



78Q8430

Software Driver Development Guidelines

**February, 2008
Rev. 1.0**

© 2008 Teridian Semiconductor Corporation. All rights reserved.
Teridian Semiconductor Corporation is a registered trademark of Teridian Semiconductor Corporation.
All other trademarks are the property of their respective owners.

Teridian Semiconductor Corporation makes no warranty for the use of its products, other than expressly contained in the Company's warranty detailed in the Teridian Semiconductor Corporation standard Terms and Conditions. The company assumes no responsibility for any errors which may appear in this document, reserves the right to change devices or specifications detailed herein at any time without notice and does not make any commitment to update the information contained herein. Accordingly, the reader is cautioned to verify that this document is current by comparing it to the latest version on <http://www.teridian.com> or by checking with your sales representative.

Teridian Semiconductor Corp., 6440 Oak Canyon Rd., Suite 100, Irvine, CA 92618
TEL (714) 508-8800, FAX (714) 508-8877, <http://www.Teridian.com>

Table of Contents

1	Introduction	4
2	List of Features	6
3	MAC Operations	8
3.1	Basic Transmit	8
3.2	Basic Receive	9
3.2.1	Hardware Jabber Protection.....	9
3.3	Transmit PAUSE.....	10
3.3.1	Specify PAUSE Frame	10
3.3.2	Specify PAUSE Time	10
3.3.3	PAUSE Transmit Immediately.....	11
3.4	Local PAUSE	11
3.4.1	Disable Local PAUSE.....	11
3.4.2	Enable Local PAUSE	11
3.4.3	Start Local PAUSE Timer.....	12
3.5	HNR Frame Transmission	12
3.5.1	Specify HNR Frame	12
3.5.2	HNR Automatic Transmission	12
3.5.3	HNR Transmit Immediately	13
3.6	Receive Interrupt Delay Timer	13
3.7	Host Drop.....	13
3.8	Append CRC.....	14
3.9	Add Padding	15
3.10	Strip CRC.....	15
3.11	Strip Padding	15
4	Packet Classification	16
4.1	CAM Configuration.....	16
4.2	Address Recognition (ARC).....	16
4.2.1	Wildcard Address Filters	16
4.2.2	Promiscuous Mode Off	17
4.2.3	Promiscuous Mode On.....	17
4.2.4	Unicast Address Filters	17
4.2.5	Multicast Address Filters	19
4.2.6	Negative Address Filters	21
4.3	OnNow Packet Recognition.....	21
4.3.1	Enable OnNow	21
4.3.2	Disable OnNow.....	21
4.3.3	Set the OnNow Pattern	21
4.4	Magic Packet Recognition	22
4.4.1	Enable Magic Packet.....	23
4.4.2	Disable Magic Packet.....	23
4.4.3	Set Magic Packet MAC Address	23
4.5	Classification Interrupts	24
5	Media Controller Offload Features	25
5.1	DMA	25
5.1.1	Slave DMA Read Mode	25
5.1.2	Slave DMA Write Mode	25
5.2	Jumbo Frames	26
5.2.1	Receive Jumbo.....	26
5.2.2	Transmit Jumbo.....	27
5.3	Watermarking.....	28
5.3.1	Headroom Watermark	28
5.3.2	PAUSE Watermark Level	28
5.3.3	Watermark Interrupt	29

5.4	Transfer Frame	30
5.4.1	Transmit ARP Response	30
5.4.2	Transmit ICMP Echo Reply	31
5.5	IP Firewall	35
5.6	IP Checksum Check	35
5.7	Transmit Priority	36
6	Counter Rollover Monitor (RMON)	37
6.1	Read Counters	37
6.2	Clear all Counters	37
6.3	RMON Interrupts	37
7	PHY Procedures	39
7.1	PHY Register Write Access	39
7.2	PHY Register Read Access	39
7.3	PHY Power Down	39
7.4	Auto-negotiation	39
7.4.1	Disable Auto-negotiation	40
7.4.2	Enable Auto-negotiation	40
7.4.3	Restart Auto-negotiation	40
7.5	PHY Speed	41
7.6	PHY Duplex Mode	41
7.7	PHY MDI/MDIX Modes	41
7.8	PHY Link Status Change	42
8	EEPROM Operations	43
8.1	EEPROM Erase	43
8.2	EEPROM Write Access	43
8.3	EEPROM Read Access	43
9	Power Management	44
9.1	Sleep Procedure	44
9.2	Wake Event Procedure	44
10	BIST	45
11	Software Reset	46
12	Related Documentation	47
13	Contact Information	47
	Appendix A – Default CAM Rule Summary	48
	Appendix B – Acronyms	52
	Revision History	54

Tables

Table 1: List of 78Q8430 Features	6
Table 2: Wildcard Address Example	17
Table 3: CAM Rules Associated with Unicast Filter Bytes	17
Table 4: CAM Rules Associated with Multicast Filter Bytes	19
Table 5: ARP Frame Contents	30
Table 6: ICMP Frame Contents	32
Table 7: Auto-negotiation Registers Default Values	40
Table 8: BIST Run Times	45
Table 9: Default CAM Rules	48
Table 10: 78Q8430 Register Acronyms	52

1 Introduction

The 78Q8430 is a 10/100 Fast Ethernet MAC and PHY controller supporting multi-media offload. The device is optimized to enhance throughput and offload network protocol tasks from the host processor for demanding multi-media applications found in Set Top Boxes, IP video, and Broadband Media Appliances.

This document provides guidelines for developing the software driver needed to implement the hardware supported features of the 78Q8430. A description of each feature and the procedure required to exercise the feature is included. The guidelines presented in this document assume a basic knowledge of the 78Q8430 architecture and operation and familiarity with the 78Q8430 register set. Appendix B provides a list of the register acronyms and descriptions.

For more information refer to the *78Q8430 Data Sheet*. For an overview of the 78Q8430 functional blocks refer to Figure 5: Device Block Diagram in the data sheet. Data sheet Figure 13: Internal Digital Block Diagram presents an overview of the functional layers, internal connections and external signals. The “Register Descriptions” section of the data sheet contains a detailed description of the registers.

2 List of Features

Table 1 lists the 78Q8430 hardware supported features described in this document. A brief description of the feature from the software driver point of view is also included.

Table 1: List of 78Q8430 Features

Feature	Description	Page
MAC Operations		8
Basic Transmit	Accept a frame and cause it to be transmitted by the 78Q8430.	8
Basic Receive	Receives a frame from the 78Q8430 and deliver it to the host.	9
Jabber Protection	Enable/disable hardware jabber protection for receive.	9
Local PAUSE	Enable, disable, and start the automatic local pause counter.	11
HNR Frame Transmission	- Specify the HNR frame to transmit when the HNR timer expires.	12
	- Set the HNR Timer for automatic HNR transmission.	12
	- Request that an HNR frame transmit immediately irrespective of the HNR timer.	13
Rx Interrupt Delay Timer	Set the Rx interrupt delay timer.	13
Host Drop	Request that next frame in the receive QUE to be dropped.	13
Strip CRC	Enable hardware removal of the Receive CRC.	15
Strip Padding	Enable/disable the automatic Receive strip padding feature.	15
Packet Classification		16
CAM Configuration		16
ARC (Address Recognition)	- Wildcard address filters.	16
	- Enable/disable promiscuous mode.	17
	- Unicast address filters.	17
	- Multicast address filters.	19
	- Negative address filters.	21
OnNow Packets	- Enable/disable Wake-on-LAN (WOL) for OnNow packet.	21
	- Set the OnNow pattern.	21
Magic Packets	- Enable/disable WOL for Magic Packet™.	22
	- Set Magic Packet MAC address.	23
Classification Interrupt	Specify an action to take when a classification interrupt is received.	24
Media Controller Offload Features		25
DMA	- Use slave DMA Read for receive.	25
	- Use Slave DMA Write for transmit.	25
Jumbo Frames (up to 64 kB)	- Receive jumbo frames.	26
	- Transmit jumbo frames.	27
Watermarking	- Set the headroom watermark level.	28
	- Set the transmit PAUSE watermark level.	28
	- Watermark interrupt: Set the level and handle the interrupt.	29
Transfer Frame	- Transmit ARP response using Snoop access.	30
	- Transmit ICMP echo reply using Snoop access.	31

Feature	Description	Page
IP Firewall	IP address filtering.	35
IP Checksum	Check the receive IP Header checksum.	35
Transmit Priority	Allow higher priority frame transmission.	36
Counter Rollover Monitor (RMON)		37
Read Counters	Read current value of one or more statistics counter.	37
Clear all Counters	Clear all statistics counters instantaneously to zero.	37
RMON Interrupts	Service counter rollover interrupts to provide additional resolution.	37
PHY Operations		39
PHY Register Access	Read/write PHY station management registers.	39
PHY Power Down	Put the PHY into power down mode for low-power operation.	39
Auto-negotiation	Enable, disable, and restart auto-negotiation.	39
PHY Configuration	- Query/set PHY speed. - Query/set PHY duplex mode.	39 41
MDI/MDIX Modes	Enable/disable MDIX.	41
Link Status Change	Update the MAC when there is a PHY link status change interrupt.	42
General		
EEPROM	- EEPROM erase. - EEPROM read/write access.	43 43
Power Management	Sleep and wake procedures	44
BIST	Self diagnostic.	45
Software Reset	When the part is low on memory, check for a memory leak and execute a software reset if needed.	46

3 MAC Operations

This section describes the procedures to implement the following 78Q8430 MAC related operations:

- Basic transmit and receive
- Hardware Jabber protection
- Pause frame transmission with watermarking
- HNR frame transmission
- Receive interrupt delay timer
- Host Drop
- Append and strip CRC
- Strip padding

3.1 Basic Transmit

The basic transmit operation is achieved using the QUE 4 registers. QUE 2 is reserved for PAUSE frame transmission, QUE 3 is reserved for high priority frames and transfer frames, and QUE 5 is reserved for HNR frames. Use the following procedure to add a frame to QUE 4 for basic transmission:

STEP 1: Write to QUE 4 PCWR.

Writing to this register sets the ID and transmission options for the frame and initializes the write logic for QUE 4. The driver should assign a unique ID number to each transmitted frame. When the write logic is initialized, it triggers an under-run interrupt for QUE 4 if the previous frame has not been completed. This has the effect of aborting the previous frame.

STEP 2: Write to QUE 4 PSZR.

Writing to PSZR indicates to the QUE 4 write logic how many bytes to expect. For example, if the part is operating in 32-bit bus mode and the byte count is not an even multiple of four, the write logic uses the size value to determine how many bytes in the last write are valid. If the host attempts to write an extra word past the end of the programmed size, an overrun interrupt results.

STEP 3: Write the frame data to QUE 4 TDR.

Each successive write to TDR adds the data to QUE 4.

STEP 4: Read the transmit status from TPSR.

Reading TPSR reads status words from the top of a 128 word deep FIFO. One status word is added to the FIFO for each transmitted frame. If the status words for transmitted frames are not read from the FIFO, it eventually fills to capacity at which time any new status words are dropped while the FIFO retains the oldest ones.

- ✓ Use of the deferral timeout feature can cause 802.3 non-conformance. To avoid this, the disable deferral timeout bit in the PCWR should always be set.

The software driver should retain the buffer memory containing the transmitted frame until the transmit status for that frame has been received. The status can be associated with the buffer that contains its data using the ID assigned in QUE 4 PCWR, which is also contained in the TPSR status word. Once the TPSR Done bit is set and the transmit status is known to be good the driver can safely de-allocate the buffer memory. In this way, if there was a problem during transmission and the status is not good, the driver can resubmit the frame if needed.

The procedure above assumes that the MAC transmitter is enabled and is not halted and that the MAC has been properly configured via the MCR.

The transmit status for a frame will not be available until the transmission completes. The driver may need to have provisions to recover the status at a later time.

3.2 Basic Receive

The basic receive operation is achieved using the QUE0 registers. Use the following procedure to read a received frame from QUE 0:

STEP 1: Read the receive status from RPSR.

When the RPSR Done bit is set, the entire frame has been received into QUE 0 and the RPSR Count field contains the number of bytes in the frame.

STEP 2: Read the frame data from QUE 0 RDR.

Each successive read from QUE 0 RDR removes data from QUE 0. The driver must be careful to read all the frame data and not to read extra data.

STEP 3: Discard the frame if the status was bad.

The driver can also use the Host Drop feature described in Section 3.7 to drop frames with bad status without reading the data.

The procedure above assumes that the MAC receiver is enabled and is not halted and that the MAC has been properly configured via the MCR.

The Receive Jumbo procedure described in Section 5.2.1 must be used instead of the basic receive procedure if the frames are too large to fit into the receive buffer (Jumbo frames).

✓ The driver must not lose track of which status goes with the next data in QUE 0. If the driver reads QUE 0 RDR after the current frame data is exhausted, it either reads the first data word from the next frame which causes the next frame to be short, or if there is no data in the QUE, it triggers a QUE 0 under-run interrupt. An under-run interrupt on QUE 0 indicates that the driver is confused. In this event issuing a software reset will realign the system.

