12. NIC is being activated and RX/ TX ready now.

5.3 How to Read/ Write EEPROM Data

The DM9010 supports the serial EEPROM interface and provides a very easy method to access it. It's only to read/ write the EEPROM&PHY control/ address/ data register (REG. 0BH ~ REG. 0EH), which are used to control and read/ write the EEPROM address or data.

For example, IC 93C46 is a 64-word (1024-bit) EEPROM and the word addresses are mapped to the EROA Bit [5:0] of the EEPROM&PHY address register (EPAR REG. 0CH). Besides, EPAR PHY_ADR Bit [7:6] must be set "00"b to enable the word address for the EEPROM. And, the setting PHY_ADR Bit [7:6] = 01 of EPAR is for the PHY address selected. Please refer to the datasheet chapter 6.12~6.14 about setting the EEPROM&PHY registers.

5.3.1 HOWTO Read EEPROM Data

The following steps are shown to read data from the serial EEPROM (SROM):

Step 1: write the word address into EPAR REG. 0CH.

Step 2: write command = 0x04 into the EEPROM&PHY Control Register (EPCR REG. 0BH) to start the SROM + READ operation:

- i. EPCR (REG. 0BH) EPOS Bit [3] = 0: select SROM mode (this is default: 0).
- ii. EPCR (REG. 0BH) ERPRR Bit [2] = 1: issue READ command.

Step 3: read EPCR REG. 0BH and wait until ERRE Bit [0] = 0 ok, or just following Step 4.

Step 4: wait 40 us at least, then write "0" into EPCR REG. 0BH to clear READ command.

Step 5: read the SROM data high byte from EE_PHY_H REG. 0EH, and the low byte from EE_PHY_L REG. 0DH in the EEPROM&PHY Data registers.

HOWTO read SROM Word 0x03, Auto Load Control, shown as follows:

1. write word address 0x03 into EPAR REG. 0CH

```
    iow ( 0x0C, 0x03 );
```

2. write the SROM + READ command = 0x04 into EPCR REG. 0BH

```
    iow ( 0x0B, 0x04 );
```

3. wait until EPCR REG. 0BH Bit [0] = 0 ok, or just following Step 4

```
    do { udelay ( 10 ); i++; } while ( ( i < 500 ) && ( inb ( I0addr + 4 ) & 1 ) );
```

4. delay 40 us at least, then write "0" into EPCR REG. 0BH

```
    udelay ( 40 );
```

```
iow ( 0x0B, 0 );
```

5. EE_PHY_H REG. 0EH for the high-byte of the SROM data, and

```
(UI NT8) e2prom[i *2+1] = i or ( 0x0E );
```

EE_PHY_L REG. 0DH for the low-byte

```
(UI NT8) e2prom[i *2] = i or( 0x0D );
```

For example, to read the MAC address from the SROM word address 0& 1& 2:

```
for ( i = 0; i < 3; i++ ) { /* read Ethernet MAC address from SROM */
    iow ( 0x0C, i );
    iow ( 0x0B, 0x04 );
    udelay ( 40 );
    iow ( 0x0B, 0 );
    /* read the SROM word data from EPDR to device address */
    (UI NT8) dev_addr[i *2+1] = i or ( 0x0E ); /* the hi gh byte of thi s 16-bi t word */
    (UI NT8) dev_addr[i *2] = i or ( 0x0D ); /* the low byte of thi s 16-bi t word */
}
```

5.3.2 HOWTO Write EEPROM Data

The following steps are to write data into the SROM:

Step 1: write the word address into EPAR REG. 0CH.

Step 2: write the data high byte into EE_PHY_H REG. 0EH, and the low byte to EE_PHY_L.

Step 3: write command= 0x12 into EPCR REG. 0BH to start the SROM + WRITE operation:

- i. EPCR REG. 0BH WEP Bit [4] = 1: enable SROM WRITE operation.
- ii. EPCR REG. 0BH EPOS Bit [3] = 0: select SROM mode (this is default: 0).
- iii. EPCR (REG. 0BH) ERPRW Bit [1] = 1: issue WRITE command.

Step 4: read EPCR REG. 0BH and wait until ERRE Bit [0] = 0 ok, or just following Step 5.

Step 5: wait 500 us at least, then, write "0" into EPCR REG. 0BH to clear WRITE command.

HOWTO write 0x1E7 into SROM Word 0x06, Pin Control, shown as follows:

1. write word address = 0x06 to EPAR REG. 0CH

```
iow ( 0x0C, 0x06 );
```

2. write the high-byte 0x01 into EE_PHY_H, and the low-byte 0xE7 into EE_PHY_L

```
iow ( 0x0E, 0x01 );
```

```
iow ( 0x0D, 0xE7 );
```

- write the SR0M + WRIT E command = 0x12 into EPCR REG. 0BH

```
iow ( 0x0B, ( 0x02 | 0x10 ) );
```

- wait until EPCR (REG. 0BH) ERRE Bit [0] = 0 ok, or just following Step 5

```
do { udelay ( 10 ); i++; } while ( ( i < 500 ) && ( inb ( IOaddr + 4 ) & 1 ) );
```

- delay 500 us at least, then write "0" into EPCR REG. 0BH

```
udelay ( 60000 );
```

```
iow ( 0x0B, 0 );
```

For example, to write Ethernet node address 00:E0:63:0E:56:78 into SR0M Word 0& 1& 2:

```
srom_wri te ( 0x00, 0xE000 );          /* Word 0: low-byte = 00, high-byte = E0 */
srom_wri te ( 0x01, 0x0E63 );          /* Word 1: low-byte = 63, high-byte = 0E */
srom_wri te ( 0x02, 0x7856 );          /* Word 2: low-byte = 56, high-byte = 78 */

void srom_wri te ( int offset, UINT16 dataW )
{
    UINT16    i, tmpv;
    /* write the SR0M word address into EPAR REG. 0CH */
    iow ( 0x0C, offset );

    /* write the high-byte to EE_PHY_H REG. 0EH, low-byte to EE_PHY_L REG. 0DH */
    iow ( 0x0D, dataW & 0xff );          /* the low-byte of this 16-bit word */
    iow ( 0x0E, (dataW >> 8) & 0xff ); /* the high-byte of this 16-bit word */

    /* issue the SR0M + WRIT E command = 0x12 into EPCR REG. 0BH */
    iow ( 0x0B, 0x02 | 0x10 );

    /* wait >500 us for SR0M + WRIT E completed then write "0" to clear command */
    do { udelay (10); tmpv= inb (IOaddr+4); i++; } while ( ( i < 500 ) && (tmpv & 1) );
    udelay ( 60000 );
    iow( 0x0B, 0 );
}
}
```

