



DM9013/DM9003 Programming Guide

DM9013/DM9003 (ISA/Local Bus)

Programming Guide

Outline:

- 1. How to Read/Write MAC Registers**
- 2. How to Read/Write PHY Registers**
- 3. How to Read/Write EEPROM**
- 4. How to Program Host MAC Filter**
- 5. How to Read/Write Per-Port Registers**
- 6. VLAN**
 - 6.1. Port Base**
 - 6.2. Tag Base**
- 7. Priority**
 - 7.1. Port Priority**
 - 7.2. VLAN Tag Priority**
 - 7.3. IP TOS Priority**
- 8. Bandwidth Control**
- 9. Initialize DM9013/DM9003**
- 10. Transmit Packets Operation**
- 11. Receive Packets Operation**



DM9013/DM9003 Programming Guide

1. How to Read/Write MAC Registers

There are two addressing ports through the access of the host interface. One is INDEX_PORT and another is DATA_PORT.

- If user wants to write register, write register offset into INDEX_PORT first then write the value into DATA_PORT.
- If user wants to read the register value, write register offset into INDEX_PORT first then read the value from DATA_PORT.

I/O base address (INDEX_PORT) :

x86 system (*only for DM9013*)

0x300 ~ 0x370

Other platform

Please refer CPU datasheet

DATA_PORT depends on CMD pin. If CMD pin connects to CPU_ADDRESS_BUS[2], DATA_PORT = INDEX_PORT + 4.

The CMD pin of DM9003/DM9013 connects to CPU_ADDRESS_BUS[2] in most condition. In some cases, the CPU_ADDRESS_BUS[0] can't work in 16bit mode (this can be checked with CPU data sheet)

CPU_ADDRESS_BUS[1].

CPU_ADDRESS_BUS[2], DATA_PORT = INDEX_PORT + 2.

X86 system :

```
/* Read DM9013 register */
static u8 ior( int reg)
{
    outb(reg, INDEX_PORT);
    return inb(DATA_PORT);
}
/* Write the value into DM9013 register */
static void iow(int reg, u8 value)
{
    outb(reg, INDEX_PORT);
    outb(value, DATA_PORT);
}
```

Version : 1.1

Author : Bill

Update : 2007/10/11

2/17



DM9013/DM9003 Programming Guide

Other platform :

```
/*
 * If your bus is 8-bit or 32-bit, you must modify below.
 * Ex. your bus is 8 bit
 *   DM9000_PPTR *(volatile u8 *)(DM9013_BASE)
 */
#define DM9013_BASE    0x12345678;    /* Depend on CPU */
#define DM9013_PPTR    *(volatile u16 *)(DM9013_BASE)
#define DM9013_PDATA   *(volatile u16 *)(DM9013_BASE + 4)

static void iow(u8 reg, u8 value)
{
    DM9013_PPTR = reg;
    DM9013_PDATA = value & 0xff;
}

static u8 ior(u8 reg)
{
    DM9013_PPTR = reg;
    return DM9013_PDATA & 0xff;
}
```

2. How to Read/Write PHY Registers

DM9013/DM9003 has two internal PHYs. The user can write PHY number (0~2) into Reg.0C_H.[7:6]. Use Reg.0B_H to control PHY read/write function.

Read PHY :

- Write PHY number (0~2) into Reg.0C_H.[7:6]
- Reg.0B_H.[3] = 1 // select PHY
- Reg.0B_H = 0x8 // clear PHY read command
- Reg.0B_H = 0xC // PHY read command
- Reg.0B_H = 0x8 // clear PHY read command
- Wait for Reg.0B_H.[0] =0 (Reg.0B_H.[0] =1 means it is busy)
- Get data from Reg.0D_H and Reg.0E_H.