If the driver is unsure about the length of the current frame being read from QUE 0, the RDSR EOF bit can be used. When this bit is set, the next read from QUE 0 RDR contains the last data in the frame and the RDSR Data Size field indicates how many valid bytes it contains.

3.2.1 Hardware Jabber Protection

If the Hardware Jabber feature is enabled (the MCR Jumbo OK bit is not set) and certain conditions are met, the hardware flags the length error and truncates the frame being received. This prevents an illegally long frame, or jabber frame, from filling the buffer.

Once the classification engine has identified the Len/Typ field, the hardware performs some frame length sanity checking.

- If the frame size exceeds the maximum VLAN allowed size, the frame is truncated at that point.
- Then, if the field is a len and not a type (Len/Type < 0x0600) and the actual frame length exceeds the length specified in the Len/Type field, the frame will be truncated.

The hardware truncates the frame by automatically dropping the bytes of the frame in excess of the maximum allowed length.

Enabling Hardware Jabber Protection

Use the following procedure to enable hardware jabber protection:

STEP 1: Clear the MCR Jumbo OK bit.

Disabling Hardware Jabber Protection

Use the following procedure to disable hardware jabber protection:

STEP 1: Set the MCR Jumbo OK bit.

3.3 Transmit PAUSE

This section describes the procedures used to access the automatic immediate PAUSE transmission feature.

3.3.1 Specify PAUSE Frame

When the PAUSE watermark is reached, the 78Q8430 automatically transmits a PAUSE frame. Section 5.3.2 describes how to set the PAUSE watermark level. Use the following procedure to write the PAUSE frame to QUE 2:

STEP 1: Write to QUE 2 PCWR.

Writing to this register sets the ID and transmission options for the frame and initializes the write logic for QUE 2. The driver should assign a unique ID number to each transmitted frame. When the write logic is initialized, it triggers an under-run interrupt for QUE 2 if the previous frame has not been completed. This has the effect of aborting the previous frame.

STEP 2: Write to QUE 2 PSZR.

Writing to PSZR indicates to the QUE 2 write logic how many bytes to expect. For example, if the part is operating in 32-bit bus mode and the byte count is not an even multiple of four, the write logic uses the size value to determine how many bytes in the last write are valid. If the host attempts to write an extra word past the end of the programmed size, an overrun interrupt results.

STEP 3: Write the frame data to QUE 2 TDR.

Each successive write to TDR adds the data to QUE 2.

STEP 4: Read the transmit status from TPSR.

Reading TPSR reads status words from the top of a 128 word deep FIFO. One status word is added to the FIFO for each transmitted frame. If the software does not read the status words for transmitted frames from the FIFO, it eventually fills to capacity at which time any new status words are dropped while the FIFO retains the oldest ones.

3.3.2 Specify PAUSE Time

It may be more convenient for the application to simply specify the PAUSE time rather than the entire PAUSE frame. Use the following procedure to specify the PAUSE time in milliseconds:

STEP 1: Assemble the following frame in host memory.

Destination Address						Source Address						Len/Typ		PAUSE OPCODE		Time	
												Hi	Lo			Hi	Lo
01	80	C2	00	00	01	??	??	??	??	??	??	88	08	00	01	??	??

STEP 2: Insert the host MAC address in the Source Address field.

STEP 3: Insert the desired 16-bit PAUSE time into the Time field.

The most significant byte is placed in the Hi field and the least significant byte in the Lo field.

STEP 4: Write to QUE 2 PCWR to set the ID and transmission options.

Writing to this register initializes the write logic for QUE 2. When the write logic is initialized, it

triggers an under-run interrupt for QUE 2 if the previous frame has not been completed. This has the effect of aborting the previous frame.

The driver should assign a unique ID number to each transmitted frame. Since the frame does not include padding or an FCS, make sure the PCWR Append CRC bit is set and the PCWR Disable Padding bit is clear.

STEP 5: Write to QUE 2 PSZR.

Writing to PSZR indicates to the QUE 2 write logic how many bytes to expect. For example, if the part is operating in 32-bit bus mode and the byte count is not an even multiple of four, the write logic uses the size value to determine how many bytes in the last write are valid. If the host attempts to write an extra word past the end of the programmed size, an overrun interrupt results.

STEP 6: Write the frame data to QUE 2 TDR.

Each successive write to TDR adds the data to QUE 2.

STEP 7: Read the transmit status from TPSR.

Reading TPSR reads status words from the top of a 128 word deep FIFO. One status word is added to the FIFO for each transmitted frame. If the software does not read the status words for transmitted frames from the FIFO, it eventually fills to capacity at which time any new status words are dropped while the FIFO retains the oldest ones.

3.3.3 PAUSE Transmit Immediately

Use the following procedure to cause the immediate transmission of the PAUSE frame:

STEP 1: Read the value of the QUE 2 QSR.

STEP 2: Set the QDR bit in the QSR value.

This can be done using a bit-wise OR operator.

STEP 3: Write the new value back to the QUE 2 QSR.

This sets the QSR QDR bit for QUE 2 and cause the MAC transmitter to select QUE 2 for transmission of the next frame.



The driver should not set the QSR QDR bit for QUE 2 if no frame has been loaded into QUE 2.

3.4 Local PAUSE

This section describes the procedures used for the automatic local PAUSE timer feature.

3.4.1 Disable Local PAUSE

The default CAM rule set will not activate the local PAUSE timer if the destination MAC address on the received PAUSE frame is not correct. Disabling the PAUSE MAC address recognition in multicast filter #1 therefore disables the local PAUSE feature. Use the following procedure to disable local PAUSE:

STEP 1: Disable multicast filter #1 in the CAM.

Modify CAM rule 0x7D (multicast filter #1 byte [0]) to disable the filter.

- Write 0x7D to the CAR Address field.
- Set the RMR Previous Hit Mask field to 0x00.

3.4.2 Enable Local PAUSE

Enabling multicast filter #1 enables the classification engine to automatically identify received PAUSE frames and set the local PAUSE timer. Use the following procedure to enable local PAUSE:

STEP 1: Enable multicast filter #1 in the CAM.

Modify CAM rule 0x7D (multicast filter #1 byte [0]) to enable the filter.

- Write 0x7D to the CAR Address field.
- Set the RMR Previous Hit Mask field to 0x7F.

3.4.3 Start Local PAUSE Timer

Use the following procedure to start the local PAUSE timer immediately:

STEP 1: Write to PDCR to activate the local PAUSE condition.

Set the PDCR Start bit to start the PAUSE timer. Set the desired local PAUSE duration in the PDCR Pause field at the same time.

3.5 HNR Frame Transmission

When the host fails to acknowledge a Wake-On-LAN (WOL) event before the specified timer expires, the 78Q8430 can be configured to automatically transmit an HNR frame. The device may also be commanded to transmit an HNR frame immediately. This section describes the procedures for HNR frame transmission.

3.5.1 Specify HNR Frame

Use the following procedure to write the desired HNR frame to QUE 5:

STEP 1: Write to QUE 5 PCWR.

Writing to this register sets the ID and transmission options for the frame and initializes the write logic for QUE 5. The driver should assign a unique ID number to each transmitted frame. When the write logic is initialized, it triggers an under-run interrupt for QUE 5 if the previous frame has not been completed. This has the effect of aborting the previous frame.

STEP 2: Write to QUE 5 PSZR.

Writing to PSZR indicates to the QUE 5 write logic how many bytes to expect. For example, if the part is operating in 32-bit bus mode and the byte count is not an even multiple of four, the write logic uses the size value to determine how many bytes in the last write are valid. If the host attempts to write an extra word past the end of the programmed size, an overrun interrupt results.

STEP 3: Write the frame data to QUE 5 TDR.

Each successive write to TDR adds the data to QUE 5.

STEP 4: Read the transmit status from TPSR.

Reading TPSR reads status words from the top of a 128 word deep FIFO. One status word is added to the FIFO for each transmitted frame. If the software does not read the status words for transmitted frames from the FIFO, it eventually fills to capacity at which time any new status words are dropped while the FIFO retains the oldest ones.

3.5.2 HNR Automatic Transmission

The Host Not Responding (HNR) frame is sent automatically in the event that the host fails to respond to a remote request for it to wake. The HNR Timer is used for this purpose. After the classification identifies a wake event, the HNR Timer is started. If the host does not respond to the wake event by clearing the Power Management Event (PME) bit in the Power Management Control and Status Register (PMCSR) before the HNR Timer expires, the HNR frame is automatically transmitted.

The duration of the HNR Timer should be selected with the speed of the host processor and the interrupt latency of the system in mind such that the host has adequate time in the worst case to receive and process the wake interrupt and clear the PME bit.

The following procedure is used to set the HNR Timer duration:

STEP 1: Write the desired HNR Timer duration to the HNRCCR.

The value written is the timer duration measured in system cycles. As an example, if the system clock is 100 MHz, then the timer is in multiples of 10 nanoseconds. In this case, the maximum timer value of 2^{32} is almost 43 seconds.

3.5.3 HNR Transmit Immediately

Use the following procedure to cause the immediate transmission of the HNR frame:

STEP 1: Read the value of the QUE 5 QSR.

STEP 2: Set the QDR bit in the QUE 5 QSR value.

This can be done using a bit-wise OR operator.

STEP 3: Write the new value back to the QUE 5 QSR.

This sets the QUE 5 QDR bit and cause the MAC transmitter to select QUE 5 for transmission of the next frame as long as no other transmit QUEs have their QDR bit set. In other words, the HNR frame transmits with the lowest priority.



The driver should not set the QDR bit for QUE 5 if no frame has been loaded into QUE 5.

3.6 Receive Interrupt Delay Timer

The early receive interrupt has a delay timer feature. This feature is intended to leverage the deep receive buffer to decrease interrupt handling overhead in the host. Normally, the early receive interrupt is triggered as soon as any data for a received frame is placed into the receive QUE. The receive interrupt delay timer delays this interrupt for a programmable amount of time to allow the receive QUE to accumulate more data. In this way, under conditions of heavy load, several frames can be serviced by a single receive interrupt.

The interrupt timer is linked to the PHY speed such that the timer value is measured in byte times, or in other words, a single tick on the interrupt delay timer is equal to the amount of time it would take the PHY to receive a single byte. The timer does not require that an actual byte be received so the interrupt delay feature will not cause small frames to be left in the QUE while waiting for more data. Anytime data is added to the receive QUE and the interrupt delay timer is enabled, the timer is started if it is not already running. If the interrupt delay timer is already running when data is added to the receive QUE, the running timer is not affected. When a BLOCK is removed from the receive QUE, the interrupt delay timer is reset. This means that the driver must completely empty the receive QUE each time it services the delay timer interrupt or risk stranding data in the QUE.

If the receive QUE is empty when the interrupt timer is initially started, the maximum number of bytes that can be in the receive QUE when the interrupt is triggered is the delay value plus 252. The driver should therefore set the timer value based on the maximum amount of data it wants to accumulate in the QUE before servicing it.

Use the following procedure to set the interrupt delay timer value:

STEP 1: Write the desired timer delay value to the IDCR.

The timer value is a 15-bit value.

3.7 Host Drop

The host drop feature is used by the driver to quickly remove the top frame from the receive QUE without having to use the basic receive frame procedure in Section 3.2. This method for dropping frames is very efficient. A frame of 1522 bytes can be dropped from the QUE in about 12 clocks.

✓ The host drop feature must never be used to drop a frame that is 4 bytes or less in size. When a frame contains 4 bytes or less then the QUE pre-fetch operation loads the entire frame such that it is no longer at the top of the QUE. In this case it is more efficient to simply read and discard the entire frame with a single read to the RDR.

Use the following procedure to entirely remove the top frame from the receive QUE:

STEP 1: Inspect the frame size.

If the RPSR Count field indicates that the frame is 4 bytes or less in total size then the host drop feature is not appropriate. In this case a single read to the RDR is sufficient to remove the frame from the QUE. If the frame size is 5 bytes or greater then proceed to STEP 2.

STEP 2: Set the FDR Drop Rx Frame bit.

Setting this bit starts the drop circuit causing the receive QUE to purge one entire frame.

STEP 3: Wait for the drop circuit to complete.

It is important that the driver not attempt to read from QUE 0 while the drop circuit is running. The Drop Rx Frame bit is self clearing. When the drop operation is complete the bit will be clear.

Any frames that are in the QUE before the frame that is to be dropped must be completely removed from the QUE before this procedure is initiated. In other words, the frame to be dropped must be at the very top of the QUE.

✓ There must be absolutely no activity on the QUE 0 RDR while the host drop procedure is active.

The host drop circuit does not effect the transmit QUEs. A driver may find that a more efficient use of its time may be to service a transmit QUE rather than wait idly in a polling loop for the drop circuit to complete.

3.8 Append CRC

The transmit circuit has the ability to calculate and append the correct CRC to the end of a transmit frame on the fly. Using this feature saves the host some compute cycles in not having to calculate the CRC itself as well as bus cycles in not having to send the CRC with the frame data. The basic transmit procedure described in Section 3.1 can be modified to use the hardware to append the CRC to transmitted frames as follows.

STEP 1: Write to QUE 4 PCWR.

Make sure the Append CRC bit is set in the value written to QUE 4 PCWR.