5.4 How to Read/ Write PHY Register

The DM9010 PHY supports only 32 registers, which are mapped to EPAR REG. 0CH Bit [4:0] and EPADR REG. 33H Bit [4:0] for external PHY address. The default value of EPAR (REG. 0CH) PHY_ADR Bit [7:6] is "01" to select the PHY mode. There are 4 selections for the DM9010 PHY address. Force PHY_ADR Bit [7:6]= 01 if the PHY address is selected. And, the setting PHY_ADR Bit [7:6] = 00 is for the EEPROM address selected. The start value of the PHY register address is 0x40 at offset=0x00. The most three significant Bit [4:2] of the PHY register address is forced to default "000"b. And the default address of the PHY is "00001"b. Please refer to the datasheet chapter 6.12~6.14 about setting the EEPROM&PHY registers.

5.4.1 HOWTO Read PHY Register

The following steps are shown to read data from the PHY register:

Step 1: write the PHY register offset into EPAR REG. 0CH Bit [4:0], and the PHY address "01"b into EPAR REG. 0CH Bit [7:6].

Step 2: write command = 0x0C into EPCR REG. 0BH to start the PHY + READ operation:

- i. EPCR (REG. 0BH) EPOS Bit [3] = 1: select PHY mode (0: select SROM, see ch.5.3)
- ii. EPCR (REG. 0BH) ERPRR Bit [2] = 1: issue READ command.

Step 3: read EPCR REG. 0BH and wait until ERRE Bit [0] = 0 ok, or just following Step 4.

Step 4: wait 5 us maximum, then write "0x08" into EPCR REG. 0BH to clear READ command.

Step 5: read the PHY data high byte from EE_PHY_H REG. 0EH, and the low byte from EE_PHY_L REG. 0DH in the EEPROM&PHY Data registers.

For example, to read the PHY register of BMCR at address offset = 0x00:

1. write offset = 0 into EPAR REG. 0CH Bit [4:0], and "01"b into EPAR REG. 0CH Bit[7:6]

```
iow ( 0x0C, ( 0x00 | 0x40 ) ); /* issue PHY address+ 40H (EPAR PHY_ADR = "01"b) */
```
2. write the PHY + READ command = 0x0C into EPCR REG. 0BH

```
iow ( 0x0B, ( 0x04 | 0x08 ) );
```
3. wait until EPCR REG. 0B ERRE Bit [0] = 0 ok, or just following Step 4

```
do { udelay ( 1 ); i++; } while ( ( i < 50 ) && ( inb ( IOaddr + 4 ) & 1 ) );
```
4. delay 5 us maximum, then write "0x8" into EPCR REG. 0BH to clear it and keep PHY

```
udelay ( 5 ); /* wait 1~5 us for the PHY+ READ command completion */
iow ( 0x0B, 0x08 ); /* clear this PHY "READ" command */
```

5. read the high byte from EE_PHY_H REG. 0EH, and the low byte from EE_PHY_L

```
(UINT8) phy[offset*2+1] = ior ( 0x0E ); /* the high byte of this 16-bit word */
(UINT8) phy[offset*2] = ior ( 0x0D ); /* the low byte of this 16-bit word */
```

5.4.2 HOWTO Write PHY Register

The following steps are to write data into the PHY register:

Step 1: write the PHY register offset into EPAR REG. 0CH Bit [4:0], and the PHY address Bit [1:0] = "01"b into EPAR REG. 0CH Bit [7:6].

Step 2: write the PHY high byte data to EE_PHY_H REG. 0EH, and low byte to EE_PHY_L.

Step 3: write command = 0x0A into EPCR REG. 0BH to start the PHY + WRITE operation:

- i. EPCR (REG. 0B) EPOS Bit [3] = 1: select PHY mode (0: select SROM, see ch.5.3)
- ii. EPCR (REG. 0B) ERPRW Bit [1] = 1: issue WRITE command.

Step 4: read EPCR REG. 0BH and wait until ERRE Bit [0] = 0 ok, or just following Step 5.

Step 5: wait 5 us maximum, then, write "0x8" into EPCR REG. 0BH to clear WRITE command.

For example, to write data = 0x101 into ANAR REG. 04 for the 100M Full-duplex support:

1. write offset= 4 into EPAR REG. 0CH Bit [4:0], and "01" into EPAR REG. 0CH Bit [7:6]

```
iow ( 0x0C, ( 0x04 | 0x40 ) ); /* issue PHY address+ 40H (EPAR PHY_ADR = "01"b) */
```

2. write the PHY high byte 0x01 into EE_PHY_H, and low byte 0x01 into EE_PHY_L

```
iow ( 0x0E, 0x01 );
```

```
iow ( 0x0D, 0x01 );
```

3. write the PHY + WRITE command = 0x0A into EPCR REG. 0BH

```
iow ( 0x0B, ( 0x02 | 0x08 ) );
```

4. wait until EPCR (REG. 0BH) ERRE Bit [0] = 0 ok, or just following Step 5

```
do { udelay ( 1 ); i++; } while ( ( i < 50 ) && ( inb ( IOaddr + 4 ) & 1 ) );
```

5. delay 5 us maximum, then write "8" into EPCR REG. 0BH to clear it and keep PHY

```
udelay ( 5 ); /* wait 1~5 us for the PHY+WRITE command completion */
```

```
iow ( 0x0B, 0x08 ); /* clear this PHY "WRITE" command */
```