DM9013/DM9003 Programming Guide

```
/*
 * Read a word from phyxcer
 */
static u16 phy_read(int reg, int port)
{
    reg = (port<<6) | reg;
    iow(0x0C, reg);

    iow(0x0B, 0x8);          /* Clear phyxcer read command */
    iow(0x0B, 0xc);         /* Issue phyxcer read command */
    iow(0x0B, 0x8);         /* Clear phyxcer read command */
    while((ior(0x0B) & 0x01);
    /* The read data keeps on REG_0D & REG_0E */
    return ( ior(0x0E << 8 ) | ior(0x0D);
}
}
```

Write PHY :

- Write PHY number (0~2) into Reg.0C_H. [7:6]
- Write data into Reg.0D_H and Reg.0E_H
- Reg.0B_H. [3] = 1 // select PHY
- Reg.0B_H = 0x8 // clear PHY write command
- Reg.0B_H = 0xA // PHY write command
- Reg.0B_H = 0x8 // clear PHY write command
- Wait for Reg.0B_H. [0] =0 (Reg.0B_H. [0] =1 means it is busy)

```
/*
 * Write a word to phyxcer
 */
static void phy_write(int reg, u16 value, int port)
{
    reg = (port<<6) | reg;
    /* Fill the phyxcer register into REG_0C */
    iow(0x0C, reg);

    /* Fill the written data into REG_0D & REG_0E */
    iow(0x0D, (value & 0xff));
    iow(0x0E, ( (value >> 8) & 0xff));

    iow(0x0B, 0x8);          /* Clear phyxcer write command */
    iow(0x0B, 0xa);         /* Issue phyxcer write command */
    iow(0x0B, 0x8);         /* Clear phyxcer write command */
    while((ior(0x0B) & 0x01); /* Wait Process over */
}
}
```



DM9013/DM9003 Programming Guide

3. How to Read/Write EEPROM

3.1. Read EEPROM Operation

- Select EEPROM operation. Set Reg.0B_H.[3]=0.
- Write a offset value into Reg.0C_H
- Set EEPROM read command. Set Reg.0B_H.[2] =1
- Wait read command completion until Reg.0B_H.[0] =0. (When Reg.0B_H.[0]=1, it indicates that EEPROM access is progress)
- Clear EEPROM read command. Set Reg.0B_H.[2] =0.
- Obtain EEPROM data from Reg.0D_H and Reg.0E_H (Reg.0D_H is Low byte data and Reg.0E_H is High byte data).

```
/*
 *   Read a word data from SROM
 */
static u16 read_eeprom_word(int offset)
{
    iow(0x0C, offset);
    iow(0x0B, 0x4);           /* Read Data */
    while(ior(0x0B)&0x1);     /* Wait for read process completion */
    iow(0x0B, 0);           /* Clear read command */
    return (ior(0x0D) + (ior(0x0E) << 8));
}
```

3.2. Write EEPROM Operation

- Select EEPROM operation. Set Reg.0B_H.[3]=0
- Write a offset value into Reg.0C_H
- Write data (16 bits) into Reg.0D_H and Reg.0E_H (Reg.0D_H is Low byte data and Reg.0E_H is High byte data).
- Set EEPROM write command. Set Reg.0B_H.[1] =1 and Reg.0B_H.[4] =1.
- Wait read command completion until Reg.0B_H.[0] =0. (When Reg.0B_H.[0]=1, it indicates that EEPROM access is progress)
- Clear EEPROM read command. Set Reg.0B_H.[1] =0 and Reg.0B_H.[4] =0.



DM9013/DM9003 Programming Guide

```
/*      Write a word data from SROM      */
static void write_eeprom_word(int offset, u16 value)
{
    iow(0x0C, offset);          /* Assign the register offset*/
    iow(0x0D, value&0xff);      /* Put Data      */
    iow(0x0E, (value<<8)&0xff);
    iow(0x0B, 0x12);           /* Write Data      */
    while(ior(0x0B)&0x1);       /* Wait for write process completion */
    iow(0x0B, 0x0);           /* Clear write command */
}
```

4. How to Program Host MAC Filter

Bellow MAC addresses are for special purpose; don't use these for MAC address.

- FF : FF : FF : FF : FF : FF, when all bits are set to 1, it is for broadcast address.
- 01 : XX : XX : XX : XX : XX, when the bit0 of first byte is set to 1, it is for multicast address.

