Freescale Semiconductor, Inc.

Freescale Semiconductor

Implementing GMII Interface on C-5

Generation Date	12/2/01
Revision	1.1
Product	C-5, C-Ware Software Toolset
Relevant version	D0, CST2.0
Description	Describes the C-5's implementation of the GMII port
Distribution	Customer
Author	Bing Cheng

Freescale Semiconductor, Inc.

© Freescale Semiconductor, Inc., 2004. All rights reserved.

C-PORT CORPORATION
PROPRIETARY & CONFIDENTIAL

For More Information On This Product,
Go to: www.freescale.com

1 INTRODUCTION

GMII stands for Gigabit Media Independent Interface. It is a standard interface specified by the IEEE Std 802.3. The purpose of this interface is to provide a simple interconnection between MAC and PHY that is independent of the physical media types.

Many commercial Gigabit Ethernet PHY devices now use GMII as means to communicate to the MAC device. GMII has become the most common MAC-to-PHY interface for Gigabit Ethernet applications. The C-5 can implement Gigabit Ethernet MAC functions internally and it can connect to external PHY devices via GMII. This document is intended to describe how the C-5 works with Ethernet PHY device via GMII.

This document is targeted for the C-5 Rev D0. Previous revisions of the C-5 do not fully support the GMII interface. It's assumed that the reader should have adequate knowledge on the architecture of the C-5.

2 GMII FUNDAMENTALS

The signals for a GMII interface and their directions are shown in Figure 1 below. Refer to the IEEE Std 802.3 document for more details.

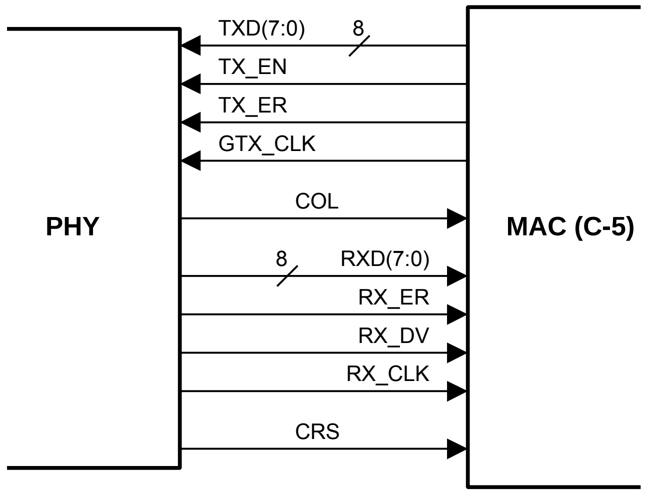

Figure 1 *GMII Signals*

As shown, there are total 24 pins required for a GMII interface. Below is a brief description of the functionality of these signals:

- GTX_CLK provides the timing reference for the transfer of the TX_EN, TX_ER and TXD. It's supplied by the MAC device, which in this case is the C-5. Its frequency is nominally 125 MHz.
- RX_CLK is the timing reference for the transfer of the RX_DV, RX_ER and RXD signals. The PHY usually recovers this clock from the receive data from the physical medium and provides it to the MAC. This clock is derived from the local clock (e.g. GTX_CLK) if the physical medium is not attached. Its frequency is also nominally 125 MHz.

For More Information On This Product,
Go to: www.freescale.com

Freescale Semiconductor, Inc.

Freescale Semiconductor, Inc.

