CD2231

Intelligent Two-Channel LAN and WAN Communications Controller

Datasheet

The CD2231 is a two-channel multi-protocol synchronous/asynchronous communications controller specifically designed to reduce host-system processing overhead and increase efficiency in a wide variety of communications applications. The CD2231 is packaged in a 100-pin MQFP, and offers eight clock/modem pins per channel. The device has two fully independent serial channels that support asynchronous, asynchronous-HDLC (PPP), synchronous HDLC/SDLC, SLIP, and MNP® 4 protocols at serial data rates up to 256 kbps, (230.4 kbps in async modes) when clocked by a 35-MHz source.

The device is based on a proprietary on-chip RISC processor that performs all time-critical, low-level tasks that are otherwise performed by the host system.

The CD2231 boosts system efficiency with on-chip DMA, on-chip FIFOs, intelligent vectored interrupts, and intelligent protocol processing. The on-chip DMA controller provides 'fire-and-forget' transmit support — the host need only inform the CCD2231 of the location of the packet to be sent. Similarly, on receive, the CD2231 automatically receives a complete packet with no host intervention or assistance required. The DMA controller also has an 'Append mode' for use in asynchronous applications.

The DMA controller uses a dual-buffer scheme that easily implements simple or complex buffer schemes. Each channel and direction has two active buffers.

The CD2231 can be programmed to interrupt the host at the completion of a frame or buffer. In applications where buffers are of a small, fixed size, the dual-buffer scheme allows large frames to be divided into multiple buffers.

For applications where a DMA interface is not desired, the device can be operated as an interrupt-driven or polled device. This choice is available individually for each channel and each direction. For example, a channel can be programmed for DMA transmit and interrupt-driven receive.

In either case, 16-byte FIFOs on each channel and in each direction reduce latency time requirements, making both software and hardware designs less time-critical. Threshold levels on FIFOs are user-programmable.

Efficient vectored interrupts are another way the CD2231 helps system efficiency. Separate interrupts are generated for transmit, receive, and modem-signal change with unique user-defined vectors for each type and channel. This allows very flexible interfacing and fast, efficient interrupt coding.

As of May 2001, this document replaces the Basis Communications Corp. document *CL-CD2231 — Intelligent Two-Channel LAN and WAN Communications Controller*.

May 2001

Information in this document is provided in connection with Intel® products. No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted by this document. Except as provided in Intel's Terms and Conditions of Sale for such products, Intel assumes no liability whatsoever, and Intel disclaims any express or implied warranty, relating to sale and/or use of Intel products including liability or warranties relating to fitness for a particular purpose, merchantability, or infringement of any patent, copyright or other intellectual property right. Intel products are not intended for use in medical, life saving, or life sustaining applications.

Intel may make changes to specifications and product descriptions at any time, without notice.

Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them.

The CD2231 may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.

Copies of documents which have an ordering number and are referenced in this document, or other Intel literature may be obtained by calling 1-800-548-4725 or by visiting Intel's website at http://www.intel.com.

Contents

Figures

Tables

Revision History

Revision	Date	Description
1.0	5/01	Initial release.

1.0 Features

- Two full-duplex multi-protocol channels, each capable of up to 256 kbits/second (230.4 kbps in async modes) at 35-MHz CLK
- Multi-protocol support: SLIP (serial-line interface protocol), MNP® 4, async, async-HDLC (high-level data link control), and HDLC/SDLC (synchronous data link control) on both channels (does not support SDLC multi-drop applications)

PPP (Point-to-Point Protocol) Features

- Supports data link level — RFC-1661
- Supports dual async control character maps (32 control characters) — RFC-1662
- Compatible with ISO 3309/4335 Addendum 1 (Async-HDLC)
- Automatic insertion and deletion of control/escape characters and bit complements
- Automatic generation and detection of 16-bit FCS (frame check sequence)

MNP® 4 V.42 Features

- AppleTalk® Remote Access Protocol 1.0/2.0

SLIP Features

- Supports data link level — RFC-1055

HDLC/SDLC (Non-Multidrop) Features

- Four 8-bit or two 16-bit frame address matching
- FCS generation and validation
- Programmable number of leading-pad characters and flags
- Supports shared flags on receive frames

Asynchronous Features

- User-programmable and automatic flow control modes
  - In-band by XON/XOFF for both receive and transmit
  - Out-of-band (hardware flow control) by RTS/CTS and DTR/DSR
  - Line-break detection and generation
  - Special-character and character-range recognition
  - Special-character transmission
  - Transmit delay
- 5- to 8-bit character plus optional parity

- Enhanced features for UNIX® environment
  - Character expansion in transmit (for example, sending <LF> will be expanded to <CR> <LF> automatically)
  - Programmable translation of receiving character with error to different pattern (for example, character with parity error can be translated into FFh, 00h, character on the system side)
  - Flow-control transparency and LNext
- Programmable timer closely coupled with character reception, especially for asynchronous receive DMA operation
- Automatic baud rate detection — single carriage return

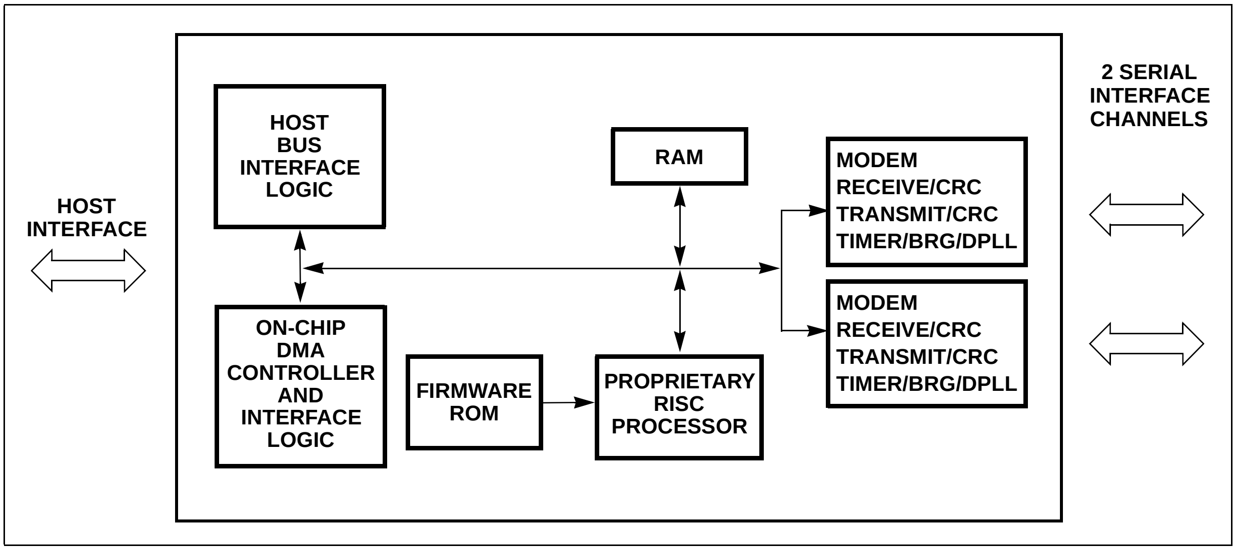
DMA Controller Features

- DMA or interrupt selectable per channel and per direction
- Dual Configuration register sets to reduce realtime constraints
- Append and Block mode DMA
- Chain/unchain of long frames into multiple buffers
- 32-bit address and 8- or 16-bit data transfer
- Programmable gap in buffers following a receive character exception

Other Features

- Improved interrupt schemes
  - Vectored interrupts per channel allow direct jump into proper service routines
  - Good Data™ interrupts eliminate need for status checks
- Easily cascadable for multiple-device configurations
- 16-byte receive and transmit FIFOs
- Local and Remote Maintenance Loopback modes
- Two independent bit-rate generators per channel for transmit and receive
- On-chip NRZ (nonreturn-to-zero), NRZI (nonreturn-to-zero inverted), and Manchester data encoding and decoding
- DPLL (digital phase locked loop) on each receiver
- Two independent timers per channel
- Byte-endian-orientation selection pin allows easy interface to 80X86 and 680X0 processors
- Eight clock/modem control signals per channel (in addition to TxD and RxD) on CD2231
- Compatible with the CD24XX family of communication controllers

Figure 1. Functional Block Diagram


HOST INTERFACE
HOST BUS INTERFACE LOGIC
ON-CHIP DMA CONTROLLER AND INTERFACE LOGIC
FIRMWARE ROM
RAM
PROPRIETARY RISC PROCESSOR
MODEM RECEIVE/CRC TRANSMIT/CRC TIMER/BRG/DPLL
MODEM RECEIVE/CRC TRANSMIT/CRC TIMER/BRG/DPLL
2 SERIAL INTERFACE CHANNELS

1.1 Benefits

- Substantially reduced host CPU overhead means more channels and faster overall throughput.
- No time-critical host software enables faster and easier software development.

Features	CD2231	CD2401	CD2431	CD2481
Number of serial channels	2	4	4	4
Interrupt on-chip DMA mechanism	√[1]	√	√	√
FIFO depth (per channel and per direction)	16	16	16	16
Data size (bits)	5-8	5-8	5-8	5-8
ASYNC	√	√	√	Downloaded[2]
SDLC/HDLC	√	√	√	Downloaded
X.21, bisync	–	√	–	Downloaded
Async-HDLC, PPP	√	–	√	Downloaded
SLIP	(Revision D)	–	(Revision D)	Downloaded
MNP® 4	(Revision D)	–	(Revision D)	Downloaded
Serial data rate (kbits/second)	256[3,4]	134.4	134.4	230.4
Number of modem leads (per channel, including RxD and TxD)	10	10	10	10
On-chip timers	√	√	√	√
UNIX® character processing[5]	√	√	√	Downloaded
In-band Rx flow control	(Revision D)	–	(Revision D)	Downloaded
Special character Tx and recognition	√	√	√	Downloaded

Features	**CD2231**	**CD2401**	**CD2431**	**CD2481**
Package	100-pin MQFP	100-pin MQFP	100-pin MQFP	100-pin MQFP
System interface	Intel/Motorola	Intel/ Motorola	Intel/Motorola	Intel/Motorola
Pin compatibility	CD2401/ CD2431/ CD2481[6]	CD2431/ CD2231	CD2401/ CD2231	CD2401/ CD2431

NOTES:

1. √ indicates identical operation and register setting.
2. Device microcode is not user-programmable; microcode is supplied by Intel Corporation.
3. A clock frequency of 35 MHz is required to obtain maximum bit-rates (60 MHz for CD2481 Rev. B or later versions).
4. 256 kbps in sync mode, 2304 kbps in async modes. Applies to Revision D and later CD2231.
5. UNIX character processing is available in ASYNC only.
6. Compatibility with all pins except those supporting channels 2 and 3 on other family members. These pins are “no connect” or must be pulled-up to VCC by a 4.7-ohm resistor on the CD2231.

2.0 Conventions

Abbreviations

Symbol	Units of measure
°C	degree Celsius
µF	microfarad
µs	microsecond (1,000 nanoseconds)
Hz	hertz (cycle per second)
Kbit	kilobit (1,024 bits)
kbits/sec., kbps	kilobit (1,000 bits) per second
Kbyte	kilobyte (1,024 bytes)
kbytes/sec.	kilobyte (1,000 bytes) per second
kHz	kilohertz
kΩ	kilohm
Mbyte	megabyte (1,048,576 bytes)
MHz	megahertz (1,000 kilohertz)
mA	milliampere
ms	millisecond (1,000 microseconds)
ns	nanosecond
pV	picovolt
V	volt
W	watt

The use of ‘tbd’ indicates values that are ‘to be determined’, ‘n/a’ designates ‘not available’, and ‘n/c’ indicates a pin that is a ‘no connect’.

Acronyms

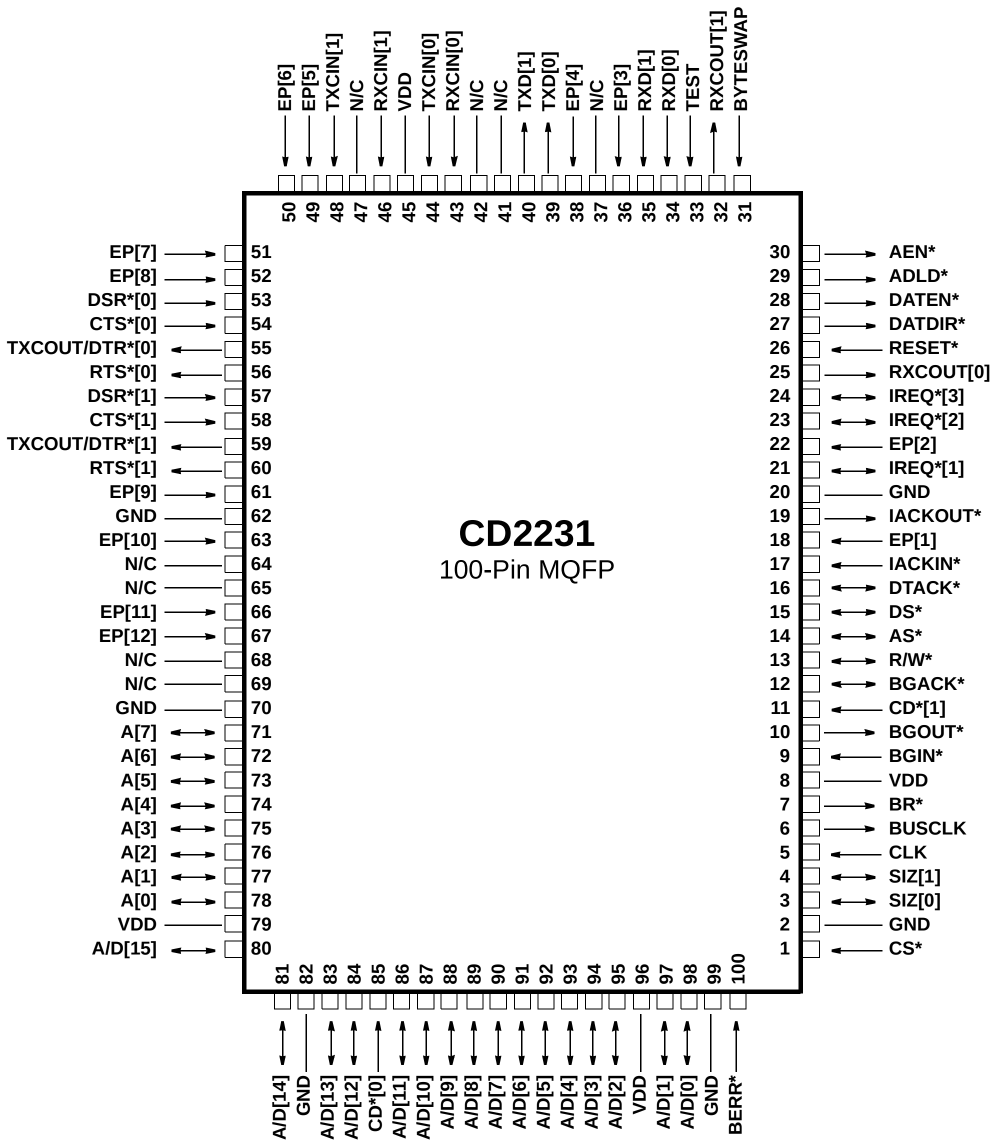
Acronym	Definition
AC	alternating current
BCC	block check character
BRG	bit rate generator
bisync	byte synchronous
CMOS	complementary metal-oxide semiconductor
CRC[1]	cyclic redundancy check
DC	direct current
DCE	data communication equipment

Acronym	Definition (Continued)
DMA	direct-memory access
DPLL	digital phase-locked loop
DRAM	dynamic random-access memory
DTE	data terminal equipment
EOF	end-of-frame
ETC	embedded transmit command
FCS[a]	frame check sequence
FCT	flow control transparency
FIFO	first in/first out
HDLC	high-level data link control
ISA	industry standard architecture
ITB	intermediate text block
LRC	longitudinal redundancy check character
LSB	least-significant bit
MSB	most-significant bit
NRZ	nonreturn to zero
NRZI	nonreturn to zero inverted
PPP	point-to-point protocol
MQFP	metric quad-flat pack
RAM	random-access memory
R/W	read/write
SDLC	synchronous data link control
TTL	transistor-transistor logic

1.The terms CRC and FCS are used interchangeably in this document.

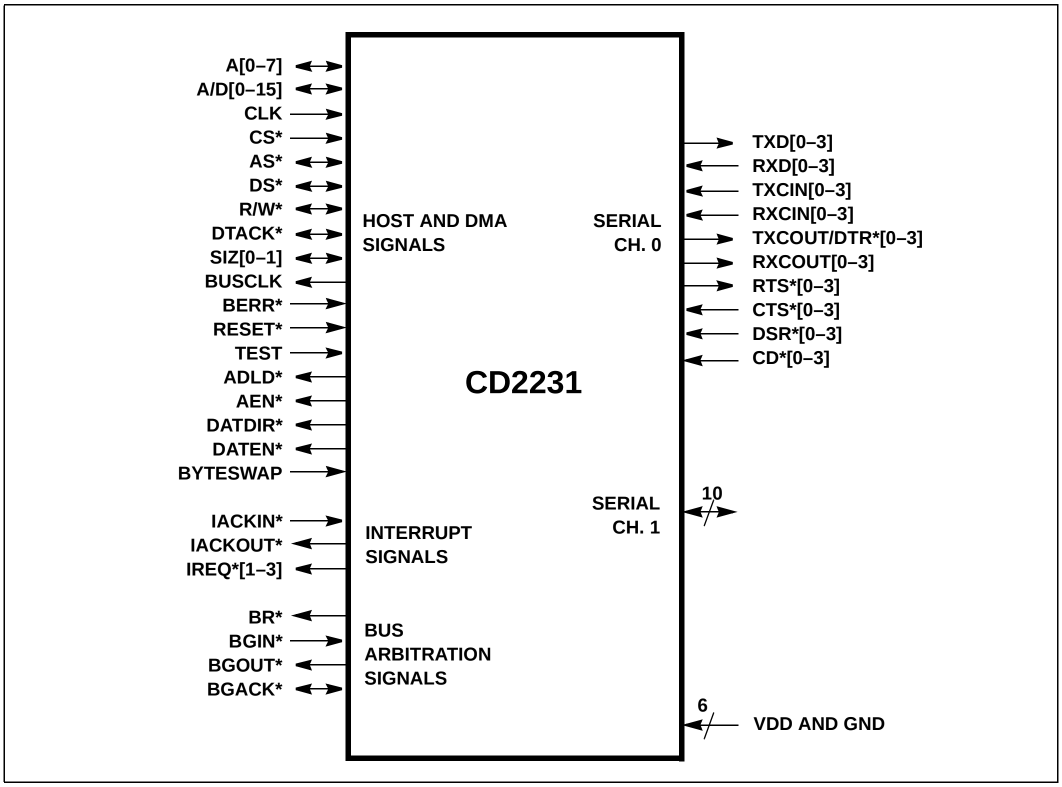
3.0 Pin Information

3.1 Pin Diagram



NOTE: N/C = No connection.
EP[1–12]:External pull-up required,
terminate to V_{CC} by a 4.7-kΩ resistor.

3.2 Pin Functions



3.3 Pin Descriptions

The following conventions are used in the pin-description tables:

- (*) after a name indicates that the signal is active-low
- ‘I’ indicates the pin is input-only
- ‘O’ indicates the pin is output-only
- ‘I/O’ indicates the pin is bidirectional
- ‘OD’ indicates open-drain; OD pins must be terminated to V_{CC} by a 2KΩ - 4.7KΩ resistor
- ‘TS’ indicates tristate
- a ‘–’ indicates ascending pin numbers
- a ‘:’ indicates descending pin numbers

Table 1. Pin Descriptions (Sheet 1 of 3)

Symbol	Pin Number	Type	Description
CS*	1	I	**CHIP SELECT*:** When low, the CD2231 registers can be read or written by the host processor.
AS*	14	I/O (TS)	**ADDRESS STROBE*:** When the CD2231 is a bus master, this pin is an output which indicates that R/W*, A[0–7], and the externally latched A[8–31] are valid.
DS*	15	I/O (TS)	**DATA STROBE*:** When the CD2231 is not a bus master, this is an input used to strobe data into registers during write cycles and enable data onto the bus during read cycles. When the CD2231 is a bus master, DS* is an output used to control data transfer to and from system memory.
R/W*	13	I/O (TS)	**READ/WRITE*:** When the CD2231 is not a bus master, this pin is an input which determines if a read or write operation is required when the CS* and DS* signals are active. When the CD2231 is a bus master, R/W* is an output and indicates whether a read from or a write to system memory is being performed.
DTACK*	16	I/O (OD)	**DATA TRANSFER ACKNOWLEDGE*:** When the CD2231 is not a bus master, this is an output and indicates to the host when a read or write to the CD2231 is complete. When BR* is driven low by the CD2231, DTACK* is an input which indicates that the system bus is no longer in use. When the CD2231 is a bus master, DTACK* is an input which indicates when system memory read and write cycles are complete.
SIZ[0–1]	3, 4	I/O (TS)	**SIZE [0–1]:** When not the active bus master, these are inputs that determine the size of the operand being read or written by the host. SIZ[1] SIZ[0] 0 1 Byte[†] 1 0 16 Bit 0 0 32 Bit[‡] 1 1 3 Bytes[‡] When the CD2231 is a bus master, this is an output determining the size of the operand being transferred to or from system memory. SIZ[1] SIZ[0] 0 1 Byte[†] 1 0 16 Bit [†] See BYTESWAP description. [‡] The CD2231 drives DTACK* even though the device does not respond to such byte alignment.
IACKIN*	17	I	**INTERRUPT ACKNOWLEDGE IN*:** This input, qualified with DS* and A[0–6], acknowledges CD2231 interrupts.
IACKOUT*	19	O	**INTERRUPT ACKNOWLEDGE OUT*:** This output is driven low during interrupt acknowledge cycles for which no internal interrupt is valid.
IREQ*[1–3]	21, 23, 24	I/O (OD)	**INTERRUPT REQUEST* [1–3]:** These outputs signal that the CD2231 has a valid interrupt for modem-lead activity (IREQ*[1]), transmit activity (IREQ*[2]), or receive activity (IREQ*[3]).
BR*	7	OD	**BUS REQUEST*:** This output is used to signal to the (open drain) host processor or bus arbiter that bus mastership is required by the CD2231.
BGIN*	9	I	**BUS GRANT IN*:** This input indicates that the bus is available after the current bus master relinquishes the bus.
BGOUT*	10	O	**BUS GRANT OUT*:** This output is asserted when BGIN* is low and no internal Bus Request has been made. A daisy-chain scheme of bus arbitration can be formed by connecting BGOUT* to BGIN* of the next device in the chain. If a priority scheme is preferred, bus requests must be prioritized externally and bus grant routed to the BGIN* of the appropriate device.

Table 1. Pin Descriptions (Sheet 2 of 3)

Symbol	Pin Number	Type	Description
BGACK*	12	I/O (OD)	**BUS GRANT ACKNOWLEDGE*:** As an input, this signal is used to determine if another alternate bus master is in control of the bus. As an output, it signals to other bus masters that this device is in control of the bus.
BERR*	100	I	**BUS ERROR*:** If this input becomes active while the CD2231 is a bus master, the current bus cycle is terminated, the bus relinquished, and an interrupt generated to indicate the error to the host processor.
A[7:0]	71–78	I/O (TS)	**ADDRESS [0–7]:** When the CD2231 is not a bus master, these pins are inputs used to determine which registers are being accessed, or which interrupt is being acknowledged. When ADLD* is low, A[0–7] output address bits 8 through 15 for external latching. When the CD2231 is a bus master, A[0–7] output the least-significant byte of the transfer address.
A/D[15:0]	80, 81, 83, 84, 86–95, 97, 98	I/O (TS)	**ADDRESS/DATA [0–15]:** When the CD2231 is not a bus master, these pins provide the 16-bit data bus for reading and writing to the CD2231 registers. When ADLD* is low, A/D[0–15] provide the upper address bits for external latching. When the CD2231 is a bus master, A/D[0–15] provide a multiplexed address/data bus for reading and writing to system memory.
ADLD*	29	O (TS)	**ADDRESS LOAD*:** This is a strobe used to externally latch the upper portion of the system address bus A[8–31]. While ADLD* is low, address bits 16–31 are available on A/D[0–15], and address bits 8 through 15 on A[0–7].
AEN*	30	O (TS)	**ADDRESS ENABLE*:** This output is used to output enable the external address bus drivers during CD2231 DMA cycles.
DATEN*	28	O (TS)	**DATA ENABLE*:** This output is active when either the CD2231 is a bus master, or the CS* and DS* pins are low. It is used to enable the external data bus buffers during host register read/write operations or during DMA operations. For operations on 32-bit buses, this signal needs to be gated with A[1] to select the correct half of the data bus.
DATDIR*	27	O (TS)	**DATA DIRECTION*:** This output is active when either the CD2231 is a bus master, or the CS* pin is low. It is used to control the external data buffers; when low, the buffers should be enabled in the CD2231 to system bus direction.
CLK	5	I	**CLOCK:** System clock.
BUSCLK	6	O	**BUS CLOCK:** This is the system clock (divided by 2) which is used internally to control certain bus operations. This pin is driven low during hardware reset.
RESET*	26	I	**RESET*:** This signal should stay valid for a minimum of 20 ns. The reset state of the CD2231 is guaranteed at the rising edge of this signal. When RESET* is removed, the CD2231 also performs a software initialization of its registers.
TEST	33	I	**TEST:** In normal operation, this pin should be kept low. For board-level testing purposes, it provides a mechanism for forcing normal output pins to High-Impedance mode. When the TEST pin is high, the following pins are in High-Impedance mode: BUSCLK, BGOUT*, IACKOUT*, RXCOUT[0–1], RTS*[0–1], DTR*[0–1], and TXD[0–1]. To ensure all CD2231 outputs are high-impedance, either of the following two conditions must be met: the RESET* pin can be driven low, and the TEST pin driven high; or, the CD2231 is kept in the bus idle state (not accessed for read/write operations nor DMA active), and the TEST pin is driven high.
RTS*[0–1]	56, 60	O	**REQUEST TO SEND* [0–3]:** This output can be controlled automatically by the CD2231 to indicate that data is being sent on the TXD pin.

Table 1. Pin Descriptions (Sheet 3 of 3)

Symbol	Pin Number	Type	Description
TXCOUT/DTR* [0–1]	55, 59	O	**TRANSMIT CLOCK OUT/DATA TERMINAL READY* [0–1]:** This output can be controlled automatically by the CD2231 to indicate a programmable threshold has been reached in the receive FIFO. It can also be programmed to output the transmit data clock. Following reset, this pin is high and stays high in Clock mode until the transmit channel is enabled for the first time; after which it remains active, independent of the state of the transmit enable. In all modes, the clock transitions every bit time, even during idle fill in Asynchronous mode. Data transitions are made on the negative-going edge of TXCOUT.
RXCOUT[0–1]	25, 32	O	**RECEIVE CLOCK OUT [0–1]:** This output provides a one-time bit rate clock for the receive data in all modes, except when an input (RXCIN) one-time receive clock is used. After reset, this pin is low until the channel is receive enabled for the first time, after which it remains active, independent of the state of receive enable. When in Asynchronous mode, the output only transitions while receiving data and not during inter-character fill. The receive data is sampled on the positive-going edge of this clock.
CTS*[0–1]	54, 58	I	**CLEAR TO SEND* [0–1]:** This input can be programmed to control the flow of transmit data, for out-of-band flow control applications.
CD*[0–1]	85, 11	I	**CARRIER DETECT* [0–1]:** This pin is always visible in the MSVR register. On the CD2231, these functions are separated onto two pins. When used as CD*, this input can be programmed to validate receive data.
TXCIN[0–1]	44, 48	I	**TRANSMIT CLOCK [0–1]:** This pin inputs the transmit clock to the bit rate generator.
RXCIN[0–1]	43, 46	I	**RECEIVE CLOCK [0–1]:** This pin inputs the receive clock to the bit rate generator.
DSR*[0–1]	53, 57	I	**DATA SET READY* [0–1]:** On the CD2231, these functions are separated onto two pins. When used as DSR*, this input can be programmed to validate receive data.
TXD[0–1]	39, 40	O	**TRANSMIT DATA [0–1]:** Serial data output for each channel.
RXD[0–1]	34, 35	I	**RECEIVE DATA [0–1]:** Serial data input for each channel.
BYTESWAP	31	I	**BYTESWAP:** This pin alters the byte ordering of data during certain 16-bit transfers and changes the half of the data bus on which byte transfers are made to comply with Intel® or Motorola® processor systems. BYTESWAP does not alter the bus handshake signals. When the BYTESWAP pin is high, the byte of A/D[0–7] precedes that of A/D[8–15] in a string of transmit or receive bytes; when BYTESWAP is low, A/D[8–15] precedes A/D[0–7]. When the BYTESWAP pin is high, bytes are transferred on A/D[0–7] when A[0] is low, and on A/D[8–15] when A[0] is high. When BYTESWAP is low, bytes are transferred on A/D[8–15] when A[0] is low, and A/D[0–7] when A[0] is high. A different register map is used, depending on the state of this pin. **Byteswap Byte Alignment** 0 Motorola® 1 Intel®
N/C	37, 41, 42, 47, 64, 65, 68, 69	–	**NO CONNECTION:** Make no connections to these pins. They must be left open for proper device operation.
EP[1–12]	18, 22, 36, 38, 49–52, 61, 63, 66, 67	I	**EXTERNAL PULL-UP [1–12]:** These pins must be terminated to V_{CC} by pull-up resistors for proper device operation. A value of 4.7 kΩ is recommended.
V_{DD}	8, 45, 79, 96	–	**POWER**
GND	2, 20, 62, 70, 99	–	**GROUND**

4.0 Register Table

Registers in the CD2231 are either Global or Per-Channel. The column ‘Address mode’ in the memory map on the following pages defines this attribute for each register. Only one set of Global registers exists, and are accessible by the host at any time. Two sets of Per-Channel registers exist, and the set accessible at any one time is determined by the currently active channel number. The channel number is selected by the host in normal (non-interrupt) processing by writing to the Channel Access register. The channel number in the Channel Access register remains in force until changed by the host. The channel number is provided automatically by the CD2231 during interrupt service routines and DMA transfers.