4.1. Receive Unicast Packets

Write MAC address into Reg.10_H~15_H. These values were also loaded from EEPROM if having EEPROM.

4.2. Receive Multicast Packets

Programmer can set Reg.05_H.[3]=1 to pass all multicast packets. If programmers configure Reg.16_H~1D_H, host only receives the multicast packets that host wants and other multicast packets will be dropped by the filter.



DM9013/DM9003 Programming Guide

```
struct dev_mc_list *mcptr = dev->mc_list; /* multicast addresses list */
int mc_cnt = dev->mc_count; /* multicast addresses count */
u32 hash_val;
u16 i, oft, hash_table[4];

/* Clear Hash Table */
for (i = 0; i < 4; i++)
    hash_table[i] = 0x0;

/* the multicast address in Hash Table : 64 bits */
for (i = 0; i < mc_cnt; i++, mcptr = mcptr->next) {
    hash_val = cal_CRC((char *)mcptr->dmi_addr, 6, 0) & 0x3f;
    hash_table[hash_val / 16] |= (u16) 1 << (hash_val % 16);
}

/* Write the hash table to MAC MD table */
for (i = 0, oft = 0x16; i < 4; i++) {
    iow(oft++, hash_table[i] & 0xff);
    iow(oft++, (hash_table[i] >> 8) & 0xff);
}
```

```
/*
Calculate the CRC valude of the Rx packet
flag = 1 : return the reverse CRC (for the received packet CRC)
      0 : return the normal CRC (for Hash Table index)
*/
static unsigned long cal_CRC(unsigned char * Data, unsigned int Len, u8 flag)
{
    u32 crc = ether_crc_le(Len, Data);

    if (flag)
        return ~crc;

    return crc;
}
```



DM9013/DM9003 Programming Guide

4.3. Receive Broadcast Packets

Set Reg.1D_H=0x80.

5. How to Read/Write Per-Port Registers

Reg.61_H ~ 6F_H and Reg.80_H ~ 84_H are per-port registers. At first, user must write port number (0~3) into Reg.60_H.[1:0]. Then, write/read the register you want.

Ex.

Read the Reg.61_H of Port1

<pre>iow (0x60, 1); //Port index =1 data= ior(0x61);</pre>

Ex.

Write the Reg.61_H of Port1

<pre>iow (0x60, 1); //Port index =1 iow(0x61, data);</pre>

6. VLAN

DM9013/DM9003 supports Port-base VLAN and Tag-base VLAN.

6.1. Port Base

DM9013/DM9003 will use per-port VID as VLAN group reference if coming packet is non-VLAN packet.

1. Enable Port Base VLAN (Reg.53_H.[0]=0)
2. Set Per-Port VID (Reg.6E_H)
3. Set VLAN mapping table (Reg.B0_H ~BF_H)

Ex.

uP (P3)		
Port 0(P0)	Port 1(P1)	Port 2(P2)

We want to divide four ports (P0~P3) into two groups. Group 1 has P0 and P3. Group 2 has P1, P2, and P3.

Step 1 :

Reg.53_H bit 0 =0

Step 2 :

Version : 1.1

Author : Bill

Update : 2007/10/11



DM9013/DM9003 Programming Guide

P0 VID=0 (Reg.60_H = 0, Reg.6E_H=0)

P1 VID=1 (Reg.60_H = 1, Reg.6E_H=1)

P2 VID=1 (Reg.60_H = 2, Reg.6E_H=1)

P3 VID=2 (Reg.60_H = 3, Reg.6E_H=2)

Step 3 :

Reg.B0_H = 1001_B

Reg.B1_H = 1110_B

Reg.B2_H = 1111_B

6.2. Tag Base

- Normal packet content (Ethernet Packet from Layer 2)

Destination Address (6 bytes)	Source Address (6 bytes)	Packet Type (2 bytes)	Data (46~1500 bytes)	CRC (4 bytes)
Byte 0~5	Byte 6~11	Byte 12~13		