- TX_EN (Transmit Enable) is used to indicate to the PHY that data is present on the GMII for transmission. It is asserted synchronously with the first octet of the preamble and remains asserted for the duration of the entire data frame.
- TXD(7:0) (Transmit Data) are the 8 data signals that are used for the MAC to supply byte-aligned data to the PHY for transmission.
- TX_ER (Transmit Coding Error) is used to tell the PHY to transmit an illegal symbol code to the physical medium.
- RX_DV (Receive Data Valid) is driven by the PHY to indicate the PHY is presenting recovered and decoded data on the RXD(7:0). It's asserted during the entire data frame, and so provide an envelope signal for a valid data frame including Preambles, Start Frame Delimiter (SFD) and Frame Check Sequence (FCS) bytes. The MAC uses this signal to delineate the frame.
- RXD(7:0) (Receive Data) are the 8 data signals for the PHY to present byte-aligned receive data to the MAC. Data byte on these signals is valid only when the RX_DV is asserted and RX_ER is de-asserted.
- RX_ER (Receive Error) is used by the PHY to indicate detection of illegal symbol code during a reception of a frame (when RX_DV is asserted), so the MAC can discard the frame. It's used to signal Carrier Extension when RX_DV is de-asserted.
- CRS (Carrier Sense) is a signal driven by the PHY. While operating in half duplex mode, the PHY will assert CRS when either transmit or receive medium is non-idle and de-assert it when both transmit and receive media are idle. While operating in full duplex mode, the behavior is the CRS is not specified by the standard, hence it can be ignored by the MAC.
- COL (Collision Detected) is asserted by the PHY upon detection of a collision on the medium when it's operating in half duplex mode. While operating in full duplex mode, the behavior is the COL is not specified by the standard, hence it can be ignored by the MAC.

Permissible code combinations on TXD(7:0), TX_EN and TX_ER are defined as follows.

TX_EN	TX_ER	TXD(7:0)	Description
0	0	0x00 through 0xFF	Normal inter-frame
0	1	0x00 through 0x0E	Reserved
0	1	0x0F	Carrier Extend
0	1	0x10 through 0x1E	Reserved
0	1	0x1F	Carrier Extend Error
0	1	0x20 through 0xFF	Reserved
1	0	0x00 through 0xFF	Normal data transmission
1	1	0x00 through 0xFF	Transmit error propagation

Table 1 *Permissible Code Combination on TX*

Permissible code combinations on RXD(7:0), RX_DV and RX_ER are defined as follows.

For More Information On This Product,
Go to: www.freescale.com

RX_DV	RX_ER	RXD(7:0)	Description
0	0	0x00 through 0xFF	Normal inter-frame
0	1	0x00	Normal inter-frame
0	1	0x01 through 0x0D	Reserved
0	1	0X0E	False Carrier indication
0	1	0x0F	Carrier Extend
0	1	0x10 through 0x1E	Reserved
0	1	0x1F	Carrier Extend Error
0	1	0x20 through 0xFF	Reserved
1	0	0x00 through 0xFF	Normal data reception
1	1	0x00 through 0xFF	Data reception error

Table 2 *Permissible Code Combinations on RX*

There are two other signals defined for PHY management purpose, which are not shown in Figure 1. These two signals are MDC and MDIO. They provide means for the MAC device to access the internal management registers on the PHY device, so that the MAC device can configure the PHY as well as check the status of the port. The IEEE standard defines a serial protocol to be applied on these two signals, where MDIO carries the serial data and MDC provides a clock reference to for the serial data. Multiple PHY devices can share the same management interface, and each of them needs to be assigned a unique PHY address.

3 GMII Implementation on the C-5

Since data transfer rate for GMII is 1 Gb/s in each direction, the CPs need to run in aggregation mode to share the load of processing the data stream at this speed. Running in aggregation mode, all the interface pins of 4 CPs in a cluster can be combined and used together to implement an 8-bit interface required by GMII. For more information on aggregation mode, refer to the C-5 Network Processor Architecture Guide.

There are a total of 28 pins within a cluster, so each cluster has enough signals to implement one GMII interface. The SDPs within a cluster are capable of handling the throughput of a full duplex Gigabit Ethernet port, so the MAC function can fully be implemented in one cluster. However, the CPRCs may not have enough cycles to process both directions of the traffic concurrently, and hence, wire speed operation cannot be achieved. Another concern is that the IMEM space within a cluster may not be big enough to hold both transmit and receive code. The C-5e, with faster processors, improved architecture and larger IMEM will likely solve these two problems.

In the current C-5, to ensure we can support the GMII interface at line rate (1 Gb/s full duplex), one of our Gigabit Ethernet reference design uses two clusters to implement one GMII interface. This application, where one cluster handles the transmission of the Gigabit Ethernet port and the other processes the reception of the port, supports Layer 2 Bridging and Layer 3 IP forwarding functions on the Gigabit Ethernet port. We also have another reference application that

Freescale Semiconductor, Inc.

For More Information On This Product,
Go to: www.freescale.com

implements two single cluster GMII interfaces. This application supports IP forwarding and MPLS switching on the two Gigabit Ethernet ports, but the two ports do not run at line rate on the C-5.