In the following list, some register locations appear twice. They have different names and functions for asynchronous and synchronous protocol operations. See Chapter 8.0 on page 88 of this datasheet for detailed descriptions of all register functions.

4.1 Memory Map

4.1.1 Global Registers

Name	Description	Addr. Mode[1]	INT[2]	MOT[3]	Size	Access	Page
GFRCR	Global Firmware Revision Code Register	G	82	81	B	R/W	88
CAR	Channel Access Register	G	EC	EE	B	R/W	88

The following notes are applicable for Section 4.1.1 through Section 4.1.8.

NOTES:

1. Address mode G: Global register — one set is always accessible.
   Address mode P: Per-Channel register — two sets, one per channel, accessible by CAR or interrupt context.
2. INT = address for Intel®-style processor.
3. MOT = address for Motorola®-style processor.

4.1.2 Option Registers

Name	Description	Addr. Mode[1]	INT[2]	MOT[3]	Size	Access	Page
CMR	Channel Mode Register	P	18	1B	B	R/W	89
COR1	Channel Option Register 1	P	13	10	B	R/W	90
COR2	Channel Option Register 2	P	14	17	B	R/W	92
COR3	Channel Option Register 3	P	15	16	B	R/W	94
COR4	Channel Option Register 4	P	16	15	B	R/W	98
COR5	Channel Option Register 5	P	17	14	B	R/W	99

Name	Description	Addr. Mode[1]	INT[2]	MOT[3]	Size	Access	Page
COR6	Channel Option Register 6	P	1B	18	B	R/W	100
COR7	Channel Option Register 7	P	04	07	B	R/W	101
SCHR1	Special Character Register 1	P	1C	1F	B	R/W Async	102
SCHR2	Special Character Register 2	P	1D	1E	B	R/W Async	102
SCHR3	Special Character Register 3	P	1E	1D	B	R/W Async	103
SCHR4	Special Character Register 4	P	1F	1C	B	R/W Async	103
SCRl	Special Character Range low	P	20	23	B	R/W Async	104
SCRh	Special Character Range high	P	21	22	B	R/W Async	104
LNXT	LNext Character	P	2D	2E	B	R/W Async	104
RFAR1	Receive Frame Address Register 1	P	1C	1F	B	R/W Sync	105
RFAR2	Receive Frame Address Register 2	P	1D	1E	B	R/W Sync	105
RFAR3	Receive Frame Address Register 3	P	1E	1D	B	R/W Sync	105
RFAR4	Receive Frame Address Register 4	P	1F	1C	B	R/W Sync	105
CPSR	CRC Polynomial Select Register	P	D4	D6	B	R/W Sync	106
TSPMAP1	Transmit Special Mapped Character 1	P	1B	18	B	R/W	106
TSPMAP2	Transmit Special Mapped Character 2	P	04	07	B	R/W	106
TSPMAP3	Transmit Special Mapped Character 3	P	2D	2E	B	R/W	107
TXACCM0	Transmit Async Control Character Map 0	P	1C	1F	B	R/W	107
TXACCM1	Transmit Async Control Character Map 1	P	1D	1E	B	R/W	107
TXACCM2	Transmit Async Control Character Map 2	P	1E	1D	B	R/W	108
TXACCM3	Transmit Async Control Character Map 3	P	1F	1C	B	R/W	108
RXACCM0	Receive Async Control Character Map 0	P	20	23	B	R/W	108
RXACCM1	Receive Async Control Character Map 1	P	21	22	B	R/W	109
RXACCM2	Receive Async Control Character Map 2	P	22	21	B	R/W	109
RXACCM3	Receive Async Control Character Map 3	P	23	20	B	R/W	109

4.1.3 Bit Rate and Clock Option Registers

Name	Description	Addr. Mode[1]	INT[2]	MOT[3]	Size	Access	Page
RBPR	Receive Baud Rate Period Register	P	C9	CB	B	R/W	110
RCOR	Receive Clock Option Register	P	CA	C8	B	R/W	110
TBPR	Transmit Baud Rate Period Register	P	C1	C3	B	R/W	111
TCOR	Transmit Clock Option Register	P	C2	C0	B	R/W	112

4.1.4 Channel Command and Status Registers

Name	Description	Addr. Mode[1]	INT[2]	MOT[3]	Size	Access	Page
CCR	Channel Command Register	P	10	13	B	R/W	112
STCR	Special Transmit Command Register	P	11	12	B	R/W	115
CSR	Channel Status Register	P	19	1A	B	R	117
MSVR-RTS	Modem Signal Value Registers	P	DC	DE	B	R/W	121
MSVR-DTR		P	DD	DF	B	R/W	121

4.1.4.1 Interrupt Registers

Name	Description	Addr. Mode[1]	INT[2]	MOT[3]	Size	Access	Page
LIVR	Local Interrupt Vector Register	P	0A	09	B	R/W	122
IER	Interrupt Enable Register	P	12	11	B	R/W	123
LICR	Local Interrupting Channel Register	P	25	26	B	R/W	124
STK	Stack Register	G	E0	E2	B	R	125

4.1.5 Receive Interrupt Registers

Name	Description	Addr. Mode[1]	INT[2]	MOT[3]	Size	Access	Page
RPILR	Receive Priority Interrupt Level Register	G	E3	E1	B	R/W	125
RIR	Receive Interrupt Register	G	EF	ED	B	R	126
RISR	Receive Interrupt Status Register	G	8A	88	W	R	127
RISRl	Receive Interrupt Status Register low	G	8A	89	B	R	127
RISRh	Receive Interrupt Status Register high	G	8B	88	B	R	130
RFOC	Receive FIFO Output Count	G	33	30	B	R	131
RDR	Receive Data Register	G	F8	F8	B	R	131
REOIR	Receive End of Interrupt Register	G	87	84	B	W	131

4.1.6 Transmit Interrupt Registers

Name	Description	Addr. Mode[1]	INT[2]	MOT[3]	Size	Access	Page
TPILR	Transmit Priority Interrupt Level Register	G	E2	E0	B	R/W	133
TIR	Transmit Interrupt Register	G	EE	EC	B	R	134
TISR	Transmit Interrupt Status Register	G	89	8A	B	R	135
TFTC	Transmit FIFO Transfer Count	G	83	80	B	R	135
TDR	Transmit Data Register	G	F8	F8	B	W	136
TEOIR	Transmit End of Interrupt Register	G	86	85	B	W	136

4.1.6.1 Modem Interrupt Registers

Name	Description	Addr. Mode[1]	INT[2]	MOT[3]	Size	Access	Page
MPILR	Modem Priority Interrupt Level Register	G	E1	E3	B	R/W	137
MIR	Modem Interrupt Register	G	ED	EF	B	R	137
MISR	Modem (/Timer) Interrupt Status Register	G	88	8B	B	R	138
MEOIR	Modem End of Interrupt Register	G	85	86	B	W	139

4.1.7 DMA Registers

Name	Description	Addr. Mode[1]	INT[2]	MOT[3]	Size	Access	Page
DMR	DMA Mode Register	G	F4	F6	B	W	139
BERCNT	Bus Error Retry Count	G	8D	8E	B	R/W	140
DMABSTS	DMA Buffer Status	P	1A	19	B	R	140

4.1.7.1 DMA Receive Registers

Name	Description	Addr. Mode[1]	INT[2]	MOT[3]	Size	Access	Page
ARBADRL	A Receive Buffer Address Lower	P	40	42	W	R/W	141
ARBADRU	A Receive Buffer Address Upper	P	42	40	W	R/W	142
BRBADRL	B Receive Buffer Address Lower	P	44	46	W	R/W	142
BRBADRU	B Receive Buffer Address Upper	P	46	44	W	R/W	142
ARBCNT	A Receive Buffer Byte Count	P	48	4A	W	R/W	143
BRBCNT	B Receive Buffer Byte Count	P	4A	48	W	R/W	143
ARBSTS	A Receive Buffer Status	P	4C	4F	B	R/W	143
BRBSTS	B Receive Buffer Status	P	4D	4E	B	R/W	144
RCBADRL	Receive Current Buffer Address Lower	P	3C	3E	W	R	144
RCBADRU	Receive Current Buffer Address Upper	P	3E	3C	W	R	145

4.1.7.2 DMA Transmit Registers

Name	Description	Addr. Mode[1]	INT[2]	MOT[3]	Size	Access	Page
ATBADRL	A Transmit Buffer Address Lower	P	50	52	W	R/W	145
ATBADRU	A Transmit Buffer Address Upper	P	52	50	W	R/W	146
BTBADRL	B Transmit Buffer Address Lower	P	54	56	W	R/W	146
BTBADRU	B Transmit Buffer Address Upper	P	56	54	W	R/W	146
ATBCNT	A Transmit Buffer Byte Count	P	58	5A	W	R/W	147
BTBCNT	B Transmit Buffer Byte Count	P	5A	58	W	R/W	147

Name	Description	Addr. Mode[1]	INT[2]	MOT[3]	Size	Access	Page
ATBSTS	A Transmit Buffer Status	P	5C	5F	B	R/W	147
BTBSTS	B Transmit Buffer Status	P	5D	5E	B	R/W	149
TCBADRL	Transmit Current Buffer Address Lower	P	38	3A	W	R	152
TCBADRU	Transmit Current Buffer Address Upper	P	3A	38	W	R	152

4.1.8 Timer Registers

Name	Description	Addr. Mode[1]	INT[2]	MOT[3]	Size	Access	Page
TPR	Timer Period Register	G	D8	DA	B	R/W	152
RTPR	Receive Timeout Period Register	P	26	24	W	R/W Async	153
RTPRl	Receive Timeout Period Register low	P	26	25	B	R/W Async	153
RTPRh	Receive Timeout Period Register high	P	27	24	B	R/W Async	153
GT1	General Timer 1	P	28	2A	W	R/W Sync	154
GT1l	General Timer 1 low	P	28	2B	B	R/W Sync	154
GT1h	General Timer 1 high	P	29	2A	B	R/W Sync	154
GT2	General Timer 2	P	2A	29	B	R/W Sync	154
TTR	Transmit Timer Register	P	2A	29	B	R/W Async	155

4.2 Register Definitions

4.2.1 Global Registers

Global Firmware Revision Code Register (GFRCR) 82 81 B R/W

Firmware Revision Code

Channel Access Register (CAR) EC EE B R/W

0	0	0	0	0	0	0	C0

4.2.2 Option Registers

Channel Mode Register (CMR) 18 1B B R/W

RxMode	TxMode	0	0	0	chmd2	chmd1	chmd0

Channel Option Register 1 (COR1) 13 10 B R/W
HDLC Mode

AFLO	ClrDet	AdMde1	AdMde0	Flag3	Flag2	Flag1	Flag0

Asynchronous Mode

Parity	ParM1	ParM0	Ignore	Chl3	Chl2	Chl1	Chl0

Channel Option Register 2 (COR2) 14 17 B R/W
Asynchronous / Async-HDLC / PPP Mode

IXM	TxIBE	0	0	RLM	RtsAO	CtsAE	DsrAE

HDLC Mode

0	FCSApd	0	CRCNinv	0	RtsAO	CtsAE	DsrAE

MNP4/SLIP Mode

0	0	0	0	RLM	RtsAO	CtsAE	DsrAE

Channel Option Register 3 (COR3) 15 16 B R/W
Async-HDLC/PPP Mode

Stop2	FCSApd	RxChk	TxGen	npad3	npad2	npad1	npad0

MNP4 Mode

Stop2	FCSApd	RxChk	TxGen	npad3	npad2	npad1	npad0

HDLC Mode

sndpad	Alt1	FCSPre	FCS	idle	npad2	npad1	npad0

Asynchronous Mode

EDCDE	RngDE	FCT	SCDE	Splstp	Stop2	Stop1	Stop0

SLIP Mode

Stop2	0	0	0	npad3	npad2	npad1	npad0

Channel Option Register 4 (COR4) 16 15 B R/W

DSRzd	CDzd	CTSzd	0	FIFO Threshold			

Channel Option Register 5 (COR5) 17 14 B R/W

DSRod	CDod	CTSod	In/Out	Rx Flow Control Threshold			

Channel Option Register 6 (COR6) 1B 18 B R/W
Asynchronous Mode

IgnCR	ICRNL	INLCF	IgnBrk	NBrkInt	ParMrk	INPCK	ParInt

Channel Option Register 7 (COR7) 04 07 B R/W
Asynchronous Mode

IStrip	LNE	FCErr	0	0	0	ONLCR	OCRNL

Special Character Registers

Special Character Register 1 (SCHR1)	**1C**	**1F**	**B**	**R/W Async**
Special Character Register 2 (SCHR2)	**1D**	**1E**	**B**	**R/W Async**
Special Character Register 3 (SCHR3)	**1E**	**1D**	**B**	**R/W Async**
Special Character Register 4 (SCHR4)	**1F**	**1C**	**B**	**R/W Async**

Special Character Ranges

Special Character Range low (SCRl)	**20**	**23**	**B**	**R/W Async**
Special Character Range high (SCRh)	**21**	**22**	**B**	**R/W Async**

LNext Character (LNXT)	**2D**	**2E**	**B**	**R/W Async**

Receive Frame Address Registers

Receive Frame Address Register 1 (RFAR1)	**1C**	**1F**	**B**	**R/W Sync**
Receive Frame Address Register 2 (RFAR2)	**1D**	**1E**	**B**	**R/W Sync**
Receive Frame Address Register 3 (RFAR3)	**1E**	**1D**	**B**	**R/W Sync**
Receive Frame Address Register 4 (RFAR4)	**1F**	**1C**	**B**	**R/W Sync**

CRC Polynomial Select Register (CPSR)	**D4**	**D6**	**B**	**R/W**

0	0	0	0	0	0	0	poly

Transmit Special Mapped Characters (PPP only)

Transmit Special Mapped Character 1 (TSPMAP1)	**1B**	**18**	**B**	**R/W PPP**
Transmit Special Mapped Character 2 (TSPMAP2)	**04**	**07**	**B**	**R/W PPP**
Transmit Special Mapped Character 3 (TSPMAP3)	**2D**	**2E**	**B**	**R/W PPP**

Transmit Async Control Character Maps (PPP only)

Transmit Async Control Character Map 0 (TXACCM0)	**1C**	**1F**	**B**	**R/W PPP**
Transmit Async Control Character Map 1 (TXACCM1)	**1D**	**1E**	**B**	**R/W PPP**
Transmit Async Control Character Map 2 (TXACCM2)	**1E**	**1D**	**B**	**R/W PPP**
Transmit Async Control Character Map 3 (TXACCM3)	**1F**	**1C**	**B**	**R/W PPP**

Receive Async Control Character Maps (PPP only)

Receive Async Control Character Map 0 (RXACCM0)	**20**	**23**	**B**	**R/W PPP**
Receive Async Control Character Map 1 (RXACCM1)	**21**	**22**	**B**	**R/W PPP**
Receive Async Control Character Map 2 (RXACCM2)	**22**	**21**	**B**	**R/W PPP**
Receive Async Control Character Map 3 (RXACCM3)	**23**	**20**	**B**	**R/W PPP**

4.2.3 Bit Rate and Clock Option Registers

Receive Bit Rate Period Register (RBPR)	**C9**	**CB**	**B**	**R/W**

Receive Bit Rate Period (Divisor)

Receive Clock Option Register (RCOR) CA C8 B R/W

TLVal	res	DpllEn	Dpllmd1	Dpllmd0	ClkSel2	ClkSel1	ClkSel0

Transmit Bit Rate Period Register (TBPR) C1 C3 B R/W

Transmit Bit Rate Period (Divisor)

Transmit Clock Option Register (TCOR) C2 C0 B R/W

ClkSel2	ClkSel1	ClkSel0	res	Ext-1X	res	LLM	res

4.2.4 Channel Command and Status Registers

Channel Command Register (CCR) 10 13 B R/W
Mode 1

0	ClrCh	InitCh	RstAll	EnTx	DisTx	EnRx	DisRx

Mode 2

1	ClrT1	ClrT2	ClrRcv	0	0	0	0

Special Transmit Command Register (STCR) 11 12 B R/W
Async-HDLC/PPP Mode

0	Abort	0	0	sndsp	frame	Xon	Xoff

SLIP/MNP4 Mode

0	Abort	0	0	sndsp	frame	0	0

Asynchronous and HDLC Modes

0	AbortTx	AppdCmp	0	SndSpc	SSPC2	SSPC1	SSPC0

Channel Status Register (CSR) 19 1A B R
HDLC Mode

RxEn	RxFlag	RxFrame	RxMark	TxEn	TxFlag	TxFrame	TxMark

Asynchronous Mode

RxEn	RxFloff	RxFlon	0	TxEn	TxFloff	TxFlon	0

Async-HDLC/PPP Mode

RxEn	RxFloff	RFram	RIdle	TxEn	TxFloff	TFram	TIdle

SLIP/MNP4 Mode

RxEn	0	RFram	RIdle	TxEn	0	TFram	TIdle

Modem Signal Value Registers (MSVR) R/W
Modem Signal Value Register (MSVR-RTS) DC DE B R/W
Modem Signal Value Register (MSVR-DTR) DD DF B R/W

DS	CD	CTS	DTRop	0	0	DTR	RTS

4.2.5 Interrupt Registers

Local Interrupt Vector Register (LIVR) 0A 09 B R/W

X	X	X	X	X	X	IT1	IT0

Interrupt Enable Register (IER) 12 11 B R/W

Mdm	0	RET	0	RxD	TIMER	TxMpty	TxD

Local Interrupting Channel Register (LICR) 25 26 B R/W

X	X	X	X	X	C0	X	X

Interrupt Stack Register (STK) E0 E2 B R

CLvl [1]	MLvl [1]	TLvl [1]	0	0	TLvl [0]	MLvl [0]	CLvl [0]

4.2.5.1 Receive Interrupt Registers

Receive Priority Interrupt Level Register (RPILR) E3 E1 B R/W
Receive Interrupt Register (RIR) EF ED B R

Ren	Ract	Reoi	0	Rvct [1]	Rvct [0]	0	Rcn [0]

Receive Interrupt Status Register (RISR) 8A 88 W R
Receive Interrupt Status Register low (RISRl) 8A 89 B R
HDLC Mode

0	EOF	RxAbt	CRC	OE	ResInd	0	ClrDct

Asynchronous Mode

Timeout	SCdet2	SCdet1	SCdet0	OE	PE	FE	Break

Async-HDLC / PPP / MNP4 Mode

0	EOF	RxAbt	CRC	OE	FE	0	Break

SLIP Mode

0	EOF	0	0	OE	FE	0	Break

Receive Interrupt Status Register high (RISRh) 8B 88 B R

Berr	EOF	EOB	0	BA/BB	0	0	0

Receive FIFO Output Count (RFOC) 33 30 B R

0	0	0	RxCt4	RxCt3	RxCt2	RxCt1	RxCt0

Receive Data Register (RDR) F8 F8 B R

D7	D6	D5	D4	D3	D2	D1	D0

Receive End of Interrupt Register (REOIR) 87 84 B W
Asynchronous and HDLC Modes

TermBuff	DiscExc	SetTm2	SetTm1	NoTrans	Gap2	Gap1	Gap0

Async-HDLC / PPP / SLIP / MNP4 Modes

TermBuff	DiscExc	SetTm2	SetTm1	NoTrans	0	0	0

4.2.5.2 Transmit Interrupt Registers

Transmit Priority Interrupt Level Register (TPILR) E2 E0 B R/W
Transmit Interrupt Register (TIR) EE EC B R

Ten	Tact	Teoi	0	Tvct [1]	Tvct [0]	0	Tcn [0]

Transmit Interrupt Status Register (TISR) 89 8A B R

Berr	EOF	EOE	UE	BA/BB	0	TxEmpty	TxDat

Transmit FIFO Transfer Count (TFTC) 83 80 B R

0	0	0	TxCt4	TxCt3	TxCt2	TxCt1	TxCt0

Transmit Data Register (TDR) F8 F8 B W

D7	D6	D5	D4	D3	D2	D1	D0

Transmit End of Interrupt Register (TEOIR) 86 85 B W

TermBuff	EOF	SetTm2	SetTm1	Notrans	0	0	0

4.2.5.3 Modem/Timer Interrupt Registers

Modem Priority Interrupt Level Register (MPILR) E1 E3 B R/W
Modem Interrupt Register (MIR) ED EF B R

Men	Mact	Meo	0	Mvct [1]	Mvct [0]	0	Mcn [0]

Modem (/Timer) Interrupt Status Register (MISR) 88 8B B R

DSRChg	CDChg	CTSChg	res	res	res	Timer2	Timer1

Modem End of Interrupt Register (MEOIR) 85 86 B W

0	0	SetTm2	SetTm1	0	0	0	0

4.2.6 DMA Registers

DMA Mode Register (DMR) F4 F6 F8 W

EnSync	0	0	0	ByteDMA	0	0	0

Bus Error Retry Count (BERCNT)	**8D**	**8E**	**B**	**R/W**

Binary Value

DMA Buffer Status (DMABSTS)	**1A**	**19**	**B**	**R**

TDAlign	RstApd	CrtBuf	Append	Ntbuf	Tbusy	Nrbuf	Rbusy

4.2.6.1 DMA Receive Registers

A Receive Buffer Address Lower (ARBADRL)	**40**	**42**	**W**	**R/W**
A Receive Buffer Address Upper (ARBADRU)	**42**	**40**	**W**	**R/W**
B Receive Buffer Address Lower (BRBADRL)	**44**	**46**	**W**	**R/W**
B Receive Buffer Address Upper (BRBADRU)	**46**	**44**	**W**	**R/W**

A Buffer Receive Byte Count (ARBCNT)	**48**	**4A**	**W**	**R**
B Buffer Receive Byte Count (BRBCNT)	**4A**	**48**	**W**	**R**

A Receive Buffer Status (ARBSTS)	**4C**	**4F**	**B**	**R/W**
B Receive Buffer Status (BRBSTS)	**4D**	**4E**	**B**	**R/W**

Berr	EOF	EOB	0	0	0	0	2231own

Receive Current Buffer Address Lower (RCBADRL)	**3C**	**3E**	**W**	**R**
Receive Current Buffer Address Upper (RCBADRU)	**3E**	**3C**	**W**	**R**

4.2.6.2 DMA Transmit Registers

A Transmit Buffer Address Lower (ATBADRL)	**50**	**52**	**W**	**R/W**
A Transmit Buffer Address Upper (ATBADRU)	**52**	**50**	**W**	**R/W**
B Transmit Buffer Address Lower (BTBADRL)	**54**	**56**	**W**	**R/W**
B Transmit Buffer Address Upper (BTBADRU)	**56**	**54**	**W**	**R/W**

A Buffer Transmit Byte Count (ATBCNT)	**58**	**5A**	**W**	**R/W**
B Buffer Transmit Byte Count (BTBCNT)	**5A**	**58**	**W**	**R/W**

A Transmit Buffer Status (ATBSTS)	**5C**	**5F**	**B**	**R/W**
B Transmit Buffer Status (BTBSTS)	**5D**	**5E**	**B**	**R/W**

Async-HDLC/PPP Mode

Berr	EOF	EOB	0	0	map32	INTR	2231own

SLIP/MNP4 Mode

Berr	EOF	EOB	0	0	0	INTR	2231own

Asynchronous and HDLC Mode

Berr	EOF	EOB	UE	Append	0	INTR	2231own

Transmit Current Buffer Address Lower (TCBADRL)	**38**	**3A**	**W**	**R**
Transmit Current Buffer Address Upper (TCBADRU)	**3A**	**38**	**W**	**R**

4.2.7 Timer Registers

Timer Period Register (TPR)	**D8**	**DA**	**B**	**R/W**

Binary Value

Receive Timeout Period Register (RTPR)	**26**	**24**	**W**	**R/W Async**
Receive Timeout Period Register low (RTPRl)	**26**	**25**	**B**	**R/W Async**

Binary Value, bits 7:0

Receive Timeout Period Register high (RTPRh)	**27**	**24**	**B**	**R/W Async**

Binary Value, bits 15:8

General Timer 1 (GT1)	**28**	**2A**	**W**	**R/W Sync**
General Timer 1 low (GT1l)	**28**	**2B**	**B**	**R/W Sync**
General Timer 1 high (GT1h)	**29**	**2A**	**B**	**R/W Sync**
General Timer 2 (GT2)	**2A**	**29**	**B**	**R/W Sync**

Transmit Timer Register (TTR)	**2A**	**29**	**B**	**R Async**

5.0 Functional Description

5.1 Host Interface

The CD2231 is a synchronous device with an asynchronous bus interface. A stable input clock is required on the CLK pin — nominally 33 MHz. The CLK is divided by two (2) internally, and the resulting signal is an output on the BUSCLK pin. The baud-rate generators and timers are also related to CLK. The "AC Electrical Characteristics" in Chapter 9.0 shows that many input signal setup and output signal transitions are related to the edges of the CLK and BUSCLK signals. It is possible, however, to use the CD2231 in a purely asynchronous bus environment.

The CD2231 can act either as a bus master during DMA transfers, or as a bus slave device during normal host read and write transfers. Both byte and word transfers are supported in each of the Bus Slave and DMA Bus Master modes. Figure 2 and Figure 3 show the signals involved in these transfers.

5.1.1 Host Read and Write Cycles

The host read and write cycles begin with the activation of the CS* (chip select) and DS* (data strobe) signals. The DATADIR* (data direction) and DATEN* (data enable) signals control external data buffers. The falling edge of the DTACK* (data transfer acknowledge) signal indicates that the transfer is complete. At that time, DTACK* is released when DS* is deasserted. CS* should also be deasserted. The AS* (address strobe) is not used during slave cycles; it is an output during DMA transfers.

Note that the following open-drain and tristate outputs should have pull-up resistors attached: AEN*, AS*, DATADIR*, DATEN*, and DTACK*.