- VLAN packet content

DA (6 bytes)	SA (6 bytes)	VLAN Tag (4 bytes)	Packet Type (2bytes)	Data (46~1500 bytes)	CRC (4 bytes)
Byte 0~5	Byte 6~11	Byte 12~15	Byte 16~17		

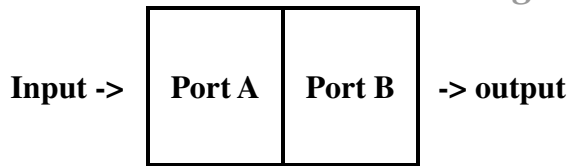
- VLAN Tag format

Tag protocol TD 0x8100 (2 bytes)	Priority (3 bits)	Canonical Format Indicator (1 bit)	VLAN ID (12 bits)
-------------------------------------	----------------------	---------------------------------------	----------------------

User can define each port as Tag port or Un-Tag port by reg.6DH.[7] (Output Packet Tagging Enable). The operation of packet between Tag and Un-Tag port can explain by following example :



DM9013/DM9003 Programming Guide



0 : Un-Tag packet 1: Tag packet

Input	Output
0	0
0	1
1	0
1	1

Example1 : 00 (Port receives Un-Tag packet and send to Un-Tag port)

DM9013/DM9003 will check the per-port VID first then check VLAN group mapping. If destination port same VLAN as receiving port then this packet will forward to destination port without any change. If destination port not same VLAN as receiving port then this packet will be dropped.

Example2 : 01 (Port receives Untag packet and send to Tag port)

DM9013/DM9003 will check the per-port VID first then check VLAN group mapping. If destination port same VLAN as receiving port then this packet will forward to destination port with four byte VLAN tag (Tag VID = input port Reg.6E_H VID) and new CRC. If destination port not same VLAN as receiving port then this packet will be dropped.

Example3 : 10 (Port receives Tag packet and send to Untag packet)

DM9013/DM9003 will check the packet VLAN ID first then check VLAN group mapping. If destination port same VLAN as receiving port then this packet will forward to destination port after remove four bytes with new CRC. If destination port not same VLAN as receiving port then this packet will be dropped.

Example4 : 11 (Port receives Tag packet and send to Tag port)

DM9013/DM9003 will check the packet VLAN ID first then check VLAN group mapping. If destination port same VLAN as receiving port then this packet will forward to destination port without any change. If destination port not same VLAN as receiving port then this packet will be dropped.



DM9013/DM9003 Programming Guide

Ex.

uP (P3)		
Port 0(P0)	Port 1(P1)	Port 2(P2)

We want to divide four ports (P0~P3) into two groups. Group 1 has P0 and P3 with VLAN ID=1. Group 2 has P1, P2, and P3 with VLAN ID=2.

Step 1 :

Reg.53_H bit 0 =1

Step 2 :

Enable P0~P3 output packet tagging

P0 : Reg.60_H = 0, Reg.6D_H. [7]=1

P1 : Reg.60_H = 1, Reg.6D_H. [7]=1

P2 : Reg.60_H = 2, Reg.6D_H. [7]=1

P3 : Reg.60_H = 3, Reg.6D_H. [7]=1

Step 3 :

Reg.B1_H = 1001_B

Reg.B2_H = 1110_B

7. Priority

DM9013/DM9003 provides three modes priority, as Port priority, VLAN Tag priority, and IP header TOS priority. DM9013/DM9003 provides four priority queues on each port and each queue is assigned a weight. DM9013/DM9003 supports two modes of weighted fair queuing. Mode 1 is 8:4:2:1 and mode 2 is queue 3 > 2 > 1 > 0.

7.1. Port Priority

Set per-port Port-Base priority (Reg.6D_H. [1:0]). Queue 3 is the highest priority and Queue 0 is the lowest priority. User can assign a priority value to each port.