STEP 2: Write to QUE 4 PSZR.

The size written to PSZR should NOT include the CRC because the CRC will not be part of the frame transferred from the host.

STEP 3: Write the frame data to QUE 4 TDR.

Do not write the CRC bytes as part of the frame data.

STEP 4: Read the transmit status from TPSR.

Reading TPSR reads status words from the top of a 128 word deep FIFO. One status word is added to the FIFO for each transmitted frame. If the software does not read the status words for transmitted frames from the FIFO, it eventually fills to capacity at which time any new status words are dropped while the FIFO retains the oldest ones.

3.9 Add Padding

The transmit circuit has the ability to add padding to transmitted frames. The append CRC feature should always be used when the add padding feature is used. The basic transmit procedure described in Section 3.1 can be modified to use the hardware to add padding and append the CRC to transmitted frames as follows.

STEP 1: Write to QUE 4 PCWR.

Make sure the Disable Padding bit is clear and the Append CRC bit is set in the value written to QUE 4 PCWR.

STEP 2: Write to QUE 4 PSZR.

The size written to PSZR should NOT include the CRC because the CRC will not be part of the frame transferred from the host.

STEP 3: Write the frame data to QUE 4 TDR.

Do not write the CRC bytes as part of the frame data.

STEP 4: Read the transmit status from TPSR.

Reading TPSR reads status words from the top of a 128 word deep FIFO. One status word is added to the FIFO for each transmitted frame. If the status words for transmitted frames are not read from the FIFO, it eventually fills to capacity at which time any new status words are dropped while the FIFO retains the oldest ones.

To disable the Add Padding feature, make sure the PCWR Disable Padding bit is set during the transmit procedure.

3.10 Strip CRC

The receive circuit has the ability to verify and remove the CRC from the end of a receive frame on the fly. Using this feature saves the host some compute cycles in not having to calculate the CRC itself as well as bus cycles in not having to retrieve the CRC with the frame data. Setting the MCR No Rx CRC bit causes the CRC to be removed from all received frames. When the Strip CRC feature is used, the basic receive procedure described in Section 3.2 can still be used to receive the frame. However, the driver must be sure to discard frames in STEP 3 when the status indicates a bad CRC.

3.11 Strip Padding

The receive circuit has the ability to use the Len/Typ field of the Ethernet frame to determine the presence of padding at the end of a frame and remove it. Frames that have the padding removed will also have the CRC removed. To maintain a consistent behavior across all frames, the Strip CRC feature must always be used in conjunction with the Strip Padding feature. Setting the MCR No Rx CRC and No Rx PAD bits causes the CRC and the padding, if any, to be removed from all frames. When this feature is used, the basic receive procedure described in Section 3.2 is still used to receive the frame. However, the driver must be sure to discard frames in STEP 3 when the status indicates a bad CRC.

4 Packet Classification

The 78Q8430 packet classification engine consists of a content addressable memory (CAM) linked to control logic (WCS) which provides extensive packet classification features. Refer to Packet Classification in the *78Q8430 Data Sheet* for more details on the packet classification features and hardware support.

This section includes a summary of the CAM configuration and describes the procedures to implement the following packet classification features:

- Address Recognition
 - Wildcard address filters
 - Promiscuous mode
 - Unicast address filters
 - Multicast address filters
 - Negative address filters
- OnNow packet recognition
- Magic Packet recognition
- Classification Interrupts

4.1 CAM Configuration

The CAM rules are accessed indirectly one at a time via the CAM Address Register (CAR). The contents of the rule whose number is indicated by the CAR are available for read and write via the Rule Match Register (RMR) and the Rule Control Register (RCR). The RMR contains a template that the CAM reference word must match in order to trigger the rule, and the RCR contains the control word passed to the control logic when the rule is triggered.

The “Configuring the CAM” section in the *78Q8430 Data Sheet* provides a complete description of the CAM registers.

4.2 Address Recognition (ARC)

The 78Q8430 CAM is loaded with a set of default rules on reset. Some of these rules are intended as a template to provide ARC functionality. They support four multicast and eight unicast address filters. The last unicast ARC rule is configured by default as a promiscuous mode rule such that the ARC is in promiscuous mode right out of reset. [Appendix A](#) summarizes the default CAM rules.

4.2.1 Wildcard Address Filters

Use the Data Mask field in the RMR register to implement wildcard address filters. Each bit in the Data Mask field masks the corresponding bit in the Data Match field. A mask bit is set to require that the corresponding data bit match a specified condition or is cleared to indicate that the data bit is ignored (wildcard bit).

Any unicast or multicast filter can be implemented with wildcards. For example, the promiscuous mode filter (Section 4.2.3) has all address bits masked as wildcard bits so that any address will match. Wildcard addresses can also be specified for the Magic Packet filter.

Table 2 illustrates how to configure the Data Mask and Data Match values for a wildcard address of 00:0010:4x:xx:xx. (x indicates a wildcard field)

Table 2: Wildcard Address Example

	Byte [5]	Byte [4]	Byte [3]	Byte [2]	Byte [1]	Byte [0]
Data Mask Value	0xFF	0xFF	0xFF	0xF0	0x00	0x00
Data Match Value	0x00	0x00	0x10	0x40	0x00	0x00

4.2.2 Promiscuous Mode Off

In promiscuous mode the 78Q8430 passes all addresses on to the host driver. This means that, so long as promiscuous mode is enabled, only negative address filters are effective. Therefore, promiscuous mode must be disabled before any positive address filtering can occur. Use the following procedure to deactivate promiscuous mode:

STEP 1: Change the NEXT field of rule 0x30 from MD to DROP.

This causes any frame with an address that does not match any other address filter to be dropped.

- Set the CAR ADDR field to 0x30.
- Change the RCR Match Control field from MD to DROP.

4.2.3 Promiscuous Mode On

Use the following procedure to activate promiscuous mode:

STEP 1: Change the NEXT field of rule 0x30 from DROP to MD.

This causes all frames to pass the address filters.

- Set the CAR ADDR field to 0x30.
- Change the RCR Match Control field from DROP to MD.

The promiscuous mode filter is a filter with all address bits masked as wildcard bits such that any address will match. Turning promiscuous mode on and off is the same as changing the promiscuous mode filter from a positive filter to a negative filter.

4.2.4 Unicast Address Filters

The default CAM rule set supports a total of eight unicast address filters. The first, unicast filter #0, is the promiscuous mode filter and is reserved for that purpose. The remaining seven are general use filters. Each filter has two components, the 48-bit address that it matches and a mask that defines which bits of the address are relevant and which bits are wildcards. Each byte of each address filter has a rule assigned to it. Table 3 summarizes the association of unicast filter bytes and CAM rules.

Table 3: CAM Rules Associated with Unicast Filter Bytes

	Byte [0]	Byte [1]	Byte [2]	Byte [3]	Byte [4]	Byte [5]
Filter #7	0x77	0x6F	0x5F	0x57	0x47	0x37
Filter #6	0x76	0x6E	0x5E	0x56	0x46	0x36
Filter #5	0x75	0x6D	0x5D	0x55	0x45	0x35
Filter #4	0x74	0x6C	0x5C	0x54	0x44	0x34
Filter #3	0x73	0x6B	0x5B	0x53	0x43	0x33
Filter #2	0x72	0x6A	0x5A	0x52	0x42	0x32
Filter #1	0x71	0x69	0x59	0x51	0x41	0x31

Bytes are in network transmit order starting with byte [0].

For an arbitrary unicast filter number N , use the following procedure to set the address and mask values.

STEP 1: Write address and mask byte [0] to the CAM.

Write CAM rule $0x70+N$ as follows:

Reg.	Field	Value to write
CAR	ADDR	0x70+N
RMR	Data Match	Value of MAC address byte [0]
	Data Mask	Value of mask byte [0] from the Wild Card setting (0xFF is for a perfect match)
	Previous Hit Match	0x00 to disable the filter
	Previous Hit Mask	0x00
RCR	Byte Offset	Retain default: 0x00
	Interrupt	Retain default: 0
	Control Logic Action	Retain default: NOP
	Match Control	Retain default: MD

STEP 2: Write address and mask byte [1] through byte [4] to the CAM.

For each byte, the CAM rule indicated by Table 3 based on the filter number, N and byte number is written as follows:

Reg.	Field	Value to write
CAR	ADDR	byte [1]: 0x68+N byte [2]: 0x58+N byte [3]: 0x50+N byte [4]: 0x40+N
RMR	Data Match	Value of MAC address byte [1] ... byte [4]
	Data Mask	Value of mask byte [1] . . . byte [4]
	Previous Hit Match	Value of the CAM rule used by the previous byte
	Previous Hit Mask	0x7F
RCR	Byte Offset	Retain default: 0x00
	Interrupt	Retain default: 0
	Control Logic Action	Retain default: NOP
	Match Control	Retain default: MD

Unlike the settings for byte [0], the RMR Previous Hit Mask field is set to 0x7F and the RMR Previous Hit Match field is always set to the value of the CAM rule used by the previous byte. For example, the Previous Hit Mask fields for filter #1 would be 0x71, 0x69, 0x59 and 0x51, for byte [1] through byte [4] respectively.

STEP 3: Write address and mask byte [5] to the CAM.

Write CAM rule 0x30+N as follows:

Reg.	Field	Value to write
CAR	ADDR	0x30+N
RMR	Data Match	Value of MAC address byte [5]
	Data Mask	Value of mask byte [5]
	Previous Hit Match	Set to the CAM rule that was used for byte [4] (0x40+N).
	Previous Hit Mask	0x7F
RCR	Byte Offset	Retain default: 0x00
	Interrupt	Retain default: 0
	Control Logic Action	Set to TAX
	Match Control	Retain default: MD

STEP 4: Enable the filter.

Enable the unicast address filter N by modifying the CAM rule for byte [0]:

- Set the CAR ADDR field to $0x70+N$.
- Set the RMR Previous Hit Mask field to $0x7F$.

This step must be done last to prevent an arriving frame from matching a partial set of filter rules. All the rules for a filter must be in place before the first rule, and then the filter itself, is enabled.

An address filter can be simply activated and deactivated by toggling the value of the RMR Previous Hit Mask field for the associated byte [0] CAM rule between $0x7F$ and $0x00$ respectively. Filter #0, the promiscuous mode filter, should not be deactivated in this way.

- ✓ It is important that STEP 1 deactivate the filter (by setting the RMR Previous Hit Mask field to $0x00$) so that no frames are filtered using a partial filter setting before all relevant rules are written. STEP 4 reactivates the filter once the new settings are in place.

4.2.5 Multicast Address Filters

The default CAM rule set supports a total of four multicast address filters. The first, multicast filter #0, serves the same purpose as the promiscuous mode filter except it applies only to multicast addresses. The last, multicast filter #3, is the broadcast filter and is reserved for that purpose. The second, multicast filter #1, is the PAUSE filter and is set up by default to match the PAUSE address. Multicast filter #2 is unused by default and available for general use. Each filter has two components, the 48-bit address that it matches and a mask that defines which bits of the address are relevant and which bits are wildcards. Each byte of each address filter has a CAM rule assigned to it. Table 4 summarizes the association of multicast filter bytes and CAM rules.

Table 4: CAM Rules Associated with Multicast Filter Bytes

	Byte [0]	Byte [1]	Byte [2]	Byte [3]	Byte [4]	Byte [5]
Filter #3	0x7F	0x67	0x63	0x4F	0x4B	0x3F
Filter #2	0x7E	0x66	0x62	0x4E	0x4A	0x3E
Filter #1	0x7D	0x65	0x61	0x4D	0x49	0x3D

Bytes are in network transmit order starting with byte [0].

For an arbitrary multicast filter number N , use the following procedure to set the address and mask values:

STEP 1: Write address and mask byte [0] to the CAM.

CAM rule $0x7C+N$ is written as follows:

Reg.	Field	Value to write
CAR	ADDR	$0x7C+N$
RMR	Data Match	Value of MAC address byte [0]
	Data Mask	Value of mask byte [0]
	Previous Hit Match	$0x00$ to disable the filter
	Previous Hit Mask	$0x00$
RCR	Byte Offset	Retain default: $0x00$
	Interrupt	Retain default: 0
	Control Logic Action	SETMC
	Match Control	Retain default: MD

The LSB of both address byte [0] and mask byte [0] must be non-zero for multicast address filters. If the LSBs are zero, the address is not multicast and belongs in the unicast filter set.

STEP 2: Write address and mask byte [1] through byte [4] to the CAM.

For each byte, write the CAM rule indicated by Table 4 based on the filter number and byte number as follows:

Reg.	Field	Value to write			
CAR	ADDR	byte [1]: 0x64+N	byte [2]: 0x60+N	byte [3]: 0x4C+N	byte [4]: 0x48+N
RMR	Data Match	Value of MAC address byte [1] ... byte [4]			
	Data Mask	Value of mask byte [1] . . . byte [4]			
	Previous Hit Match	Value of the CAM rule used by the previous byte ¹			
	Previous Hit Mask	0x7F			
RCR	Byte Offset	Retain default: 0x00			
	Interrupt	Retain default: 0			
	Control Logic Action	Retain default: NOP ²			
	Match Control	Retain default: MD			

¹ For example, the Previous Hit fields for filter #1 would be 0x7D, 0x65, 0x61 and 0x4D, for byte [1] through byte [4] respectively.