5.5 How to Transmit Packets

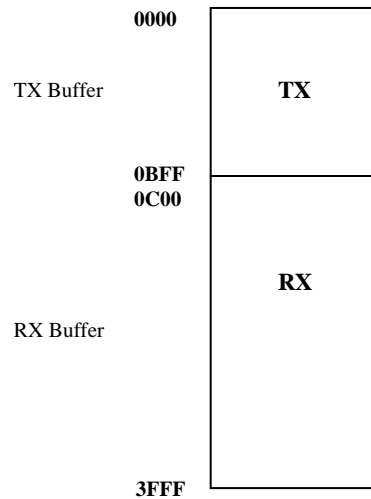


Figure 5.1 Packet Transmitting Buffer

Before transmitting a packet, the data of the packet must be kept in the TX FIFO, which is the internal SRAM address 0 ~ 0x0BFF in the DM9010 MAC and use the output port byte command to select MWCMD REG. F8H, the memory data write command with address increment register. Then, the length of the packet is defined as the high byte put in MDRAH REG. FDH and the low byte put in MDRAL REG. FCH. The final step is to set the Bit [0] TXREQ (Transmit Request bit in TCR REG. 02) for transmitting the packet automatically.

The DM9010 will generate an interrupt latched at PTS Bit [1] = 1 in ISR REG. FEH, if setting PTM Bit [1] = 1 of IMR REG. FFH, and also to set a completion flag indicated at NSR REG. 01 either TX1END Bit [2] = 1 or TX2END Bit [3] = 1 in toggle to indicate that the packet is transmitted. The DM9010 supports byte/ word/ dword memory data command to quickly write the packet's data into the TX FIFO SRAM, dependent on the system hardware application.

5.5.1 Packet Transmission

Step 1: check the memory data 8/ 16/ 32-bit DATA width.

```
(UINT8) IO_mode = ior ( 0xFE ) >> 6; /* ISR Bit[7:6] IOMODE keep I/O DATA mode */
```

Step 2: write the packet's data into TX FIFO, depended on the Byte/ Word/ Dword mode.

```
outb ( 0xF8 ); /* trigger MWCMD REG. F8H with write-pointer++ */
/* UINT8 TX_data[]: the transmitting data, int TX_length: the length of TX_data[] */
if ( IO_mode == 2 ) { /* I/O 8-bit Byte mode if ( IO_mode == DM9010_BYTE_MODE ) */
```

```

for ( i = 0; i < TX_Length; i++ ) /* Loop to write a Byte data i into TX FIFO */
    outb ( TX_data[i], IOaddr + 4 );
}

else if ( IO_mode == 0 ) { /* I/O 16-bit Word mode if(IO_mode== DM9010_WORD_MODE) */
    (int) length_tmp = ( TX_Length + 1 ) / 2;
    for ( i = 0; i < length_tmp; i++ ) /* Loop to write a Word data i into TX FIFO */
        outw ( ( (UINT16 *) TX_data)[i], IOaddr + 4 );
}

else if ( IO_mode == 1 ) { /* I/O 32-bit Dword mode if(IO_mode== DM9010_DWORD_MODE)*/
    (int) length_tmp = ( TX_Length + 3 ) / 4;
    for ( i = 0; i < length_tmp; i++ ) /* Loop to write a Dword data to TX FIFO */
        /* the outl () command is outputting a long word to the 32-bit I/O port */
        outl ( ( (UINT32 *) TX_data)[i], IOaddr + 4 );
}

```

Step 3: write the transmitting length into MDRAL REG. FCH and MDRAH REG. FDH

```

/* write the high byte of the TX data length into MDRAH REG. FDH */
iow ( 0xFD, (TX_Length >> 8) & 0xff );

/* write the low byte of the TX data length into MDRAL REG. FCH */
iow ( 0xFC, TX_Length & 0xff );

```

Step 4: start to transmit this packet

```

iow ( 0x02, 1 ); /* set the TX request command, TXREQ, Bit [0] in TCR REG. 02 */

```

5.5.2 Check a Completion Flag

If the driver is used the polling/ interrupt method, the program segment can be inserted into the TX routine for detecting a packet transmission completed or even double-check interrupt:

```

(UINT8) TX_status = ior ( 0x01 ); /* read Bit [3:2] of NSR REG. 01 for TX done */
if ( TX_status & (4 | 8) ) { /* success */ }; /* check if TX1END or TX2END=1, TX ok*/

```

The program segment shown as follows can be inserted into the interrupt handler, if the driver is used the interrupt driven and setting the IMR PTM Bit [1] = 1 enable:

```

(UINT8) INT_status = ior ( 0xFE ); /* got DM9010 interrupt status in ISR */
if ( INT_status & 0x02 ) { /* success */ }; /* check if PTS Bit [1] = 1, TX ok */
iow ( 0xFE, 0x02 ); /* clear PTS Bit [1] latched in ISR */

```

5.6 Transmit Second Packet

The DM9010 could automatically send out two TX packets with the maximum length 3072-byte, while the first packet is sending and the second is queued in the TX FIFO 3KB SRAM. Two-packet-mode is to fill two packets into the TX FIFO SRAM continuously. And, it's the default mode to latch the two packet's length in the DM9010 for TX high performance. The DM9010 MAC will auto send the second packet, when the first packet transmitted completion.

The DM9010 can issue transmitting-two-packets or two TX packets, as one packet mode, by setting TX Control Register 2 (TCR2 REG. 2DH) Bit [4] ONEPM: 1 or Two Packet Mode. When cleared at most, two-packets-transmit "command" can be issued before TX completed.

While to set-bit ONEPM Bit [4] of TCR2 REG. 2DH put in the DM9010 initial function, it's to enable One-packet-mode and to disable the TX Two-packet-mode at default.