7.2. VLAN Tag Priority

The weight of VLAN priority is assigned according to Reg.D0_H and Reg.D1_H. The default value is the following :

Queue 0 : VLAN priority 00h and 01h

Queue 1 : VLAN priority 02h and 03h

Queue 2 : VLAN priority 04h and 05h

Queue 3 : VLAN priority 06h and 07h

Version : 1.1

Author : Bill

Update : 2007/10/11

11/17



DM9013/DM9003 Programming Guide

Reg. D0_H

Bit	Default	Description
7:6	1, R/W	VLAN priority =03h
5:4	1, R/W	VLAN priority =02h
3:2	0, R/W	VLAN priority =01h
1:0	0, R/W	VLAN priority =00h

Reg. D1_H

Bit	Default	Description
7:6	3, R/W	VLAN priority =07h
5:4	3, R/W	VLAN priority =06h
3:2	2, R/W	VLAN priority =05h
1:0	2, R/W	VLAN priority =04h

7.3. IP TOS priority

Must enable Per-port Reg.6D_H. [4]=1. In default setting, DM9013/DM9003 checks most significant 3-bit only of TOS. The user can set Reg.53_H. [7] =1 to check full 6-bit of TOS. If the Reg.53_H. [7] =0, the weight of VLAN priority is assigned in Reg.C0_H and Reg.C1_H. If Reg.53_H. [7] =1, the weight of VLAN priority is assigned from Reg.C0_H to Reg.CF_H.

8. Bandwidth Control

DM9013/DM9003 support two modes bandwidth control by Reg.61_H. [3]. If Reg.61_H. [3]=0, the bandwidth of ingress and egress are controlled by Reg.66_H separately. If Reg.61_H. [3]=1, the total (TX and RX) bandwidth of this port is equal to the value you set and is controlled by Reg.67_H. [3-0].

Bandwidth control setting steps :

- Set per-port Bandwidth control (Reg.61_H. [3])
- Set Bandwidth (Reg.66_H or Reg.67_H)

Ex. Set port 1 ingress bandwidth=64K bps and egress bandwidth= 16M bps

Step 1 : Reg.60_H = 1, Reg.61_H. [3]=0

Step 2 : Reg.66_H = **00011001**_B

Version : 1.1

Author : Bill

Update : 2007/10/11

12/17



DM9013/DM9003 Programming Guide

9. Initialize DM9013/DM9003

1. Switch reset, set Reg.52_H.[6]=1, and wait for this bit clear.
2. Software reset, set Reg.0_H.[0]=1, and wait for this bit clear.
3. Read Reg.FE_H.[7:6] to access I/O mode(0 0: 16-bit mode, 0 1: 32-bit mode, 1 0: 8-bit mode)
4. Check Reg.41_H.[6]. Reg.41_H.[6]=1 is 2 port mode; Reg.41_H.[6]=0 is 3 port mode.
5. Reset PHY.
6. Enable flow control, set Reg.0A_H.[5]=1.
7. If having EEPROM, reload EEPROM (set Reg.0B_H.[5]=1), else set switch configurations.
8. Set address filter hash table.
 - Write MAC address into Reg.10_H~ Reg.15_H.
 - Set multicast address into Reg.16_H~ Reg.1D_H. (refer to chapter 4)
9. Enable receive packet.
10. Enable TX/RX interrupt mask.