² EXCEPTION: multicast filter #3, byte [4] should have the Action field set to TAX.

STEP 3: Write address and mask byte [5] to the CAM.

Write CAM rule 0x3C+N as follows:

Reg.	Field	Value to write
CAR	ADDR	0x3C+N
RMR	Data Match	Value of MAC address byte [5]
	Data Mask	Value of mask byte [5]
	Previous Hit Match	Set to the CAM rule that was used for byte [4] (0x48+N).
	Previous Hit Mask	0x7F
RCR	Byte Offset	Retain default: 0x00
	Interrupt	Retain default: 0
	Control Logic Action	Set to TAX ¹
	Match Control	Retain default: MD

¹ EXCEPTION: multicast filter #3, byte [5] should have the Action field set to SETBC.


STEP 4: Enable the filter.

Enable the multicast address filter *N* by modifying the CAM rule for byte [0]:

- Set the CAR ADDR field to 0x7C+N.
- Set the RMR Previous Hit Mask field to 0x7F.

This step must be done last to prevent an incoming frame from matching a partial set of filter rules. All the rules for a filter must be in place before the first rule, and therefore the filter itself, is enabled.

A multicast address filter can be simply activated and deactivated by toggling the value of the RMR Previous Hit Mask field for the byte [0] CAM rule between 0x7F and 0x00 respectively. Multicast filter #0 should not be deactivated in this way.

 It is important to deactivate the filter in STEP 1 (by setting the Previous Hit Mask field to 0x00) so that no frames are filtered using a partial filter setting before all relevant rules are written. STEP 4 reactivates the filter once the new settings are in place.

4.2.6 Negative Address Filters

Any address filter, either multicast or unicast, can be set as either a positive or negative filter. A positive filter is a filter that passes frames with a source MAC address that matches the filter. A negative filter is a filter that blocks frames with a source MAC address that matches the filter. By default, all filters are positive acting. Use the following procedure to change a filter to negative action:

STEP 1: Change the Match Control field for the CAM rule for byte [5] from MD to DROP.

- Set the CAR Address field to the address for the CAM rule byte [5].
- Change the RCR Match Control field from MD to DROP.

To change a filter back to a positive acting filter, change the same Match Control field back to MD.

4.3 OnNow Packet Recognition

Identifying an OnNow frame is based on matching a sequence of bytes, some of which may be masked off. The CAM contains six rules (rules 0x15 down through 0x10) reserved for OnNow pattern matching such that it can match a sequence of up to 6 bytes. The OnNow rules are set up such that they will start matching in the IP payload section of a frame that has already passed the address filters and IP header checks. If rule 0x15 is disabled, the CAM no longer identifies the specified OnNow pattern.

4.3.1 Enable OnNow

Use the following procedure to enable the OnNow feature:

STEP 1: Modify CAM rule 0x15 to enable the filter.

- Set the CAR Address field to 0x15.
- Set the RMR Previous Hit Mask field to 0x7F.

4.3.2 Disable OnNow

Use the following procedure to disable the OnNow feature:

STEP 1: Disable CAM rule 0x15.

- Set the CAR Address field to 0x15.
- Set the RMR Previous Hit Mask field to 0x00.

4.3.3 Set the OnNow Pattern

Use the following procedure to set the OnNow pattern:

STEP 1: Write byte [0] and mask to the CAM.

Write CAM rule 0x15 as follows:

Reg.	Field	Value to write
CAR	ADDR	0x15
RMR	Data Match	Pattern byte [0]
	Data Mask	0x00 to mask, else 0xFF
	Previous Hit Match	Retain default: 0x23
	Previous Hit Mask	0x00
RCR	Byte Offset	Retain default: 0x00
	Interrupt	Retain default: 0
	Control Logic Action	Retain default: NOP
	Match Control	Retain default: MD

STEP 2: Write pattern and mask byte [1] through byte [4] to the CAM.

For each byte, CAM rule 0x15-*N*, where *N* is the byte number, is written as follows:

Reg.	Field	Value to write			
CAR	ADDR	byte [1]: 0x14	byte [2]: 0x13	byte [3]: 0x12	byte [4]: 0x11
RMR	Data Match	Pattern byte [1] ... byte [4]			
	Data Mask	0x00 to mask, else 0xFF			
	Previous Hit Match	Value of the CAM rule used by the previous byte ¹			
	Previous Hit Mask	0x7F			
RCR	Byte Offset	Retain default: 0x00			
	Interrupt	Retain default: 0			
	Control Logic Action	Retain default: NOP			
	Match Control	Retain default: MD			

¹ The Previous Hit Match fields for byte [1] through byte [4] are 0x15, 0x14, 0x13 and 0x12.

STEP 3: Write address and mask byte [5] to the CAM.

Write CAM rule 0x10 as follows.

Reg.	Field	Value to write
CAR	ADDR	0x10
RMR	Data Match	Pattern byte [5]
	Data Mask	0x00 to mask, else 0xFF
	Previous Hit Match	Set to the CAM rule that was used for byte [4] (0x11).
	Previous Hit Mask	0x7F
RCR	Byte Offset	Retain default: 0x00
	Interrupt	Retain default: 0
	Control Logic Action	Retain default: WAKE
	Match Control	Retain default: DONE

STEP 4: Enable OnNow.

STEP 1 of this procedure disabled the OnNow feature. Modify CAM rule 0x15 to enable the filter.

- Set the CAR Address field to 0x15.
- Set the RMR Previous Hit Mask field to 0x7F.



It is important that STEP 1 disable the OnNow feature while the pattern is being set so that no frames are matched against a partial pattern before the entire pattern is set.

The existing rules will only match the OnNow pattern starting with the first byte of the IP payload. In some cases it may be desirable for the OnNow pattern to match an ARP request or other standard frame. In this case, it may be possible to leverage other rules to accomplish this with only a few rules. This cannot be done by the procedure specified here.

4.4 Magic Packet Recognition

Identifying a Magic Packet frame is based on matching a specific sequence of bytes based on the MAC address of the target. CAM rules 0x0F down through 0x02 are reserved for Magic Packet recognition. Rules 0x0F, 0x0E and 0x0D can all be entry points for the Magic Packet rules. If rules 0x0F, 0x0E and 0x0D are all disabled, the CAM no longer identifies Magic Packets.

4.4.1 Enable Magic Packet

Use the following procedure to enable the Magic Packet feature:

STEP 1: Enable CAM rule 0x0F.

- Set the CAR Address field to 0x0F.
- Set the RMR Previous Hit Mask field to 0x7F.

STEP 2: Enable CAM rule 0x0E.

- Set the CAR Address field to 0x0E.
- Set the RMR Previous Hit Mask field to 0x7F.

STEP 3: Enable CAM rule 0x0D.

- Set the CAR Address field to 0x0D.
- Set the RMR Previous Hit Mask field to 0x7F.

4.4.2 Disable Magic Packet

Use the following procedure to disable the Magic Packet feature:

STEP 1: Disable CAM rule 0x0F.

- Set the CAR Address field to 0x0F.
- Set the RMR Previous Hit Mask field to 0x00.

STEP 2: Disable CAM rule 0x0E.

- Set the CAR Address field to 0x0E.
- Set the RMR Previous Hit Mask field to 0x00.

STEP 3: Disable CAM rule 0x0D.

- Set the CAR Address field to 0x0D.
- Set the RMR Previous Hit Mask field to 0x00.

4.4.3 Set Magic Packet MAC Address

Use the following procedure to set the MAC address that the Magic Packet pattern must contain:

STEP 1: Disable the Magic Packet feature.

Execute procedure 4.4.2 to disable the CAM from identifying Magic Packets. This prevents a frame from incorrectly matching a partial MAC address while it is being set.

STEP 2: Write MAC address byte [0] to the both CAM rule 0x09 and rule 0x08.

Byte [0] is the only byte that is matched in two rules.

- Set the CAR ADDR field to 0x09.
- Replace the RMR Data Match field with MAC address byte [0].
- Set the CAR ADDR field to 0x08.
- Replace the RMR Data Match field with MAC address byte [0].

STEP 3: Write MAC address byte [1] through byte [5] to the CAM.

Replace the Data Match field of CAM rule 0x08-N with MAC address byte [N], for N=1 to N=5. Complete the following 2 writes for N=1 to N=5.

- Set the CAR ADDR field to 0x08-N.
- Set the RMR Data Match field to MAC addresses byte [N].

STEP 4: Enable the Magic Packet feature.

Execute procedure 4.4.1 to enable the CAM to identify Magic Packets with the specified MAC address.

4.5 Classification Interrupts

The classification engine has an interrupt feature. The interrupt can be programmed to identify certain types of frames or payloads and trigger an interrupt in the event that such a frame or payload is received. The implementation of this feature is highly application dependent and there is no specified procedure at this time. In order for an application to leverage this feature, the driver must provide a way for the application to set its own CAM rules, possibly with the interrupt bit set, and register some type of call back routine with the interrupt service routine so that the application is notified when the classification interrupt is received.

5 Media Controller Offload Features

The 78Q8430 has several features that support offloading some IP work from the host processor. This section describes the following media controller offload procedures:

- Slave DMA Read and Write
- Jumbo frame transmit and receive
- Watermarking
 - Headroom Watermark
 - PAUSE watermark
 - Watermark Interrupt
- Transfer Frame using SNOOP access
 - Transmit ARP response
 - Transmit ICMP echo reply
- IP Firewall
- IP Checksum Check
- Transmit priority

5.1 DMA

The use of a DMA engine in the host to deliver data to the 78Q8430 is desirable. This allows the transfer of data to the part with minimal involvement of the central processor. The implementation of a DMA engine in the host is platform specific. This section describes the 78Q8430 DMA read and write mode operations.

5.1.1 Slave DMA Read Mode

The 78Q8430 slave DMA mode facilitates the use of various host-side utilities by the driver to read large blocks of data with improved IO efficiency. Use the following procedure to read a block of data using the slave DMA read mode:

STEP 1: Set the Read Mode bit and Address field in the DMA register.

The Address field should contain the address of the RDR register for the QUE to be read.

STEP 2: Read the desired amount of data.

Once the slave DMA Read Mode bit is set, all reads on the host interface, regardless of the address used, read from the programmed RDR indicated by the Address field of the DMA register. The DMA address is auto-incremented when the host makes successive reads to the RDR.

STEP 3: Write to the DMA register and clear the Read Mode bit to terminate DMA Read Mode.

The Read Mode bit is self-clearing and any write operation on the host interface bus while in slave DMA read mode terminates DMA mode and clears the bit.

✓ Care must be taken while the 78Q8430 is in DMA mode. If an interrupt service routine is entered while in DMA mode, the read/write bus operations executed by the interrupt routine may not have the desired effect. The driver software should consider using a semaphore to the ISR to indicate the DMA mode to the ISR, or blocking all interrupts, possibly at the HIMR, before entering DMA mode.

5.1.2 Slave DMA Write Mode

Use the following procedure to write a block of data using the slave DMA write mode:

STEP 1: Set the Write Mode bit and Address field in the DMA register.

The Address field should contain the address of the TDR register for the QUE to be written.

STEP 2: Write the desired amount of data.

Once the slave DMA Write Mode is set, all writes on the host interface, regardless of the address, write to the programmed TDR indicated by the Address field of the DMA register. The DMA address is auto-incremented when the host makes successive writes to the RDR.

STEP 3: Read the DMA register to clear the Write Mode bit and terminate DMA Write Mode.

The Write Mode bit is self-clearing and any read operations on the host interface bus while in slave DMA Write Mode clear the bit and terminate the DMA mode.

✓ Care must be taken while the 78Q8430 is in DMA mode. If an interrupt service routine is entered while in DMA mode, the read/write bus operations executed by the interrupt routine may not have the desired effect. The driver software should consider using a semaphore to the ISR to indicate the DMA mode to the ISR, or blocking all interrupts, possibly at the HIMR, before entering DMA mode.

5.2 Jumbo Frames

Normally frames in excess of the maximum size allowed by 802.3 are flagged as bad. The Jumbo Frame feature allows these frames to be transmitted and received. This section describes the procedures for transmitting and receiving jumbo frames.

5.2.1 Receive Jumbo

Use the following procedure to receive frames that are too large to fit into QUE 0 all at once.

STEP 1: Read the status from RPSR.

If the RPSR Done bit is set, the entire frame is already in the receive QUE and the basic receive procedure 3.2 can be used to receive the frame.

If the RPSR Done bit is not set, follow STEPs 2 through 8 to receive the jumbo frame.

STEP 2: Allocate a memory buffer large enough to hold a maximum jumbo frame.

Since the final frame size is unknown, assume the maximum size so that the buffer will be large enough for the entire frame. As an alternative, the driver may use any scatter/gather algorithm to allow it to allocate multiple buffers each of the correct size for the amount of data available. Implementation of this may depend on what the IP stack supports.

STEP 3: Read all available data from QUE 0.