3.1 Pin Connections

To implement GMII interface, the CP pins are configured to operate at LVTTL levels. The 24 GMII signals are divided into two groups, TX Group and RX Group. The TX Group includes GTX_CLK, TXD(7:0), TX_EN, TX_ER, CRC and COL. The other signals, RX_CLK, RXD(7:0), RX_DV and RX_ER are in the RX Group. Whether one cluster or two clusters are used for a GMII interface, the TX Group signals must be connected to the two lower-numbered CPs in a cluster, and the RX Group signals must be on the two higher-numbered CPs in a cluster. If two clusters are used, they can be any two of the four clusters on a C-5.

Shown below is the pin connection map of GMII signals to CP pins.

C-5 Pin Name	GMII Signal Name	MII Signal Name	I/O Direction on C-5
CPn_0	GTX_CLK	Not Used	O
CPn_1	Not Used	TXCLK	I
CPn_2	TXD(0)	TXD(0)	O
CPn_3	TXD(1)	TXD(1)	O
CPn_4	TXD(2)	TXD(2)	O
CPn_5	TXD(3)	TXD(3)	O
CPn_6	TX_EN	TX_EN	O
CPn+1_0	No Connect	No Connect	
CPn+1_1	COL	COL	I
CPn+1_2	TXD(4)	Not Used	O
CPn+1_3	TXD(5)	Not Used	O
CPn+1_4	TXD(6)	Not Used	O
CPn+1_5	TXD(7)	Not Used	O
CPn+1_6	TX_ER	TX_ER	O
CPm+2_0	No Connect	No Connect	
CPm+2_1	RX_CLK	RX_CLK	I
CPm+2_2	RXD(0)	RXD(0)	I
CPm+2_3	RXD(1)	RXD(1)	I
CPm+2_4	RXD(2)	RXD(2)	I
CPm+2_5	RXD(3)	RXD(3)	I
CPm+2_6	RX_DV	RX_DV	I
CPm+3_0	No Connect	No Connect	

Freescale Semiconductor, Inc.

For More Information On This Product,
Go to: www.freescale.com

CPm+3_1	CRS	CRS	I
CPm+3_2	RXD(4)	Not Used	I
CPm+3_3	RXD(5)	Not Used	I
CPm+3_4	RXD(6)	Not Used	I
CPm+3_5	RXD(7)	Not Used	I
CPm+3_6	RX_ER	RX_ER	I

Note: n and m in column 1 of the above table can be 0, 4, 8 or 12. If a single cluster is used, n = m; if two are used, n ≠ m.

Table 3 *GMII to CP Pin Map*

Freescale Semiconductor, Inc.

3.2 Mode Register Configuration

In order to configure the CPs to operate as a GMII MAC, the following registers in the CPs need to be set up: SDP_Mode3, SDP_Mode4, SDP_Mode5 and PIN_Mode. All the bits in these registers need to be set properly for the GMII interface to function.

There is a CPI function that can be used to do this. The CPI function is psEthernetportConfigure(), which is part of the Protocol Services of the CPI. The application software just need to call this function for each CP with the proper parameters, the registers mentioned above for the CP will be configured appropriately. The parameters that need to be specified include: 1) the processor ID of the CP to configure, 2) what type of PHY is used (GMII, MII, TBI or RMII), and 3) whether you need to configure the RX side, TX side or both of the CP. Therefore, the software doesn't need to deal with what value need to be programmed onto each register. Refer to the C-Ware CPI User Guide for details on how to use this function.

The settings for these registers are given as a reference in 4 Appendix section.

3.3 Auto-Negotiation Implementation

Most if not all of the GMII PHY devices support Auto-Negotiation, so the C-5 functioning as a MAC device needs to adjust its operation according to the Auto-Negotiation results. The communication of the results is done via the PHY management interface.

3.3.1 PHY Management

The C-5 has a serial interface port that can implement the MDC and MDIO management protocol specified by the standard. This interface is used to access management registers in the PHY device. There is a set of 16-bit registers implemented in any GMII PHY device. Some of them are specified and required by the IEEE standard, and some are specific to the PHY. Via the serial interface, the C-5 can write to these registers to configure the PHY as well as read from these registers to check status of the PHY or gather statistics for the network port.