For example, to set the TX One-packet-mode:

```
if ( db->tx_pkt_cnt > 1 )      return 1;
iow ( 0x2D, 0x10 );          /* TCR2 REG. 2DH One-packet-mode setting */

data_ptr = (char *)skb->data; /* move data into TX FIFO SRAM */
outb ( 0xF8, IOaddr );      /* trigger TX MWCMD port number 0xF8 locking */

db->sent_pkt_len = skb->len;
tmplen = (skb->len + 1) / 2; /* I/O DATA Word mode */
for ( i = 0; i < tmplen; i++ ) outw ( ( (u16 *)data_ptr )[i], IOaddr );

/* TX control: first packet immediately send, or second packet queue */
if ( db->tx_pkt_cnt <= 1 ) { /* first packet sending */
    db->tx_pkt_cnt++;

    iow ( 0xFD, (skb->len >> 8) & 0xff ); /* set TX length into MDRAH REG. FDH */
    iow ( 0xFC, skb->len & 0xff ); /* set TX length into MDRAL REG. FCH */

    /* issue TX polling command */
}
```

```

    iow ( 0x02, 0x01 );          /* auto cleared TXREQ bit after TX completed */
} else {                        /* second packet queued */
    db->tx_pkt_cnt++;
    db->queue_pkt_len = skb->len;
}

/* TX event if one packet sent completed, queued TX packet check & send */
(UINT8) TX_status = ior ( 0x01 ); /* got TX1END & TX2END status bits from NSR */
if ( ( TX_status & 0xc ) && ( db->tx_pkt_cnt > 0 ) ) {
    iow ( 0xFD, ( db->queue_pkt_len >> 8 ) & 0xff ); /* set TX length to MDRAH */
    iow ( 0xFC, db->queue_pkt_len & 0xff ); /* set TX length into MDRAL */
    iow ( 0x02, 0x01 ); /* issue TX polling command and auto cleared TXREQ */
}

```

5.7 Early Transmit

Early-transmit and Two-packet-mode (above) are mutually exclusive for the DM9010 chip.

For Early-transmit feature, it's very easy to set ETXCSR REG. 30H: enable ETE Bit [7] = 1 and setting ETT Bit [1:0] value for the Threshold percentage %. Then, to fill the TX packet length into MDRAH REG. FDH (the high byte) and MDRAL REG. FCH (the low byte), and it's unnecessary to issue the TX polling command, TXREQ Bit [0] in TCR REG. 02.

But, also don't enable the function of Transmit IP/TCP/UDP Checksums generations in TCSCR REG. 31H at the same time.

Finally, ETS1 Bit [5] and ETS2 Bit [6] are indicating Early TX status I/ II under-run or not. "1": Early TX under-run while the packet I/ II transmitting, "0": normal. And, the ISR (REG. FEH) UDRUN Bit [4] is also latched "1" for TX under-run status.

The Early TX Threshold percentage table is shown as follows,

ETXCSR Bit [1]	ETXCSR Bit [0]	Early TX Threshold %
0	0	12.5%
0	1	25%
1	0	50%
1	1	75%

Please refer to the datasheet chapter 6.26 about ETXCSR REG. 30H setting.

For example, to set the Early TX Threshold 75%:

```
#define ETXCSR 0x30
#define TCSCR 0x31

static int EarlyTxFlag = 1;

iow ( TCSCR, 0 ); /* disable TX Checksums generations */
iow ( ETXCSR, 0x80 | 0x03 ); /* ETXCSR REG. 30H Early TX Threshold 75% ("11"b) */
/* Then, ETS1/2 indicating Early TX status I/II, while packet 1/2 underrun or not */
```

```
static int dmfe_start_xmit(struct sk_buff *skb, struct net_device *dev)
{
    /* .. */
    iow ( 0xFC, skb->len & 0xFF );
    iow ( 0xFD, ( skb->len >> 8 ) & 0xFF );
    /* move TX data into the DM9010 TX SRAM */
    for ( i = 0; i < (skb->len + 1)/2; i++)
        outw ( ((u16 *)data_ptr)[i], IOaddr+4 );
    /* .. */
    /* issue TX request command, TXREQ Bit[0] in TCR REG. 02, if Early TX disable */
    if ( !EarlyTxFlag ) iow ( 0x02, 0x01 );
    /* .. */
}
```

5.8 How to Receive Packets

The received and filtered packets are kept in the RX FIFO, which is the internal SRAM address 0x0C00 ~ 0x3FFF (13K-byte) in the DM9010 MAC. There are four bytes for the MAC header of each packet put in the RX FIFO SRAM, and to command the MRCMDX REG. F0H and MRCMD REG. F2H registers for got the information of the packet incoming.

The first byte is used to check whether a packet is received and filtered into the RX FIFO. If the Bit [1:0] are "01"b, it means there is a packet received. If the Bit [1:0] are "00"b, it means there is no packet received in the RX FIFO. Before reading the other bytes, make sure the Bit [1:0] are "01"b at the first byte of the MAC header. But, if neither "01"b nor "00"b, the DM9010 MAC/ PHY must do a software-reset in order to recover from the un-stable states between the system bus and the DM9010 LAN chip. The second byte keeps the "status" information of the received packet. The format of the status "second byte" is the same as RSR REG. 06. Please refer to the datasheet chapter 6.7. According to this format, the received packet can be verified as either a correct packet or an error packet. The third and fourth bytes are the "length" of the received packet. The others bytes are the received packet's data, which is RX payload and 4-byte CRC. The following diagram is shown the format of the received package:

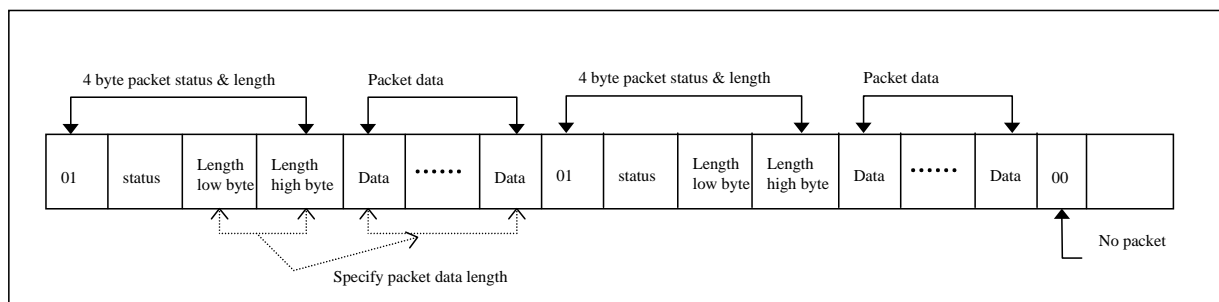


Figure 5.2 Block Diagram of the Received Packets

5.8.1 Receive Interrupt Service Routine

The following program segment can be inserted to the interrupt handler if the driver is designed as the interrupt driven or the polling method for waiting packets received ready:

```
(UINT8) INT_status = ior ( 0xFE );      /* got DM9010 interrupt status in ISR */
if ( INT_status & 0x01 ) { /* doing ch. 5. 8. 2 */ }; /* check if PRS Bit[0]=1, RX ok*/
ior ( 0xFE, 0x01 );      /* clear PRS Bit[0] latched in ISR (see chapter 5. 5. 2) */
```

5.8.2 Packet Reception

The definition of MRCMDX REG. F0H is the memory data read command without address increment register. MRCMDX is only used to read the RX ready flag for Bit [1:0] = "01"b and Bit [7:5] = "000"b for IP/TCP/UDP checksum status indicated ok, which is read from the first byte of the received packet in the RX FIFO SRAM.

Here is an example to get RX ready:

```
(UINT8) RX_ready = ior ( 0xF0 );          /* dummy READ the packet RX ready flag */
RX_ready = (UINT8) inb ( IOaddr + 4 );    /* got the most updated data */
if ( (RX_ready & 0x01)==1 ) { /* RX ready check: Bit [1:0] must be "01"b or "00"b */
/* check the RX status & length (see ch. 5.8.3) and income packets (see ch. 5.8.4) */
} else if ( ( RX_ready & 0x1f ) != 0 ) {
/* stop NIC interface and wait the system kernel timer to *reset device */
iow ( 0xFF, 0x80 );          /* stop INT request */
iow ( 0xFE, 0x0F );          /* clear ISR status */
iow ( 0x05, 0x00 );          /* stop RX function */
(UINT8) device_wait_reset = TRUE;      /* raise the MAC/PHY software-reset flag */
/* it's quick software-reset to replace the *reset device program above */
/*
iow ( 0x1F, 0x00 );          // GPR REG. 1FH GEPI00 Bit[0] = 0 to activate internal PHY
iow ( 0x00, 0x01 );          // NCR REG. 00 RST Bit[0] = 1 reset on
udelay ( 10 );              // wait >10 us
iow ( 0x00, NCR_set );      // #define NCR_set 0x00
iow ( 0x0A, 0x29 );          // RTFCR REG. 0AH enable TXPEN, BKPM (TX_Half), FLCE (RX)
iow ( 0x2D, 0x10 );          // TCR2 setting One-Packet-Mode
iow ( 0x2F, 0x00 );          // SMCR REG. 2FH special mode cleared
iow ( 0xFE, 0x3F );          // ISR REG. FEH clear all interrupt status latched
iow ( 0xFF, 0x80 );          // IMR REG. FFH only enable PAR: Pointer Auto Return
iow ( 5, RCR_set | 1 );     // #define RCR_set 0x30
*/
/* then, re-new system variables and counters for dropped and queued packets... */
}
```

5.8.3 Check the Packet Status and Length

MRCMD (REG. F2H): memory data read command with the increment of the RX read pointer. The read pointer will be increased after reading the memory read command, MRCMD REG. F2H. The size, the increment of the RX read pointer, is depended on the system application, which the I/O bus width is byte/ word/ dword to increase 1/ 2/ 4 bytes respectively. MRCMD REG. F2H is only used to read the RX status, length and the packet's data from the RX FIFO.

Here is an example to get the RX status and length:

```
(UINT8) IO_mode = ior ( 0xFE ) >> 6; /* ISR Bit[7:6] IOMODE keep I/O DATA mode*/
outb ( 0xF2, IOaddr ); /* trigger MRCMD REG. F2H with read_ptr++ */
/* int RX_status: the RX packet status, int RX_length: the length of the RX packet*/

if ( IO_mode == 2 ) { /* I/O 8-bit byte mode if ( IO_mode == DM9010_BYTE_MODE ) */
RX_status = inb ( IOaddr + 4 ) + ( inb ( IOaddr + 4 ) << 8 );
RX_length = inb ( IOaddr + 4 ) + ( inb ( IOaddr + 4 ) << 8 ); }

else if ( IO_mode == 0 ) { /* I/O 16-bit word mode if ( IO_mode == DM9010_WORD_MODE ) */
RX_status = inw ( IOaddr + 4 ); /* the high byte is the status as RSR REG. 06 */
RX_length = inw ( IOaddr + 4 ); }

else if ( IO_mode == 1 ) { /* I/O 32-bit dword mode if ( IO_mode == DM9010_DWORD_MODE ) */
(UINT32) status_tmp = inl ( IOaddr + 4 ); /* got the RX 32-bit Dword data */
RX_status = (UINT16)( status_tmp & 0xff00 ); /* got the high byte of the status */
RX_length = (UINT16)( status_tmp >> 16 ); }
```

5.8.4 Receive the Packet's Data

The read pointer will be increased after reading the memory read command, MRCMD REG. F2H. According to the length of the received packet, dump out the RX payload and the 4-byte CRC checksums.