Each successive read from the QUE 0 RDR removes data from QUE 0. Take care not to read extra data. The driver must track the current position in the buffer so that when additional data is available it is appended to the buffer at the correct location.

STEP 4: Reread the status from RPSR.

This procedure does not specify whether reading the RPSR is polling or interrupt driven. Whatever scheme is used should be engineered to ensure that the receive buffer does not overflow between reads of RPSR.

STEP 5: Examine the status bits in the new RPSR value.

If a fatal error has occurred in the frame since the last read of the RPSR, the Host Drop procedure, described in Section 3.7, can be used to discard the rest of the frame in the receive QUE and free the buffer. This avoids unnecessarily reading data for a bad frame from the receive QUE.

STEP 6: Examine the new frame size (RPSR Count field) value.

If the new frame size is less than the amount of data already read into the buffer than the size field has rolled over and the frame should be treated as a jabber frame and discarded.

STEP 7: Read all available new data from QUE 0.

The amount of new data in the QUE is the new frame size (Count field) from the RPSR value minus the amount of data already read into the buffer. Append the new frame data to the buffer at the correct location.

STEP 8: Examine the new Done bit in the RPSR value.

If the Done bit is not set, repeat the procedure starting at STEP 4. If the Done bit is set, the entire jumbo frame has been loaded into the buffer.

5.2.2 Transmit Jumbo

Use the following procedure to transmit jumbo frames that are too large to fit into the transmit QUE all at once:

STEP 1: Write to the QUE 4 PCWR.

This initializes the write logic for QUE 4 and set the ID and transmission options for the frame. The software should assign a unique ID number to each transmitted frame. Initializing the write logic triggers an under-run interrupt for QUE 4 if the previous frame transmission is not complete. This has the effect of aborting the previous frame.

STEP 2: Write the QUE 4 PSZR Packet Size field.

This indicates to the QUE 4 write logic how many bytes to expect. For example, if the part is operating in 32-bit bus mode and the byte count is not an even multiple of four the write logic uses the Packet Size value to determine how many bytes in the last write are valid. An overrun interrupt results if the host attempts to write an extra word past the end of the programmed size.

STEP 3: Set the QSR Threshold and Mode fields for QUE 4.

Set the QSR Mode field to 3 and the Threshold field to 5. With these settings, anytime the amount of data in the QUE is 1024 bytes or less, the Below Condition will be true.

STEP 4: Enable the B Rise interrupt for QUE 4.

Set the QSMR bit corresponding to the QUE 4 B Rise interrupt to enable the interrupt.

STEP 5: Enable the QSIR interrupt.

Set the HIMR Que Status bit to enable the QSIR interrupt.

STEP 6: Write the first 2048 bytes of the frame to the QUE 4 TDR.

Now that the QUE contains data the Below Condition should be clear.

STEP 7: Read QSIR.

This clears the B Rise interrupt that was set when the QUE was empty.

STEP 8: Wait for a QSIR interrupt.

The interrupt occurs when the MAC transmitter has removed data from the transmit QUE such that it is below the threshold set in STEP 1.

STEP 9: Write an additional 2048 bytes of the frame to the QUE 4 TDR.**STEP 10: Clear the QSIR interrupt by reading QSIR.**

The QSIR is clear-on-read.

STEP 11: Repeat STEP 8 through STEP 10 until the entire frame is written to the QUE 4 TDR.

The final amount of data written in STEP 9 will likely be less than 2048 bytes.

STEP 12: Clear the QSMR so that the QSIR interrupts will no longer be received.

Now that the final bytes of the frame are in the QUE, the MAC transmitter is free to empty it without any more involvement of the driver.

STEP 13: Read the transmit status from TPSR.

Reading the TPSR reads status words from the top of a 128 word deep FIFO. If the status words for transmitted frames are not read from the FIFO, it eventually fills to capacity at which time any new status words are dropped while the FIFO retains the old ones.

The software driver should retain the buffer memory containing the transmitted frame until the transmit status for that frame has been received. The status can be associated with the buffer that contains its data using the Packet ID assigned in the PCWR, which is also contained in the status. Once the TPSR Done bit is set and the transmit status is known to be good, the driver can safely de-allocate the buffer memory. In this way, if there was a problem during transmission and the status is not good, the driver can resubmit the frame if needed.

The procedure described above assumes that the MAC transmitter has been enabled and is not halted and that the MAC has been properly configured via the MCR.

The transmit status for a frame will not be available until the transmission completes. The driver may need to have provisions to recover the status at a later time.

5.3 Watermarking

The Headroom, PAUSE, and Interrupt watermarks, accessed by WMVR, are used to manage memory usage based on the size of the free memory pool. The three watermark levels affect each other. Refer to the "Watermarking" section of the *78Q8430 Data Sheet* for more information on this feature.

5.3.1 Headroom Watermark

The Headroom watermark specifies the number of free memory BLOCKS below which the MAC receiver is halted. WMVR contains three different watermark values. Change only the Headroom watermark value while keeping the values of the other two unchanged. This is accomplished using a read-modify-write operation. Use the following procedure to set the Headroom watermark.

STEP 1: Read the WMVR value.

STEP 2: Clear only the Headroom field in the WMVR value.

This can be done using a bit-wise AND operator.

STEP 3: Set the value of the Headroom field in the WMVR value.

This can be done using a bit-wise OR operator. The watermark value needs to be shifted to make it line up with the position of the Headroom field in the WMVR value.

STEP 4: Write the new value to WMVR.

5.3.2 PAUSE Watermark Level

The PAUSE watermark specifies the minimum number of free memory BLOCKS that triggers the automatic transmission of the PAUSE frame. Refer to Section 3.3 for more information on the transmit PAUSE operation. Use the following procedure to set the PAUSE watermark level.

STEP 1: Read the WMVR value.

WMVR contains three different watermark values. Change only the PAUSE watermark value while keeping the values of the other two unchanged. This is accomplished using a read-modify-write operation.

STEP 2: Clear only the PAUSE field of the WMVR value.

This can be done with the use of a bit-wise AND operator.

STEP 3: Set the value of the PAUSE field in the WMVR value.

This can be done with the use of a bit-wise OR operator. The desired watermark value needs to be shifted to make it line up with the position of the PAUSE field in the WMVR value.

STEP 4: Write the new value to WMVR.

5.3.3 Watermark Interrupt

The watermark interrupt is a key part of 78Q8430 memory management. Proper setting and servicing of the watermark interrupt helps to ensure smooth Ethernet operation under the most severe loads.

The watermark values are set based on the minimum number of free memory blocks that must be available to avoid the specified action. In the case of the interrupt watermark, the specified action is an interrupt. The value of the interrupt watermark should be set low enough that it is not triggered under ordinary circumstances, as this would increase the interrupt service load of the system. The interrupt watermark should also be set high enough that the interrupt is triggered while there is still enough free memory to keep the system moving long enough to take action and avoid data loss due to a lack of memory.

The following procedures describe how to set the interrupt watermark and service the watermark interrupt.

Set the Interrupt Watermark Level

STEP 1: Read the WMVR value.

WMVR contains three different watermark values. Change only the Interrupt watermark value while keeping the values of the other two unchanged. This is accomplished using a read-modify-write operation.

STEP 2: Clear only the Interrupt field of the WMVR value.

This can be done with the use of a bit-wise AND operator.

STEP 3: Set the value of the Interrupt field in the WMVR value.

This can be done with the use of a bit-wise OR operator. The desired watermark value needs to be shifted to make it line up with the position of the Interrupt field in the WMVR value.

STEP 4: Write the new value to WMVR.

Handle the Watermark Interrupt

STEP 1: Set the HIMR Water Mark bit to enable the watermark interrupt.**STEP 2: Wait for a watermark interrupt.****STEP 3: Calculate the number of memory blocks currently in use.**

The number of blocks in use is the sum of the number of blocks used by each QUE. Each QUE's QSR Count field contains the number of blocks currently used. It is not necessary to read the static QUEs as they always use exactly one block.

STEP 4: Check the receive QUE size for excessive use.

If the number of memory blocks currently in use by the receive QUE is large, the driver should address this before taking any other action. The driver should stop loading the transmit QUEs at the next opportunity without risking a transmit under-run and focus all host bus activity into reducing the receive QUE. If transmit frames are higher priority than receive frames, the driver should consider dropping low priority receive frames as an alternate method for drawing down the receive QUE.

STEP 5: Check transmit QUE size for excessive use.

The only appropriate response to a full transmit QUE is for the driver to stop loading the transmit QUE for some amount of time while the transmitter catches up. The driver should verify that the QSR QDR bit for the transmit QUE is set. If the QDR bit is not set but should be based on the QSR Mode and Threshold settings, or the QDR bit is set but the transmit QUE memory usage is not decreasing over time, the QUE may be stalled. A stalled transmit QUE requires a software reset.

5.4 Transfer Frame

The software driver can offload ARP responses and ICMP echo replies from the IP stack using the transfer frame feature. Once the driver discovers that the frame is an ARP request or ICMP echo request, the SNOOP feature is used to modify the frame while it is still in the receive QUE. Once the frame is converted from a request into the appropriate response, the transfer frame feature is used to transfer the entire frame from the receive QUE to a transmit QUE where it is transmitted out without any involvement of the IP stack.

Refer to the “Snoop Mode Access” section in the *78Q8430 Data Sheet* for additional information on the SNOOP feature.

5.4.1 Transmit ARP Response

Table 5 illustrates the contents of an ARP frame. The bottom row of the table contains the SNOOP addresses used to access the ARP frame fields above them. All table values are hexadecimal.

Table 5: ARP Frame Contents

Dest. Addr.	Src. Addr.	Type	ETH/IP Code	Addr. Size	Opcode ¹	HW Src.	Prot. Src	HW Dest. ²	Prot. Dest.	PADF CS
6 Bytes	6 Bytes	08 06	00 01 08 00 06 04		00 0?	6 Bytes	4 Bytes	6 Bytes	4 Bytes	22 Bytes
304	308	30C		310		314		318	31C	320
								324	328	32C

¹ The Opcode field is 0x0001 for a request and 0x0002 for a response.

² Typically, the hardware destination address is unknown in an ARP request. In this case the HW Dest. field is filled with zeros.

Use the following procedure to transmit an ARP response in reply to an ARP request:

STEP 1: Get the BLOCK number that contains the ARP request.

When the ARP request is the next frame in the receive QUE, the QUE 0 QFLR First field contains the BLOCK number of the frame.

STEP 2: Configure SNOOP to access the ARP request frame.

Write the value contained in the QFLR First field (from STEP 1) to the SNCR BLOCK field to configure the SNOOP interface to access the ARP request frame.

STEP 3: Use SNOOP access to retrieve the ARP Opcode field.

The two LSBs of SNOOP address 0x318 are the opcode. If the opcode is not 0x0001, the ARP is not a request and the standard receive frame procedure is used to pass the frame up to the stack, otherwise, proceed to STEP 4.

STEP 4: Use SNOOP access to retrieve the ARP protocol destination address field.

The two MSBs of SNOOP address 0x328 and the two LSBs of SNOOP address 0x32C contain the destination IP address. If the address is not one assigned to this interface, use the host drop procedure to drop the frame, otherwise, continue to STEP 5.

STEP 5: Use SNOOP access to retrieve the hardware and protocol source address fields.

The protocol source address is contained in SNOOP address 0x320. The hardware source address is contained in SNOOP address 0x31C plus the two MSBs of SNOOP address 0x318. There is no need to read address 0x318 since it was already read in STEP 3.

STEP 6: Set the destination address and source address fields in the frame.

Write the hardware source address retrieved in STEP 5 and the MAC address of this interface to the destination and source address fields, respectively, using SNOOP addresses 0x304, 0x308 and 0x30C.

STEP 7: Set the Opcode field and the hardware and protocol address fields in the frame.

SNOOP addresses 0x318, 0x31C, 0x320, 0x324, 0x328 and 0x32C are written with the response Opcode field set to 0x0002, the MAC address for this interface, the IP address gathered in STEP 4, the hardware source address and the protocol source address gathered in STEP 5 respectively. In other words, the source and destination addresses are swapped and the opcode is changed from a request to a response.

STEP 8: Set the transmit options for the ARP response.

The value normally written to the PCWR is written to SNOOP address 0x300 instead. The driver should set the PCWR Fix CRC bit because the existing CRC is incorrect after the changes made in this procedure. The driver may also wish to use a unique ID for this frame so that it does not try to free any buffers when it retrieves the transmit status for this frame.

STEP 9: Transfer the frame from the receive QUE to transmit QUE 3.

Set the RTTR Transfer bit to enable the transfer. The RTTR Transfer bit is self-clearing. When the bit is clear, it indicates that the First BLOCK from QUE 0 is now the last BLOCK in QUE 3. For frames that are 252 bytes or smaller in length, only one transfer is required. If the frame size is greater, more than one transfer will be required. The transfer of a single BLOCK should complete quickly. The driver must ensure that no data is read from QUE 0 RDR until the transfer operation is complete.