The speed of the serial interface is programmable. It can also be configured to run with or without Preamble suppression, which is specified by the IEEE standard. To adjust to the

For More Information On This Product,
Go to: www.freescale.com

implementation of some PHY device in the market, the C-5 provides an option to run the serial interface with 1 turn around bits in MDIO read operation. Note that the standard requires two turn around bits. All of these options are configured through the Serial Bus Configuration Register. In the program, the user can simply call the ksSerialBusConfigMdio() CPI function to set up these options. CPI functions, ksMdioRead() and ksMdioWrite() can be used to access the PHY registers.

In some cases, it's desirable to configure the physical port to operate in a certain mode, and some PHY devices need to be initialized before it can be put to use. The C-5 can configure the PHY by writing to its management registers at initialization time and command it to start the Auto-Negotiation process.

Most of the GMII PHY devices can be programmed to generate hardware interrupts to the MAC it connects to upon changes of certain PHY status. The C-5 has an interrupt input pin, XPUHOT that can connect to the PHY interrupt. Changes in the PHY status will interrupt the XP of the C-5 so that the XP program can take the appropriate measures. Of course, the alternative way for the C-5 to find out status changes in the PHY is to use polling. In that case, the XP program will implement a timed loop that polls the internal management registers periodically to check for status changes.

The serial port is only accessible by the XP or the host processor (via address mapping on the PCI bus).

3.3.2 Adjust to Auto-Negotiation Results

Auto-Negotiation is a process where the PHY device negotiates with its link partner to reach the most optimum operation mode between them. The important options to negotiate are: at what speed data is transported on the line; whether the line is full duplex or half duplex; and whether flow control (pause frames) can be applied to the link. Upon detection of the physical link, the PHY device automatically performs this process. The result of Auto-Negotiation is stored in the management registers.

The C-5 can access the PHY management registers to check the Auto-Negotiation result. The XP is the only internal processor that can access this serial port. The way our reference application handles Auto-Negotiation is that the XP will read the PHY registers. If any thing changes in the operation mode on a physical interface, it will send a control message to the corresponding CP. The CP will make the adjustment to adapt the new operation mode. This adjustment can be as simple as setting a control register bit or as complicated as reloading different piece of micro-code to its SDPs.

3.3.3 Switching to MII interface

When the PHY device determines during the Auto-Negotiation process that it needs to run at 10 or 100 Mb/s speed, it will switch to MII type of interface. MII is also specified in the IEEE802.3 standard. The differences between MII and GMII is the TX and RX data buses are 4-bit wide in MII and TX clock is output from the PHY device (see Table 3 for details). To adjust to these differences, the mode registers on the CPs need to be reprogrammed. Again, it can be done by calling the CPI function, psEthernetportConfigure() in the XP program. The settings for these registers when the interface operates as a MII type are given as a reference in 4 Appendix section.

Freescale Semiconductor, Inc.

For More Information On This Product,
Go to: www.freescale.com

Moreover, the CPRC needs to load different microcode to its SDP. The difference in the microcode is the TxBit and RxBit processor code where the data bus width is differentiated.

3.4 Clock and Signal Timing

A GMII PHY device requires a 125 MHz clock for the timing reference for the TX signals, and the MAC provides this clock on the GTX_CLK (CPn_0, where n = 0, 4, 8 or 12). The C-5 has a clock input pin, CCLK6, which can take in a 125 MHz reference clock. This clock source is supplied to each CP cluster to provide individual transmit reference clock for each GMII interface.

On the RX side, the C-5 expects the PHY to provide the 125 MHz clock reference for the RXD, RX_ER and RX_DV signals. This clock signal is driven on the RCLK pin (CPn+2_1, where n = 0, 4, 8 or 12). The C-5 used this clock to latch in the other input signals.

The input and output signal timing relative to their reference clocks on the C-5 is shown below.