Here is an example to get the RX packet's data:

```
/* UINT8 RX_data[]: the data of the received packet with the 4-byte CRC checksums */
if ( IO_mode == 2 ) { /* I/O 8-bit Byte mode if ( IO_mode == DM9010_BYTE_MODE ) */
for ( i = 0 ; i < RX_length ; i++ ) /* Loop to READ a Byte data from RX FIFO */
RX_data[ i ] = (UINT8) inb ( IOaddr + 4 );      }

else if ( IO_mode == 0 ) { /* I/O 16-bit Word mode if ( IO_mode == DM9010_WORD_MODE ) */
(int) length_tmp = ( RX_length + 1 ) / 2;
for ( i = 0 ; i < length_tmp ; i++ ) /* Loop to READ a Word data from RX FIFO */
( (UINT16 *) RX_data)[ i ] = inw ( IOaddr + 4 );      }

else if ( IO_mode == 1 ) { /* I/O 32-bit Dword mode if ( IO_mode == DM9010_DWORD_MODE ) */
(int) length_tmp = ( RX_length + 3 ) / 4;
for ( i = 0 ; i < length_tmp ; i++ ) /* Loop to READ a Dword data from RX FIFO */
( (UINT32 *) RX_data)[ i ] = inl ( IOaddr + 4 ); } /* inl () is input Long Word for 32-bit */
```

5.8.5 RX Perfect-Filter with Hash Table

Besides, the RX machine supports Perfect-Filter by setting RCR Bit [7] = 1 with Hash table:

```
low ( 0x05, ior(5) | 0x80 ); /* HASHALL Bit [7]=1 in REG. 05, Hash-table filter all */
```

While the Ethernet MAC address of incoming packet is matched to 64-bit Hash table (or it's just the MAC address itself), this packet would be filtered and put into the RX FIFO SRAM as one received packet. Please refer to the datasheet chapter 6.6 about RCR REG. 05 setting.

6 The Others

6.1 Priority of the EEPROM, the Strap Pins, and the Default Setting

The default setting of DM9010 can be changed by the strap pins or the EEPROM with higher priority at EEPROM setting. For example, the default I/O based address of DM9010 is 0x0300 (i.e. SA4~9 are set to 0x0300). If the strap pin TXD [0] is pulled high and the strap pins TXD [1] and TXD [2] are floating (default setting), the I/O based address set by the strap pins is 0x0310. And if the Word 3 Bit [5:4] is "01"b, and Word 6 Bit [11:9] are "111"b for EEPROM setting, the IO based address of EEPROM setting is 0x0370. These settings are different but based on the priority rule; the DM9010 will be operated in the IO based address 0x0370 by the EEPROM 93C46/ LC46 setting.

6.2 Identify How Many DM9010 Chips

In an open system, the following program segment can identify how many DM9010 chips are used on this embedded system.

```
for (IOaddr= 0x0300 , IOdata = 0x0304; IOaddr< 0x380; IOaddr+= 0x10, IOdata+= 0x10)
{
    if ( ( ior ( 0x28 ) == 0x46 ) && ( ior ( 0x29 ) == 0x0A ) )
        if ( ( ior ( 0x2A ) == 0x00 ) && ( ior ( 0x2B ) == 0x90 ) )
        {
            printf("\n DM9010 is found and the location is %04X \n", IOaddr);
            result ++;      /* success found one NIC */
        }
}
return  result;
```

Because the Vendor ID and the Product ID might be changed by the EEPROM 93C46/ LC46 setting, there is another way for your reference as follows,

```
for (IOaddr= 0x0300 , IOdata = 0x0304; IOaddr< 0x380; IOaddr+= 0x10, IOdata+= 0x10)
{
    if ( ( ior ( 0x08 ) == 0x37 ) && ( ior ( 0x09 ) == 0x38 ) )
    {
        printf("\n DM9010 is found and the location is %04X \n", IOaddr);
        result ++;      /* success found one NIC */
    }
}
return  result;
```

6.3 How to Transmit and Receive more than 2048-Byte Packets

If a system transmits or receives more than 2048-byte packets, which size is although over 802.3 spec., the TCR Bit [6] and RCR Bit [6] should be set to "1". Ex. to set iow(RCR | 0x40).

6.4 Performance of the DM9010 LAN Chip

The performance of the DM9010 LAN chip is depended on the capability of the MPU/ MCU. If the MPU/MCU is so fast to match the timing of the maximum speed for the DM9010 IOR#/ IOW#, the performance will be: 10ns + 10ns = 20ns -> 50 Mbps * IOMODE.

8-bit mode	Receive or transmit data	->	8 * 50Mbps	->	*400 Mbps
	RX & TX data at the same time	->	8 * 50Mbps / 2	->	*200 Mbps
16-bit mode	Receive or transmit data	->	16 * 50Mbps	->	*800 Mbps
	RX & TX data at the same time	->	16 * 50Mbps/ 2	->	*400 Mbps
32-bit mode	Receive or transmit data	->	32 * 50Mbps	->	*1600 Mbps
	RX & TX data at the same time	->	32 * 50Mbps/ 2	->	*800 Mbps

*Note: The DM9010 is the 10/ 100 Mbps Ethernet NIC, so the maximum speed is 100 Mbps.

6.5 WOL (Wakeup on LAN)

The DM9010 chip also supports the WOL (Wakeup on LAN) function. There are three methods to generate WOL signals, which are the Magic Packet, Link Change, and Sample Frame. This section will discuss HOWTO implement the WOL function in the DM9010 MAC.

(1) Magic Packet

If DM9010 receives a broadcast packet which content is "FF:FF:FF:FF:FF:FF"+"16 times of MAC address", then the wakeup pin 79 WAKEUP will be activated. Please note that the MAC address must be the same with the unicast address in the PAR REG. 10H ~ REG. 15H.