This procedure assumes that the driver has determined in advance that the next frame in the receive QUE is an ARP frame, or specifically, an ARP request. This can be done using classification or through inspection. The driver can read a portion of the frame in the standard way and the BLOCK will still be available for a transfer operation. However, once the first 252 bytes of the frame have been read, or the entire frame contents have been read, whichever comes first, the BLOCK is no longer available for a transfer operation.

If the ARP request is VLAN tagged, the procedure must be modified to account for the additional VLAN data in the Ethernet frame. To accomplish this, the SNOOP addresses in all STEPS except those in STEP 6 must be incremented by four.

5.4.2 Transmit ICMP Echo Reply

This section describes how to configure the CAM filters to enable recognition of ICMP frames and how to process the ICMP echo reply using the Snoop feature to access the ICMP frame.

Table 6 illustrates the contents of an ICMP frame. The bottom row of the table contains the SNOOP addresses used to access the ICMP frame fields above them. All values in the table are hexadecimal.

Table 6: ICMP Frame Contents

MAC Header			IP Header . . .						
Dest. Addr.	Src. Addr.	Len/Type ¹	Ver / IHL ²	TOS	Total Len	Identification	Flags / Frag Offset	TTL	Prot. ³
6 bytes	6 bytes	08 00	45	??	2 bytes	2 bytes	2 bytes	1 byte	01
304	308	30C	310		314		318		

. . . IP Header, cont.				ICMP Header		
Checksum	Src. Address	Dest. Address ⁴	Type ⁵	Code ⁶	Checksum	
2 bytes	4 bytes	4 bytes	0?	00	2 bytes	
31C		320		324		328 – 2 LSBs

¹ Type = 0x0800 indicates that an IP header follows.

² Based on the IHL value, there may be extra data at the end of the IP header before the ICMP header.

³ The Protocol field is always 0x01 for an ICMP frame.

⁴ The driver should verify that the Dest. Address belongs to it.

⁵ A Type value of 0x08 indicates an ICMP echo request and 0x00 indicates an ICMP echo reply.

⁶ The Code value should always be 0x00.

This procedure is only suitable for IP packets that have not been fragmented. When the MF flag is clear and the fragment offset value is zero, there are no fragments. This is the case when the last 14 bits of the Flags/Frag Offset field in the IP Header are all zeros.

Enable ICMP Classification

Use the following procedure to enable the identification of ICMP echo requests in the CAM:

STEP 1: Change the offset field of CAM rule 0x24 from 0x00 to 0x05.

- Set the CAR ADDR field to 0x24
- Change the RCR Byte Offset field to 0x05. This causes the next byte looked at by the classification engine to be the Flags and MSBs of the Fragment Offset.

STEP 2: Set CAM rule 0x7B to verify the MF flag is clear and the fragment offset MSBs are zero.

Reg.	Field	Value to write
CAR	ADDR	0x7B
RMR	Data Match	0x00
	Data Mask	0x3F
	Previous Hit Match	0x24
	Previous Hit Mask	0x7F
RCR	Byte Offset	Retain default: 0x00
	Interrupt	Retain default: 0
	Control Logic Action	NOP
	Match Control	MD

STEP 3: Set CAM rule 0x7A to verify the fragment offset LSBs are all zero.

Reg.	Field	Value to write
CAR	ADDR	0x7A
RMR	Data Match	0x00

	Data Mask	0xFF
	Previous Hit Match	0x7B
	Previous Hit Mask	0x7F
RCR	Byte Offset	0x01 (to skip over the TTL field and look at Protocol)
	Interrupt	Retain default: 0
	Control Logic Action	NOP
	Match Control	MD

STEP 4: Set CAM rule 0x79 to verify the protocol is ICMP.

Reg.	Field	Value to write
CAR	ADDR	0x79
RMR	Data Match	0x01
	Data Mask	0xFF
	Previous Hit Match	0x7A
	Previous Hit Mask	0x7F
RCR	Byte Offset	0x3F (to skip over the IP header)
	Interrupt	Retain default: 0
	Control Logic Action	NOP
	Match Control	MD

STEP 5: Set rule 0x78 to verify the ICMP type is echo request.

Reg.	Field	Value to write
CAR	ADDR	0x78
RMR	Data Match	0x08
	Data Mask	0xFF
	Previous Hit Match	0x79
	Previous Hit Mask	0x7F
RCR	Byte Offset	0x00
	Interrupt	Retain default: 0
	Control Logic Action	NOP
	Match Control	DONE

Process the ICMP Echo Reply

Use the following procedure to transmit an ICMP echo reply in response to an ICMP echo request:

STEP 1: Check that the classification result is 0x78.

The RPSR Classification field value is 0x78 when the frame is an ICMP echo request.

STEP 2: Read the QUE 0 QLFR to obtain the BLOCK number for the ICMP echo request frame.

When the ICMP request is the next frame in the receive QUE, the QUE 0 QLFR First field contains the BLOCK number of the frame.

STEP 3: Configure SNOOP to access the ICMP request frame.

Write the value contained in the QLFR First field to the SNCR BLOCK field to configure the SNOOP interface to access the ICMP frame.

STEP 4: Use SNOOP access to retrieve the IP destination address.

The two MSBs of SNOOP address 0x320 plus the two LSBs of SNOOP address 0x324 contain the destination IP address. If the address is not one assigned to this interface, use the host drop procedure to drop the frame, otherwise, continue on to STEP 5.

STEP 5: Swap the source and destination addresses in the IP header.

- The two LSBs of SNOOP address 0x320 (already read in STEP 4) plus the two MSBs of address 0x31C contain the IP source address. The IP destination address was already retrieved in STEP 4.
- Replace the Src. Addr field with the destination address value and the Dst. Addr field with the source address value. Leave the two LSBs of 0x31C and the two MSBs of 0x324 unchanged.
- Write the new values back using snoop addresses 0x31C, 0x320 and 0x324.

STEP 6: Swap the source and destination addresses in the MAC header.

- Read the MAC source and destination addresses using SNOOP addresses 0x304, 0x308 and 0x30C. The destination address is in Snoop address 0x304 plus the two LSBs of 0x308. The source address is in the two MSBs of 0x308 plus address 0x30C.
- Replace the Src. Addr field with the destination address value and the Dst. Addr field with the source address value.
- Write the new values back using SNOOP addresses 0x304, 0x308 and 0x31C.

STEP 7: Update the TTL field in the IP header.

A fresh TTL value needs to be written to the TTL field of the IP header as this will be a new frame when it is transmitted.

STEP 8: Set the ICMP Type and Code fields to echo reply.

Set the two MSBs of SNOOP address 0x324 to 0x0000 and leave the two LSBs unchanged. This converts the ICMP echo request into an ICMP echo reply.

STEP 9: Correct the ICMP checksum.

The two LSBs of SNOOP address 0x328 access the ICMP Header Checksum field. Only a single bit change was made to the ICMP header, so the checksum can be easily corrected without recalculation. The corrected checksum is the existing checksum +0x0800.



It is important to note that the first checksum byte to arrive is the most significant byte. This means it is in the lower byte position in the snoop register.

STEP 10: Set the transmit options for the ICMP echo reply.

The value normally written to PCWR is written to SNOOP address 0x300 instead. Set the PCWR Fix CRC bit because the existing CRC is incorrect after the changes made. The PCWR IP Header Offset field must also be set as changes to the IP header have invalidated the IP header checksum. The driver may wish to use a unique ID for this frame so that it does not try to free any buffers when it retrieves the transmit status for this frame.

STEP 11: Transfer the frame from the receive QUE to transmit QUE 3.

Set the RTTR Transfer bit to enable the transfer. The RTTR Transfer bit is self-clearing. When the bit is clear, it indicates that the First BLOCK from QUE 0 is now the last BLOCK in QUE 3. For frames that are 252 bytes or smaller in length, only one transfer is required. If the frame size is greater, more than one transfer is required. The transfer of a single BLOCK should complete quickly. The driver must ensure that no data is read from the QUE 0 RDR until the transfer operation is complete.

This procedure assumes that the driver has determined in advance that the next frame in the receive QUE is an ICMP frame, or more specifically, an ICMP echo request. This can be done via classification (see Enable ICMP Classification above) or through inspection. The driver can read a portion of the frame

in the standard way and the BLOCK will still be available for a transfer operation. Once the first 252 bytes of the frame have been read, or the entire frame contents have been read, whichever occurs first, the BLOCK is no longer available for a transfer operation.

If the ICMP echo request is VLAN tagged, the procedure must be modified to account for the additional VLAN data in the Ethernet frame. To accomplish this, all the SNOOP addresses except those in STEP 7 must be incremented by four.

5.5 IP Firewall

The CAM can be configured to drop frames based on source IP address. This saves the host from wasting many cycles processing a frame that will ultimately be dropped by the host firewall anyway. For example, the class A IP subnet 127.0.0.0 is reserved for the localhost domain. It is always an error for an internet datagram with a source IP address in the localhost domain to be received on an outside interface.

Use the following procedure to leverage the spare rules in the default CAM rule set to automatically drop all Ethernet frames that contain illegal internet datagrams with a source IP address in the 127.0.0.0 class A subnet.

STEP 1: Change the Offset for CAM rule 0x23 to 0x0A.

- Set the CAR ADDR field to 0x23.
- Set the RCR Byte Offset field to 0x0A.

Normally, rule 0x23 skips over the IP header. Changing the Offset prevents this to create an opportunity to act on the IP addresses. An offset of 10 causes the next byte matched by the CAM to be the first byte in the IP source address field.

STEP 2: Configure CAM rule 0x1A to check for the localhost domain.

Reg.	Field	Value to write
CAR	ADDR	0x1A
RMR	Data Match	Value of IP source byte [0] (0x7F)
	Data Mask	0xFF
	Previous Hit Match	0x23
	Previous Hit Mask	0x7F
RCR	Byte Offset	Retain default: 0x00
	Interrupt	Retain default: 0
	Control Logic Action	Retain default: NOP
	Match Control	DROP

5.6 IP Checksum Check

RFC 791 specifies that, if the header checksum fails, the internet datagram be discarded at once by the entity which detects the error. If the receive packet status indicates that the IP header checksum is bad, the driver should use the host drop procedure to drop the frame. This saves the overhead of reading the frame and calculating the checksum on the host side.

The default CAM rule set checks for IP header checksums in Ethernet payloads, even when the Len/Typ field is not IP. If desired, the CAM can be reconfigured to test IP header checksums only on Ethernet frames with a Len/Typ value that indicates an IP payload.

5.7 Transmit Priority

The standard transmit procedure uses transmit QUE 4. Frames can also be transmitted at a higher priority by using QUE 3. Use the following procedure to transmit a frame at a higher priority than the standard frames:

STEP 1: Write to QUE 3 PCWR.

Writing to this register sets the ID and transmission options for the frame and initializes the write logic for QUE 3. The driver should assign a unique ID number to each transmitted frame. When the write logic is initialized, it triggers an under-run interrupt for QUE 3 if the previous frame has not been completed. This has the effect of aborting the previous frame.

STEP 2: Write to QUE 3 PSZR.

Writing to PSZR indicates to the QUE 3 write logic how many bytes to expect. For example, if the part is operating in 32-bit bus mode and the byte count is not an even multiple of four, the write logic uses the size value to determine how many bytes in the last write are valid. If the host attempts to write an extra word past the end of the programmed size, an overrun interrupt results.

STEP 3: Write the frame data to QUE 3 TDR.

Each successive write to TDR adds the data to QUE 3.

STEP 4: Read the transmit status from TPSR.

Reading TPSR reads status words from the top of a 128 word deep FIFO. One status word is added to the FIFO for each transmitted frame. If the status words for transmitted frames are not read from the FIFO, it eventually fills to capacity at which time any new status words are dropped while the FIFO retains the oldest ones.

6 Counter Rollover Monitor (RMON)

The 78Q8430 includes a block of hardware counters to monitor transmit and receive statistics. All hardware statistics counters are 32-bits. Since the driver is required to maintain 64-bit statistics counts, it is responsible for maintaining the upper 32 bits of each 64-bit counter. Rollover interrupt hardware that can be individually enabled or disabled for each counter is provided to assist the driver to implement this.

Refer to the “Counters” section in the *78Q8430 Data Sheet* for additional information on the statistics counters and RMON interrupts. The following sections describe how to use the RMON feature.

6.1 Read Counters

Use the following procedure to read the statistics counter value:

STEP 1: Configure the CCR for the counter to be read.

- Fill the CCR Address field with the address of the counter to be read.
- Clear the Access Mode bit to specify read mode.
- Make sure the Clear on Read bit is clear (if this bit is set, the counter is automatically cleared when it is read).
- The CCR Auto Increment bit should always be set.

STEP 2: Read CDR to retrieve the counter value.

The CCR Address field is automatically incremented after each read or write access to the CDR. This allows many counters to be accessed through repeated reads or writes of the CDR without reconfiguring the CCR.

6.2 Clear all Counters

Use the following procedure to instantaneously clear all the hardware statistics counters to zero at exactly the same time:

STEP 1: Set the CMR Clear Receive and Clear Transmit bits.

No further action is necessary.