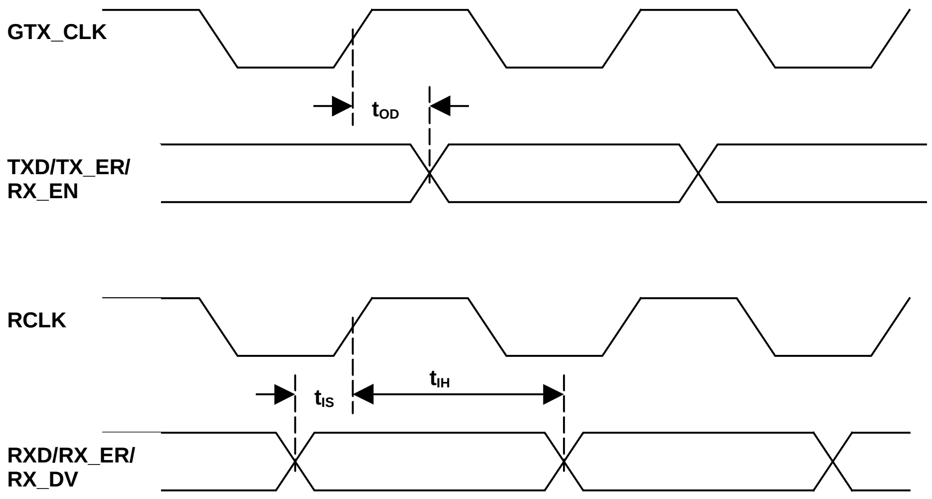

Figure 2 *C-5 GMII Signal Timing*

Symbol	Parameter	Min	Typ	Max	Unit
t_{OD}	Output delay	3.2		5.8	ns
t_{IS}	Input setup time	2.0			ns
t_{IH}	Input hold time	2.5			ns

Table 4 *C-5 GMII Signal Timing Parameters*

3.5 Implementation-specific Issues

Since the MAC function is implemented by the microcode running on the SDPs, the PHY status and control pins such as RX_DV, RX_ER, CRC, COL, TX_EN and TX_ER need to be accessed

For More Information On This Product,
Go to: www.freescale.com

Freescale Semiconductor, Inc.

and controlled by the RxBit and TxBit processors. The table below outlines the mapping of these pins to the internal bit processor signals with the mode register settings in GMII interface.

Bit Processor Signal		GMII PHY Pin
RxBit	PhyStatus0	RX_DV
	PhyStatus1	RX_ER
TxBit	PhyControl0	TX_EN
	PhyControl1	TX_ER
TxBit	PhyStatus0	CRS
	PhyStatus1	COL

Table 5 *Bit Processor Signal Mapping*

The CP clusters operatein aggregation mode for GMII. In this mode, all of the 4 RxBit processors of the RX cluster are running four copies of the same piece of code simultaneously. They all process the entire data stream. They delineate each data frame being received and send it up stream to the RxByte processor. Therefore, the 4 RxByte processors in the cluster all see every frame from the data stream. However, there is a hardware token being passed among them in round robin fashion. Only the processor who has the token will process the data frame that's currently being received and send it up stream to the payload memory. Those who don't have the token just discard the frame. After receiving the entire frame, this RxByte processor releases the token, so that the next RxByte processor will have the token and process the next frame. That way, each RxByte processor only receives every 4th frame.

To delineate the data frames, every RxBit processor monitorsthe PhyStatus0 signal that represents RX_DV for valid data. The logic combination of RX_DV*(~RX_ER) is brought in as the 9th data bit in the Small FIFO, so the processor uses the CAM to examine the data and check for an error on every data byte received. This is to ensure data can be received and examined fast enough for wire speed operation on the GMII interface. The RxBit processor also uses the PhyStatus1 that represents RX_ER to confirm error data within a valid frame.

In the TX direction, only one TxBit processor is operating. It takes data frame from one of the 4 TxByte processors and sends it out to the GMII interface with proper preambles and delimiters. There is a hardware token being passed among the 4 TxByte processors in round robin fashion. The TxBit processor can only take data from the large FIFO of the TxByte processor who currently owns the token. Therefore, it transmits a frame from each TxByte processor in round robin fashion. When there is no data frame to be transmitted, the TxBit processor will generate IDLE patterns, and it also ensures a minimum IPG is inserted between frames. Since TX_EN is mapped to the PhyControl0 signal, when the TxBit processor transmits a frame, it asserts the PhyControl0 so that the TX_EN is driven high for the duration of the frame. For every byte it sends out, it can choose to assert or de-assert the TX_EN. The PhyControl1 signal, which maps to TX_ER, can be controlled the same way, so the TxBit processor can assert the TX_ER for any data byte it needs.