Figure 2. Host Read Cycle

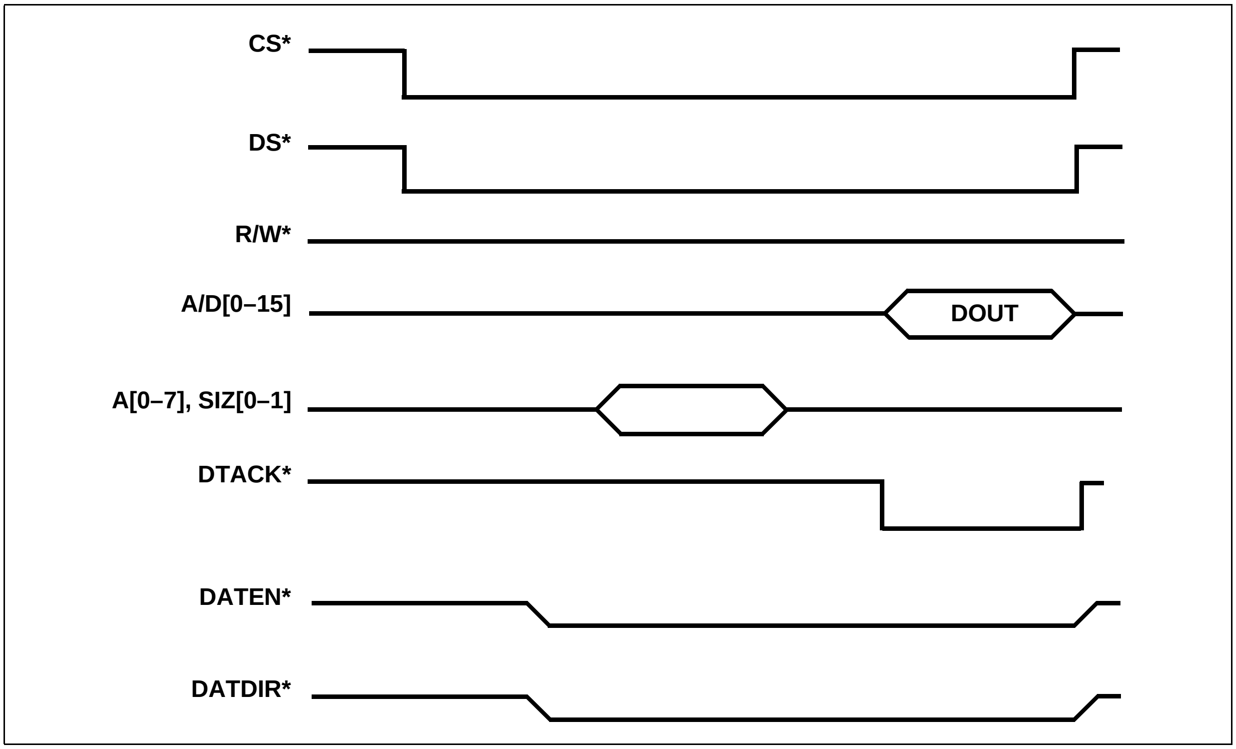
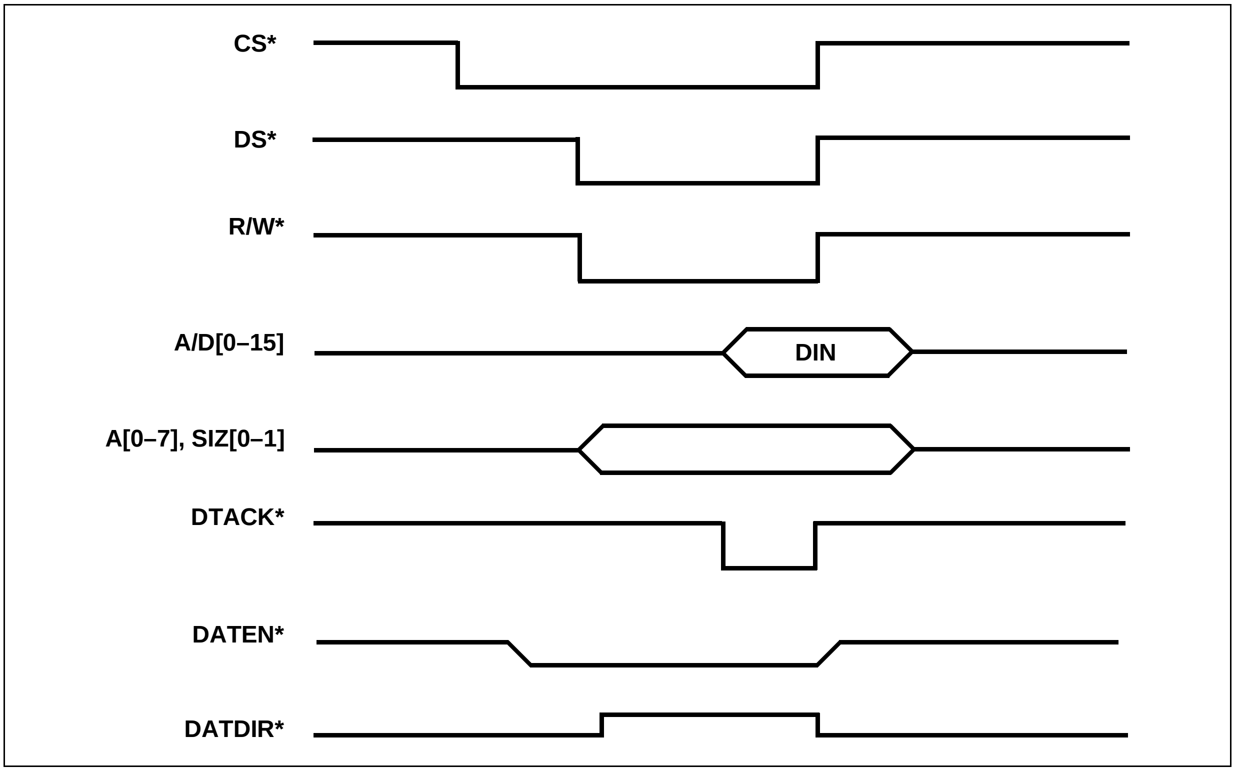


Figure 3. Host Write Cycle



5.1.2 Byte and Word Transfers

Data can be moved to and from the CD2231 in either byte or word transfers. To accommodate various families of host processors, the BYTESWAP input pin is set to indicate the system byte-ordering scheme. The SIZ pins (SIZ[1, 0]) are used to indicate whether the transfer is 1 or 2 bytes wide.

In systems where the even addresses represent the most-significant byte, the BYTESWAP input pin should be tied low, and byte transfers occur on the A/D[15:8] pins for even addresses and on the A/D[7:0] pins for odd addresses. In systems where the most-significant byte is on the odd address, the situation is reversed, and BYTESWAP should be tied high. Byte transfers to even addresses occur on the A/D[7:0] pins, and to odd addresses on the A/D[15:8] pins.

5.2 Interrupts

The CD2231 uses interrupt requests to alert the host that certain events have occurred. Interrupt operations on the CD2231 are tightly coupled with several registers described later. The concept of context affects the accessibility of these and other registers.

5.2.1 Contexts and Channels

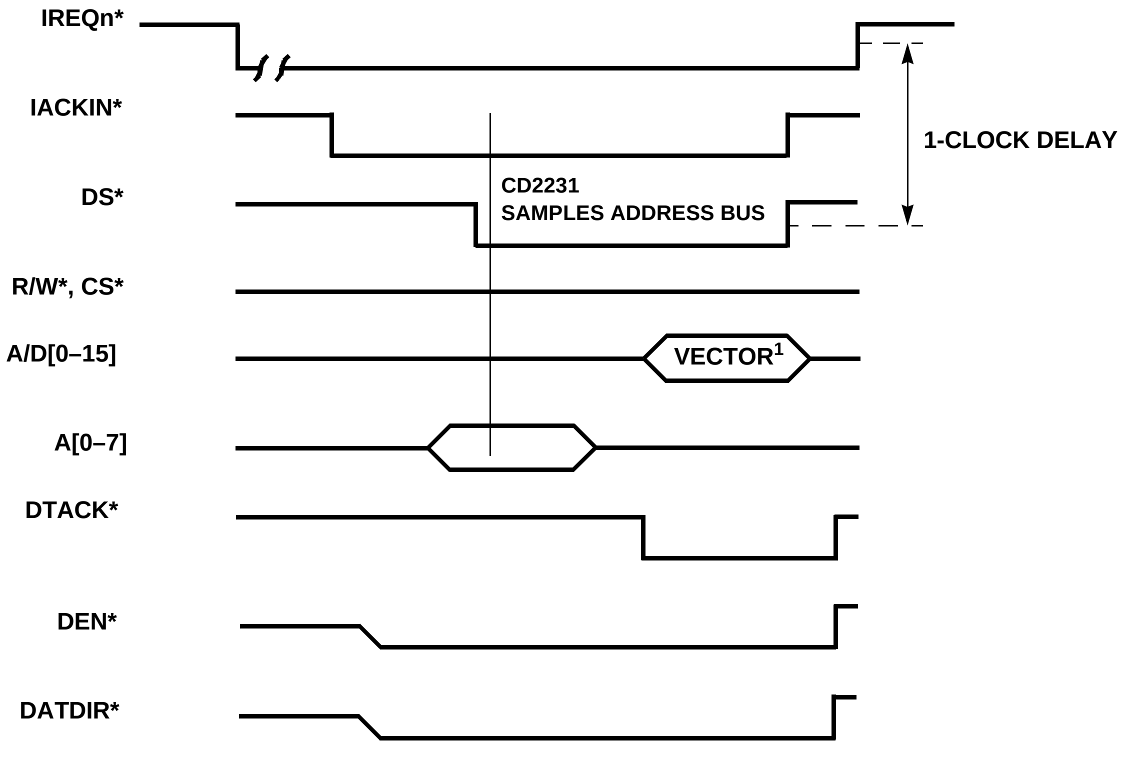
The registers in the CD2231 are grouped into Global, Virtual, and two sets of Per-Channel registers. The CD2231 is normally in the background context, where the CAR (Channel Access register) selects the channel number for the Per-Channel registers. The interrupt context begins with the interrupt acknowledge bus cycle, and ends with a write access to the appropriate End of Interrupt register. In the interrupt context, only the Per-Channel registers for the channel number being serviced are available; the CAR has no effect. Most Global registers are available at all times, but some are shared by the two channels, such as the FIFO registers. These are called Virtual registers, and must be accessed only during an interrupt context.

Interrupt contexts can be nested so that a higher-priority interrupt service can preempt a lower-priority interrupt already in progress. The CD2231 pushes the current interrupt context onto the stack, visible in the STK (Stack register), and enters the context for the newly acknowledged interrupt. Any register accesses are in the new interrupt context until the host performs a write to the appropriate EOIR for the top-level context. The CD2231 then pops the top-level context off the stack and returns to the previous interrupt context.

5.2.2 Interrupt Registers

The IER (Interrupt Enable register) and the LIVR (Local Interrupt Vector register) are Per-Channel registers. IER contains bits to enable or disable the various interrupt sources within the CD2231. The LIVR value is output on the data bus during the interrupt acknowledge cycle. There are sets of three Global registers that correspond to the three types of interrupts: Receive, Transmit, and Modem. The Priority Interrupt Level registers — RPILR, TPILR, and MPILR are programmed to contain the value that is present on the address bus during the interrupt acknowledge bus cycle for each type of interrupt. The Interrupt Status registers — RISR, TISR, or MISR are examined during the interrupt service routine to determine the cause of each type of interrupt. The TDR (Transmit Data) and RDR (Receive Data) registers provide access to the FIFO buffers for each channel. These registers must not be accessed outside of the proper interrupt context. A write operation to the End of Interrupt registers — REOIR, TEOIR, or MEOIR must be the last access to the CD2231 at the end of this handler routine to return it to its background context.

Figure 4. Interrupt Acknowledge Cycle



[1] Interrupt vector is always on A/D[0–7].

5.2.3 Groups and Types

There are two general reasons for the CD2231 to request service from the host processor — data transfer and exceptional conditions. Furthermore, interrupts are grouped into three categories, each with an associated Interrupt Request signal — IREQ1*, IREQ2*, and IREQ3*.

- Group 1 — Modem signal change/timer events
- Group 2 — Transmit interrupts
- Group 3 — Receive interrupts

Group 1 is only used for exceptions. Groups 2 and 3 include both data transfer and exceptions. Table 2 shows the possible causes of transmit and receive interrupt service requests. The cause of an interrupt request is encoded into the two least-significant bits of the vector presented on the data bus during the interrupt acknowledge cycle. The most-significant six bits of the vector come from the LIVR:

Interrupt Vector LSBs

00	Receive exception
01	Modem signal change or timer event
10	Transmit data or exception
11	Receive Good Data™

Table 2. Transmit and Receive Interrupt Service Requests

Interrupt Cause	ASYNC	HDLC	PPP	SLIP	MNP®4	Comments
Receive Good Data™	•	•	•	•	•	Not in DMA mode
Break detect	•		•	•	•	
Framing error	•		•	•	•	
Parity error	•					
Receive timeout, no data	•		•	•	•	
Special character match	•					
Transmitter empty	•	•	•	•	•	
Tx FIFO threshold	•	•	•	•	•	Not in DMA mode
Receive overrun	•	•	•	•	•	
Clear detect		•				
CRC error		•	•		•	
Residual bit count		•				
Receive abort		•	•	•	•	
End of frame		•	•	•	•	
Transmit underrun		•				
Bus error	•	•	•	•	•	DMA mode only
End of buffer	•	•	•	•	•	DMA mode only

5.2.4 Hardware Signals and IACK Cycles

The IACK (interrupt acknowledge) bus cycle begins with the IACKIN* (interrupt acknowledge in) and DS* asserted, and a value matching the appropriate PILR contents on the least-significant seven address bus bits, A[6:0]. If the IACK cycle is valid (that is, the PILR values match), the corresponding vector from the interrupting channel LIVR is driven onto the data bus and DTACK* is asserted. DTACK* is released after DS* is removed.

Figure 4 on page 38 shows the interrupt acknowledge cycle timing. It is similar to the basic host read cycle, except that IACKIN* is active and CS* is inactive.

The three IREQn* pins are open-drain outputs requiring external pull-up resistors, nominally 4.7 kΩ. The IACKOUT* (interrupt acknowledge out) is used to form a daisy chain in systems with more than one CD2231.

5.2.4.1 Programming the PILR Registers

The three PILRs (Priority Interrupt Level Registers) must be programmed with values that correspond to the least-significant seven address bits present on A[6:0] during the interrupt acknowledge bus cycle. Some CPUs output the priority level of the interrupts that are being acknowledged on the bus during the IACK cycle. In these systems the three PILR values are unique. In other systems that do not use this scheme, the PILR values can be the same or different depending on the specific design. When all of the PILRs contain the same value and multiple IREQn* lines are asserted, the CD2231 imposes the following priority scheme to determine which interrupt request are acknowledged:

Highest priority:	Receive Interrupt register Transmit Interrupt register
Lowest priority:	Modem Interrupt register

5.2.4.2 Systems with Interrupt Controllers

Some systems use an interrupt controller that supplies its own vector during the interrupt acknowledge cycle. To function properly, the CD2231 needs an IACK cycle in response to its interrupt request. These systems can decode three distinct locations from the CD2231 to produce an IACKIN* instead of CS*. The PILRs should be programmed with the addresses of these three locations.

Alternatively, a single location can be decoded and the three PILRs given identical values as described earlier. In either case, the host should read one of these locations before the first access to the device in an interrupt service routine. The CD2231 enters its interrupt acknowledge context for the proper type and channel, and the data returned is the device interrupt vector from the LIVR.

5.2.5 Multi CD2231 Systems

Multiple CD2231s can be chained together for systems requiring more than two channels. Each group of interrupt request lines — IREQn* — can be connected in a parallel wired-OR fashion. The system Interrupt Acknowledge signal is connected to the IACKIN* pin of the first device, its IACKOUT* is then connected to the IACKIN* of the next device, and so on, forming a chain of CD2231s.

5.2.5.1 Keep-and-Pass Logic

The acceptance of an interrupt acknowledge cycle by the CD2231 depends on whether the part is requesting service and whether the least-significant seven address bits match the contents of the appropriate PILR. The following rules apply to the keep-and-pass logic:

1. If the CD2231 does not have an interrupt asserted, the interrupt acknowledge is passed out on IACKOUT*.
2. If the CD2231 is asserting one or more of its interrupts, but the interrupt priority levels driven on the address bus by the host do not match the contents of the appropriate PILR, this interrupt acknowledge is also passed out on the IACKOUT*.
3. If the CD2231 is asserting an interrupt and the interrupt priority level on the address bus matches the PILR for that interrupt type, the interrupt acknowledge is accepted by the CD2231, and the vector from the LIVR is driven onto the data bus.

5.2.5.2 Fair Share Scheme

When multiple CD2231s are chained, the Fair Share logic in these devices guarantees that the interrupts from all CD2231s in the system are presented to the host with equal urgency. There is no positional hierarchy in the interrupt scheme. For example, the CD2231 farthest from the host has an equal chance of getting its interrupts through as the CD2231 nearest to the top of the interrupt chain. The Fair Share scheme is transparent to the user, and no enabling or disabling is required.

When an interrupt request line is asserted, the Fair bit for that type of interrupt on the asserting device is cleared. The Fair bit remains cleared until the interrupt line returns to a high state. The CD2231 does not assert a new interrupt of that type while the corresponding Fair bit is cleared. Therefore, when multiple CD2231s assert interrupts together, each one is serviced in turn, before they can reassert the same interrupt type.

The IREQn* lines are open-drain outputs that can be tied together in groups of the same type, creating a Fair Share scheme for each group of interrupts. Alternatively, all three groups can be tied to a common request using the CD2231 internal-priority scheme (see Section 5.2.4.1).

5.3 FIFO and Timer Operations

Each channel in the CD2231 has a 16-byte receive FIFO and a 16-byte transmit FIFO. The FIFOs are accessible through RDR (Receive Data register) and TDR (Transmit Data register) registers. These Virtual registers are shared among the two channels; therefore, they cannot be accessed outside an interrupt context.

The threshold level of each channel is common for both FIFOs and is set by COR4 (Channel Option Register 4), with a maximum threshold value of 12. The FIFO threshold is meaningful in both DMA and non-DMA modes. In DMA mode, the FIFO threshold determines when transfer bursts should occur. In non-DMA mode, the threshold level determines when transfer interrupts are asserted.

5.3.1 Receive FIFO Operation

In the Asynchronous mode, a Good Data interrupt is initiated when the number of characters in the FIFO is greater than the FIFO threshold. Note that receive timeout and receive data exception conditions also cause an interrupt to the host.

In the Synchronous mode, an interrupt request for data transfer is initiated when the number of characters is greater than the FIFO threshold or an end of frame is reached.

5.3.2 Transmit FIFO Operation

The TxDat and TxEmpty bits in the IER control the generation of transmit FIFO interrupts. The CD2231 initiates an interrupt request for more data when the number of empty bytes in the FIFO is greater than the threshold set. During synchronous operation when the last byte of the frame is transferred to the FIFO, the CD2231 stops asserting transmit interrupts until the frame is sent.

5.3.3 Timers

The global TPR (Timer Period register) provides a timer prescale ‘tick’ as a clock source for the timers. The TPR counter is clocked by the system clock (CLK) divided by 2048. To maintain timer accuracy, the TPR should not be programmed with a value less than 16 (10 hex) — a ‘tick’ of about 1 millisecond when CLK is 33 MHz.

Each channel has two timers: one 16-bit general timer 1 (GT1), and one 8-bit general timer 2 (GT2). Their operation and programming are different in synchronous and asynchronous protocols.

5.3.4 Timers in Synchronous Protocols

In synchronous protocols, the timers have no special significance for the CD2231; they are available to support the protocols. They are started by host commands or by interrupts generated by the CD2231. General timers 1 and 2 can be started in either of two ways:

1. By loading a new value to GT1 or GT2 when the timer is not running.
2. By setting the SetTm1 or SetTm2 bits in the End of Interrupt register when terminating an interrupt service routine. In this case, the value should be written to the appropriate Interrupt Status register (RISR, TISR, or MISR).

These timers can be disabled by a command through the CCR (Channel Command register).

5.3.5 Timers in Asynchronous Protocols

The receive timer is restarted from the value programmed in RTPR every time a character is received and loaded into the FIFO, or data is read by the host. For example, the receive FIFO threshold is set to eight, and six characters are stored in the receive FIFO. If no more characters are received and the receiver timer times-out, a receive interrupt is asserted (in DMA mode, DMA transfer occurs). The host is expected to retrieve all six characters from the receive FIFO. Assuming the host is still enabling this feature (that is, RET bit (IER[5]) from the IER register bit 5 is still set), and if there is no character being received and receiver timer times-out, a receive exception timeout interrupt (a group 3 interrupt) is asserted. The timer can be disabled if the value in RTPR is set to ‘0’ or the RET bit is cleared.

5.3.6 Transmit Timer

The TTR (Transmit Timer register) is used only if the embedded transmit command is enabled in the COR2. The delay transmit command specifies the delay period loaded in the TTR; no further transmit operations are performed until this timer reaches zero. The current state of the line is held at either ‘0’ for send break or ‘1’ for inter-character fill.

5.4 DMA Operation

The CD2231 uses a simple, but powerful, double-buffering method readily compatible with higher-level buffer control procedures, such as circular queues, link lists, and buffer pools. Each transmitter and each receiver is assigned an ‘A’ and a ‘B’ buffer. When transmitting, the host processor alternately fills the A and B buffers, and commands the CD2231 to transmit the buffers one at a time. When receiving, the CD2231 fills the A and B buffers and informs the host processor when each is ready.

A simple Ownership Status bit is used for each buffer; this ensures that there are no deadlocks between the host and the CD2231 regarding the use of a particular buffer.

By using the simple and flexible DMA management of the CD2231, the user host processor is concerned with transmit/receive data on a block-by-block basis. The user need not be concerned with character-by-character transfers, or even filling and emptying the FIFOs. The DMA controls are user-selectable per-channel and operate independently of one another.

The CD2231 can perform DMA operations in any of the supported line protocols. A special Append mode feature can reduce host CPU overhead for asynchronous datastreams. DMA operations are channel- and direction-specific. In each channel, either the transmitter and the receiver, or both, can be independently programmed for DMA mode by the CMR (Channel Mode register).

When the CD2231 acquires the bus for a DMA transfer, only data for one channel and in one direction is transferred; then, bus ownership is relinquished. A maximum of 16 bytes — the depth of the transmit and receive FIFOs — are transferred during any ownership cycle.

Whenever possible, DMA cycles are 16 bits wide, and buffers have the proper byte alignment. Unaligned buffers are sent using only 8-bit-wide transfers. If the buffer begins on an even address and contains an odd number of bytes, the CD2231 uses 16-bit transfers for all the words in the buffer except the last transfer, which is 8 bits.

If one buffer in a chain ends on an odd address, the next buffer in the chain should also start on an odd address to maintain proper alignment for most efficient bus usage. In this case, only the last transfer of the first buffer and the first transfer of the next buffer is 8 bits wide; all others are 16 bits.

The CD2231 can be forced to perform only byte-wide DMA operations by setting the byteDMA (DMR[3]) bit in the DMR (DMA Mode register).

5.4.1 Bus Acquisition Cycle

1. CD2231 asserts BR* and waits for BGIN*.
2. When BGIN* is detected, the CD2231 can access the bus after the current bus owner relinquishes control of the bus.
3. If BGACK* is high when BGIN* goes low, then the bus is free to access. Go to step 5.
4. If BGACK* is low when BGIN* goes low, then the bus is in use. The CD2231 waits for BGACK* to go high.
5. Once the CD2231 senses that BGACK* is high, the CD2231 waits for the current bus cycle to terminate (DS* and DTACK* high) and then assert BGACK* by driving it low. At that time, the CD2231 owns the bus. After driving BGACK* low, the CD2231 drives BR* high.

In Figure 5, the CD2231 was required to wait to access the bus.

5.4.2 DMA Data Transfer

After the CD2231 acquires the bus, it pulses ADLD* once. This loads the upper 24 address bits to the external 24-bit latch. This happens only once per DMA grant cycle. The AD[0–15] bits are remapped to memory address (MA) bits MA[16–31] and A[0–7] are mapped to MA[8–15]. If during DMA the upper 24 bits need to change, the CD2231 relinquishes the bus and then re-acquires the bus.

During each DMA read and write cycle, the least-significant eight memory address bits, MA[0–7], come from A[0–7].

In Figure 6, one DMA access is shown after the bus is acquired.

Figure 5. Bus Acquisition Cycle

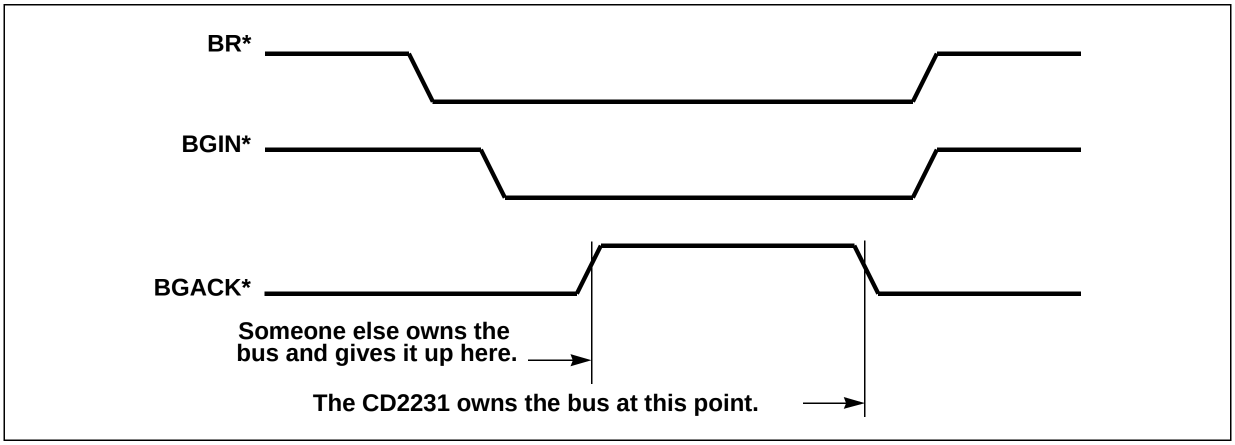
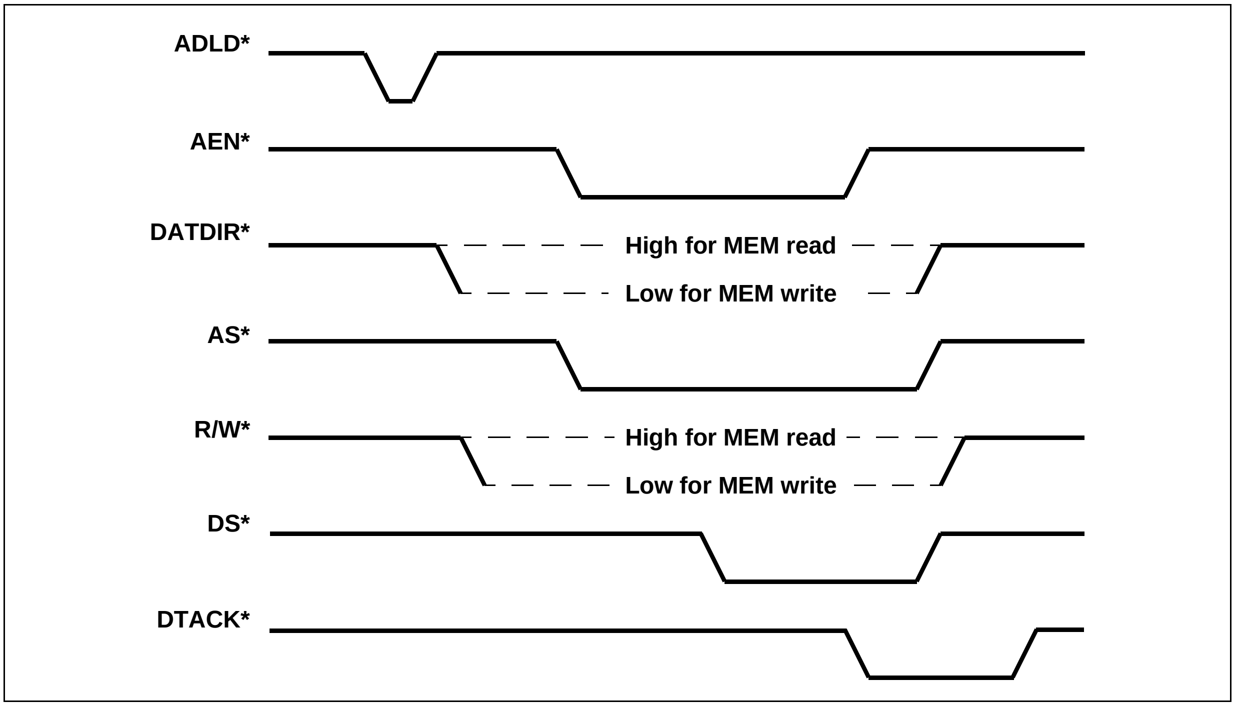

BR*
BGIN*
BGACK*
Someone else owns the bus and gives it up here.
The CD2231 owns the bus at this point.

Figure 6. Data Transfer Timing



5.4.3 Bus Error Handling

When a bus error is detected during a DMA sequence, the CD2231 terminates the current bus cycle and relinquishes the bus. Any data transfer in the bus ownership cycle is ignored, and the original conditions are restored. A subsequent retry attempt would start again from these original conditions.

If there is a non-zero value in the BERCNT (Bus Error Retry Count register), the register is decremented and the failed transfer is retried automatically. If the BERCNT is zero, a bus error interrupt is generated and DMA transfers are suspended on the failing buffer until the interrupt is serviced.

5.4.4 A and B Buffers and Chaining

The buffer management of the CD2231 uses a dual-buffer scheme. There is an A and B buffer pair for each transmitter and each receiver. Each buffer is controlled by an Ownership Status bit, called 2231own. When 2231own is set to '1', the CD2231 'owns' the buffer. When 2231own is set to '0', the host 'owns' the buffer. A simple rule prevents confusion in the buffer management — neither the CD2231 nor the host seizes buffer ownership. Each always relinquishes ownership to the other.

The host relinquishes ownership of a receive buffer to the CD2231 when the receive buffer is ready. The CD2231 is then free to write received data into the buffer. The CD2231 returns ownership of the receive buffer after the receive data is in the buffer. The host gives ownership of a transmit buffer to the CD2231 when the transmit buffer is ready to transmit. The CD2231 then transmits the contents of the buffer. When this is complete, the CD2231 returns ownership back to the host.

The CD2231 keeps track of which buffer (A or B) is to be used next in the status bits — Ntbuf for transmit and Nrbuf for receive. The relationship between the 2231own bit and the 'next' bits is shown later. The receive buffers are handled in the same way using the Nrbuf (next receive buffer).