For example:

```
iow ( WCR , ior ( WCR ) | 0x08 ); /* WCR REG. 0FH Link Change event enable */
iow ( NCR , ior ( NCR ) | 0x40 ); /* NCR REG. 00 Wake-up function enable */
/* If a physical address in DM9010 PAR REG. 10H ~ REG. 15H is 00:60:6e:90:00:01 */
/* DM9010 WOL would be active, while received the Magic packet as follows, */
```

```

/* FF FF FF FF FF FF 00 60 6E 90 00 01 00 60 6E 90 00 01 00 60 6E 90 00 01 */
/* 00 60 6E 90 00 01 00 60 6E 90 00 01 00 60 6E 90 00 01 00 60 6E 90 00 01 */
/* 00 60 6E 90 00 01 00 60 6E 90 00 01 00 60 6E 90 00 01 00 60 6E 90 00 01 */
/* 00 60 6E 90 00 01 00 60 6E 90 00 01 00 60 6E 90 00 01 00 60 6E 90 00 01 */
/* 00 60 6E 90 00 01 (+ another DM9010 TX auto 4-byte CRC appends) */

```

(2) Link Change

No matter the internal PHY or the external MII of DM9010 is used, if the link status has been changed, the wakeup pin will be active.

For example,

```

iow ( WCR , ior ( WCR ) | 0x20 ); /* WCR REG. 0FH Link Change event enable */
iow ( NCR , ior ( NCR ) | 0x40 ); /* NCR REG. 00 Wake-up function enable */

```

(3) Sample Frame

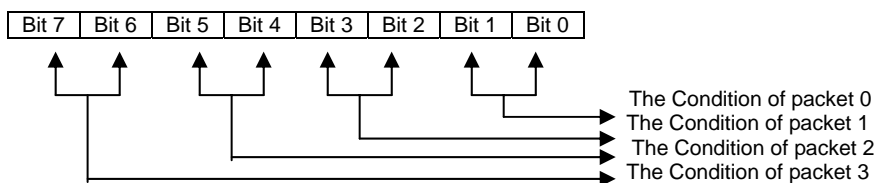
The Sample Frame would be made up of 6 packets and the maximum size of each packet can be 2048-byte. The wakeup pin will be active if DM9010 receives any set of the sample frame packet. The following description will show how to set the Sample Frame in detail.

- First close the receiver function: clear-bit the Bit [0] RXEN of RCR REG. 05 to be "0".
- Stop the self recover function: clear-bit the PAR Bit [7] = 0 in IMR REG. FFH.
- Keep the packet (up to 6 max packets) in the DM9010 FIFO. The format is shown below,

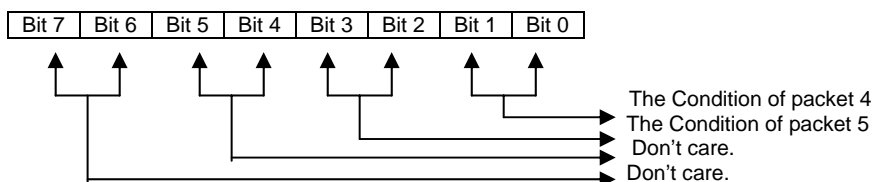
m-byte3 / packet 3	m-byte2 / packet 2	m-byte1 / packet 1	m-byte0 / packet 0	DM9010 Memory location
packet 3-byte 0	packet 2-byte 0	packet 1-byte 0	packet 0-byte 0	0000h
packet 3-byte 1	packet 2-byte 1	packet 1-byte 1	packet 0-byte 1	0004h
:	:	:	:	0008h
:	:	:	:	000Ch
:	:	:	:	0010h
:	:	:	:	:
packet 3-byte 2047	packet 2-byte 2047	packet 1-byte 2047	packet 0-byte 2047	1FFCh
m-byte3 / packet 5	m-byte2 / packet 4	m-byte1 / Condition 1	m-byte0 / Condition 0	

packet 5-byte 0	packet 4-byte 0	Cond. 1-byte 0	Cond. 0-byte 0	2000h
packet 5-byte 1	packet 4-byte 1	Cond. 1-byte 1	Cond. 0-byte 1	2004h
:	:	:	:	2008h
:	:	:	:	200Ch
:	:	:	:	2010h
:	:	:	:	:
packet 5-byte 2047	packet 4-byte 2047	Cond. 1-byte 2047	Cond. 0-byte 2047	3FFCh

The format of Condition 0



The format of Condition 1



If the condition is "00"b or "01"b, don't care the byte; if "10"b, check the byte is matched or not; if "11"b, the DM9010 MAC stops checking.

- Restart the Sample Frame function by set-bit the Bit [4] SAMPLEEN of WCR REG. 0FH to "1", and enable the WOL function by set-bit the WAKEEN Bit [6] = 1 in NCR REG. 00.
- Restart the receiver function by set-bit the Bit [0] RXEN of RCR REG. 05 to be "1".

For example,

```

iow ( RCR , ior ( RCR ) & 0xfe ); /* RCR REG. 05 disable RX machine/ filter */
iow ( IMR , ior ( IMR ) & 0x0f ); /* IMR REG. FFH disable TX&RX Pointer Auto Return */
/* Here, the Sample Frame put in the DM9010 FIFO SRAM 16KB: 0x0000~ 0x3FFF */
for ( i = 0, sample_ptr = 0; i < sample_length; i++, sample_ptr += 4 ) {
/* set sample_ptr to MDWAL as low byte and to MDWAH as high byte in DM9010 SRAM */
iow ( 0xFA, sample_ptr & 0xff );
iow ( 0xFB, ( sample_ptr >> 8 ) & 0xff );
outb ( 0xF8, I0addr ); /* output port number MWCMD= 0xF8 TX WRITE Locking */
if ( i % 2 ) outw ( ( ( (UINT16 *) sample_frame)[(i+1)/2] >> 8 ), I0addr + 4 );
else outw ( ( (UINT16 *) sample_frame)[(i+1)/2], I0addr + 4 ); }
/* set the condition byte[0] & byte[1] of the Sample Frame in 0x2000~ 0x3FFD */
for ( i = 0, sample_ptr = 0x2000; i < sample_length; i++, sample_ptr += 4 ) {
/* set sample_ptr to MDWAL as low byte and to MDWAH as high byte in DM9010 SRAM */
iow ( 0xFA, sample_ptr & 0xff );
iow ( 0xFB, ( sample_ptr >> 8 ) & 0xff );
outb ( 0xF8, I0addr ); /* output port number MWCMD= 0xF8 TX WRITE Locking */
outw ( 0x0002, I0addr + 4 ); } /* Fill Bit[1:0]= "10" of condition 0 in pkt 0 */
// iow ( 0xFA, (sample_ptr+4) & 0xff ); iow ( 0xFB, ( (sample_ptr+4) >> 8 ) & 0xff );