CRS and COL, which are mapped to PhyStatus0 and PhyStatus1 respectively, can also be examined by the TxBit processor. They only need to be considered when the interface is in half duplex mode. Currently the GMII microcode provided by C-Port does not support half duplex.

For More Information On This Product,
Go to: www.freescale.com

Freescale Semiconductor, Inc.

Because of the flexibility in microcode programming, the GMII MAC function can be implemented in different ways. The current GMII microcode as shipped with the CST has the following characteristics:

- On TX, Carrier Extend and Carrier Extend Error code are never generated.
- The TX_ER signal is never asserted, so the C-5 will not command the PHY to send illegal symbols on the line on purpose.
- The C-5 will receive data only when the RX_DV is asserted. Therefore, False Carrier Indication, Carrier Extend and Carrier Extend Error conditions are ignored by the microcode. These errors are not counted.
- If during reception of a normal packet (RX_DV is asserted), RX_ER is high for any clock cycle, the packet will be discarded. The packet is counted as one miss-alignment error for now. The code can be changed to report this error to be a more appropriate type.
- Half duplex operation for 1000 or 10/100 Mbps is not supported. Both CRS and COL signals are ignored by the microcode. We have no plans in supporting half duplex at this point in time. If the user is interested in that, please consult with C-Port.
- There are two way to avoid TX underrun: 1) using the WaterMark feature, and 2) using the byte processor to bit processor handshaking mechanism. In 1), the application program will set the Large FIFO Watermark value in the SDP_Mode4 Register. When the number of bytes sent to the Large FIFO exceeds this water mark value, the TxBit processor will start transmitting data out of the Large FIFO. In 2), when the TxByte processor loads up the Large FIFO with enough data bytes, it will set the TxByteToBit flag to tell the TxBit processor to start sending the data. Both solutions work to ensure all the data in a frame will be sent to the GMII continuously. Mechanism 2) is currently used in the microcode provided by C-Port.

Obviously, the microcode can be modified easily to add more features or change its implementation. The user should be aware that any change to the microcode may alter the behavior of the code, hence IEEE standard compliance will not be guaranteed. Note that the microcode C-Port provides will be tested by the InterOperability Lab of University of New Hampshire (test results can be provided upon request).

Freescale Semiconductor, Inc.

For More Information On This Product,
Go to: www.freescale.com

Freescale Semiconductor, Inc.

4 APPENDIX

Shown below are the settings for these registers in the CPs of cluster n when this cluster is used to implement full duplex Gigabit Ethernet port (one GMII interface), where n is 0, 4, 8 or 12.

CP Number	Register Name	Setting for GMII	Setting for MII	Notes
CPn+0	SDP_Mode3	0x00931941	0x00931141	All enable bits for the RxSDP are not set yet.
	SDP_Mode4	0x00000000	0x00000000	Large FIFO Watermark is not used.
	SDP_Mode5	0x10801940	0x10801140	TxBit processor is **enabled** the base CP.
	PIN_Mode	0x005e407d	0x005dc07d	
CPn+1	SDP_Mode3	0x00931941	0x00931141	All enable bits for the RxSDP are not set yet.
	SDP_Mode4	0x00000000	0x00000000	Large FIFO Watermark is not used.
	SDP_Mode5	0x00801940	0x00801140	TxBit processor is **not** enabled.
	PIN_Mode	0x005e407c	0x005dc07c	
CPn+2	SDP_Mode3	0x00931941	0x00931141	All enable bits for the RxSDP are not set yet.
	SDP_Mode4	0x00000000	0x00000000	Only the Large FIFO Watermark bits are set.
	SDP_Mode5	0x00801940	0x00801140	TxBit processor is **not** enabled.
	PIN_Mode	0x001e4000	0x001dc000	
CPn+3	SDP_Mode3	0x00931941	0x00931141	All enable bits for the RxSDP are not set yet.
	SDP_Mode4	0x00000000	0x00000000	Large FIFO Watermark is not used.
	SDP_Mode5	0x00801940	0x00801140	TxBit processor is **not** enabled.
	PIN_Mode	0x001e4000	0x001dc000	

Table 6 *Register Settings for Single Cluster Implementation*

When two clusters (n and m) are used to implement a Gigabit Ethernet port, the setting for the two clusters are shown in the following table, where n and m can be 0, 4, 8 or 12, and n ≠ m. Here cluster n implements RX and cluster m implements TX.