Chaining is used to break up relatively long frames into shorter blocks in memory, and is useful where there are frequent smaller frames and occasional long frames. Chaining allows more efficient use of the user RAM.

The EOF Status bit controls chaining in Synchronous modes. Chaining applies to both transmit and receive. For transmit, the host determines the EOF bit; for receive, the CD2231 determines the EOF bit.

In Transmit DMA, when the first buffer is supplied to the CD2231, it is treated as the start of frame — the CRC is reset and leading pad/flag/syn characters are transmitted, followed by the data. If the EOF bit is set, the CRC and closing flag/syn is appended, and the next buffer is again treated as the start of frame. If the EOF bit is not set, the CD2231 treats the buffer as the first part of a larger frame and chains into the next buffer (does not reset CRC); this process continues until a buffer is supplied with the EOF bit set.

Table 3. A and B Buffers Chaining (Sheet 1 of 2)

Ntbuf	2231own Buffer A	2231own Buffer B	Transmit Action
0	0	0	Send nothing
0	1	0	Host sets up Buffer A
1	1	0	CD2231 accepts Buffer A and marks B as next
1	0	0	CD2231 completes A Tx, and passes it to host
1	0	1	Host sets up Buffer B
0	0	1	CD2231 accepts B and marks A as next

Table 3. A and B Buffers Chaining (Sheet 2 of 2)

Ntbuf	2231own Buffer A	2231own Buffer B	Transmit Action
0	1	1	Host sets up Buffer A
1	1	0	CD2231 completes B Tx, passes to host, accepts A and marks B as next
1	0	0	CD2231 completes A Tx and passes it to host

5.4.5 Transmit DMA Transfer

As in receive data transfers, two buffers are available for DMA transmit transfers. The ATBADR/BTBADR and ATBCNT/BTBCNT (Transmit Buffer Address and Transmit Buffer Count registers) contain the start address of and the byte count in the buffers. These registers are set by the host when initiating a transfer. The CD2231 makes a copy of the registers to perform the transfer, leaving the originals unchanged. The transfer of buffers between the host and the CD2231 is controlled by the ATBSTS/BTBSTS (Transmit Buffer Status) registers.

Buffers can contain either complete frames or blocks of data, linked together to form a complete frame or a block, or used in an Append mode to transmit data as it arrives from another process. The first two transfer types are Block mode transfers, the last is the Append mode. Both are described further below. The management of the buffers reduces the processor overhead associated with short data transfers and increases the minimum response time requirements for frame-based transmissions.

Chain Mode Transfer

In Chain mode, the frame should be complete in buffers in memory before transmission is started. The Append Status bit should not be set; the Start of Frame bit must be set to begin transmission, and the Last Buffer bit must be set if this buffer is the last in a chained block or is a complete frame or block.

When the CRC bit is set, the CD2231 generates and transmits a cyclic redundancy check word for the frame using the polynomial selected by the CPSR (CRC Polynomial Select register). If the Interrupt Required bit is set, a host interrupt is generated after the buffer is transmitted.

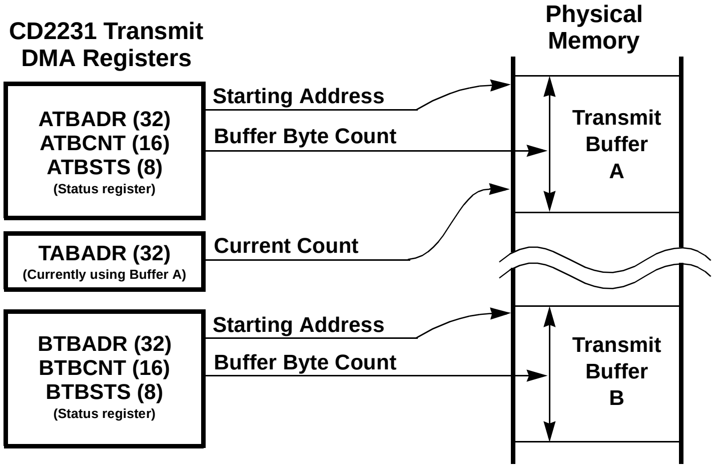
Transmit buffers can be chained to support large frames. To minimize bus usage, the first buffer of the chain should begin on an even address in host memory. The CD2231 begins fetching a frame from a buffer performing DMA transfer, reading two bytes at a time. The CD2231 cannot realign data between external memory and the FIFO. If one buffer of the chain ends on an odd address, the next buffer in the chain should begin on an odd address. Otherwise, only single-byte transfers are made for the rest of the buffer.

Append Mode Transfer (Buffer A Only)

Append mode transfers are available for Buffer A in Asynchronous mode only. If Buffer A is set to Append mode, the host can enable the CD2231 to transmit data in the buffer before it is completely filled. The CD2231 starts transmitting new data when it is appended to the buffer.

This mode is useful for terminal echo routines that do not wait for a complete block to be formed before starting transmission. In this mode, transmission is started when the buffer is made available to the CD2231 by the host; ATBADR[0–3] and ATBCNT[l, h] are initialized. Subsequent triggering of DMA transfer occurs by programming the ATBCNT[l, h] with the accumulated byte count. In this case, ATBCNT should be written as a 16-bit word to avoid confusion between two byte operations. The ATBADR[0–3] should not be reprogrammed during the Append mode. If the memory space has to be moved, the Append mode must be disabled first. When the final data is added to the append buffer and ATBCNT has been updated, the host should set the AppdCmp bit (STCR[5]). When the CD2231 has completed the final transmission, it clears the 2231own bit in the ATBSTS register, and generates an end-of-buffer interrupt.

Figure 7. Transmitter A and B Buffers



NOTE: Number of bits in each register is shown in parentheses ( ). Buffer A and Buffer B do not need to be the same length.

5.4.6 Synchronous Transmitter Examples

In Figure 7, buffers A and B are contained in RAM external to the CD2231. All others (DMABSTS, ATBADR, TCBADR, ATBCNT, ATBSTS, BTBADR, BTBCNT, and BTBSTS) are inside the CD2231.

Example 1

Transmit a frame out of channel 1, with no chaining.

1. The host checks the Ntbuf bit in the DMABSTS register for channel 1 to determine which buffer is next. In this example, Ntbuf is set to ‘0’ indicating that Buffer A is used next.
2. The host sets up the buffer data, the starting address — ATBADR, and the buffer byte count — ATBCNT.
3. The host sets up the ATBSTS (‘A’ Buffer Status) register. The EOF bit is set to indicate that there is no chaining. The 2231own bit is set to give ownership to the CD2231. By setting 2231own, the host commands the CD2231 to start transmission. Thus, everything must be ready (starting address, buffer data, and byte count) prior to setting 2231own.
4. The CD2231 starts frame transmission out of channel 1. When transmission is started, the CD2231 sets Tbusy bit in DMABSTS. As transmission progresses, the current buffer pointer (TCBADR) is updated by the CD2231. Also, at the start of transmission, the Ntbuf bit (Next Buffer) is set to ‘1’ to notify the host that Buffer B is next.

5. The CD2231 completes frame transmission by adding any necessary CRCs and trailing frame delimiters.
6. When the CD2231 completes the transmission, it clears the Tbusy bit. Then, it sets the EOB bit and clears the 2231own bit in the ATBSTS. This notifies the host that the transmission is complete, and returns ownership of the buffer back to the host.
7. The CD2231 optionally interrupts the host, with EOF and EOB in the TISR both set to indicate that the transmission is complete and there was no chaining.

Example 2

Transmit out of channel 0 and chain three buffers into one frame. The frame is 240 bytes long, and the maximum buffer size is 100.

1. The host checks the Ntbuf bit in the DMABSTS register for channel 0 to determine which buffer is next. In this example, Ntbuf is set to ‘1’ indicating that Buffer B is used next.
2. The host sets up the buffer data, the starting address (BTBADR), and the buffer byte count (BTBCNT) for the first ‘link’ of the chain to be transmitted. For this example, BTBCNT is set to ‘100’.
3. The host sets up the BTBSTS (‘B’ Buffer Status) register. The EOF bit is cleared to indicate that this buffer is the first link in a chain. The 2231own bit is set to give ownership to the CD2231. By setting 2231own, the host commands the CD2231 to start transmission. Thus, everything must be ready (starting address, buffer, and data count) prior to setting 2231own.
4. At this point, the host has enough time to transmit 100 bytes to set up the next buffer link. If the host fails to do this in time, there is a transmitter underrun, and the frame is aborted in HDLC.
5. The CD2231 starts transmitting Buffer B from channel 0. When this is started, the Ntbuf bit is cleared to ‘0’ to indicate that Buffer A is next. This helps the host keep track of which buffer is next. As transmission progresses, the current buffer pointer, TCBADR, is updated by the CD2231. During or prior to this update, the host has readied Buffer A. For Buffer A, the EOF bit in the ATBSTS register is cleared by the host, indicating that the buffer is not at the end of the chain.
6. At the end of transmission of this buffer, the CD2231 does not add any CRCs nor end of frame delimiters because there is more data for the current frame.
7. After the CD2231 has completed transmission of the first link out of Buffer B, the CD2231 sets the EOB bit and clears the 2231own bit in the BTBSTS. This notifies the host that the transmission is complete, and returns ownership of the buffer back to the host.
8. The CD2231 optionally interrupts the host with EOF clear and EOB set in the TISR to indicate that the transmission is completed, and that there was chaining.
9. The ATBSTS register indicates that the CD2231 has ownership of Buffer A for transmission of the next ‘link’. The EOF is cleared so that this link is not the last link in the transmitted chain.
10. The CD2231 continues transmission of the current frame, but now transmission is from Buffer A. This is the second link, which is 100 bytes long. During this time, the host must set up a new Buffer B for the third and final link. The BTBCNT for the last link is set to 40 bytes.
11. After the CD2231 has completed transmission of the second link out of Buffer A, it sets the EOB bit and clears the 2231own bit in the ATBSTS. This notifies the host that the transmission has complete, and returns ownership of the buffer back to the host. As with the first link, the CD2231 does not add CRCs or ending frame delimiters to this link.

12. The CD2231 optionally interrupts the host with bit EOF clear and bit EOB set in the TISR to indicate that the transmission has completed, and that there was chaining.
13. By this time, the host has set up a new buffer for Buffer B. The EOF bit in the BTBSTS is set to indicate that this is the last link in the chain.
14. The CD2231 transmits Buffer B in the same manner as explained earlier. As before, the CD2231 transmits the number of bytes indicated in the BTBCNT, which is 40 bytes for the third segment.
15. When the CD2231 completes transmission, any necessary CRCs and ending frame delimiters are transmitted.
16. The CD2231 optionally interrupts the host with EOF and EOB bits set in the TISR to indicate that the transmission has completed, and that this was the last link in the chain.

5.4.7 Receive DMA Transfer

In all protocol modes, two host memory buffers can be made available to each receive channel, by the ARBADR/BRBADR and ARBCNT/BRBCNT (Receive Buffer Address and Receive Buffer Count registers) registers. To make a buffer available, the user must supply the buffer address in the Receive Buffer Address registers; the number of free bytes in the buffer must be written in the Receive Buffer Count registers, and the buffer status must be updated in the ARBSTS/BRBSTS (Receive Buffer Status register) registers. The CD2231 is then free to use the buffer for receive data, and updates the Buffer Status register as appropriate. When the buffer is no longer in use, the CD2231 writes the number of bytes stored in the buffer in RBCNT and updates status in RBSTS. This frees the host to take control of this buffer and supply a new buffer in its place. The CD2231 automatically switches to the other buffer whenever one buffer becomes full, or the end of a frame has been reached. If the other buffer has not been allocated, the host still has the time required to fill the CD2231 16-byte FIFO, to respond, and to avoid loss of data.

Special actions are taken depending on the channel protocol. In HDLC, PPP, SLIP and MNP 4, the end-of-frame/data block boundaries are recognized by the CD2231. When a data-block boundary is detected, the current buffer is automatically terminated. If the other buffer is allocated and owned by the CD2231, it becomes the current buffer. End-of-frame and block interrupts are also generated to the host.

In Asynchronous mode, a host interrupt is generated when there are receive exceptions (framing error, special character, etc.), but the buffer is not terminated. The data and exception status are made available to the host, just as when the Asynchronous mode is purely interrupt-driven. New data is buffered internally in the FIFO until the host services the exception interrupt. The host has the following three options when terminating an exception interrupt:

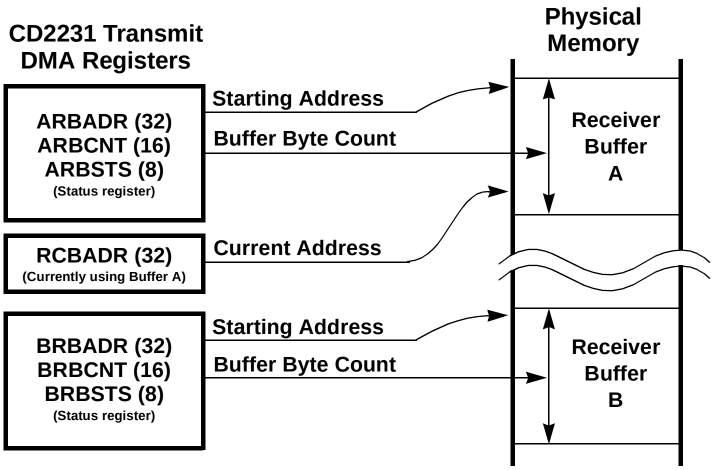
1. The exception character can be discarded.
2. The buffer can be terminated (if there is no additional interrupt to be generated). The transfer count is not provided in ARBCNT/BRBCNT, but can be calculated by RCBADR.
3. A user-defined gap can be left in the buffer.

These selections are communicated to the CD2231 by the value written by the host to the Receive End of Interrupt register, when the Receive Interrupt service is completed. Leaving an ‘n’-byte gap enables the host to insert status of its own in the current buffer, while continuing to receive data in the same buffer. This eliminates the overhead of allocating a new buffer. The host must have noted the starting location of the gap while in the exception interrupt. This is done by reading the Receive Current Buffer Address register. The address in this register is guaranteed to be stable during the Receive interrupt, and point to the next free character location in the current DMA buffer. If the size of the gap supplied by the host is sufficient to fill or complete the current buffer, the CD2231 automatically switches to the other buffer and advances the Receive Current Buffer Address enough to complete the desired gap. The CD2231 readjusts data alignment in its internal FIFO as needed to maintain alignment with the external buffer.

Receiver A and B Buffers

In Figure 8, buffers A and B are contained in RAM external to the CD2231. All others (DMABSTS, ARBADR, ARBCNT, ARBSTS, RCBADR, BRBADR, BRBCNT, and BRBSTS) are inside the CD2231.

Figure 8. Receiver A and B Buffers



NOTE: Number of bits in each register is shown in parentheses ( ).
Buffer A and Buffer B do not need to be the same length.

Example 1

Receive a frame from channel 1, no chaining.

1. The host must first make a receive buffer available before a frame can be received. Thus, the host checks the Nrbuf bit in the DMABSTS register for channel 1 to determine which buffer is next. In this example, Nrbuf is set to ‘0’, indicating that Buffer A is used next.

2. The host sets up the starting address — ARBADR, and the buffer byte count — ARBCNT. When the host writes the count — ARBCNT, the host has defined the size limit for the buffer.

3. The host then gives the buffer to the CD2231 by setting the 2231own bit in the status register — ARBSTS. This notifies the CD2231 that it is now alright to write received.

4. The Rbusy bit (DMABSTS[0]) for channel 1 is ‘0’ until a frame starts to be received. When frame data starts coming in, the CD2231 sets Rbusy to notify the host that Buffer B is next. As data bytes are written into the buffer, the current buffer pointer, RCBADR, is updated by the CD2231.

5. At the end of the received frame, the CD2231 tests for correct end of frame delimiter and CRC. When the received frame is complete, the CD2231 clears the Rbusy bit. In this example, there is no receive chaining, so the received frame byte count is less than or equal to the buffer

size count — ARBCNT. The CD2231 writes the value of the actual received byte count into the same register — ARBCNT. (Note that the host has written the maximum buffer size in ARBCNT when the buffer is given to the CD2231. But when the buffer is returned back to the host, the CD2231 has written the actual byte count of the received buffer into ARBCNT.)

6. The CD2231 sets the EOB and EOF bits. This notifies the host that the end of the buffer and frame have been reached. The CD2231 also clears the 2231own bit to return the buffer to the host.

Example 2

Receive a frame on channel 0, which consists of three buffers chained together. The frame is 240 bytes long, and the maximum buffer size is 100.

1. The host checks the Nrbuf bit (DMABSTS[1]) register for channel 0 to determine which buffer is next. In this example, Nrbuf is set to ‘1’ indicating that Buffer B is used next.
2. The host sets up the starting address — BRBADR. Buffer size is set to ‘100’ in this example. Thus, the host sets BRBCNT to ‘100’.
3. The host then sets the 2231own bit to relinquish ownership to the CD2231.
4. The host should know the amount of time it takes to receive 100 bytes, because this is the minimum time the host has to set up the next buffer link. If the host fails to do this in time, there is a receiver overrun, and the received frame is lost.
5. Suppose that the CD2231 starts receiving data into Buffer B of channel 0. When this is started, the Nrbuf bit is cleared to ‘0’ by the CD2231 to help the host keep track of which buffer is next. (During or prior to this, the host has made Buffer A ready.)
6. After the CD2231 has received the first link of the frame into Buffer B, it sets the EOB bit and clears the EOF bit. This indicates that the first link in a chain has been received. Also, the CD2231 clears the 2231own bit, and returns ownership of the buffer to the host.

   For the first received link, the received byte count (BRBCNT) remains unchanged at 100, since the received data filled the buffer.
7. The CD2231 optionally interrupts the host with bit EOF clear and bit EOB set in the RISR to indicate that the received buffer is complete, and that there was chaining.
8. The ARBSTS register indicates that the CD2231 has ownership of Buffer A for transmission of the next link.
9. As the frame continues to be received, the data goes into Buffer A. This is the second link, which is 100 bytes long. During this time, the host must set up a new Buffer B for the third and final link.
10. After the CD2231 has received the second link into Buffer A, the CD2231 sets the EOB bit and clears the 2231own bit in the ARBSTS. This returns ownership of the buffer to the host.

    As with the first link, the received byte count — ARBCNT, remains unchanged at 100 since the received data filled the buffer.
11. The CD2231 optionally interrupts the host with bit EOF clear and bit EOB set in the RISR to indicate that the received buffer is complete and that there was chaining.
12. By this time the host has set up a new buffer for Buffer B.
13. The CD2231 receives data into Buffer B in the same manner, as previously explained.

14. In this example, the third link does not fill the buffer. Thus, when the end-of-frame delimiter is detected by the CD2231, the value of 40 (for 40 received bytes) is written into the received byte count — BRBCNT.
15. Next, the CD2231 sets the EOB and EOF bits to show that the buffer is complete, and that this is the last link in the chain.
16. The CD2231 optionally interrupts the host with EOF and EOB set in the RISR to indicate that the received frame is complete, and this was the last link in the chain.

5.4.8 Transmit DMA Transfer

The CD2231 contains two DMA descriptors that can be loaded by the CPU to specify transmit buffers. These descriptors are designated A and B, and each consists of a 32-bit address (ATBADR/BTBADR), a 16-bit count (ATBCNT/BTBCNT), and an 8-bit status (ATBSTS/ BTBSTS).

The Status register contains an Ownership Status bit — 2231own. When this bit is set the CD2231 owns the descriptor, and it should not be written to by the CPU. When this bit is clear, the descriptor is owned by the CPU.

When DMA is selected and the channel is enabled, the CD2231 waits for ownership of Buffer A. When ownership of A is given by setting the 2231own bit, the buffer is transmitted, and the Ownership bit is cleared. The CD2231 waits for ownership of Buffer B; this process continues, toggling between the two buffer descriptors.

The DMABSTS register contains a status bit (NtBuf) that informs the CPU of the next buffer to transmit and to ensure that the CPU and CD2231 stay in synchronization. This procedure ensures that a pipeline of data is available for the CD2231 to send, maximizing the bandwidth utilization and minimizing the possibility of underruns. Figure 9 illustrates this procedure.

5.4.8.1 Interrupts for Transmit DMA Buffers

Two types of transmit interrupts are available in DMA mode; they are enabled by the IER and controlled by the TxD and TxMpty bits.

When the TxMpty interrupt is enabled, interrupts are generated when there is no transmit data available to send. For example, the TxMpty interrupt can be used by the CPU to determine when line turn-around can occur on half-duplex lines.

Normally, the TxDat interrupt is used to indicate the end of each transmit buffer. The interrupt is scheduled internally when the last data is read from the transmit buffer into the FIFO.

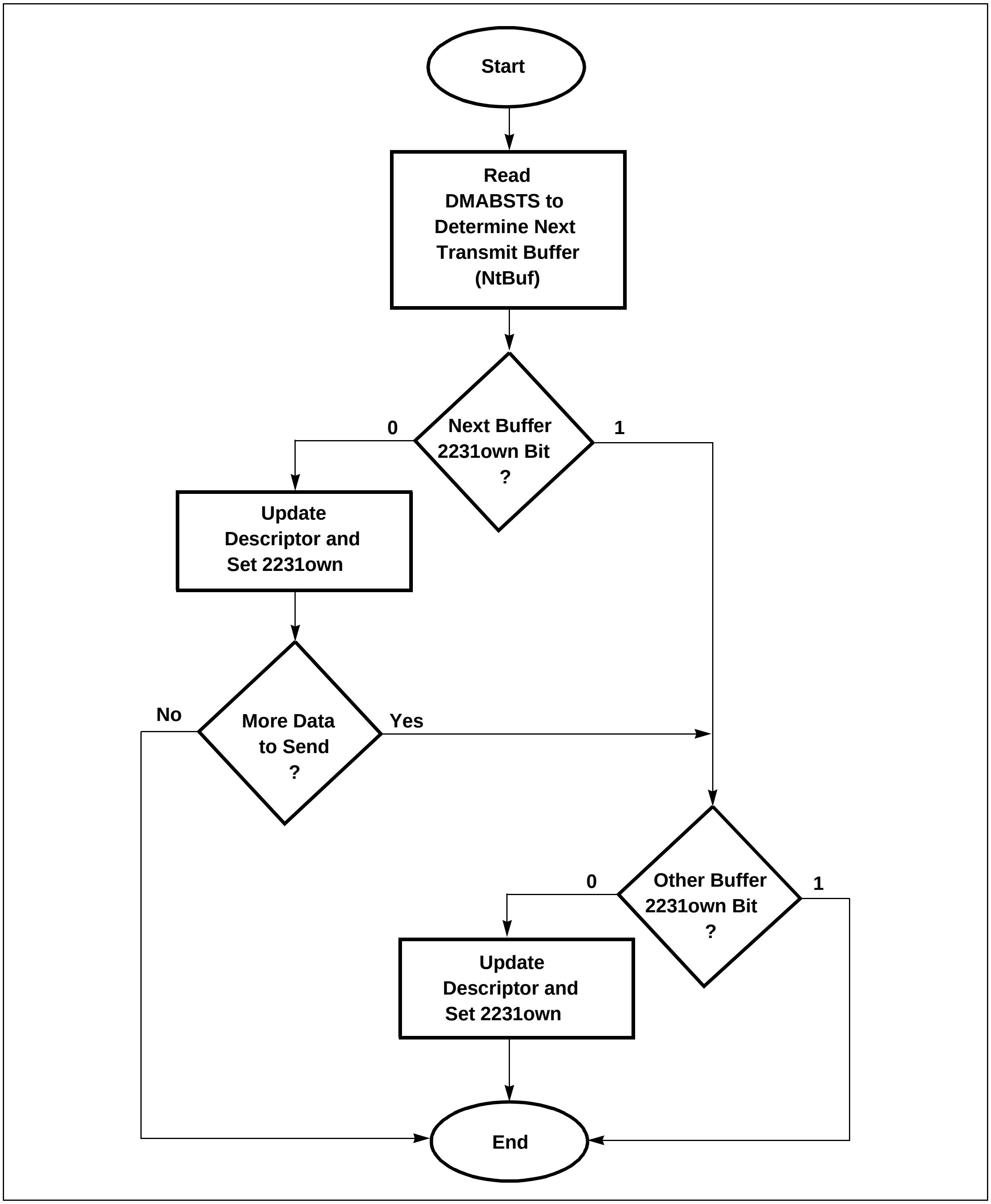
Because only one interrupt is generated for each buffer, the TxD bit (IER[0]) register can be left permanently enabled. If interrupts are required selectively for individual buffers, the INTR bit in the ATBSTS/BTBSTS registers can be used to selectively enable interrupts.

5.4.8.2 Chained Buffers

In Synchronous modes, when the frame size exceeds the maximum buffer size, a frame can be transmitted from a number of separate buffers. This is achieved simply by **not** setting the EOF bit in the ATBSTS/BTBSTS (Transmit Buffer Status register) until the last buffer of the frame. The CD2231 transmits the buffers as one frame; it appends the CRC only when all the data is transmitted from the buffer with the EOF flag set.

If the above procedure for allocating buffers is used, the CPU has the transmission time of the last buffer to allocate the next to avoid possible underrun. The EOF bit (TISR[6]) is set for the interrupt associated with the last buffer.

Figure 9. DMA Transmit Buffer Selection


Start
Read DMABSTS to Determine Next Transmit Buffer (NtBuf)
Next Buffer 2231own Bit ?
0
1
Update Descriptor and Set 2231own
More Data to Send ?
No
Yes
Other Buffer 2231own Bit ?
0
1
Update Descriptor and Set 2231own
End

5.4.8.3 Append Mode

The Append mode reduces the CPU overhead required to provide asynchronous terminal echoing functionality; this is also necessary for any similar application that involves an unpredictable datastream. Buffer A can be set into Append mode by the ATBSTS register. This buffer can then be used for the echoed data, while Buffer B is used for all other output data. The append buffer allows data transmission to start from a buffer before all data is available for transmission. For example, terminal echoing requires that each character is echoed (or translated and echoed) before the complete line is typed.

To operate in Append mode, the ATBADR and ATBCNT are set as normal (the ATBCNT can be zero), and the 2231own and Append bits are set in the ATBSTS. When any data is available for transmission, it is placed in the RAM buffer by the CPU, and the total buffer byte count is updated in the ATBCNT. The CD2231 can now scan the ATBCNT register for any changes; if new data is found, it is read from the buffer and transmitted.

When no more data is found in the append buffer, the CD2231 scans Buffer B for ownership. If Buffer B is owned by the CD2231, the data in that buffer is transmitted uninterrupted; at the end of the transmission, Buffer A count continues to be scanned for new data.

For correct operation of this feature, the ATBCNT register should be updated with a word-write operation. If only byte access is possible, the value should not exceed 256 bytes. This mode allows multiple transfers to be performed through a single buffer; it saves CPU overhead by either processing multiple buffers or in handling interrupts with every character.

Line retransmission becomes as simple as ‘stepping back’ in the buffer and resending. To terminate Append mode, a command can be given by the STCR to cause the Buffer A to be terminated when all current data has been sent.

5.4.8.4 Transmit Bus Errors

When a transmit bus error interrupt is generated, the TISR and ATBSTS/BTBSTS registers indicate a bus error status. The current transfer address is available in the TCBADR[0–3] registers, and the bus error occurred on the last transfer that started at this address. This means the actual error address can be up to 16 bytes further in the buffer.

Following a bus error condition, the CPU can either discontinue the current buffer or retry from the start of the last transfer. To discontinue the current buffer, the TermBuff bit should be set when TEOIR is written to at the end of the interrupt. In Synchronous mode, the frame is still in progress and needs to be aborted by the STCR (Special Transmit Command register).

To retry the frame, the CPU should set the 2231own bit in the ATBSTS/BTBSTS register, and not set the TermBuff bit when writing to TEOIR at the end of the interrupt. This causes the last transfer to be retried; should a bus error occur again, the above procedure is repeated. The CPU should check to ensure that a bad location is not continually retried.

5.4.9 Receive Buffer Interrupts

When a receive buffer is complete, the CD2231 generates an end-of-frame receive exception interrupt. It provides the CPU with RISR status and information on which buffer is complete.

When a receive error occurs, the device stops DMA at the point of error and generates a bus error receive exception interrupt. RISR indicates the cause of the exception, and RCBADR provides the next location in the receive buffer.

The CPU has the following five options:

1. Terminate the buffer.
2. Discard the exception.
3. Terminate the buffer and discard the exception.
4. Continue from the current position in the buffer.
5. Leave an ‘n’ byte gap in the buffer and then continue.

The required option is written to the REOIR (Receive End-of-Interrupt register) by the CPU to terminate the interrupt. If the terminate buffer option is chosen, the 2231own bit in the ARBSTS/BRBSTS register should first be cleared by the CPU, or a new buffer can be supplied by the CPU.

5.4.9.1 Receive Timeout in Asynchronous DMA Mode

In Asynchronous DMA mode, the only way that the CD2231 releases the ownership is by reaching the end-of-buffer. Receive timeout, or any exceptions, do not release the ownership if end-of-buffer condition is not met. The following illustrates recommended procedures to handle a receive timeout in Asynchronous DMA mode.