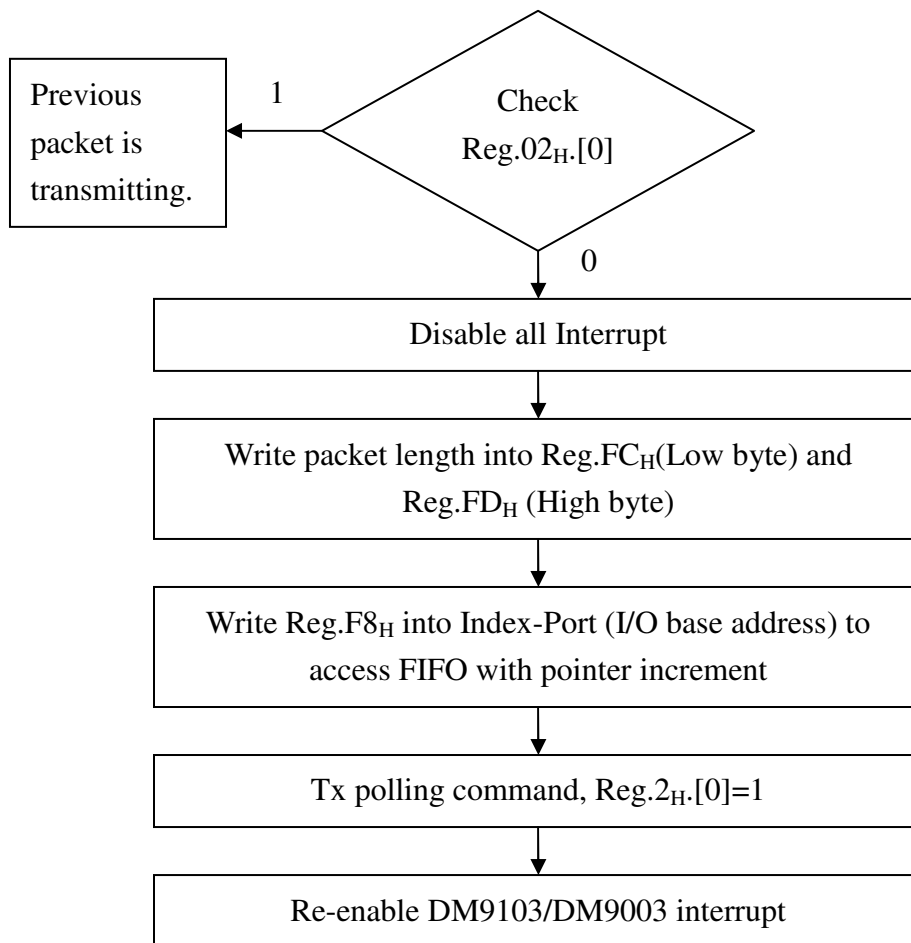
DM9013/DM9003 Programming Guide

10. Transmit Packets Operation

DM9013 has 8-bit, 16-bit, and 32-bit data modes to access internal memory. It indicates the width of data bus between CPU and DM9013. But DM9003 has 8-bit and 16-bit data modes. The data mode can be set by strap pins (DM9003 : pin 28 ; DM9013 : pin 50 and 51).

- Disable DM9013/DM9003 Interrupt.
- Check Reg.02_H. [0]=0. If Reg.02_H. [0]=1, previous packet is transmitting.
- Write packet length into Reg.FC_H and Reg.FD_H (Reg.FC_H is Low byte ; Reg.FD_H is High byte).
- Write 0xF8 into Index port (I/O base address) to access internal memory (FIFO) with pointer increment. The write pointer is increased by 1, 2, or 4, depends on data mode (8-bit, 16-bit, or 32-bit respectively).
- Transmitted polling command. Set Reg.02_H. [0]=1.
- Re-enable DM9013/DM9003 interrupt.

After transmitting the packet completion, DM9013/DM9003 will generate interrupt signal. Programmers can read Reg.FE_H to get the interrupt status if Reg.FF_H was set when DM9013/DM9003 generates TX packet transmitted interrupt.





DM9013/DM9003 Programming Guide

11.Receive Packets Operation

DM9013 has 8-bit, 16-bit, and 32-bit data modes to access internal memory. It indicates the width of data bus between CPU and DM9013. But DM9003 has 8-bit and 16-bit data modes. The data mode can be set by strap pins (DM9003 : pin 28 ; DM9013 : pin 50 and 51).

The RX SRAM is a ring data structure. Each packet has a 4-byte header, which was defined by DAVICOM, in front of the received packet. The format of 4-byte header is the following.

<i>First Byte</i>	<i>Second Byte</i>	<i>Third Byte</i>	<i>Fourth Byte</i>
Ready or not (1: ready; 0:non-ready)	Status (refer to Reg.F6 _H)	Length (low-byte)	Length (high-byte)

The follow char of receiving packets is bellow.