CP Number	Register Name	Setting for GMII	Setting for MII	Notes
CPn+0	SDP_Mode3	0x00931941	0x00931141	All enable bits for the RxSDP are not set yet.

For More Information On This Product,
Go to: www.freescale.com

	SDP_Mode4	Not set	Not set	TxSDP is not used.
	SDP_Mode5	Not set	Not set	TxSDP is not used.
	PIN_Mode	0x005e4000	0x005c0000	
CPn+1	SDP_Mode3	0x00931941	0x00931141	All enable bits for the RxSDP are not set yet.
	SDP_Mode4	Not set	Not set	TxSDP is not used.
	SDP_Mode5	Not set	Not set	TxSDP is not used.
	PIN_Mode	0x005e4000	0x005c0000	
CPn+2	SDP_Mode3	0x00931941	0x00931141	All enable bits for the RxSDP are not set yet.
	SDP_Mode4	Not set	Not set	TxSDP is not used.
	SDP_Mode5	Not set	Not set	TxSDP is not used.
	PIN_Mode	0x001e4000	0x001c0000	
CPn+3	SDP_Mode3	0x00931941	0x00931141	All enable bits for the RxSDP are not set yet.
	SDP_Mode4	Not set	Not set	TxSDP is not used.
	SDP_Mode5	Not set	Not set	TxSDP is not used.
	PIN_Mode	0x001e4000	0x001c0000	
CPm+0	SDP_Mode3	0x00000000	0x00000000	Turn off processors in the RxSDP.
	SDP_Mode4	0x00000000	0x00000000	Large FIFO Watermark is not used.
	SDP_Mode5	0x10801940	0x10801140	TxBit processor is **enabled** the base CP.
	PIN_Mode	0x00801940	0x0055c07d	
CPm+1	SDP_Mode3	0x00000000	0x00000000	Turn off processors in the RxSDP.
	SDP_Mode4	0x00000000	0x00000000	Large FIFO Watermark is not used.
	SDP_Mode5	0x00801940	0x00801140	TxBit processor is **not** enabled.
	PIN_Mode	0x0056407c	0x0055c07c	
CPm+2	SDP_Mode3	0x00000000	0x00000000	Turn off processors in the RxSDP.
	SDP_Mode4	0x00000000	0x00000000	Large FIFO Watermark is not used.
	SDP_Mode5	0x00801940	0x00801140	TxBit processor is **not** enabled.
	PIN_Mode	0x00064000	0x0015c000	
CPm+3	SDP_Mode3	0x00000000	0x00000000	Turn off processors in the RxSDP.
	SDP_Mode4	0x00000000	0x00000000	Large FIFO Watermark is not used.
	SDP_Mode5	0x00801940	0x00801140	TxBit processor is **not** enabled..

For More Information On This Product,
Go to: www.freescale.com

Freescale Semiconductor, Inc.

	PIN_Mode	0x00064000	0x0015c000	

Table 7 *Register Settings for Dual Cluster Implementation*

How to Reach Us:

Home Page:
www.freescale.com

E-mail:
support@freescale.com

USA/Europe or Locations Not Listed:
Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
+1-800-521-6274 or +1-480-768-2130
support@freescale.com

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
support@freescale.com

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800-441-2447 or 303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

Information in this document is provided solely to enable system and software implementers to use Freescale Semiconductor products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.
Freescale Semiconductor reserves the right to make changes without further notice to any products herein. Freescale Semiconductor makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale Semiconductor assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters which may be provided in Freescale Semiconductor data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including “Typicals” must be validated for each customer application by customer’s technical experts. Freescale Semiconductor does not convey any license under its patent rights nor the rights of others. Freescale Semiconductor products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Freescale Semiconductor product could create a situation where personal injury or death may occur. Should Buyer purchase or use Freescale Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify and hold Freescale Semiconductor and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Freescale Semiconductor was negligent regarding the design or manufacture of the part.

Freescale Semiconductor, Inc.