Scenario 1: Buffer A is currently selected, receive timeout occurs, host wants to continue on.
Recommendation: Do nothing in the receive timeout interrupt service routine.

Scenario 2: Buffer A is currently selected, receive timeout occurs, host no longer requires DMA.
Recommendation: Reset ownership bits in ARBSTS/BRBSTS, and set TermBuff in REOIR in the receive timeout interrupt service routine.

Scenario 3: Buffer A is currently used, receive timeout occurs, host wants to start DMA in Buffer B.
Recommendation: Set TermBuff in REOIR in the receive timeout interrupt service routine. The CD2231 switches to Buffer B.

Note: When a receive timeout occurs in Buffer B, the CD2231 pops back to Buffer A, unless the host clears both Ownership Status bits.

The above scenarios applies if Buffer B is selected first.

5.4.9.2 Receive Bus Errors

When a receive bus error interrupt is generated, the RISR and ARBSTS/BRBSTS registers indicate a bus error status. The current transfer address is available in RCBADR[0–3], the bus error occurred on the last transfer that started at this address. This means that the actual error address can be up to 16 bytes further in the buffer.

Following a bus-error condition, the CPU can either discontinue the current buffer or retry from the start of the last transfer. If the buffer is to be discontinued, the number of valid receive bytes can be calculated by subtracting the starting address A/BRBADR[0–3] from the current address RCBADR[0–3]. The CPU should set the TermBuff bit in REOIR to terminate this buffer and move to the next.

The transfer that failed to the first buffer, due to the bus error, is still in the receive FIFO and is transferred to the next buffer following the end of interrupt.

To retry the buffer from the failure point, the CPU should set the 2231own bit in the ARBSTS/ BRBSTS register. The CPU should not set the TermBuff bit when writing to REOIR at the end of the interrupt, this causes the last transfer to be retried. Should a bus error occur again, the above procedure is repeated. The CPU should check to ensure that a bad location is not continually retried.

5.5 Bit Rate Generation and Data Encoding

5.5.1 BRG and DPLL Operation

Data clocks are generated in the CD2231 by feeding one of a number of clock sources into a programmable divider. The clock source and divisor are separately programmable for each channel and direction by the user. Clock options are programmed in the Transmit Clock Option and Receive Clock Option registers. The divisors are programmed in the Transmit Bit Rate Period and Receive Bit Rate Period registers. The possible clock sources are as following:

Transmit

1. Clk 0 – CLK input ÷ 8
2. Clk 1 – CLK input ÷ 32
3. Clk 2 – CLK input ÷ 128
4. Clk 3 – CLK input ÷ 512
5. Clk 4 – CLK input ÷ 2048
6. TXCIN pin
7. Receive bit clock

Receive

1. Clk 0 – CLK input ÷ 8
2. Clk 1 – CLK input ÷ 32
3. Clk 2 – CLK input ÷ 128
4. Clk 3 – CLK input ÷ 512
5. Clk 4 – CLK input ÷ 2048
6. RXCIN pin

The CLK input is nominally 33 MHz.

The divisor can be programmed for values from 1–255. To maximize the accuracy of edge detection in Asynchronous and DPLL (digital phase locked loop) modes, select the highest frequency clock and largest divisor combination.

An external clock input can be used and used as a multiple of the desired bit rate. If so, the appropriate divisor value must be loaded into the Bit Rate Period register. If the external clock is at the desired bit rate (1× clock), a value of 01h must be loaded into the associated Bit Rate Period register.

The receive bit rate generator can also be programmed to act as a DPLL . In that mode, the clock select and divisor are programmed to be as near as possible to the nominal receive bit rate. Clock phase adjustments are made by the DPLL logic to lock to the incoming datastream. The receive bit clock is an optional input to the transmitter. This makes it possible to use the DPLL-derived clock to synchronize the transmit datastream.

Section 5.2 includes examples for programming standard bit rates. The value to be loaded to set a given bit rate is determined with:

$$\text{Bit rate divisor} = \frac{\text{Frequency of chosen clock source}}{\text{Desired bit rate}} - 1$$

The above equation, in general, yields a non-integer result. The nearest integer value, along with the clock source, is the optimum choice for that bit rate. The value loaded in the Period register must be that integer expressed as an 8-bit binary value. The bit-rate error is the difference between the integer value and the ideal value, expressed as a percentage.

Example 1

This example illustrates programming the bit rate generator at 19.2 kbits/second using the internal clock, at a system clock frequency of 33 MHz.

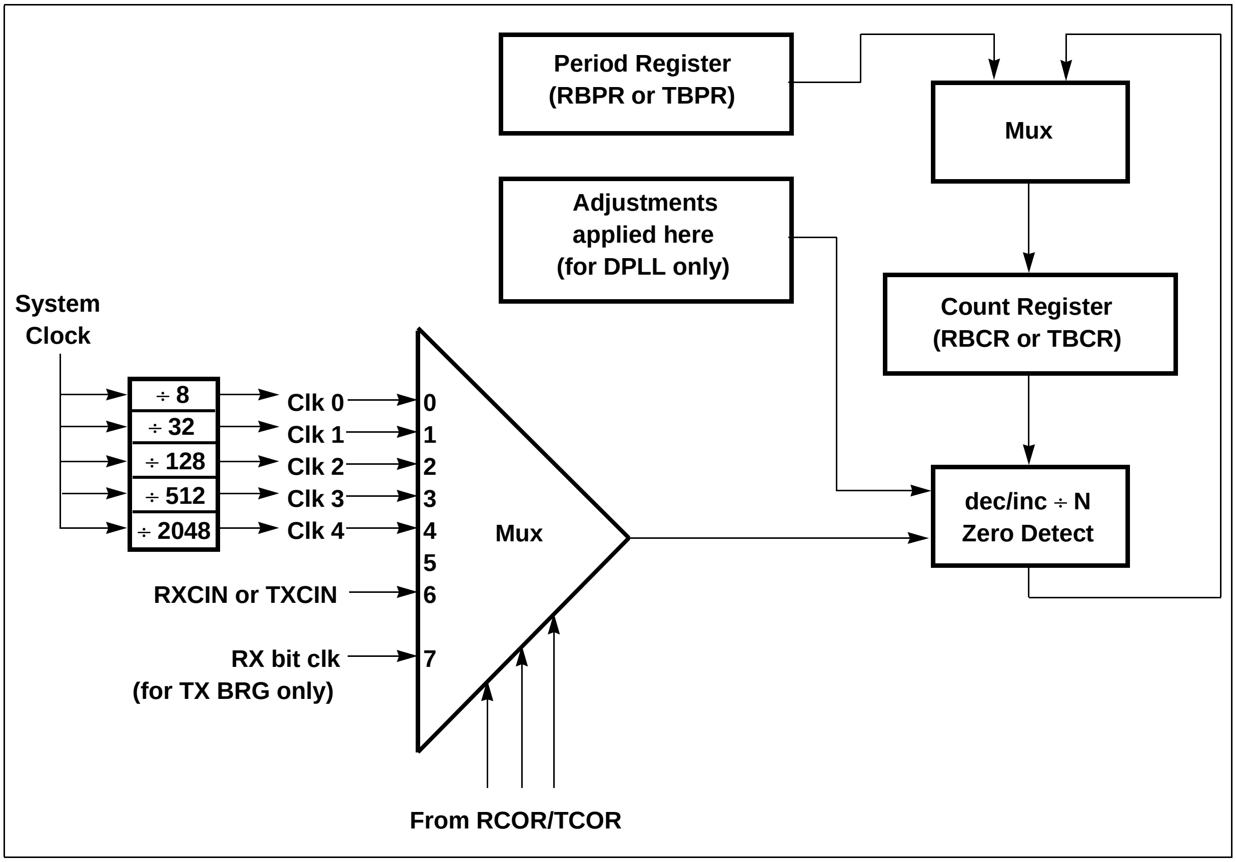
Divisor loaded into R(T)BPR = 214 or d6h
Value loaded into R(T)COR = 00h, to select Clk 0

Example 2

This example illustrates programming the bit rate generator at 56,000 bps using external clock, at a system clock frequency of 33 MHz.

The user provides a 1.25-MHz clock on the RXCIN or TXCIN pin.
Divisor loaded into R(T)BPR = 21 or 15h
Value loaded into RCOR = 06h, to select External Clock mode
Value loaded into TCOR = C0h, to select External Clock mode

Figure 10. Bit Rate Generator/DPLL



Receive Clock Option Register (RCOR) **CA C8 B R/W**

TLVal	res	dpllEn	Dpllmd1	Dpllmd0	ClkSel2	ClkSel1	ClkSel0

Transmit Clock Option Register (TCOR) **C2 C0 B R/W**

ClkSel2	ClkSel1	ClkSel0	res	Ext-1X	res	LLM	res

Table 4. Clock Source Select (Sheet 1 of 2)

ClkSel2	ClkSel1	ClkSel0	Select	
0	0	0	Clk 0	
0	0	1	Clk 1	
0	1	0	Clk 2	
0	1	1	Clk 3	
1	0	0	Clk 4	
1	0	1	Reserved	

Table 4. Clock Source Select (Sheet 2 of 2)

ClkSel2	ClkSel1	ClkSel0	Select	
1	1	0	External clock	
1	1	1	Reserved	(RCOR)
1	1	1	Receive clock	(TCOR)

Table 5. Bit Rate Constants, CLK = 20 MHz

Bit Rate	Divisor	Clock	Error
50	c2	Clk 4	0.16%
110	58	Clk 4	0.25%
150	40	Clk 4	0.16%
300	81	Clk 3	0.16%
600	40	Clk 3	0.16%
1200	81	Clk 2	0.16%
2400	40	Clk 2	0.16%
3600	ad	Clk 1	0.22%
4800	81	Clk 1	0.16%
7200	56	Clk 1	0.22%
9600	40	Clk 1	0.16%
19200	81	Clk 0	0.16%
38400	40	Clk 0	0.16%
56000	2c	Clk 0	0.80%
64000	26	Clk 0	0.16%

NOTE: All divisors are in hexadecimal.

Table 6. Bit Rate Constants, CLK = 25 MHz (Sheet 1 of 2)

Bit Rate	Divisor	Clock	Error
50	f3	Clk 4	0.06%
110	6e	Clk 4	0.02%
150	50	Clk 4	0.47%
300	a2	Clk 3	0.15%
600	50	Clk 3	0.47%
1200	a2	Clk 2	0.15%
2400	50	Clk 2	0.47%
3600	d8	Clk 1	0.01%
4800	a2	Clk 1	0.15%
7200	6c	Clk 1	0.45%
9600	50	Clk 1	0.47%

Table 6. Bit Rate Constants, CLK = 25 MHz (Sheet 2 of 2)

Bit Rate	Divisor	Clock	Error
19200	a2	Clk 0	0.15%
38400	50	Clk 0	0.47%
56000	37	Clk 0	0.35%
64000	30	Clk 0	0.35%
76800	28	Clk 0	0.76%

NOTE: All divisors are in hexadecimal.

Table 7. Bit Rate Constants, CLK = 30 MHz

Bit Rate	Divisor	Clock	Error
110	84	Clk 4	0.13%
150	61	Clk 4	0.35%
300	c2	Clk 3	0.16%
600	61	Clk 3	0.35%
1200	c2	Clk 2	0.16%
2400	61	Clk 2	0.35%
3600	40	Clk 2	0.16%
4800	c2	Clk 1	0.16%
7200	81	Clk 1	0.16%
9600	61	Clk 1	0.35%
19200	c2	Clk 0	0.16%
38400	61	Clk 0	0.35%
56000	42	Clk 0	0.05%
64000	3a	Clk 0	0.69%
76800	30	Clk 0	0.35%
115200	20	Clk 0	1.38%

NOTE: All divisors are in hexadecimal.

Table 8. Bit Rate Constants, CLK = 35 MHz (Sheet 1 of 2)

Bit Rate	Divisor	Clock	Error
110	9a	Clk 4	0.23%
150	71	Clk 4	0.06%
300	e3	Clk 3	0.06%
600	71	Clk 3	0.06%
1200	e3	Clk 2	0.06%
2401	71	Clk 2	0.06%
3600	4b	Clk 2	0.06%

Table 8. Bit Rate Constants, CLK = 35 MHz (Sheet 2 of 2)

Bit Rate	Divisor	Clock	Error
4800	e3	Clk 1	0.06%
7200	97	Clk 1	0.06%
9600	71	Clk 1	0.06%
19200	e3	Clk 0	0.06%
38400	71	Clk 0	0.06%
56000	4d	Clk 0	0.16%
64000	43	Clk 0	0.53%
76800	38	Clk 0	0.06%
115200	25	Clk 0	0.06%
12800	21	Clk 0	0.53%
134400	20	Clk 0	1.38%

NOTE: All divisors are in hexadecimal.

Transmit and receive data can be encoded and decoded in NRZ, NRZI, or Manchester formats. For NRZI, at the start of transmission, a learning datastream of contiguous zeros achieves bit synchronization; for Manchester, an alternating pattern of ones and zeros is required.

NRZ, NRZI, and Manchester are data encoding schemes used in various synchronous protocols. In NRZ, the signal condition represents the data type, high for a logic ‘1’ and low for a logic ‘0’. In NRZ and NRZI encoding, the transitions of the datastream occur at the beginning of the bit cell. In NRZI encoding, the signal condition switches to the opposite state to send a binary ‘0’. In Manchester encoding, the transitions are always in the middle of the bit cell. A high-to-low transition is made to send a logic ‘1’, and a low-to-high transition to send a logic ‘0’. The timing diagrams (Figure 11 to Figure 13) illustrate the encoding method. The data bits are ‘0110010’.

Example 3

This example illustrates programming the DPLL at 128 kbits/second in NRZI mode, using the internal clock at a system clock frequency of 33 MHz.

Divisor loaded into RCOR = 38 or 26h
Value loaded into RCOR = 28h, to enable the DPLL, NRZI framing and select Clk 0

Example 4

This example illustrates programming the DPLL in the ×1 External Clock mode, with Manchester encoding.

Divisor loaded into RBPR = 01h, to enable ×1 external clock
Value loaded into RCOR = 36h, to enable the DPLL, select Manchester framing, and external clock

When using an n-times external clock, the highest possible clock frequency and largest divisor combination is recommended. The frequency of an external clock should be less than the system CLK input divided by 16, (that is, for 33-MHz operation, the data clock should be less than 2.0 MHz). Note that R(T)BPR is an 8-bit register, therefore the largest divisor value is 255.

The following equation computes the divisor value:

$$\text{bit rate divisor} = \frac{\text{Frequency of external clock source}}{\text{Desired bit rate}} - 1$$

Figure 11. Data Encoding

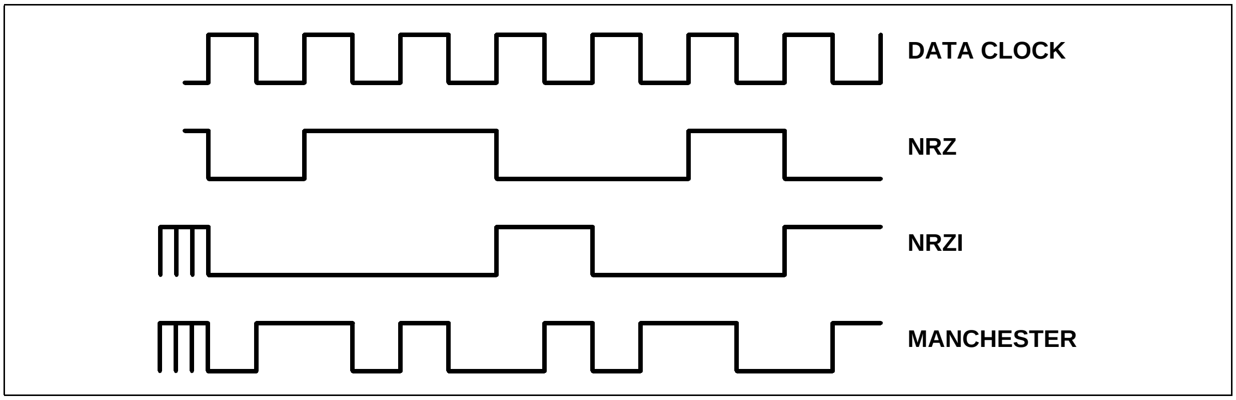
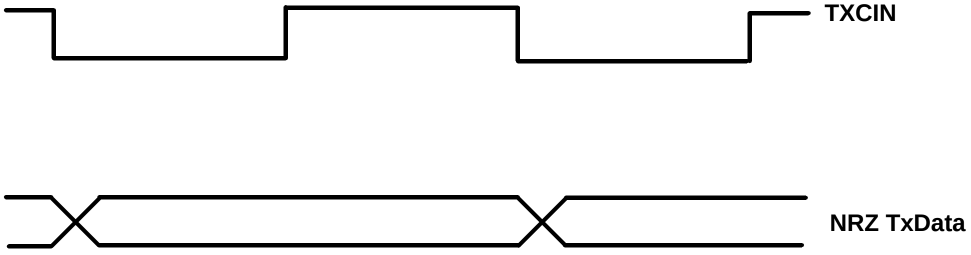


Figure 12. Transmit Data With External Clock In



NOTE: When using the external receive clock in Receive mode, data is sampled on the low-to-high-going edge of RXCIN.

Figure 13. Transmit Data With External Clock Out

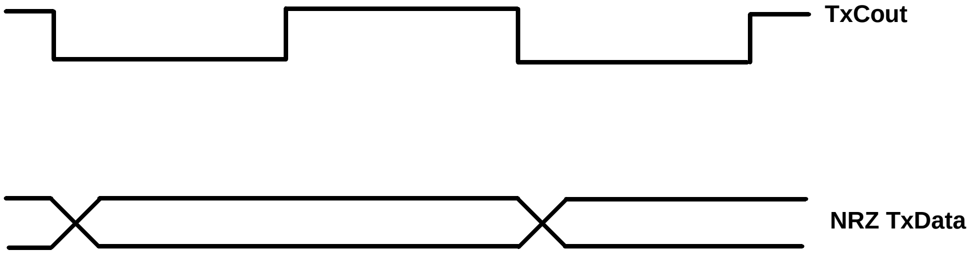


Table 9. Data Clock Selection Using External Clock

Bit Rate	External Clock Frequency	Divisor (hex)
Clock = 33 MHz		
50	9.765 kHz	c2
110	9.765 kHz	57
150	9.765 kHz	40
300	39.062 kHz	81
600	39.062 kHz	40
1200	156.250 kHz	81
2400	156.250 kHz	40
3600	625.00 kHz	ef
4800	625.00 kHz	81
7200	1.250 MHz	ac
9600	1.250 MHz	81
19200	1.250 MHz	40
38400	1.250 MHz	1f
56000	1.250 MHz	15
64000	1.250 MHz	12
76800	1.250 MHz	0f
115200	2.00 MHz	10
128000	2.00 MHz	0f

5.6 Hardware Configurations

To demultiplex the A/D[15:0] bus into separate address and data buses, external buffers and latches are required. To reduce external circuitry, these external devices can be shared in multi–CD1400 applications. The common control lines (ADLD*, AEN*, DATDIR*, DATEN*) to the external devices are wire-OR’ed together. These pins are tristate, not open collector, but an external pull-up resistor (2.2–5.0 k¾) must be connected to each line to ensure logic 1 when no CD1400 is a bus master.

When no higher-priority alternate bus masters are present, a daisy-chain priority scheme can be implemented by wire OR’ing BR* and BGACK* and connecting directly to the 680X0. The 680X0 BG* signal is then connected to the first device in the chain and daisy-chained to the remaining devices. A lower-priority bus master can then be connected at the end of the chain.

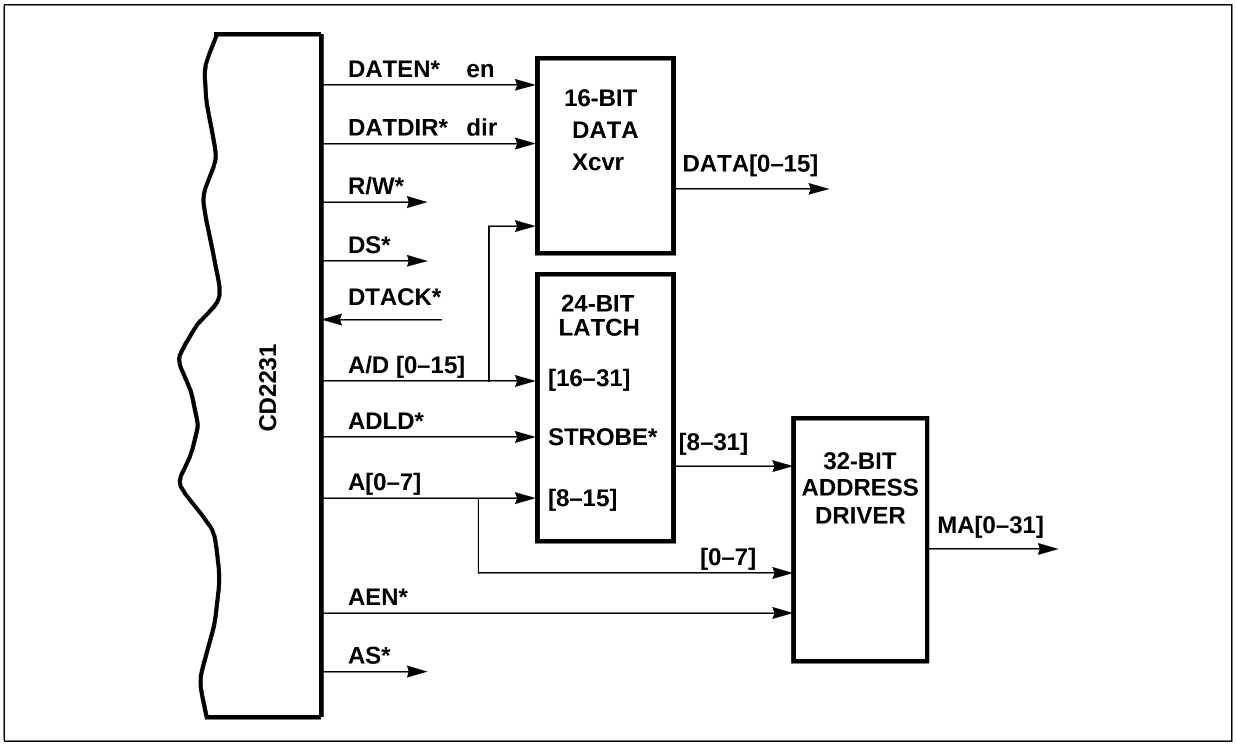
If a higher-priority bus master is present, the BG* signal must be qualified before being passed into the highest priority CD1400. If a priority encoded scheme is required, the BR* signals must be prioritized externally and BG* signals routed to individual devices.

5.6.1 Interface to a 32-Bit Data Bus

To interface to a 32-bit data bus, two 16-bit data buffers must be used to isolate the CD2231 A/D[0–15] pins from either half of the 32-bit bus. The A[1] address pin determines if the lower or upper half of the data bus is in use for a particular bus cycle. The CD2231 always drives all 16 data bits during a register-read or DMA-write operation, regardless of the size of the actual transfer.

5.6.2 DMA Connections for the CD2231

Figure 14. DMA Connections for the CD2231



NOTES:

1. The 24-bit latch is *required*.
2. The 16-bit Xcvr is *optional* depending on application.
3. The 32-bit driver is *optional* depending on drive needs.

5.6.3 CD2231 as a DTE and DCE Interface[†]

Table 10 shows the recommended DTE (data terminal equipment) connections between the CD2231 and RS-232C standard interfaces.

[†] Reference: CCITT 1988 Blue Book.

Table 10. DTE Connections

CD2231	RS-232C
RXD	BB
TXD	BA
RTS*	CA
CTS*	CB
DSR*	CC
TXCOUT/DTR*	–/CD
RXCIN	DD
TXCIN	DB
RXCOUT	DA
CD*	–/CF

Table 11 shows the recommended DCE (data communications equipment) connections between the CD2231 and RS-232C standard interfaces.

Table 11. DCE Connections

CD2231	RS-232C
RXD	BA
TXD	BB
RTS*	CB
CTS*	CA
DSR*	CD
TXCOUT/DTR*	DB/CC
RXCIN	–
TXCIN	DA
RXCOUT	DD
CD*	DA/–

6.0 Protocol Processing

6.1 HDLC Processing

6.1.1 Frame Check Sequence

The FCS is a 16-bit standard computation used in HDLC, and defined in ISO 3309. This FCS algorithm is the same that is used with the synchronous HDLC operation of the CD2231. The basic characteristics of the FCS are the following:

Accumulation: FCS computation starts after the opening flag and continues to the closing flag.

Polynomial: The standard polynomial is $x^{16} + x^{12} + x^5 + 1$.

Pre-load: The FCS 16-bit accumulator is pre-set to all ‘1’s.

Transmit order: The FCS bits are identified as X15 to X0. The most-significant bit is X15, and is transmitted first. Thus, the first FCS character transmitted has bits X15–X8 in character positions D1–D8, respectively. The second FCS character has bits X7–X0 in character positions D1–D8, respectively.

Transmit polarity: Inverted.

Correct remainder: The receiver calculates the entire received frame, including the received FCS field. If the frame is received error free, then the correct remainder in the FCS accumulation is ‘F 0 B 8’ (X15 is the leftmost bit).

The FCS can be individually enabled or disabled for the transmitter and receiver.

If enabled for the transmitter, the device appends the FCS on transmitted frames. If disabled, the device adds no FCS at the end of the frame.

If enabled for the receiver, the device computes the received FCS and reports the results. If the FCS append is also enabled, the device includes the 2-byte FCS in the received data presented to the host. If disabled, the device does not test the received FCS.

6.1.2 HDLC Transmit Mode

The transmitter can be programmed to idle in either Flag (01111110) or Mark (continuous 1’s) mode by the Idle bit (COR3[3]) in Channel Option Register 3. When idle in Mark mode, frame transmission can be programmed to be prepended by a programmable number of pad characters and flags. The pad character can be selected as either 00 or AA. The pad characters allow the remote receivers phase locked loop to synchronize quickly to the data. When NRZI encoding is used for Manchester encoding, the 00 character guarantees a transition every bit time, and the AA character guarantees exactly one transition per bit time.

If the transmitter is idle in Mark mode, frame transmission is started when data is made available to the transmitter, either by the TDR (Transmit Data register) or a DMA buffer. First, the programmable number of pad characters are transmitted, then the programmable number of flag characters. Data characters are then transmitted and a CRC value accumulated using each data character.

When end of frame status is passed to the CD2231 by the TEOIR or the A/BTBSTS, and the remaining data transmitted, the CRC and a closing flag are appended to the frame. If a new frame is available immediately, the correct number of opening flags are transmitted and data transmission starts. If data is not available, the line is returned to its idle condition.

If data underrun occurs, the CD2231 does not append a CRC, but aborts the transmission by sending eight continuous ‘1’s, and then reverts to the idle condition. An underrun interrupt is generated, and if interrupt transfer is being used, the CPU should provide an EOF response in TEOIR. If DMA Transfer mode is being used, the CD2231 discards DMA buffers until an EOF buffer is found; transmission then resumes from the next buffer. This ensures correct operation when a multiple buffer frame underruns.

When programmed in NRZI mode and idle in Mark mode, after the closing flag and the first eight ‘1’s are transmitted, the transmit data line is sampled to determine if it is a logic high or low. If it is low, an extra ‘0’ is transmitted to force the line to be a logic high.

When idle in Flag mode is selected, the send pad and opening number of flags have no significance; transmission is started when data is first made available in the FIFO. If no data underrun occurs, the frame is terminated normally with a CRC, and then continuous flags are generated. If an underrun does occur, then no CRC is appended, eight ‘1’s are transmitted, and then continuous flags and an underrun interrupt are generated.

6.1.3 HDLC Receive Mode

When enabled, the receiver enters Flag Hunt mode. When the first flag is detected, the next non-flag/abort character is treated as the start of frame. If no address recognition is enabled, frame reception then continues; if Address Recognition mode is enabled, the incoming data is compared with the receive address registers. The following two modes of address recognition are available:

- First byte of address field only, (four possible matches available against RFAR[1–4]).
- First and second byte address field, (two possible matches available against RFAR[1–2], RFAR[3–4]).

For the purposes of address matching, the Address Extension bit is not interpreted by the device. The address matching occurs on either the complete first byte, or the complete first and second byte of the frame. If no address match is recognized, Flag Hunt mode is once again entered, thereby discarding the current frame. If a match is found, normal frame reception continues. When the closing flag of the frame is detected, the data remaining in the FIFO is passed to the CPU, either through DMA transfers or Good Data interrupts, and then an EOF (end of frame) interrupt is generated. The CRC can be either validated or ignored. If the CD2231 does not check the CRC, it is passed onto the host. A validated CRC can be discarded or passed onto the host for diagnostic purposes.