6.3 RMON Interrupts

The RMON interrupts are the best way to maintain the upper 32 bits of a 64 bit count (rollover count). Each transmit and receive statistics counter has a corresponding rollover bit which can be individually enabled to trigger a host interrupt each time the corresponding hardware 32-bit counter rolls over. Use the following procedure to implement the RMON interrupts:

STEP 1: Enable the rollover interrupts.

Set the TRMR and RRMR bits corresponding to the transmit and receive statistics counters to be monitored for rollover. Set the HIMR RMON bit.

TRMR bits 0 through 14 correspond to transmit counters 0 through 14. RRMR bits 0 through 24 correspond to receive counters 15 through 39.

STEP 2: Wait for an RMON interrupt.

The HIR RMON bit will be set to indicate an RMON interrupt has occurred.

STEP 3: Read TRIR.

A single bit will be set in TRIR for each transmit counter that has rolled over. The driver increments by one each rollover counter that corresponds to a TRIR set bit.

STEP 4: Read RRIR.

A single bit will be set in RRIR for each receive counter that has rolled over. The driver increments by one each rollover counter that corresponds to a RRIR set bit.

All of the RRIR and TRIR bits are cleared on read. Once the RRIR and TRIR bits clear, the RMON bit in the HIR will be clear.

7 PHY Procedures

This section describes the following procedures related to PHY operation:

- PHY Register read and write access
- PHY power down
- Auto-negotiation
- Setting PHY speed
- Setting PHY duplex mode
- Setting PHY MDI/MDIX mode
- Response to PHY link status change

7.1 PHY Register Write Access

Use the following procedure to write a value to a PHY register:

STEP 1: Write the value for the PHY register to the MDDAR SMI Data field.

STEP 2: Write to MDCAR.

Fill the MDCAR PHY Reg field with the number of the PHY register to write. Set the MDCAR PHY Addr field to 0x01. Set the MDCAR RegWr bit and the Busy bit.

STEP 3: Read MDCAR.

STEP 4: Repeat STEP 3 until the MDCAR Busy bit is clear.

7.2 PHY Register Read Access

The following procedure is used to read a value from a PHY register:

STEP 1: Write to MDCAR.

Fill the MDCAR PHY Reg field with the number of the PHY register to read. Set the MDCAR PHY Addr field to 0x01. Clear the MDCAR RegWr bit and set the Busy bit.

STEP 2: Read MDCAR.

STEP 3: Repeat STEP 2 until the MDCAR Busy bit is clear.

STEP 4: Read the PHY register value from the MDDAR SMI Data field.

7.3 PHY Power Down

In an effort to save power the host may wish to power down the PHY. Use the following procedure to enable or disable the PHY power down mode:

STEP 1: Use procedure [7.2](#) to read the value of the MR0 register.

STEP 2: Modify the value of the MR0 Power-down bit.

For low power mode the Power-down bit should be set. For normal mode this bit should be clear.

STEP 3: Use procedure [7.1](#) to write the modified value back to the MR0 register.

7.4 Auto-negotiation

The 78Q8430 supports the auto-negotiation functions of IEEE 802.3 Clause 28 for 10/100 Mbps operation over copper wiring. The auto-negotiation function defaults to ON (the MR0 Auto-negotiation Enable bit is high after reset). Table 7 lists the default values for the auto-negotiation registers.

Table 7: Auto-negotiation Registers Default Values

Register	Field / Function	Default Value
MR0	Speed Selection	1 (100Base-TX)
	Auto-negotiation Enable	1 (enabled)
	Duplex Mode	1 (full duplex)
MR4 / MR1	100BASE-TX Full Duplex	1
	100BASE-TX Half Duplex	1
	10BASE-T Full Duplex	1
	10BASE-T	1

When auto-negotiation is enabled, the 78Q8430 sends the contents of the Auto-negotiation Advertisement Register, MR4, to its link partner using fast link pulse coding at power on, loss of link or on a command to restart auto-negotiation. At the same time, it looks for either 10BASE-T idle, 100BASE-TX idle, or fast link pulses from its link partner. If either idle pattern is detected, the 78Q8430 configures itself in half-duplex mode at the appropriate speed. If it detects fast link pulses, it decodes and analyzes the link code transmitted by the link partner. When three identical link code words are received (ignoring the acknowledge bit) the link code word is stored in register MR5. Upon receiving three more identical link code words, with the acknowledge bit set, the 78Q8430 configures itself to the highest priority technology common to the two link partners. The technology priorities are, in descending order:

100BASE-TX, Full Duplex
 100BASE-TX, Half Duplex
 10BASE-T, Full Duplex
 10BASE-T, Half Duplex

When auto-negotiation is complete, the MR18 Rate Indication and Duplex Indication bits reflect the actual negotiated speed and duplex mode.

If auto-negotiation fails to establish a link for any reason, the MR18 Auto-negotiation Fail Indication bit reflects this and auto negotiation restarts from the beginning.

7.4.1 Disable Auto-negotiation

Use the following procedure to disable the auto-negotiation function:

- STEP 1: Use procedure 7.2 to read the value of the MR0 register.**
- STEP 2: Clear the MR0 Auto-negotiation Enable bit to disable auto-negotiation.**
- STEP 3: Use procedure 7.1 to write the modified value back to the MR0 register.**

7.4.2 Enable Auto-negotiation

The auto-negotiation function defaults to enabled after reset. Use the following procedure to re-enable the function if it has been disabled:

- STEP 1: Use procedure 7.2 to read the value of the MR0 register.**
- STEP 2: Set the MR0 Auto-negotiation Enable bit to enable auto-negotiation.**
- STEP 3: Use procedure 7.1 to write the modified value back to the MR0 register.**

7.4.3 Restart Auto-negotiation

Normally, the auto-negotiation process is started at power up or loss of link. Use the following process to restart the process at other times:

STEP 1: Use procedure 7.2 to read the value of the MR0 register.

STEP 2: Set the MR0 Restart Auto-negotiation bit.

This bit is self-clearing.

STEP 3: Use procedure 7.1 to write the modified value back to the MR0 register.

7.5 PHY Speed

Use the following procedure to set the PHY speed:

STEP 1: Use procedure 7.2 to read the value of the MR0 register.

STEP 2: Clear the MR0 Auto-negotiation Enable bit to disable auto-negotiation.

The speed cannot be set if auto-negotiation is enabled.

STEP 3: Modify the value of the MR0 Speed Selection bit.

Clear the Speed Selection bit for 10Mbps mode. Set the Speed Selection bit for 100Mbps mode.

STEP 4: Use procedure 7.1 to write the modified value back to the MR0 register.

When the link speed is set in this way, auto-negotiation is disabled and the PHY only operates in the selected speed and selected duplex mode even when a new link is established.

7.6 PHY Duplex Mode

Use the following procedure to set the PHY duplex mode:

STEP 1: Use procedure 7.2 to read the value of the MR0 register.

STEP 2: Clear the MR0 Auto-negotiation Enable bit to disable auto-negotiation.

The duplex mode can not be set if auto-negotiation is enabled.

STEP 3: Modify the value of the MR0 Duplex Mode bit.

Set the Duplex Mode bit for full-duplex mode. Clear the Duplex Mode bit for half-duplex mode.

STEP 4: Use procedure 7.1 to write the modified value back to the MR0 register.

When the duplex mode is set in this way, auto-negotiation is disabled and the PHY only operates in the selected duplex mode and selected speed even when a new link is established.

7.7 PHY MDI/MDIX Modes

Use the following procedure to set the PHY MDI/MDIX mode:

STEP 1: Use procedure 7.2 to read the value of the MR16 register.

STEP 2: Clear the MR16 Auto Polarity bit to disable auto polarity.

The MDI/MDIX mode can not be set if auto polarity is enabled.

STEP 3: Modify the value of the MR16 Reverse Polarity bit.

Clear the Reverse Polarity bit for MDI mode. Set the Reverse Polarity bit for MDIX mode.

STEP 4: Use procedure 7.1 to write the modified value back to the MR16 register.

When the polarity is set in this way, auto polarity is disabled and the PHY only operates in the selected polarity even when a new link is established.

7.8 PHY Link Status Change

The behavior of the MAC in half duplex mode is not the same as its behavior in full duplex mode. It is critical for proper operation of the MAC that it be kept up to date as to the duplex nature of the currently established link on the PHY at all times. To ensure this, the driver must respond to all link change interrupts from the PHY and update the MAC duplex mode based on the nature of the new link.

Use the following procedure to handle a link status change interrupt from the PHY:

STEP 1: Enable link status change interrupts in the PHY.

Set the Link Status Change Interrupt Enable bit in the MR17 register to enable the interrupt.

STEP 2: Enable the PHY interrupt.

Set the HIMR PHY bit to enable PHY interrupts to the driver.

STEP 3: Wait for a link status change interrupt.

The interrupt service routine should identify interrupts from the PHY by reading the MR17 register. The MR17 Link Status Change Interrupt bit confirms that a link status change has occurred. Reading MR17 also clears the interrupt.

STEP 4: Get the status of the new link.

Read the MR1 Link Status bit to determine the link status. If the Link Status bit is clear, there is no link and the driver should disable the MAC to prevent frames being sent to the PHY. In this case, no further action is required.

STEP 5: Get the mode of the new link.

Read the MR18 register to determine the actual status of the current link. The MR18 Duplex Indication bit will be set for full duplex and clear for half duplex.

STEP 6: Set the MCR FullDup bit to the value contained in the MR18 Duplex Indication bit.

The value of the MR0 Duplex Mode bit is not an accurate indication of the state of the current link. When auto-negotiation is used, only the MR18 register contains the state of the current link.

8 EEPROM Operations

The 78Q8430 provides logic for reading and writing an optional external EEPROM or ROM device. Refer to the *78Q8430 Data Sheet* for supported devices. EEPROM Erase, Write, and Read procedures are described in this section.

8.1 EEPROM Erase

Use the following procedure to erase the EEPROM:

STEP 1: Write to PRCR.

Fill the PRCR Operation field with the Erase command (b11) and set the PRCR Busy bit.

STEP 3: Read PRCR.

STEP 4: Repeat STEP 3 until the PRCR Busy bit is clear.

8.2 EEPROM Write Access

Use the following procedure to write a value to the EEPROM:

STEP 1: Write the data to PRDR.

Write the desired data to the lower 16 bits of PRDR.

STEP 2: Write to PRCR.

Fill the PRCR PROM Addr field with the EEPROM address to write. Fill the Operation field with the Write command (b01) and set the Busy bit.

STEP 3: Read PRCR.

STEP 4: Repeat STEP 3 until the PRCR Busy bit is clear.

8.3 EEPROM Read Access

Use the following procedure to read a value from the EEPROM:

STEP 1: Write to PRCR.

Fill the PRCR PROM Addr field with the EEPROM address to read. Fill the Operation field with the Read command (b10) and set the Busy bit.

STEP 2: Read PRCR.

STEP 3: Repeat STEP 3 until the PRCR Busy bit is clear.

STEP 4: Read the EEPROM value from PRDR.

The EEPROM data will be in the lower 16 bits.

9 Power Management

9.1 Sleep Procedure

Use the following procedure to setup the sleep state and specify the wake event interrupt:

STEP 1: Read the PMCSR value.

STEP 2: Modify the PS bits to anything non-zero (e.g. 0x01).

The wake event only occurs if one or both of the PS bits are set (non-zero).

STEP 3: Set the wake event interrupt.

If desired, enable the PME interrupt by setting the PME_ENB bit in the PMCSR value. If not using the PME interrupt, enable the WAKE interrupt in the HIMR.

STEP 4: Write the new value back to the PMCSR.

Do not modify PMCSR bits 16-31 in this procedure.

9.2 Wake Event Procedure

Use the following procedure to process a wake event:

STEP 1: Read the PMCSR value.

STEP 2: Verify that the PME bit is set.

STEP 3: Clear the PS bits.

STEP 4: Write the modified value back to the PMCSR.

Do not modify PMCSR bits 16-31 in this procedure.

STEP 5: Read the WUSR to determine the CAM rule that triggered the wake event.

This is useful if the host wants to know the source of the wake event.

STEP 6: Execute a software reset.

✓ It is important that the software execute steps 1 through 4 of this procedure as soon as possible after the wake event because this stops the HNR counter and potentially prevent the transmission of a Host Not Responding frame. Steps 5 and 6 of the procedure are not time critical.

10 BIST

Use the following procedure to run the built in self-check for the embedded memories:

STEP 1: Set the BCR BIST Enable bit.

STEP 2: Set the BCR BIST Mode field for the desired BIST test mode.

STEP 3: Set the BCR BIST Start bit.

This bit initiates the currently selected BIST test and then auto-clears. For BYPASS test mode this step is necessary to go into BYPASS. Once in BYPASS mode, BYPASS mode is maintained until a new BIST Mode is selected and the BIST Start bit is set again to initiate the new mode.

STEP 4: Poll BCR in order to determine if the current BIST test is done.

When the selected test is complete, the BCR Pass bit or Fail bit will be set. If they are both clear, the currently selected BIST test is still in progress.

Use BYPASS mode when a manual test is needed. Using BYPASS mode to test the RAM is not covered here. For reference purposes, in BYPASS Mode, the RAM Data is read or written using the BBDR register. If auto-incrementing is selected, the address increments after each BYPASS RAM cycle (read or write of BBDR register). The current address pointer can be determined by reading back the RAM Address contained in the BCR RAM Address field.