```

```
// outw ( 0x0003, I0addr + 4 ); /* Fill Bit [1:0]= "11"b to stop it checking */
iow ( WCR , ior ( WCR ) | 0x10 ); /* REG. 0FH Link Change event enable WCR */
iow ( NCR , ior ( NCR ) | 0x40 ); /* NCR REG. 00 Wake-up function enable */
iow ( RCR , ior ( RCR ) | 0x01 ); /* RCR REG. 05 enable RX machine/ filter */
```

6.6 IP/TCP/UDP Checksums Offload

The DM9010 LAN chip supports IP/TCP/UDP checksums generation and checking.

Please refer to the datasheet chapter 6.27 & 6.28 about setting TCSCR and RCSCSR.

List the powerful function bits for the TCSCR and RCSCSR registers:

TX Check Sum Control Register (TCSCR REG. 31H):

Bit	Name	Description
2	UDPCSE	UDP Check Sum Enable Generation
1	TCPSE	TCP Check Sum Enable Generation
0	IPCSE	IP Check Sum Enable Generation

RX Check Sum Control Status Register (RCSCSR REG. 32H):

Bit	Name	Description
7	UDPS	UDP Check Sum Status 0: checksum OK, if UDP packet received.
6	TCPS	TCP Check Sum Status 0: checksum OK, if TCP packet received.
5	IPS	IP Check Sum Status 0: checksum OK, if IP packet received.
4	UDPP	UDP packet if indicating "1"
3	TCPP	TCP packet if indicating "1"
2	IPP	IP packet if indicating "1"
1	RCSEN	Receive Check Sum Enable Checking When set "1", the checksum status is inserted in status first byte of packet header.
0	DCSE	Discard Check Sum Error Packets When set "1", if IP/TCP/UDP checksum field is error, this packet will be discarded.

For example,

```
iow ( 0x31 , 0x07 ); /* TX IPCSE+ TCPSE+ UDPCSE checksums generations enable */
iow ( 0x32 , 0x03 ); /* RX checksums checking and bad packet discard enable */
/* IPS/TCPS/UDPS indicating checksums status, while RX IP/TCP/UDP (good) packets */
```

6.7 Auto-MDIX and Application Circuit

The DM9010/ 9000A supports Auto-MDIX powerful feature, while chip power-up at default. When design Auto-MDIX circuit of the DM9010 in the embedded system, the following layout guide must be cared: Traces routed from RXI± and TXO± to the transformer should run in close pairs directly to the transformer such as YCL-PH163539. The network interface should be void of any signals other than the TXO± and RXI± pairs between the RJ-45 to the transformer and the transformer to the DM9010.

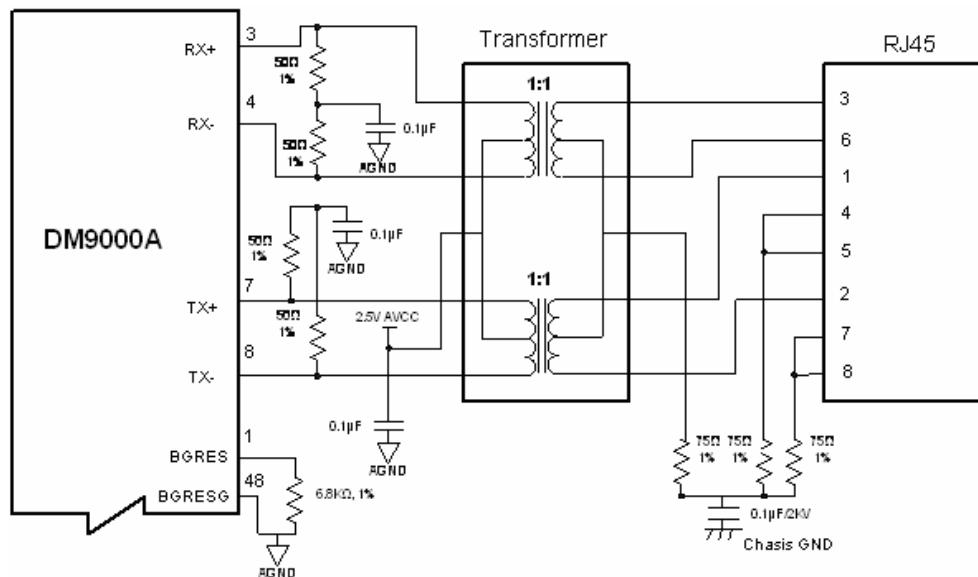


Figure 6.1 Auto-MDIX 10Base-T/100Base-TX Application.

The DM9010 is default turning on Auto-MDIX, and there are two ways to disable Auto-MDIX:

1. To set the SROM Word 7, Wake-up Mode Control, Bit [14] = 1 then zero to re-load it,

```
srom_wri te (0x07, 0x180); /* PHY power-on but di sable Auto-MDI X setting i nto SROM */
i ow ( 0x0B, 0x20 );      /* REEP Bi t [5] of EPCR REG. 0BH to re-load EEPROM val ue */
i ow ( 0x0B, 0x00 );      /* REEP Bi t [5] = 0 normal */
```

2. To write high "1" to the Bit [4] Mdx-down in the PHY REG. 20,

```
phy_wri te(20, 0x0010); /* Mdi x_down Bi t [4]= 1 wi th Mdi x_fi x_Val ue Bi t [5]=0 for MDI */
phy_wri te(20, 0x0030); /* Mdi x_down Bi t [4]=1 wi th Mdi x_fi x_Val ue Bi t [5]=1 for MDI X */
```