The next non-flag/abort character restarts the process; the current state of the receive process is visible to the CPU by the CSR register, which indicates whether data, flag, or mark are currently being received. To support the data phase of an X.21 connection, a clear detect feature can be enabled by COR1. When enabled, the receive data and CTS* pin are monitored for the clear indication (0, off) from the remote. If detected, the remainder of the current frame is discarded, and a clear detect indication is passed to the CPU by the RISR. However, the channel remains in HDLC mode until modified by the CPU.

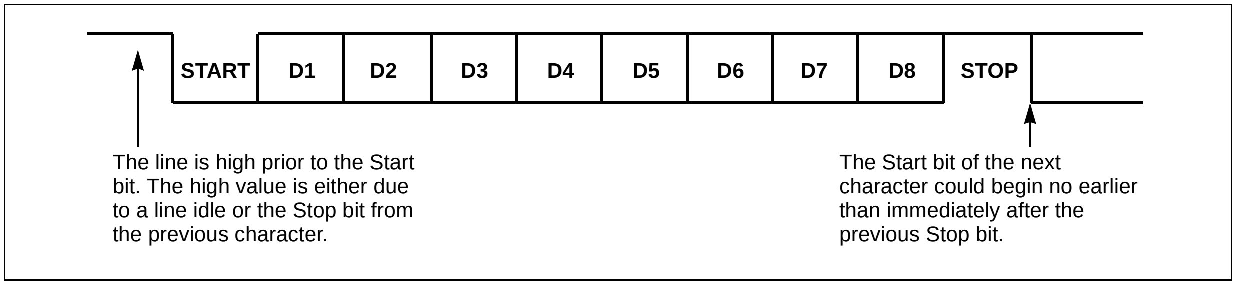
6.2 PPP (Point-to-Point Protocol) Mode

6.2.1 Character Format

The PPP mode uses the async-HDLC character format, which is fixed as one start bit, eight data bits, and one stop bit. There is no parity bit. The character format is as shown in Figure 15.

Using the bit definitions from the standard format (Figure 16), the data bits are identified as D1–D8. D1 is the LSB. Characters are identified as either bits (D1–D8) or as hexadecimal values showing the hex value for bits D5–D8 first, followed by the hex value for D1–D4. Thus, a flag character is ‘01111110’, and is indicated as a hex 7E. A control-escape character is ‘10111110’ or 7D.

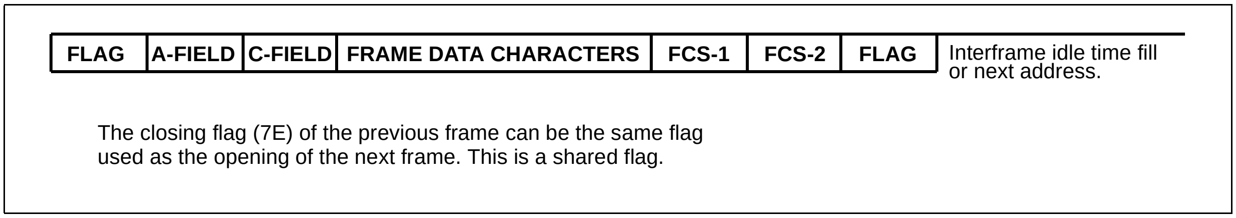
Figure 15. Character Format


START
D1
D2
D3
D4
D5
D6
D7
D8
STOP
The line is high prior to the Start bit. The high value is either due to a line idle or the Stop bit from the previous character.
The Start bit of the next character could begin no earlier than immediately after the previous Stop bit.

6.2.2 Frame Format

The standard frame format is as follows:

Figure 16. Point-to-Point Protocol Frame


FLAG
A-FIELD
C-FIELD
FRAME DATA CHARACTERS
FCS-1
FCS-2
FLAG
Interframe idle time fill or next address.
The closing flag (7E) of the previous frame can be the same flag used as the opening of the next frame. This is a shared flag.

A and C fields

The device passes the A and C fields to and from the host. The device does no special processing on these fields.

6.2.3 Frame Check Sequence

The PPP mode uses the same 16-bit CRC as HDLC mode (V.41).

Everything between the flags is included in the calculation with two exceptions: control-escape (7D) characters added for transparency, and mapped characters received without a preceding control-escape. For characters preceded by a control-escape, the FCS calculation is made after bit 6 is inverted.

6.2.4 Transparency

Transparency means that there is a protocol method to prevent confusion and ambiguity between control characters and data characters in the frame.

For PPP mode, there is a control-escape mechanism. Specific characters are identified as 'control mapped' characters. The control map is called the ACCM (async-control-character map). Whenever there is a mapped character in the data stream, the transmitter precedes that character with a control-escape character of 7D. After the control-escape, the character itself is transmitted with bit 6 inverted. For example, if the character 13 is a mapped character, then the transmission of 13 is 7D–33.

When the receiver sees the 7D control-escape character, the 7D is removed and bit 6 of the following character is inverted. The resultant reconstructed character is passed on to the host as one received character.

6.2.4.1 Mapped Characters from 00–1F

When a channel is selected for PPP mode, two ACCMs are assigned. Each ACCM consists of four registers (32 bits) to define mapped characters in the range 00–1F. One ACCM is for the transmitter (TXACCM), and one for the receiver (RXACCM). Each bit within the ACCM points to a particular character within the range. When the bit is set, that character is a mapped character. When the bit is clear, that character is not a mapped character.

For example, suppose the TXACCM bit pointing to the character 12 is set, and that the TXACCM bit pointing to the character 0B is clear. Then whenever a 12 is present for transmission, the actual transmission is 7D–32. Whenever a 0B is present for transmission, the 0B is transmitted without modification.

Continuing the example, if the receiver ACCM bits pointing to 12 and 0B are also set and clear, respectively. Then a received 12 (without a preceding 7D) is discarded, and a received 0B (without a preceding 7D) is passed through to the host unchanged.

6.2.4.2 Mapped Characters from 20 and Above

Three characters above 20 can be mapped. These characters are defined in the Channel Specific registers TSPMAP[1], TSPMAP[2], and TSPMAP[3].

6.2.4.3 Characters 7D and 7E as Transmitted Data

Whenever the transmitter sees either 7D or 7E as data for transmission, the transmitter treats these as mapped characters. Thus, a 7D as data is transmitted as 7D–5D, and a 7E as data is transmitted as 7D–5E.

6.2.4.4 Mapped Characters in the FCS Field

Whenever the transmitter sees that the FCS result to be transmitted contains a mapped character, it handles that character as any other mapped character. See Section 5.2.4.

For example, if an FCS field is A7–7E, the transmitter would send three characters in the FCS field, A7–7D–5E. The receiver would convert the received FCS back to A7–7E before completing the FCS computation.

6.2.5 Definition of a Valid Frame

This section discusses valid frames from the viewpoint of the CD2231 devices.

All characters are formatted as in the standard async-HDLC format shown in Section 6.2.1. When a channel is placed in the PPP mode, that channel transmits and expects received characters to be as shown in Section 6.2.1. There is one exception of an option to transmit a framing error. See Section 6.2.6.2 and Section 6.2.6.4.

Async-HDLC and PPP protocols have minimum frame size requirements. However, the CD2231 devices makes no requirement of a minimum frame size.

The frame opens and ends with a flag (7E). The device complies with this in transmit, and requires opening and closing flags on the receiver. The closing flag from a preceding frame can be the same flag as the opening flag of the next frame. This is a shared flag. The device can send and receive both shared or non-shared flags.

The frame never ends with a control-escape followed by a flag (7D–7E). The device does not send a 7D–7E at the end of a frame in the normal PPP mode. The device can be commanded to send an abort of either a 7D–7E or a character with a bad stop bit.

If the device receives a frame that ends in a 7D–7E, that frame is indicated to the host as being in error.

The PPP mode requires transparency as described in Section 6.2.4. The transparency is always enabled when the channel is in the PPP mode.

6.2.6 Transmitter

6.2.6.1 Fixed Transmitter Operations

For PPP mode, all transmitted characters are of the format shown in Section 6.2.1, and the transmitter always sends an opening flag.

6.2.6.2 Transmitter Options

The device transmitter can be control-bit selected for the following options:

Option	Description
map32 (ATBSTS) (BTBSTS)	When map32 is set, all the characters in the TXACCM (00–1F) are mapped. All 32 characters are transmitted with a preceding 7D, and with bit 6 flipped. When map32 is clear, the normal TXACCM is used.
npad3, 2, 1, 0 (COR3)	The minimum number idle character times between transmitted frames is programmable from 0–15 character times.
TxGen (COR3)	If TxGen is set, the device adds the two character FCS at the end of each frame. If TxGen is clear, the device ends the frame with a closing flag after the last data byte from the host.
frame (STCR)	When commanded by setting the frame bit in STCR, the device sends one character in the frame with the Stop bit forced to ‘0’.

6.2.6.3 Transmission of Abort

When commanded through a bit in the STCR (Special Transmit Command register), the device ends the transmission of the current frame with an abort sequence of 7D–7E. After executing the abort, the device clears the STCR. The rules for shared flag transmission in Section 6.2.6.2 are followed for the trailing flag (7E) of the abort sequence (7D–7E).

If the device is not sending a frame when the Command bit is set, the device clears the STCR and does not send an abort sequence.

6.2.6.4 Transmit Framing Error

For test purposes, one character with a framing error can be transmitted inside a frame. A Command bit in the STCR notifies the device to transmit one character with Stop bit forced to '0'. If the channel is transmitting a frame, one framing error character is inserted. After transmission, the channel continues with the frame transmission. After executing the command, the device clears the STCR.

If the device is not sending a frame when the Command bit is set, the device clears the STCR and does not send a framing error character.

6.2.7 Receiver

6.2.7.1 Fixed Receiver Operations

The receiver accepts a frame or a character when the received data is brought through the device and presented to the host.

In Async-HDLC mode, the receiver accepts only characters of the format shown in Section 6.2.1.

The receiver accepts only frames that have an opening flag; there can be more than one opening flag.

6.2.7.2 Receiver Options

The device receiver can be control-bit selected for the following options:

Option	Description
RxChk (COR3)	If RxChk is set, the receiver tests the FCS at the end of each frame and reports the result. If RxChk is clear, the receiver makes no FCS computation.
RTPR	The RTPR timer is disabled when all bits are zero. RTPR is enabled with a non-zero value. See Section 6.5.

6.3 SLIP Processing

Note: SLIP, MNP$^{®}$4, and Automatic In-Band Flow Control modes are only available on Revision B and later devices.

6.3.1 Framing

As defined in the original implementation, SLIP frames end with an ‘END’ character and have no beginning character. However, RFC-1055 suggests that all frames begin and end with ‘END’ characters. The CD2231 uses the ‘END’ character essentially as opening and closing flags. The defined characters (see table below) are fixed (hardcoded) and cannot be changed by the user.

Defined Character	Hex Encoding
END	0xC0
ESC	0xDB
ESC_END	0xDC
ESC_ESC	0xDD

The CD2231 uses the following conventions when transmitting a SLIP frame:

- When an ‘END’ character is to be sent, it is replaced by the character sequence ‘ESC’, ‘ESC_END’.
- When an ‘ESC’ character is to be sent, it is replaced by the character sequence ‘ESC’, ‘ESC_ESC’.

During receipt of a frame, the CD2231 makes the following substitutions:

- When an ‘ESC’ character is found in the data stream, only the ‘ESC_END’ and ‘ESC_ESC’ characters can follow. These two character sequences are replaced with a single character:
  - — The sequence ‘ESC’, ‘ESC_END’ is replaced with ‘END’.
  - — The sequence ‘ESC’, ‘ESC_ESC’ is replaced with ‘ESC’.
- Even though the characters ‘ESC_END’ and ‘ESC_ESC’ are the only valid characters following ‘ESC’, RFC-1055 suggests that when other characters are encountered, the ‘ESC’ should be discarded and the second character should be kept unmodified. The CD2231 follows this convention.

The SLIP protocol prohibits in-band flow control. As such, the CD2231 does not respond to XON and XOFF characters in any special way, they are treated as normal data.

6.3.2 Debugging Aids

For debug purposes, the CD2231 can send the sequence ‘ESC’, ‘END’, by the STCR (Special Transmit Command register). This is intended as an abort frame function. The STCR also has a command for sending a bad (0 value) Stop bit, which causes a framing error at the receiving end.

When the CD2231 receives the sequence ‘ESC’, ‘END’, it is reported as ‘receive abort’ in the RISR register. A bad Stop bit is reported as a FE (framing error) in the RISR.

6.4 MNP®4/ARAP Protocol Processing

Note: SLIP, MNP4, and Automatic In-Band Flow Control modes are only available on Revision B and later devices.

6.4.1 Framing

An MNP4 (V.42) frame consists of a start flag, data octets, a stop flag, and a 16-bit FCS (frame check sequence). The FCS uses the polynomial ($x^{16} + x^{12} + x^5 + 1$), preset to all ‘1’s, transmitted, and inverted. The character format uses asynchronous framing with 8 data bits, no parity, and one Stop bit. In-band flow control (XON/XOFF) is not permitted in this mode.

The start flag is a three octet sequence consisting of the start character, escape character, and STX (0x02). The stop flag is a two octet sequence consisting of the escape character and ETX (0x03).

During transmit, if an escape character is encountered in the data stream, it is duplicated. Conversely, the receiver discards the second of two sequential escape characters.

MNP4 is the data-link layer of ARAP 1.0 (AppleTalk™ Remote Access Protocol). ARAP 2.0 is the same as MNP4 except for the two start and escape characters.

The CD2231 uses two Special Character registers (SCHR1 and SCHR2) to hold the definition of the start and escape characters. There is no mode selection within the CD2231 that allows it to determine whether it is in an ARAP 1.0 or ARAP 2.0 environment. It builds and detects frames using the values in the two Special Character registers. The user must load the two Special Character registers with the appropriate start and escape characters for the version in use during channel initialization. The two special characters for each protocol are shown in Table 12.

Table 12. Special Character Definition

Special Character Register 1 and 2	ARAP 1.0	ARAP 2.0
SCHR1 contains the start character	SYN	SOH
SCHR2 contains the escape character	DLE	ESC

For both versions of ARAP, frames begin with SCHR1, SCHR2, STX, and end with SCHR2 and ETX:

- ARAP 1.0— SYN, DLE, STX, data, data, data, ... DLE, ETX
- ARAP 2.0— SOH, ESC, STX, data, data, data, ... ESC, ETX

Both versions escape the escape character (in SHCR2) by duplicating it if it appears within the data stream.

6.4.2 MNP®4/ARAP FCS (Frame Check Sequence) Calculation

Both versions use the ($x^{16} + x^{12} + x^5 + 1$) polynomial, preset to all ‘1’s, transmitted and inverted with remainder equal to 0x1D0F. The frame body and ETX octet of the stop flag are included in the FCS calculation for both versions. The start flag and all DLE (ARAP 1.0)/ESC (ARAP 2.0) octets that are used for transparency are excluded from the FCS calculation. Figure 17 and Figure 18 illustrate the characters used in the FCS calculation. Data used in FCS calculation is in bold print.

Figure 17. ARAP 1.0 Frame

	Start Flag					T			End Flag			
SYN	DLE	STX	**data**	**data**	**DLE**	DLE	**data**	**data**	DLE	**ETX**	FCS1	FCS2

Figure 18. ARAP 2.0 Frame

	Start Flag					T			End Flag			
SOH	ESC	STX	**data**	**data**	**DLE**	DLE	**data**	**data**	ESC	**ETX**	FCS1	FCS2

NOTE: The DLE (ARAP 1.0) and ESC (ARAP 2.0) characters in the middle of the data stream, indicated by the 'T' column, are inserted for transparency and thus not included in the FCS calculation.

6.5 Async Processing

Data is transmitted according to the format options defined in the Channel Option registers. These options determine the character length, parity, and Stop bit length. New data sent from the host is transmitted in a continuous stream, unless one of the following occurs:

- Transmitter disabled — transmission terminated at the end of the current character until transmitter enabled.
- XOFF received from line — transmission terminated at end of the current character until XON received or transmitter enabled.
- Out-of-band flow control — transmission terminated at the end of the current character until out-of-band flow control removed.
- In-line command received in data stream from host — in-line command is executed and transmission resumed.
- Send special character command from host — the current character is completed and the special character is transmitted after which normal transmission is resumed.

6.5.1 Transmitter In-Band Flow Control

For in-band flow control modes to be active, the Special Character Detect mode must be enabled.

Transmit in-band flow control is enabled when the TxIBE (Transmit In-Band Enable) bit (COR2[6]) is set to '1'. When TxIBE is set to '0', in-band flow control is disabled, the IXM (Implied XON Mode) bit, also in COR2[7], has no meaning. The XON and XOFF characters are defined in the Special Character registers SCHR[1] and SCHR[2].

When in-band flow control is enabled (TxIBE = 1), and an XOFF character is received, the channel stops transmission after the current character in the Transmit Shift register and the current character in the Transmit Holding register are transmitted. When IXM = 0, transmission restarts after an XON character is received. When IXM = 1, transmission restarts after any character is received.

The FCT (flow control transparency) Mode bit (COR3[5]) determines whether to pass the received flow control characters to the host. If FCT = 1, the characters are not passed to the host. If FCT = 0, they are passed to the host as exception characters. This bit does not affect non-flow control special characters.

Additional status information about transmitter in-band flow control is available in the CSR (Channel Status register). The TxFloff (Transmit Flow Off) and TxFlon (Transmit Flow On) bits CSR[2:1] are used.

TxFloff = 0 is normal. TxFloff = 1 indicates that the channel has been requested by the remote to stop transmission. This bit is reset to '0' when the channel receives restart, as previously described. This bit is reset to '0' when the transmitter is enabled or disabled, or the channel is reset.

TxFlon = 0 is normal. TxFlon = 1 indicates that the channel has been requested by the remote to restart transmission. This bit is reset to '0' once the channel has restarted transmission. This bit is reset to '0' when the transmitter is enabled or disabled, or the channel is reset.

6.5.1.1 Receiver In-Band Flow Control

The channel can request the remote to stop transmission by sending an XOFF character. Likewise, the channel can request the remote to restart transmission by sending an XON characters. The XON/XOFF characters is transmitted by setting STCR[3] to a '1'.

The CSR contains status bits RxFloff (Receive Flow Off) and RxFlon (Receive Flow On), which are used for receiver in-band flow control.

RxFloff = 0 is normal. RxFloff = 1 indicates the channel has requested that the remote stop transmission. This bit is reset to '0' when the channel requests that the remote restart transmission. This bit is reset to '0' when the receiver is enabled or disabled, or the channel is reset.

RxFlon = 0 is normal. RxFlon = 1 indicates that the channel has requested that the remote restart transmission. This bit is reset to '0' when the next non-flow control character is received. This bit is reset to '0' when the receiver is enabled or disabled, or the channel is reset.

6.5.1.2 Automatic Receive In-Band Flow Control

Note: SLIP, MNP4, and Automatic In-Band Flow Control modes are only available on Revision B and later devices.

The CD2231 can perform automatic in-band flow control, if desired. Automatic in-band flow control means that the device sends XON and XOFF characters based on the level of characters in the receive FIFO. This function is identical to the automatic out-of-band (hardware) flow control that uses the DTR bin.

As with automatic hardware flow control, when the number of characters in the FIFO exceeds the number programmed in COR5 (Channel Option Register 5), the automatic feature is activated and the CD2231 transmits an XOFF character (as defined by SCHR2). When the number of characters falls back to equal or below the programmed value, an XON character (as defined by SCHR1) is transmitted. The CD2231 keeps track of XON/XOFF characters that it has sent so that erroneous flow control characters are not transmitted. For example, it does not transmit an XON simply because the number of characters is below the threshold; it only does so if it had previously sent an XOFF due to the threshold being exceeded. For this reason, the user should not use the Send Special Character command in the STCR (Special Transmit Command register) to send XON/XOFF characters because the CD2231 does not keep track of flow control characters that it did not send automatically. The result could cause confusion on the other end of the connection due to conflicting flow control commands.

Automatic in-band flow control is functional only in standard Async and Async-HDLC/PPP modes; SLIP and MNP4 expressly forbid in-band flow control. See the COR5 description on page 99 for programming details.

6.5.2 Out-of-Band Flow Control

Receive out-of-band flow control is enabled when the CtsAE (CTS Automatic Enable) bit (COR2[1]) is set to ‘1’. In this mode, character transmission begins only after the CTS* pin is active (low). In asynchronous transmission, if CTS* goes inactive (high) after transmission starts, the channel stops transmission after the current character in the Transmit Shift register, and the current characters in the Transmit Holding register are transmitted. In Synchronous modes, if CTS* goes inactive, the channel stops transmission after the current frame. In either case, transmission restarts after CTS* goes active.

The CD2231 can automatically flow-control the remote device by the DTR* pin. This mode is selected by setting a non-zero DTR* threshold in COR5; when both thresholds in COR4 and COR5 are exceeded, the CD2231 sets the DTR* pin high. When the data in the FIFO falls below the DTR* threshold, the DTR* pin is automatically driven low.

Each channel of the CD2231 has four pins that can be used either as a modem control or GPIO (general-purpose input/output) pins. The modem signal names assigned to these four pins were chosen to provide an easy reference for system designers. In fact, they are all simply general-purpose inputs and outputs (if automatic out-of-band flow-control is not used) that can be individually controlled by the Modem Signal Value register(s). Since the pins are general-purpose, system designers can choose to connect the pins in any way appropriate to the application.

However, when the system software design employs automatic out-of-band flow control with the pins, the signal naming convention no longer holds true in some cases, depending on whether the device is used as DCE or DTE. In this case, it is best to think of the pins in terms of their actual uses within the CD2231 and connect them accordingly, without regard to their names; the RTS* and CTS* pins are associated with transmitter and the DTR* and DSR* pins are associated with the receiver. Table 13 shows the recommended signal hook-up if automatic out-of-band flow control is desired.

Table 13. Signal Hookup for Flow Control

Mode		CD2231 Pin Name	Out-of-Band Flow Control
DCE	DTE		
CTS		DTR	Signal remote to transmit
RTS			Not implemented in this direction
	RTS	RTS	Request remote permission to transmit
	CTS	CTS	Enable transmitter

For example, if the CD2231 is designed to be DCE and automatic out-of-band flow control is desired, connect the DTR pin to the remote CTS input. If the CD2231 is to be used as the DTE side, then connect the CD2231 CTS output to the remote CTS input.

Note that if automatic out-of-band flow control is implemented, the activity of the DTR and DSR pins do not implement the function assigned to those signal names by the signaling conventions of the CCITT and other standards organizations. These names would only apply to these pins if they are under program control and not under automatic CD2231 control. In fact, the DTR function enables the modem to go on- and off-line, depending on the state of the pin. If automatic control is used, then DTR goes inactive when the receive FIFO reached the programmed threshold, thus causing the modem to drop the connection (carrier) to the remote; this would not be the correct function based on the state of the receive FIFO.

6.5.3 Line Break Detection and Generation

A line break on the receiver occurs when the input at the RXD (receive data) pin is all zeros (low) for at least one full character time. This is indicated when the Break bit (RISRL[0]) is set to ‘1’.

Line-break generation out of the transmitter is possible when the ETC (Embedded Transmit Command) bit (COR2[5]) is set to ‘1’. A line break is generated when the output at the TXD (transmit data) pin is all zeroes (low) for at least one full character time.

Line breaks can be transmitted by embedding certain sequences in the data stream as defined in Table 14. These sequences are valid for transmitting breaks only if ETC is set to ‘1’. The embedded sequences to transmit a break are listed in Table 14.

The ETC mechanism works in ASYNC mode only, though breaks can be detected in ASYNC, PPP, SLIP, and MNP4 modes.

Table 14. Datastream Transmission Breaks

00h–81h	Send BREAK – Send a line break for at least one character time.
00h–82h-xxh	Insert delay – To increase the break generation beyond one character time, the insert delay sequence can be used. The inserted delay is xx, where xx is a binary number. The delay is xx times the ‘tick’ set by the TPR (Timer Period register). The minimum period of TPR should be 1 millisecond. If the insert delay sequence is not preceded by a send BREAK sequence, there is an inserted delay of all ‘1’s (high) on the output for duration xx.
00h–83h	Stop BREAK – This must follow the send BREAK sequence, or the insert delay sequence.
00h–00h	Send NUL – If the user needs to send a NUL character and ETC = 1, the user can embed 00h–00h to send one NUL character. If there are less than 8 bits per character, the user can also send a NUL character by ‘sending’ an 80h.

NOTE: In addition to insert delay, a ‘break’ can also be increased beyond one full character by transmitting more than one ‘send BREAK’ sequence at a time.

6.5.4 Special Character Transmission

Selected special characters can be sent preemptively by setting the SndSpc (Send Special Character Command) bit in the STCR. The CD2231 channel acknowledges the command by clearing the STCR. Along with the SndSpc bit, the host needs to set-up the three Special Character Select (SSPC0, SSPC1, SSPC2) bits, also in the STCR, to select which character is to be sent.

When the host commands a special character transmission, the channel completes transmitting any characters in the Transmit Shift register and Transmit Holding register, and then transmit the special character sequence. Any other characters awaiting transmission in the FIFO or through DMA are transmitted after the special character.

If the transmitter is off due to in-band flow control, the special characters override and are sent. Special characters override out-of-band flow control. Also if the transmitter is disabled, the special character send command overrides and the character are sent.

Table 15. SSPC[x] Settings

SSPC2	SSPC1	SSCP0	Function
0	0	1	Send Special Character #1
0	1	0	Send Special Character #2
0	1	1	Send Special Character #3
1	0	0	Send Special Character #4
0	0	0	Reserved
1	0	1	Reserved
1	1	0	Reserved
1	1	1	Reserved

6.5.5 Special Character Recognition and Special Character Range

Special character recognition is enabled when the SCDE (Special Character Detection) bit (COR3[4]) is set to ‘1’. The special characters are programmed in the SCHR registers, and are the same characters used for the transmitter.

If the FCT bit (COR3[5]) is set to ‘1’, the channel processes the flow-control characters and discards them. If FCT is set to ‘0’, the received flow-control characters is processed and passed onto the host by exception interrupt.

In the event of an error (framing and/or parity) in a received character sequence, the channel does not interpret this character as a special character. But, if an overrun condition occurred after a special character is detected, the new character is lost and the overrun status is set. In this condition, the CD2231 gives both an overrun exception and a special character recognition status.

6.5.6 Special Character Range

The Special Character Range low and high (SCRl and SCRh) registers define an inclusive range for special character recognition in Asynchronous mode. It is useful for identifying that a received character is within a certain range, such as a control character. To disable this function, if special character detection is enabled, make both SCRl and SCRh equal to Special Character #1 (SCHR1).

Special characters and range detection is through the three Special Character Detect bits (RISR[6:4]) in the RISR register. The meanings of these bits are listed in Table 16.

Table 16. SCdet Settings

SCdet2	SCdet1	SCdet0	Function
0	0	0	No special characters/range detected
0	0	1	Special character 1 matched
0	1	0	Special character 2 matched

Table 16. SCdet Settings

SCdet2	SCdet1	SCdet0	Function
0	1	1	Special character 3 matched if character 1 and 3 sequence not enabled
1	0	0	Special character 4 matched if character 2 and 4 sequence not enabled
1	1	1	The hex value of the receive character is within the range SCRl ≤ receive character ≤ SCRh.

6.5.7 UNIX® Support Features

The COR6 (Channel Option Register 6) provides several functions useful for UNIX TTY drivers, to further reduce the amount of character-by-character processing that the CPU is required to perform. Separate receive and transmit bits are provided to perform CR/NL (carriage return/new line) translations. In transmit, NL can be converted to CR NL or CR converted to NL. In receive, CR can be discarded, NL converted to CR, or CR converted to NL.

In receive processing, separate modes are provided to handle break conditions and character error conditions. Break conditions can be handled in the normal way (by a receive status interrupt), or the condition can be discarded, or the break can be translated to a NULL (00) and passed as normal data to the CPU. Parity and framing errors can either be handled as normal (by receive status interrupts), discarded, translated to a NULL (00) and passed to the CPU as normal data, or the character can be passed to the CPU as normal data preceded by the sequence FF 00.

The LNext option (COR7[6]) provides a mechanism to transfer flow control and other special characters without invoking flow control or special character interrupts at the receiver. If the LNext option is enabled when the LNext character is received, the following character is just passed to the CPU as a normal character. The LNext character is programmed by the LNext register. The ‘Strip’ feature (COR7[7]) strips the eighth bit off each error-free received character. It has no effect on the transmitted data. The flowchart in Figure 19 on page 81 shows the exact order of the CD2231 character processing steps.

6.6 Non-8-Bit Data Transfers

In Asynchronous mode, it is possible to transmit and receive less than 8 bits per character. There can be 5, 6, 7, or 8 bits per character.

For HDLC mode, there are always 8 bits per character transmitted. The CD2231 transmits only byte-aligned frames. The CD2231 receives HDLC frames using transfers of 8 bits per character, except for the last character received before the FCS. If this last character is not aligned to an 8-bit boundary, the ResInd (Residual Indication) bit is set, along with the EOF bit in RISR.