In BYPASS mode the Pass bit is set immediately. It is up to the host to determine the pass/fail criteria.

For best fault coverage, the recommended test mode sequence is FILL 1, wait 10 ms, READ 1, PATTERN, FILL 0, wait 10 ms, READ 0. The 10 ms delays are needed to catch slow bit failures.

As a reference point the following times were measured in simulation with a 100 Mhz system clock.

Table 8: BIST Run Times

BIST Mode	Time to Complete @ 100 MHz
PATTERN (001)	6300 us
FILL 0 (010)	85 us
READ 0 (011)	170 us
FILL 1 (100)	85 us
READ 1 (101)	170 us
BYPASS* (110)	0 us

11 Software Reset

Use the following procedure to reset the QUEs and the MAC:

STEP 1: Set the MCR MACRST bit.

STEP 2: Read the MCR to verify the reset.

Verify that both Tx and Rx are halted (Tx Halt and Rx Halt bits are set). Verify that both Tx and Rx are disabled (Tx Enable and Rx Enable bits are clear). Verify that the MACRST bit is set.

STEP 3: Write the MCR and clear the MACRST bit.

STEP 4: Release all buffers and reset the state of the driver.

STEP 5: Write the MCR Tx Enable and Rx Enable bits to enable transmit and receive.

12 Related Documentation

The following 78Q8430 documents are available from Teridian Semiconductor Corporation:

78Q8430 Preliminary Data Sheet
78Q8430 Layout Guidelines
78Q8430 Software User Guide for ST/OS-20
78Q8430 Software User Guide for Linux

13 Contact Information

For more information about Teridian Semiconductor products or to check the availability of the 78Q8430, contact us at:

6440 Oak Canyon Road
Suite 100
Irvine, CA 92618-5201

Telephone: (714) 508-8800
FAX: (714) 508-8878
Email: lan.support@teridian.com

For a complete list of worldwide sales offices, go to <http://www.teridian.com>.

Appendix A – Default CAM Rule Summary

Table 9: Default CAM Rules

Rule#	Previous Hit Match	Previous Hit Mask	Data Match	Data Mask	Match Control	Byte Offset	Action	Inter-rupt	Comment
0x7F	0x00	0x7F	0xFF	0xFF	MD	0x00	NOP	0	
0x7E	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x7D	0x00	0x7F	0x01	0xFF	MD	0x00	SETMC	0	
0x7C	0x00	0x7F	0x01	0x01	MD	0x00	SETMC	0	
0x7B	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x7A	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x79	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x78	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x77	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x76	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x75	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x74	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x73	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x72	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x71	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x70	0x00	0x7F	0x00	0x00	MD	0x00	NOP	0	
0x6F	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x6E	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x6D	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x6C	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x6B	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x6A	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x69	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x68	0x70	0x78	0x00	0x00	MD	0x00	NOP	0	
0x67	0x7F	0x7F	0xFF	0xFF	MD	0x00	NOP	0	
0x66	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x65	0x7D	0x7F	0x80	0xFF	MD	0x00	NOP	0	
0x64	0x7C	0x7C	0x00	0x00	MD	0x00	NOP	0	
0x63	0x67	0x7F	0xFF	0xFF	MD	0x00	NOP	0	
0x62	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x61	0x65	0x7F	0xC2	0xFF	MD	0x00	NOP	0	
0x60	0x64	0x7C	0x00	0x00	MD	0x00	NOP	0	
0x5F	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x5E	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x5D	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x5C	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x5B	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x5A	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x59	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	

Rule#	Previous Hit Match	Previous Hit Mask	Data Match	Data Mask	Match Control	Byte Offset	Action	Inter-rupt	Comment
0x58	0x68	0x78	0x00	0x00	MD	0x00	NOP	0	
0x57	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x56	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x55	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x54	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x53	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x52	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x51	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x50	0x58	0x78	0x00	0x00	MD	0x00	NOP	0	
0x4F	0x63	0x7F	0xFF	0xFF	MD	0x00	NOP	0	
0x4E	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x4D	0x61	0x7F	0x00	0xFF	MD	0x00	NOP	0	
0x4C	0x60	0x7C	0x00	0x00	MD	0x00	NOP	0	
0x4B	0x4F	0x7F	0xFF	0xFF	MD	0x00	TAX	0	
0x4A	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x49	0x4D	0x7F	0x00	0xFF	MD	0x00	NOP	0	
0x48	0x4C	0x7C	0x00	0x00	MD	0x00	NOP	0	
0x47	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x46	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x45	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x44	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x43	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x42	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x41	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x40	0x50	0x78	0x00	0x00	MD	0x00	NOP	0	
0x3F	0x4B	0x7F	0xFF	0xFF	MD	0x00	SETBC	0	BC
0x3E	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x3D	0x49	0x7F	0x01	0xFF	MD	0x00	TAX	0	PAUSE
0x3C	0x48	0x7C	0x00	0x00	MD	0x00	TAX	0	MC
0x3B	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x3A	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x39	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x38	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x37	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x36	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x35	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x34	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x33	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x32	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x31	0x00	0x00	0x00	0x00	MD	0x00	NOP	0	
0x30	0x40	0x78	0x00	0x00	MD	0x00	TAX	0	promiscuous

Rule#	Previous Hit Match	Previous Hit Mask	Data Match	Data Mask	Match Control	Byte Offset	Action	Inter-rupt	Comment
Process source address. Negative filter on multicast.									
0x2F	0x30	0x70	0x01	0x01	DROP	0x00	NOP	0	MC drop
0x2E	0x30	0x70	0x00	0x00	MD	0x05	NOP	0	pass other
Process Len/Type field, MAC control frames and start IP header checksum check.									
0x2D	0x2B	0x7F	0x00	0xFF	MD	0x02	VLAN	0	
0x2C	0x00	0x00	0x00	0x00	DONE	0x00	NOP	0	
0x2B	0x2E	0x7F	0x81	0xFF	MD	0x00	TDLTH	0	
0x2A	0x2B	0x7F	0x00	0x00	DONE	0x00	TDLTL	0	
0x29	0x2F	0x7C	0x88	0xFF	MD	0x00	TDLTH	0	
0x28	0x29	0x7F	0x08	0xFF	MD	0x00	MCTL	0	
0x27	0x29	0x7F	0x00	0x00	DONE	0x00	TDLTL	0	
0x26	0x2F	0x7C	0x00	0x00	MD	0x00	TDLTH	0	
0x25	0x26	0x7F	0x00	0x00	MD	0x00	TDLTL	0	
0x24	0x25	0x7F	0x40	0xF0	MD	0x00	IPCK	0	
0x23	0x24	0x7E	0x00	0x00	MD	0x3F	TXA	0	
0x22	0x28	0x7F	0x00	0xFF	MD	0x00	NOP	0	
0x21	0x28	0x7F	0x00	0x00	DONE	0x00	NOP	0	
0x20	0x22	0x7F	0x01	0xFF	MD	0x00	NOP	0	MCTL pause
0x1F	0x22	0x7F	0x00	0x00	DONE	0x00	NOP	0	MCTL other
0x1E	0x20	0x7F	0x00	0x00	MD	0x00	TDPH	0	
0x1D	0x1E	0x7F	0x00	0x00	MX	0x00	TDPL	0	
0x1C	0x1D	0x7F	0x3D	0xFF	DONE	0x00	PAUSE	0	
0x1B	0x1D	0x7F	0x00	0x00	DONE	0x00	NOP	0	MCTL pause with bad SRC
0x1A	0x00	0x00	0x00	0x00	DONE	0x00	NOP	0	
0x19	0x00	0x00	0x00	0x00	DONE	0x00	NOP	0	
0x18	0x00	0x00	0x00	0x00	DONE	0x00	NOP	0	
0x17	0x00	0x00	0x00	0x00	DONE	0x00	NOP	0	
0x16	0x00	0x00	0x00	0x00	DONE	0x00	NOP	0	
OnNow									
This section identifies the OnNow example from the <i>78Q8430 Data Sheet</i> which is based on an example taken from the Microsoft OnNow specification. Actual OnNow is likely to be more complex.									
0x15	0x23	0x7F	0x54	0xFF	MD	0x00	NOP	0	
0x14	0x15	0x7F	0x00	0x00	MD	0x02	NOP	0	
0x13	0x14	0x7F	0x04	0xFF	MD	0x00	NOP	0	
0x12	0x13	0x7F	0x05	0xFF	MD	0x00	NOP	0	
0x11	0x12	0x7F	0x06	0xFF	MD	0x00	NOP	0	
0x10	0x11	0x7F	0x07	0xFF	DONE	0x00	WAKE	0	

Rule#	Previous Hit Match	Previous Hit Mask	Data Match	Data Mask	Match Control	Byte Offset	Action	Inter-rupt	Comment
Magic Packet									
0x0F	0x25	0x7F	0xFF	0xFF	MD	0x05	TOC	0	
0x0E	0x23	0x7F	0xFF	0xFF	MD	0x05	TOC	0	bare ethernet
0x0D	0x01	0x7F	0xFF	0xFF	MD	0x05	TOC	0	IP payload
0x0C	0x0F	0x7C	0xFF	0xFF	MD	0x00	DEC	0	offset payload
0x0B	0x0B	0x7F	0xFF	0xFF	MD	0x00	NOP	0	
0x0A	0x00	0x00	0x00	0x00	DONE	0x00	NOP	0	
0x09	0x0B	0x7F	0x10	0xFF	MD	0x0F	TOC	0	
0x08	0x03	0x7F	0x10	0xFF	MD	0x00	NOP	0	
0x07	0x08	0x7E	0x20	0xFF	MD	0x00	NOP	0	
0x06	0x07	0x7F	0x30	0xFF	MD	0x00	NOP	0	
0x05	0x06	0x7F	0x40	0xFF	MD	0x00	NOP	0	
0x04	0x05	0x7F	0x50	0xFF	MD	0x00	NOP	0	
0x03	0x04	0x7F	0x60	0xFF	MD	0x00	DEC	0	
0x02	0x02	0x7F	0x00	0x00	DONE	0x00	WAKE	0	
0x01 and 0x00 are reserved									

Appendix B – Acronyms

Miscellaneous Acronyms

ARC	Address Resolution Controller
CAM	Content Addressable Memory
GBI	Generic Bus Interface
HNR	Host Not Responding
ICMP	Internet Control Message Protocol
MAC	Media Access Control
MDI	Media Dependent Interface
MDIX	Media Dependent Interface Crossover
RMON	Rollover Monitor
WOL	Wake-on-LAN

Table 10: 78Q8430 Register Acronyms

Register Set	Register Acronym	Description
QUE	PCWR	Packet Control Word Register
	PSZR	Packet Size Register
	QFLR	QUE First/Last pointers Register.
	QSR	QUE Status Register
	RDR	Receive Data Register
	STDR	Setup Transmit Data Register
	TDR	Transmit Data Register
CTL	DMA	DMA Slave Mode Control and Status
	FDR	Frame Disposition Register
	GBI_CS	Configuration Register
	GBI_ID	Part ID Register
	HIMR	Host Interrupt Mask Register
	HIR	Host Interrupt Register
	HNRCR	Host Not Responding Count Register
	IDCR	Interrupt Delay Count Register
	MCR	MAC Control Register
	MDCAR	Station Management Control and Address Register
	MDDAR	Station Management Data Register
	OUIR	Overflow/Under-run Interrupt Register
	OUMR	Overflow/Under-run Mask Register
	PDCR	Pause Delay Count Register
	PMCAP	Power Management Capabilities
	PMCSR	Power Management Control and Status
	PRCR	PROM Control
	PRDR	PROM Data
QSIR	QUE Status Interrupt Register	
QSMR	QUE Status Mask Register	

Register Set	Register Acronym	Description
CTL	RCR	Rule Control Register
	RDSR	Receive Data Status Register
	RFBSR	Receive FIRST BLOCK Status Register
	RMR	Rule Match Register
	RPROS	Receive Producer Status
	RPSR	Receive Packet Status FIFO
	RRIR	Receive RMON Interrupt Register
	RRMR	Receive RMON Mask Register
	RTTR	Receive to Transmit Transfer Register
	SNCR	Snoop Control Register
	TPROS	Transmit Producer Status
	TPSR	Transmit Packet Status FIFO
	TRIR	Transmit RMON Interrupt Register
	TRMR	Transmit RMON Mask Register
	WMVR	Water Mark Values Register
	WUSR	Wake Up Status Register
PHY	MR0	Control
	MR1	Status
	MR2	PHY Identifier 1
	MR3	PHY Identifier 2
	MR4	Auto-Negotiation Advertisement
	MR5	Auto-Negotiation Link Partner Ability
	MR6	Auto-Negotiation Expansion
	MR16	Vendor Specific
	MR17	Interrupt Control/Status
	MR18	Diagnostic Register
	MR19	Transceiver Control
	MR23	LED Configuration
	MR24	MDI/MDIX Control

Revision History

Revision	Date	Description
1.0	2/13/2008	First publication.