Figure 19. CD2231 Receive Character Processing

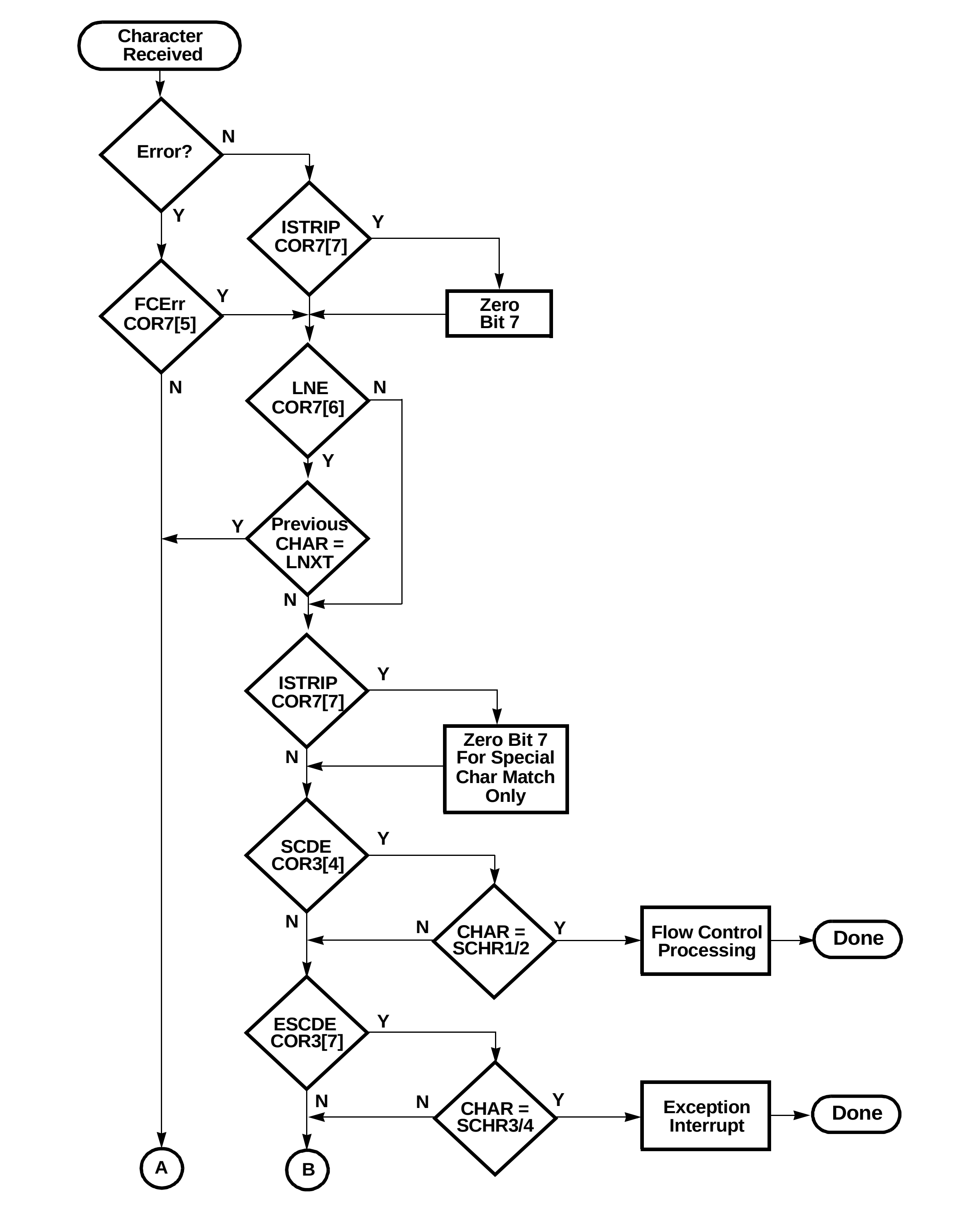
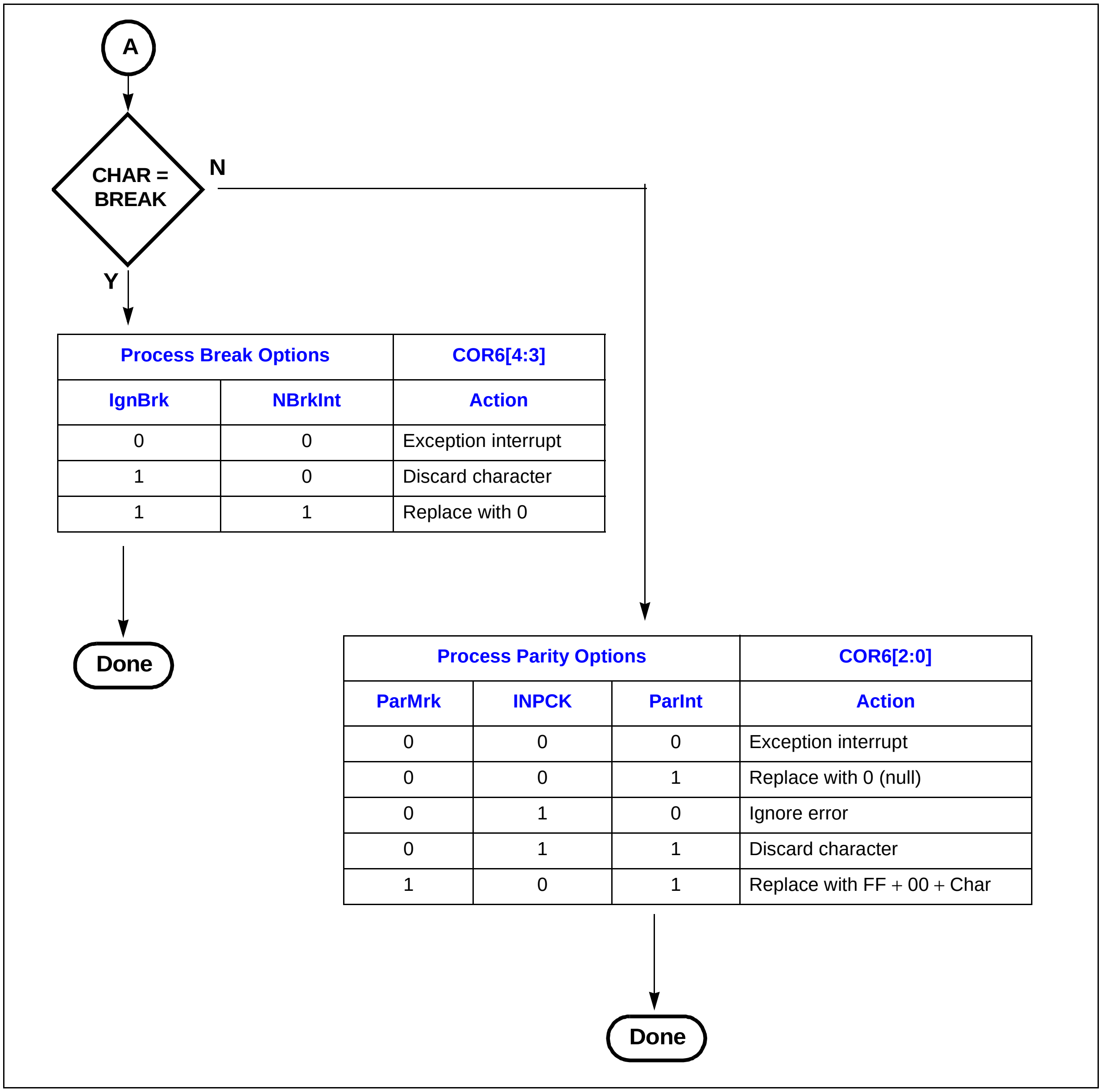


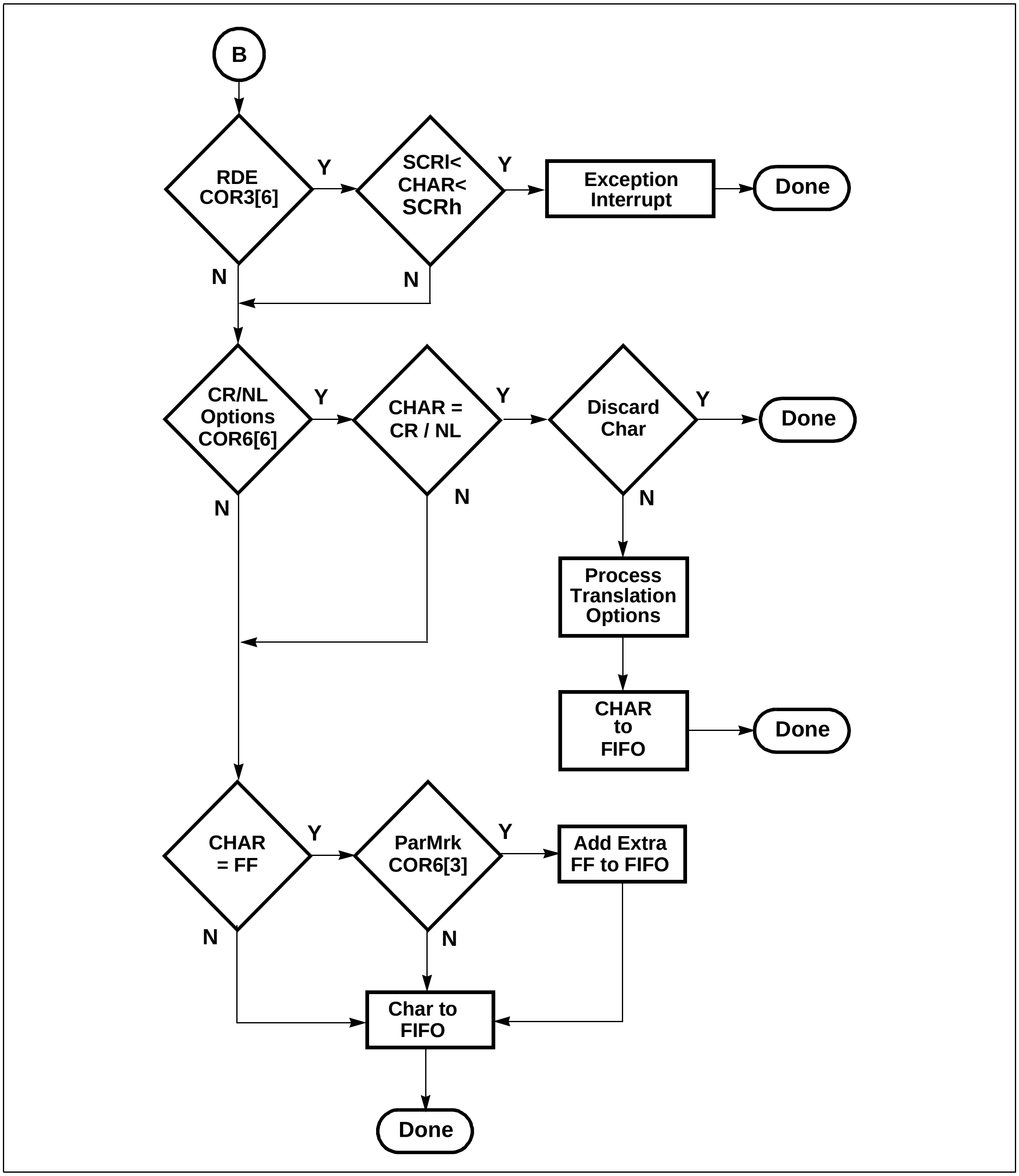
Figure 19. CD2231 Receive Character Processing (Continued)



Process Break Options		COR6[4:3]
IgnBrk	NBrkInt	Action
0	0	Exception interrupt
1	0	Discard character
1	1	Replace with 0

Process Parity Options			COR6[2:0]
ParMrk	INPCK	ParInt	Action
0	0	0	Exception interrupt
0	0	1	Replace with 0 (null)
0	1	0	Ignore error
0	1	1	Discard character
1	0	1	Replace with FF + 00 + Char

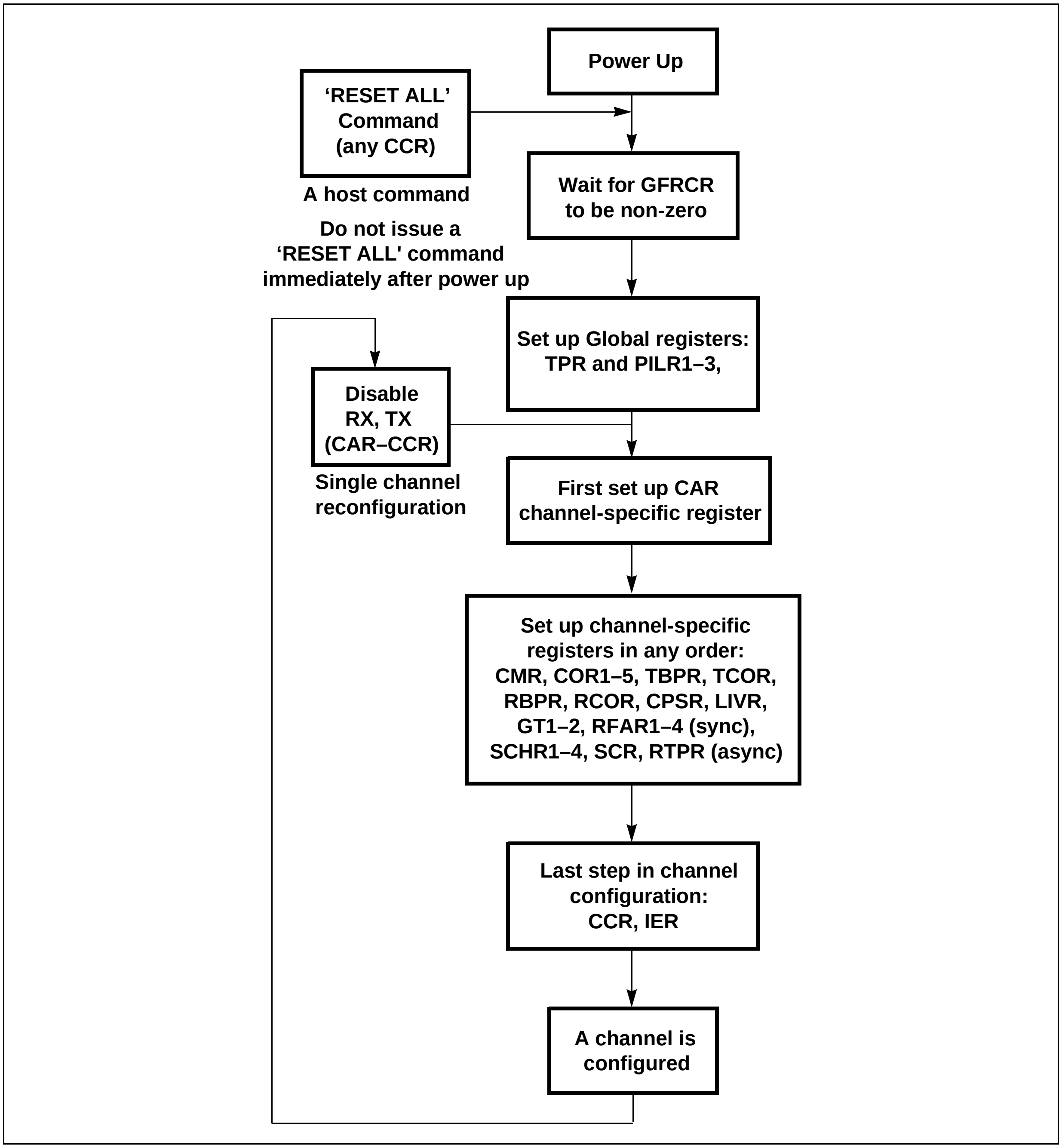
Figure 19. CD2231 Receive Character Processing (Continued)



7.0 Programming Examples

This section provides some examples of the CD2231 programming. Included are examples of global and per-channel initialization, and two interrupt service routines. The code is written in Borland® Turbo C^{++}.

Figure 20. Initialization Sequence for the CD2231


Power Up
'RESET ALL' Command (any CCR)
A host command
Do not issue a 'RESET ALL' command immediately after power up
Wait for GFRCR to be non-zero
Set up Global registers: TPR and PILR1–3,
Disable RX, TX (CAR–CCR)
Single channel reconfiguration
First set up CAR channel-specific register
Set up channel-specific registers in any order: CMR, COR1–5, TBPR, TCOR, RBPR, RCOR, CPSR, LIVR, GT1–2, RFAR1–4 (sync), SCHR1–4, SCR, RTPR (async)
Last step in channel configuration: CCR, IER
A channel is configured

7.1 Global Initialization

The following code segment is an example of global initialization. The host waits for a hardware reset, determined by a non-zero value in the GFRCR (Global Firmware Revision Code register). A 'RESET ALL' command is sent to the CD2231 through the CCR (Channel Command register). The internal processor puts a non-zero value into the GFRCR when initialization is complete. It is recommended to manually clear the GFRCR before issuing the Reset All command, as it takes a small amount of time for the internal processor to detect the command and clear the register. If the host is sufficiently fast, it might read the GFRCR before the command execution commences and incorrectly assumes the command is completed.

The Priority Interrupt Level registers should be loaded with the value of the seven address lines (A[6:0]) during interrupt acknowledge cycles. The TPR (Timer Prescale register) loads the dividing counter that inputs each of the other timers in the CD2231. The DMA Mode register and the Bus Error Count register are used in DMA modes only. After the global portion is done, the Per-Channel registers need to be initialized. Transfers and interrupts should be enabled after all other initialization is complete.

```
// Global Initialization
        while( !inportb( GFRCR ) )// wait for hardware reset
                   ; // wait
        outportb( GFRCR, 0x00 );// manually clear GFRCR

        outportb( CCR, RESET_ALL );// Reset command

        while( !inportb( GFRCR ) ) // wait for reset command
                   ; // wait

        outportb( PILR1, 0x02 );// Priority Interrupt
        outportb( PILR2, 0x04 );// Level Registers
        outportb( PILR3, 0x06 );
        outportb( TPR, 0x40 ); // Set timer prescale
        outportb( BERCNT, 0 ); // Bus error count
        outportb( DMR, 0 );    // DMA mode – 16-bit

// per-channel initialization

        for( i=0; i<2; i++ ) {
              outportb( CAR, i );// set channel number
              init_chan( cor, bpr );// initialize channel
              outportb( CCR, INIT_CH | EN_RX | EN_TX );
              while( inportb(CCR) )
                   ; // wait
              outportb( IER, TX_DATA|RX_DATA );// enable interrupts
        }
```

7.2 Async Interrupt Setup Example

This section contains a code example for an asynchronous channel running at 19,200 bps, with 8 bits/character, 1 Stop bit, and no parity. The sample program enables In-Band Flow Control and Implied Xon mode. This code assumes that the proper channel is set by the CAR (Channel Access register).

```
        outportb( LIVR, 0x40 );
        outportb( RCOR, 0 );   // Receive clock option
        outportb( RBPR, 0x81 );// Baud Rate divisor
        outportb( TCOR, 0 );   // Transmit clock option
        outportb( TBPR, 0x81 );// Baud Rate divisor
        outportb( CMR, ASYNC );// Async Mode, interrupt

        outportb( COR1, PARIGN | CHAR8 );// 8 bit chars, no parity
        outportb( COR2, IXM | TXIBE );// in-band flow,implied XON
        outportb( COR3, STOP1 | FCT );// 1 stop, flow control
        outportb( COR4, thresh );// FIFO threshold
        outportb( COR5, 0 );
```

7.3 HDLC DMA Channel Setup Example

This per-channel initialization code example is for the HDLC protocol at 64 kbps (with CLK = 33 MHz) with NRZI encoding. The setup specifies two extra opening flags before frames, no address matching, and that DMA transfers should be used.

```
        outportb( LIVR, 0x30 );// Set interrupt vector
        outportb( RCOR, DPLL_NRZI );// Receive clock option
        outportb( RBPR, x’3F );// Baud rate divisor
        outportb( TCOR, 0 );  // Transmit clock option
        outportb( TBPR, x’3F );// Baud rate divisor

        outportb( CMR, RX_DMA | TX_DMA | HDLC );// Mode register
        outportb( CPSR, CPSR_CRC_V41 );// CRC polynomial select

        outportb( COR1, NO_ADDR | FLAG_2 );// No address matching,
        outportb( COR2, CRC_V41 );// 2 opening flags
        outportb( COR3, 0 );
        outportb( COR4, thresh );// FIFO threshold
        outportb( COR5, 0);
```

7.4 Receive DMA Interrupt Service Routine

The following code example shows an interrupt service routine for the CD2231 in DMA mode. The buffer class array ib[ ] is used for notational convenience, and its exact implementation is user-defined. The upper ( ) and lower ( ) functions should return the upper and lower 16 bits of the DMA address for the current buffer segment. The ‘nxt_buf()’ accesses the next segment.

If the system uses separate interrupt handlers for receive, transmit, and modem interrupts, the channel number can be obtained from the least-significant bit of the Interrupt register (RIR, TIR, MIR). Otherwise, first use the LIVR to determine the type of interrupt. Receive Good Data interrupts should not occur during DMA transfers. The normal exception is when end-of-frame is received.

The DMABSTS register shows which buffer the CD2231 expects to use next. Fill the descriptor registers for that buffer, including the 2231own bit and return. The last access to the CD2231 during the service routine is the REOIR.

```
int risrl = inportb( RISRL ); // low status
int ch = inportb( RIR ) & 0x01;// channel number

switch( inport(LIVR) & 0x03 ) {
case LIVR_GOODDATA:           // shouldn’t happen in DMA
        break;

case LIVR_EXCEPTION:          // EOF is ’normal’ exception
        if( risrl & RISR_EOF ) {
              if( inportb(DMABSTS) & DMABS_NRBUF ) {// buffer B next
                    outport( BRBADRU, ib[ch].upper() );
```

```
                   outport( BRBADRL, ib[ch].lower() );
                   outport( BRBCNT, BUF_MAX );
                   outport( BRBSTS, OWN_2231 );
                   ib[ch].nxt_buf();// get next buffer
             } else {         // buffer A next
                   outport( ARBADRU, ib[ch].upper() );
                   outport( ARBADRL, ib[ch].lower() );
                   outport( ARBCNT, BUF_MAX );
                   outport( ARBSTS, OWN_2231 );
                   ib[ch].nxt_buf();// get next buffer
             }
        }
}
outportb( REOIR, ZERO );
```

7.5 Transmit Interrupt Service Routine

This example code is a transmit interrupt service handler example. When using a synchronous protocol, transmitters must declare an end of frame if an underrun occurs. If the end of buffer is encountered before data is transferred by this interrupt service, then the Notrans bit (TEIOR[3]) should be set along with EOF (TEIOR[6]). TEOIR is always the last access of an interrupt service routine.

```
int teoir = ZERO;      // default
int tisr = inportb( TISR );// status
int ch = inportb( TIR ) & 0x01;// channel number

switch( tisr ) {
case TISR_UE:
      teoir = TEOIR_EOF;// underflow
      break;
case TISR_TXDATA:
      tftc = inportb( TFTC ); // FIFO count
      for( i=0; i<tftc; i++) {
            if( ob[ch].is_eob() ) {// end of buffer ?
                 ob[ch].nxt_buf();// get next buffer
                 teoir = TEOIR_EOF;
                 if( i==0 )
                       teoir |= NOTRANS;
                 break;
            }
            else outportb( TDR, ob[ch].nxt_char() );//send next character
      }
}
outportb( TEOIR, teoir );
```

8.0 Detailed Register Descriptions

8.1 Global Registers

8.1.1 Global Firmware Revision Code Register (GFRCR)

Register Name: GFRCR
Register Description: Global Firmware Revision Code
Default Value: x’0D
Access: Byte Read/Write

Intel Hex Address: x’82
Motorola Hex Address: x’81

Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Firmware revision code							

This register serves two functions in providing the host with information about the CD2231. When a hardware RESET* signal, or a software RESET ALL command is issued through either of the two Channel Command registers, it initializes the CD2231 and zeroes this register at the start of the initialization. At the end of the initialization, the CD2231 writes its firmware revision code to the GFRCR. All valid CD2231 revision codes are non-zero and the revision code is incremented by one with each new release (for example, GFRCR for Revision D = 34 hex).

Host software must confirm that the GFRCR contents are non-zero before proceeding to configure the CD2231 for normal operation.

8.1.2 Channel Access Register (CAR)

Register Name: CAR
Register Description: Channel Access
Default Value: x’03
Access: Byte Read/Write

Intel Hex Address: x’EC
Motorola Hex Address: x’EE

Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
0	0	0	0	0	0	0	C0

This register contains the channel number for the channel-oriented host read or write operations when the host is not in an interrupt service routine. The CD2231 supplies the interrupting channel number during all interrupt service operations. The Channel Access register contents are not used during an interrupt service. Note that this means that an interrupt service routine is restricted to accessing only the register set of the Interrupting Channel and Global registers.

Bits 7:1	Reserved – *must be ‘0’*.
Bit 0	Channel number

C0	Channel number
0	Channel 0
1	Channel 1

8.2 Option Registers

8.2.1 Channel Mode Register (CMR)

Register Name: CMR Register Description: Channel Mode Default Value: x’02 Access: Byte Read/Write						Intel Hex Address: x’18 Motorola Hex Address: x’1B	
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
RxMode	TxMode	0	0	0	chmd2	chmd1	chmd0

Bit 7 Receive Transfer mode
0 – Interrupt
1 – DMA

Bit 6 Transmit Transfer mode
0 – Interrupt
1 – DMA

Bits 5:3 Reserved – *must be ‘0’*.

Bits 2:0 Protocol mode select
If these options are changed, an initialize command must be given to the CD2231 through the Channel Command register.

chmd2	chmd1	chmd0	
0	0	0	HDLC
0	0	1	Reserved
0	1	0	Async
0	1	1	Reserved
1	0	0	Async–HDLC/PPP
1	0	1	SLIP
1	1	0	MNP4/ARAP
1	1	1	Reserved

8.2.2 Channel Option Register 1 (COR1)

8.2.2.1 COR1 — HDLC Mode

Register Name: COR1
Register Description: Channel Option Register 1
Default Value: x'00
Access: Byte Read/Write

Intel Hex Address: x'13
Motorola Hex Address: x'10

Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
AFLO	ClrDet	AdMd1	AdMd0	Flag3	Flag2	Flag1	Flag0

If any options specified in this register are changed, an initialize command must be given to CD2231 through the Channel Command register.

Bit 7 Address field length option
0 = Address field is 1 octet in length
1 = Address field is 2 octets in length

Bit 6 Clear detect for X.21 data transfer phase
0 = Clear detect disabled
1 = Clear detect enabled
A clear is defined as two consecutive all-zero receive characters with the CTS* pin high.

Bits 5:4 Addressing modes
00 = no address recognition
01 = 4 * 1 byte
10 = 2 * 2 byte
If this bit is set, RFAR1, RFAR2, RFAR3, and RFAR4 should contain the address to be matched. If AFLO is set to '1', an address match is made against the RFAR1 and RFAR2 pair or the RFAR3 and RFAR4 pair.

Bits 3:0 Inter-frame flag option
Defines the minimum number of flags transmitted before a frame is started.

Flags 3	**Flags 2**	**Flags 1**	**Flags 0**	
0	0	0	0	minimum of 1 opening flag, with shared closing/opening flags permitted
0	0	0	1	
	through			minimum number of opening flags sent
1	1	1	1	

The minimum number of opening flags always precede a frame when idle in Mark mode is set, or is always separated by two consecutively transmitted frames. No restriction is placed on the number of flags between received frames.

8.2.2.2 COR1 — Asynchronous Mode

Register Name: COR1 Register Description: Channel Option Register 1 Default Value: x'00 Access: Byte Read/Write							Intel Hex Address: x'13 Motorola Hex Address: x'10
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Parity	ParM1	ParM0	Ignore	Chl3	Chl2	Chl1	Chl0

Bit 7 Parity
1 = odd parity
0 = even parity

Bits 6:5 Parity mode 1 and 0
Defines Parity mode for both transmitter and receiver

ParM1	ParM0	Parity
0	0	none
0	1	force (odd = force 1, even = force 0)
1	0	normal
1	1	reserved

Bit 4 Ignore – Ignore parity
0 = evaluate parity on received characters
1 = do not evaluate parity on received characters

Bits 3:0 Character Length

Chl3	Chl2	Chl1	Chl0	Character Length Bits
0	1	0	0	5
0	1	0	1	6
0	1	1	0	7
0	1	1	1	8

NOTE: Not used in PPP, MNP®4, and SLIP modes.

8.2.3 Channel Option Register 2 (COR2)

8.2.3.1 COR2 — HDLC Mode

Register Name: COR2 Register Description: Channel Option Register 2 Default Value: x’00 Access: Byte Read/Write							Intel Hex Address: x’14 Motorola Hex Address: x’17
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
0	FCSApd	0	CRCNinv	0	RtsAO	CtsAE	DsrAE

Bit 7 Reserved – *must be ‘0’.*

Bit 6 FCS append
0 = Receive CRC is not passed to the host at end of frame
1 = Receive CRC is passed to the host at end of frame

Bit 5 Reserved – *must be ‘0’.*

Bit 4 CRCNinv
0 = CRC is transmitted inverted (that is, CRC V.41)
1 = CRC is not transmitted inverted (that is, CRC-16)

Bit 3 Reserved – *must be ‘0’.*

Bit 2 RTS automatic output enable
When set, if the channel is enabled, the CD2231 automatically asserts the RTS* output when it has characters to send. When Idle-in Mark mode is selected, RTS* is asserted prior to opening flags and remains asserted until after a closing flag has been transmitted.

Bit 1 CTS automatic enable
Enables CTS* input to be used as the automatic transmitter enable/disable. If enabled, the CTS input is checked before frame transmission starts.

Bit 0 DSR Automatic Enable
Enable the DSR* input as the automatic receiver enable/disable. If enabled, DSR* is checked at the beginning of each received frame.

8.2.3.2 Asynchronous / Async-HDLC / PPP Mode

Register Name: COR2 Register Description: Channel Option Register 2 Default Value: x’00 Access: Byte Read/Write							Intel Hex Address: x’14 Motorola Hex Address: x’17
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
IXM	TxIBE	0	0	RLM	RtsAO	CtsAE	DsrAE

Bit 7 IXM – Implied XON mode
IXM has meaning only when TxIBE is set.
If transmission has been stopped due to a received XOFF character, and:

If IXM = 0, transmission is resumed only after the receipt of an XON character or a transmit enable command by the CCR (Channel Command register).

If IXM = 1, transmission is resumed after the receipt of any character or a transmit enable command by the CCR.

Bit 6 TxIBE – Transmit in-band flow control enable
If TxIBE is clear, there is no in-band flow control.

If TxIBE is set, transmission is stopped after the receipt of an XOFF character (cntl-S or hex 13). Immediately after receiving an XOFF, any character in the Transmit Shift register or Holding register is transmitted, and then character transmission is halted. Thus, no more than two characters are sent after receiving an XOFF.

Depending on the IXM bit, either the receipt of an XON (cntl-Q or hex 11) character or any other character (IXM = 1) restarts the transmission. A transmit enable command by the CCR also restarts the transmission.

Bits 5:4 Reserved – *must be '0'*.

Bit 3 RLM – Remote loop back
RLM = 1, enables Remote Loopback mode
RLM = 0, disables Remote Loopback mode

Bit 2 RtsAO – RTS automatic output enable
If RtsAO = 1, the RTS* output pin remains enabled during DMA or character bursts from the transmit FIFO. If the CTS* input pin goes high, RTS* goes high and transmission is stopped after the current burst is completed.

Bit 1 CtsAE – CTS automatic enable
When clear, the transmitter output enable is independent of the CTS* input pin.

When set, the CTS* input pin is evaluated prior to the transmission of each character. If CTS* is asserted low, that character is transmitted completely. If CTS* is high, that character transmission is held until CTS* goes low.

Bit 0 DsrAE – DSR automatic enable
When clear, the receiver input enable is independent of the DSR* input pin.

When set, the DSR* input pin is evaluated at the end of each received character. If DSR* is asserted low, the receiver input is enabled for the next character. If DSR* is high, the receiver is disabled until DSR* goes low.

8.2.3.3 COR2 — MNP4/SLIP Mode

Register Name: COR2 Intel Hex Address: x'14
Register Description: Channel Option Register 2 Motorola Hex Address: x'17
Default Value: x'00
Access: Byte Read/Write

Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
0	0	0	0	RLM	RtsAO	CtsAE	DsrAE

SLIP, MNP4, and Automatic In-Band Flow Control modes are only available on Revision B and later devices.

Bits 7:6 Reserved – *must be '0'*. No in-band flow control in MNP4 mode.

Bits 5:4 Reserved – *must be '0'*.

Bit 3 RLM – Remote Loop Back mode
RLM = 1 enables Remote Loopback mode
RLM = 0 disables Remote Loopback mode

Bit 2 RtsAO – RTS automatic output enable
If RtsAO = 1, then the RTS* output pin remains enabled during DMA or character bursts from the transmit FIFO. If the CTS* input pin goes high, then RTS* goes high and transmission is stopped after the current burst is completed.

Bit 1 CtsAE – CTS automatic enable
When clear, the transmitter output enable is independent of the CTS* input pin.

When set, the CTS* input pin is evaluated prior to the transmission of each character. If CTS* is asserted low, that character is transmitted completely. If CTS* is high, that character transmission is held until CTS* goes low.

Bit 0 DsrAE – DSR automatic enable
When clear, the receiver input enable is independent of the DSR* input pin.
When set, the DSR* input pin is evaluated at the end of each received character. If DSR* is asserted low, the receiver input is enabled for the next character. If DSR* is high, the receiver is disabled until DSR* goes low.

8.2.4 Channel Option Register 3 (COR3) — Async-HDLC/PPP Mode

Register Name: COR3 Intel Hex Address: x'15
Register Description: Channel Option Register 3 Motorola Hex Address: x'16
Default Value: x'00
Access: Byte Read/Write

Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Stop2	FCSApd	RxChk	TxGen	npad3	npad2	npad1	npad0

Bit 7 Stop2
0 = 1 Stop bit
1 = 2 Stop bit

Bit 6 FCS append
0 = Receive CRC is not passed to the host at the end of the frame
1 = Receive CRC is passed to the host at the end of the frame

Bit 5 RxChk – Receive FCS check enabled
When clear, the channel does not test the 2-byte FCS field. All frame data characters are given to the host.
When set, the channel tests the 2-byte FCS field.

Bit 4 TxGen – Transmit FCS enabled
When clear, the channel does not add the 2-byte FCS field.
When set, the channel adds the 2-byte FCS field at the end of the frame.

Bits 3:0 npad3, npad2, npad1, npad0 – Transmit frame leading pads
The number of character times preceding any frame transmission. A character time is 10 bit times. All zeros in this field disables the leading pads

npad3	npad2	npad1	npad0	Number of leading pads
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
1	1	1	1	15

8.2.4.1 MNP®4 Mode

Register Name: COR3 Intel Hex Address: x’15
Register Description: Channel Option Register 3 Motorola Hex Address: x’16
Default Value: x’00
Access: Byte Read/Write

Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Stop2	FCSApd	RxChk	TxGen	npad3	npad2	npad1	npad0

SLIP, MNP4, and Automatic In-Band Flow Control modes are only available on Revision B and later devices.

Bit 7 Stop2
0 = 1 Stop bit
1 = 2 Stop bit

Bit 6 FCS append
0 = Receive CRC is not passed to the host at the end of the frame
1 = Receive CRC is passed to the host at the end of the frame

Bit 5 RxChk – Receive FCS check enabled
When clear, the channel does not test the 2-byte FCS field. All frame data characters are given to the host.
When set, the channel tests the 2-byte FCS field.

Bit 4 TxGen – Transmit FCS enabled
When clear, the channel does not add the 2-byte FCS field.
When set, the channel adds the 2-byte FCS field at the end of the frame.

Bits 3:0 npad3, npad2, npad1, npad0 – Transmit frame leading pads
The number of character times preceding any frame transmission. A character time is 10 bit times. All zeros in this field disables the leading pads.

npad3	npad2	npad1	npad0	Number of leading pads
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
1	1	1	1	15

8.2.4.2 HDLC Mode

Register Name: COR3 Register Description: Channel Option Register 3 Default Value: x'00 Access: Byte Read/Write							Intel Hex Address: x'15 Motorola Hex Address: x'16
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
sndpad	Alt1	FCSPre	FCS	idle	npad2	npad1	npad0

In Synchronous mode, COR3 is used to specify the learning pattern (pad character) sent by the CD2231 to synchronize the DPLL at the remote end. The pad character (00h or AAh) sent depends on the kind of encoding used.

Bit 7 Sends pad character(s)
1 = CD2231 sends pad character(s) before sending flag when coming out of the Idle-in Mark mode.
0 = CD2231 does not send any pad character.

Bit 6 alt1 – send sync pattern
1 = AAh (Manchester/NRZ encoding) is sent as pad character.
0 = 00h (NRZI encoding) is sent as pad character.

Bit 5 FCS preset
0 = FCS is preset to all '1's (CRC V.41).
1 = FCS is preset to all '0's (CRC-16).

Bit 4 FCS mode
1 = Disables FCS generation and checking. The CD2231 treats the entire frame as data.
0 = Normal FCS mode. The CD2231 generates and appends CRC on transmit and validates CRC on receive using the CRC polynomial selected through the CRC Polynomial Select register.

Bit 3 Idle mode
0 = Idle-in Flag mode
1 = Idle-in Mark mode

Bits 2:0 Character Count – specifies the number of synchronous characters sent.

npad2	npad1	npad0	
0	0	0	Reserved
0	0	1	1 pad character sent
0	1	0	2 pad characters sent
0	1	1	3 pad characters sent
1	0	0	4 pad characters sent
101–111 are reserved.			

8.2.4.3 SLIP Mode

Register Name: COR3 Register Description: Channel Option Register 3 Default Value: x'00 Access: Byte Read/Write							Intel Hex Address: x'15 Motorola Hex Address: x'16
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Stop2	0	0	0	npad3	npad2	npad1	npad0

SLIP, MNP4, and Automatic In-Band Flow Control modes are only available on Revision B and later devices.

Bit 7 Stop2
0 = 1 Stop bit
1 = 2 Stop bit

Bits 6:4 Reserved – *must be '0'*.

Bits 3:0 npad3, npad2, npad1, npad0 – Transmit frame leading pads
The number of character times preceding any frame transmission. A character time is 10 bit times. All zeros in this field disables the leading pads.

npad3	**npad2**	**npad1**	**npad0**	**Number of leading pads**
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
1	1	1	1	15

8.2.4.4 Asynchronous Mode

Register Name: COR3 Register Description: Channel Option Register 3 Default Value: x'00 Access: Byte Read/Write							Intel Hex Address: x'15 Motorola Hex Address: x'16
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
ESCDE	RngDE	FCT	SCDE	Splstp	Stop2	Stop1	Stop0

Bit 7 ESCDE – Extended special character detect enable
0 = Special character detect for SCHR3 and SCHR4 is disabled.
1 = Special character detect for SCHR3 and SCHR4 is enabled; a special character interrupt is generated following the receipt of a character matching SCHR3 or SCHR4.

Bit 6 RngDE – Range detect enable
0 = Range detect disabled.
1 = Characters between SCRl and SCRh (inclusive) generate special character interrupts.

Bit 5
FCT – Flow Control Transparency mode
0 = Flow control characters received are passed to the host by receive exception interrupts.
1 = Flow control characters received are not passed to the host.
This bit has no effect unless both TxIBE (COR2[6]) and SCDE (COR3[4]) are set.

Bit 4
SCDE – Special character detection
0 = Special character detect for SCHR1 and 2 is disabled.
1 = Special character detect for SCHR1 and 2 is enabled.

This bit must be set along with TxIBE (COR2[6]) before FCT (COR3[5]) becomes effective.

Bit 3
Splstp – Special character I-strip
When set, this bit causes the receive character to be I-stripped (COR3[7] set to ‘0’) for the special character matching functions only. The character passed to the host is unaffected. This function allows special character processing of data without knowing if the data is 8 bits with no parity or 7 bit with parity.

Bits 2:0
Stop2, Stop1, Stop0 – Stop bit length
Specifies the length of the Stop bit.

Stop2	Stop1	Stop0	Stop Bit Length
0	1	0	1 stop bit
0	1	1	1.5 stop bits
1	0	0	2 stop bits
000–001 and 110–111 are reserved.			

8.2.5 Channel Option Register 4 (COR4)

Register Name: COR4
Register Description: Channel Option Register 4 (Modem Change Options and FIFO Transfer Threshold)
Default Value: x’00
Access: Byte Read/Write

Intel Hex Address: x’16
Motorola Hex Address: x’15

Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
DSRzd	CDzd	CTSzd	0	FIFO threshold			

(Modem Change Options and FIFO Transfer Threshold)

Bit 7
DSRzd = 1
Detect one-to-zero transition on the DSR* input (zero-to-one transition of DSR (MSVR) bit)

Bit 6
CDzd = 1
Detect one-to-zero transition on the CD* input (zero-to-one transition of CD (MSVR) bit)

Bit 5
CTSzd = 1
Detect one-to-zero transition on the CTS* input (zero-to-one transition of CTS (MSVR) bit)

Bit 4
Reserved – *must be ‘0’*.

Bits 3:0 FIFO Threshold in characters
Note that the maximum value allowed for this field is 12 (0C hex). This 4-bit binary encoded field, sets the FIFO transfer threshold for both transmit and receive FIFOs for both Interrupt and DMA Transfer modes.

In Asynchronous mode, a Good Data transfer is initiated for the number of characters in the FIFO greater than the specified threshold. Receive timeout and the occurrence of a receive data exception are also cause to initiate a receive transfer. In Synchronous modes, data transfer is initiated when the number of characters in the FIFO is greater than the specified threshold. An end of frame also initiates a receive transfer.

For transmit operation, the CD2231 attempts to refill the transmit FIFO when the empty space in the FIFO is greater than the set threshold. In synchronous frame transmissions, the CD2231 stops refilling the transmit FIFO once the last character in the frame transfers to the FIFO.

8.2.6 Channel Option Register 5 (COR5)

Register Name: COR5
Register Description: Channel Option Register 5
Default Value: x'00
Access: Byte Read/Write

Intel Hex Address: x'17
Motorola Hex Address: x'14

Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
DSRod	CDod	CTSod	In/Out	Rx flow control threshold			

This register is used to define the current-state change options to be monitored.

Bit 7 DSRod = 1
Detect zero-to-one transition on DSR* input (one-to-zero transition of DSR (MSVR) bit)

Bit 6 CDod = 1
Detect zero-to-one transition on CD* input (one-to-zero transition of CD (MSVR) bit)

Bit 5 CTSod = 1
Detect zero-to-one transition on CTS* input (one-to-zero transition of CTS (MSVR) bit)

Bit 4 In/Out – Automatic receive flow control select
This bit is ignored when bits 3:0 are all zeros.
0 = Use out-of-band flow control (DTR pin).
1 = Use in-band flow control (automatic transmission of XOFF/XON characters)

Bit 4	Number of characters in FIFO	CD2231 Action
0	Less than or equal to threshold	DTR asserted
0	Greater than threshold	DTR deasserted
1	Less than or equal to threshold	XON transmitted
1	Greater than threshold	XOFF transmitted

Note: Do not use the STCR (Special Transmit Command register) to send XON and XOFF characters while using automatic in-band flow control.

Bits 3:0 Receive flow control FIFO threshold
These four bits define the threshold for automatic flow control activation based on the contents of the receive FIFO. A threshold value of zero disables this function and the setting of bit 4 is ignored. Bit 4 determines whether the out-of-band (DTR pin) or the in-band (XOFF/XON characters) is used to stop the flow of incoming data from the remote transmitter.
When the number of characters in the FIFO exceeds this threshold, the DTR pin deasserts or an XOFF character is transmitted. When the number of characters in the FIFO is less than or equal to the threshold, the DTR asserts or and XON is transmitted.

8.2.7 Channel Option Register 6 (COR6) — Async Mode Only

Register Name: COR6 Register Description: Channel Option Register 6 Default Value: x’00 Access: Byte Read/Write							Intel Hex Address: x’1B Motorola Hex Address: x’18
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
IgnCR	ICRNL	INLCF	IgnBrk	NBrkInt	ParMrk	INPCK	ParInt

CR is defined as 0D hex, NL as 0A hex, and NULL as 00 hex.

Bits 7:5 These three bits are used to enable translation of received CR/NL characters as follows:

IgnCr	ICrRNL	INLCR	
0	0	0	No special action on CR and NL
0	0	1	NL translated to CR
0	1	0	CR translated to NL
0	1	1	CR translated to NL and NL translated to CR
1	0	0	CR discarded
1	0	1	CR discarded and NL translated to CR
1	1	0	CR discarded
1	1	1	CR discarded and NL translated to CR

Bits 4:3 Break Action – These bits determine the action taken after a break condition is received.

IgnBrk	NBrkInt	
0	0	Generate an exception interrupt
0	1	Translate to a NULL character
1	0	Reserved
1	1	Discard character

Bits 2:0 Parity/framing error actions – These bits determine the action taken when a parity or framing error is received.

Following the generation of a break-exception interrupt, a receive exception interrupt is generated with RET bit (RISRl[7]) set, when the end of break is detected. The RET interrupt must be enabled in IER[5] to enable this feature.

ParMrk	INPCK	ParInt	
0	0	0	Generated an exception interrupt
0	0	1	Translated to a NULL character
0	1	0	Ignore error; character passed on as good data
0	1	1	Discard error character
1	0	0	Reserved
1	0	1	Translate to a sequence of FF NULL and the error character and pass on as Good Data
1	1	0	Reserved
1	1	1	Reserved

When ParMrk = 1 and ParInt = 1, each occurrence of FF hex in the data stream is preceded by FF hex to distinguish it from a parity error sequence.

8.2.8 Channel Option Register 7 (COR7) — Async Mode Only

Register Name: COR7 Intel Hex Address: x’04
Register Description: Channel Option Register 7 Motorola Hex Address: x’07
Default Value: x’00
Access: Byte Read/Write

Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
IStrip	LNE	FCErr	0	0	0	ONLCR	OCRNL

CR is defined as 0D hex, NL as 0A hex and NULL as 00 hex.

Bit 7 IStrip – when this bit is set, the most-significant bit of receive characters is stripped, leaving 7-bit characters. IStrip is applied after all other character processing, but before special character processing.

Bit 6 LNext – this bit enables the LNext option
0 = All receive characters are processed for special character detection.
1 = The character following the LNext character is not processed for special character matching or flow control.

This provides a mechanism to transfer flow control and special characters as normal data, without invoking flow control action in the CD2231, and without generating special interrupts. The LNext character is defined in the LNXT register, and when processed, is always passed to the host CPU as normal data.

Bit 5 Flow control on error characters
0 = Characters received with an error are not processed for special character/flow control matching.

1 = All receive characters, even those with errors, are processed for special character/flow control processing.

Bits 4:2 Reserved – *must be '0'*.

Bits 1:0 Transmit processing for CR and NL; these bits define Translation mode when CR and/or NL are present in the transmit data.

ONLCR	OCRNL	
0	0	No special action
0	1	CR translated to NL
1	0	NL translated to the sequence CR NL
1	1	CR translated to NL and NL translated to the sequence CR NL

8.2.9 Special Character Registers — Async Modes Only

Special Character registers can be used for detecting specific receive characters in the incoming data stream, and can be used to transmit characters (by STCR) preempting any data in the transmit FIFO.

8.2.9.1 Special Character Register 1 (SCHR1)

Register Name: SCHR1 Register Description: Special Character Register 2 Default Value: x'00 Access: Byte Read/Write						Intel Hex Address: x'1C Motorola Hex Address: x'1F	
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
User-defined special character, protocol-defined special characters (see below).							

8.2.9.2 Special Character Register 2 (SCHR2)

Register Name: SCHR2 Register Description: Special Character Register 2 Default Value: x'00 Access: Byte Read/Write						Intel Hex Address: x'1D Motorola Hex Address: x'1E	
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
User-defined special character, protcol-defined special characters (see below).							

Asynchronous Mode

Special characters 1 and 2 are used in conjunction with the SCDE bit COR3[4] to detect incoming characters; when both SCDE and TxIBE (COR2[6]) are set, they define the in-band flow control characters XON and XOFF.

SCHR1 = XON
SCHR2 = XOFF

In addition to the SCDE and TxIBE bits, if the FCT bit (COR3) is set when flow control characters are received, they are stripped from the data stream.

MNP®4 Mode

SCHR1 holds the start character.
SCHR1 holds the escape character

	MNP®4/ARAP 1.0	**ARAP 2.0**
SCHR1	SYN 16 hex	SOH 01 hex
SCHR2	DLE 10 hex	ESC 1B hex

8.2.9.3 Special Character Register 3 (SCHR3)

<table>
<tr><td colspan="4">Register Name: SCHR3
Register Description: Special Character Register 3
Default Value: x’00
Access: Byte Read/Write</td><td colspan="4">Intel Hex Address: x’1E
Motorola Hex Address: x’1D</td></tr>
<tr><td><i>Bit 7</i></td><td><i>Bit 6</i></td><td><i>Bit 5</i></td><td><i>Bit 4</i></td><td><i>Bit 3</i></td><td><i>Bit 2</i></td><td><i>Bit 1</i></td><td><i>Bit 0</i></td></tr>
<tr><td colspan="8">User-defined special character</td></tr>
</table>

8.2.9.4 Special Character Register 4 (SCHR4)

<table>
<tr><td colspan="4">Register Name: SCHR4
Register Description: Special Character Register 4
Default Value: x’00
Access: Byte Read/Write</td><td colspan="4">Intel Hex Address: x’1F
Motorola Hex Address: x’1C</td></tr>
<tr><td><i>Bit 7</i></td><td><i>Bit 6</i></td><td><i>Bit 5</i></td><td><i>Bit 4</i></td><td><i>Bit 3</i></td><td><i>Bit 2</i></td><td><i>Bit 1</i></td><td><i>Bit 0</i></td></tr>
<tr><td colspan="8">User-defined special character</td></tr>
</table>

Special characters 3 and 4 are used in conjunction with the ESCDE bit of COR3 to detect characters in the receive data stream and to generate receive special character interrupts.

Note: Special characters 3 and 4 are not stripped from the data stream if Flow Control Transparency (FCT) mode is enabled.

8.2.10 Special Character Range — Async Mode Only

8.2.10.1 Special Character Range low (SCRl)

Register Name: SCRL Register Description: Special Character Range, low Default Value: x’00 Access: Byte Read/Write							Intel Hex Address: x’20 Motorola Hex Address: x’23
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
User-defined special character detect range, low							

8.2.10.2 Special Character Range high (SCRh)

Register Name: SCRH Register Description: Special Character Range, high Default Value: x’00 Access: Byte Read/Write							Intel Hex Address: x’21 Motorola Hex Address: x’22
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
User-defined special character detect range, high							

These registers define an inclusive range for special character recognition in the Asynchronous mode. It can be useful for identifying that a received character is within a user-defined range and is, for example, a control character.

8.2.11 LNext Character (LNXT) — Async Mode Only

Register Name: LNXT Register Description: Literal Next Character Default Value: x’00 Access: Byte Read/Write							Intel Hex Address: x’2D Motorola Hex Address: x’2E
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
User-defined literal next character							

This register defines the LNext character. If the LNext function is enabled (COR7[6]), the CD2231 examines received characters and compare them against this value. If a match occurs, this character and the following are placed in the FIFO without any special processing. In effect, the LNext function causes the CD2231 to ignore characters with special meaning, such as flow control characters. There are two exceptions: a ‘break’ or an ‘errored’ character. If the character following the LNext character is either a ‘break’ or an ‘errored’ character, LNext is placed in the FIFO, and the following character is treated as it normally would be for these error conditions.

8.2.12 Receive Frame Address Registers — HDLC Sync Mode Only

8.2.12.1 Receive Frame Address Register 1 (RFAR1)

Register Name: RFAR1 Register Description: Receive Frame Address Register 1 Default Value: x’00 Access: Byte Read/Write							Intel Hex Address: x’1C Motorola Hex Address: x’1F
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Frame Qualification Address 1							

8.2.12.2 Receive Frame Address Register 2 (RFAR2)

Register Name: RFAR2 Register Description: Receive Frame Address Register 2 Default Value: x’00 Access: Byte Read/Write							Intel Hex Address: x’1D Motorola Hex Address: x’1E
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Frame Qualification Address 2							

8.2.12.3 Receive Frame Address Register 3 (RFAR3)

Register Name: RFAR3 Register Description: Receive Frame Address Register 3 Default Value: x’00 Access: Byte Read/Write							Intel Hex Address: x’1E Motorola Hex Address: x’1D
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Frame Qualification Address 3							

8.2.12.4 Receive Frame Address Register 4 (RFAR4)

Register Name: RFAR4 Register Description: Receive Frame Address Register 4 Default Value: x’00 Access: Byte Read/Write							Intel Hex Address: x’1F Motorola Hex Address: x’1C
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Frame Qualification Address 4							

Reception of an HDLC frame can be qualified with a matched 1- or 2-byte address field either as four 1-byte alternatives or two 2-byte alternatives. The use of RFAR registers for address recognition is described in the Channel Option registers (COR1) on page 90.

8.2.13 CRC Polynomial Select Register (CPSR)

Register Name: CPSR Register Description: CRC Polynomial Select Default Value: x’00 Access: Byte Read/Write							Intel Hex Address: x’D4 Motorola Hex Address: x’D6
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
0	0	0	0	0	0	0	Poly

Bits 7:1 Reserved – *must be ‘0’*.

Bit 0 Polynomial select
0 = CRC V.41 polynomial (normally used for HDLC protocol and preset to 1’s)
x**16 + x**12 + x**5 + 1

1= CRC-16 polynomial (generally used for Bisync but will work in HDLC mode, preset to 0’s)
x**16 + x**15 + x**2 + 1

8.2.14 Transmit Special Mapped Characters — PPP Mode only

8.2.14.1 Transmit Special Mapped Character 1 (TSPMAP1)

Register Name: TSMAP1 Register Description: Special Mapped Transmit Character 1 Default Value: x’00 Access: Byte Read/Write							Intel Hex Address: x’1B Motorola Hex Address: x’18
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
User-defined mapped transmit character							

8.2.14.2 Transmit Special Mapped Character 2 (TSPMAP2)

Register Name: TSMAP2 Register Description: Special Mapped Transmit Character 2 Default Value: x’00 Access: Byte Read/Write							Intel Hex Address: x’04 Motorola Hex Address: x’07
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
User-defined mapped transmit character							

8.2.14.3 Transmit Special Mapped Character 3 (TSPMAP3)

<table>
<tr><td colspan="5">Register Name: TSMAP3
Register Description: Special Mapped Transmit Character 3
Default Value: x’00
Access: Byte Read/Write</td><td colspan="3">Intel Hex Address: x’2D
Motorola Hex Address: x’2E</td></tr>
<tr><td>Bit 7</td><td>Bit 6</td><td>Bit 5</td><td>Bit 4</td><td>Bit 3</td><td>Bit 2</td><td>Bit 1</td><td>Bit 0</td></tr>
<tr><td colspan="8">User-defined mapped transmit character</td></tr>
</table>

The three TSPMAP registers are used to provide control character escape processing on characters outside the 00–1f (hex) range. Each of these three registers are scanned to match the character currently being transmitted; if a match occurs, that character is ‘escaped’ before transmission. If a zero value is found in any of them, the scan is terminated. (Zero is already covered in the standard TXACCM.)

8.2.15 Transmit Async Control Character Maps — PPP Mode Only

8.2.15.1 Transmit Async Control Character Map 0 (TXACCM0)

<table>
<tr><td colspan="5">Register Name: TXACCM0
Register Description: Transmit Async Control Character Map 0
Default Value: x’00
Access: Byte Read/Write</td><td colspan="3">Intel Hex Address: x’1C
Motorola Hex Address: x’1F</td></tr>
<tr><td>Bit 7</td><td>Bit 6</td><td>Bit 5</td><td>Bit 4</td><td>Bit 3</td><td>Bit 2</td><td>Bit 1</td><td>Bit 0</td></tr>
<tr><td>Char. 07</td><td>Char. 06</td><td>Char. 05</td><td>Char. 04</td><td>Char. 03</td><td>Char. 02</td><td>Char. 01</td><td>Char. 00</td></tr>
</table>

8.2.15.2 Transmit Async Control Character Map 1 (TXACCM1)

<table>
<tr><td colspan="5">Register Name: TXACCM1
Register Description: Transmit Async Control Character Map 1
Default Value: x’00
Access: Byte Read/Write</td><td colspan="3">Intel Hex Address: x’1D
Motorola Hex Address: x’1E</td></tr>
<tr><td>Bit 7</td><td>Bit 6</td><td>Bit 5</td><td>Bit 4</td><td>Bit 3</td><td>Bit 2</td><td>Bit 1</td><td>Bit 0</td></tr>
<tr><td>Char. 0F</td><td>Char. 0E</td><td>Char. 0D</td><td>Char. 0C</td><td>Char. 0B</td><td>Char. 0A</td><td>Char. 09</td><td>Char. 08</td></tr>
</table>

8.2.15.3 Transmit Async Control Character Map 2 (TXACCM2)

Register Name: TXACCM2 Register Description: Transmit Async Control Character Map 2 Default Value: x’00 Access: Byte Read/Write							Intel Hex Address: x’1E Motorola Hex Address: x’1D
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Char. 17	Char. 16	Char. 15	Char. 14	Char. 13	Char. 12	Char. 11	Char. 10

8.2.15.4 Transmit Async Control Character Map 3 (TXACCM3)

Register Name: TXACCM3 Register Description: Transmit Async Control Character Map 3 Default Value: x’00 Access: Byte Read/Write							Intel Hex Address: x’1F Motorola Hex Address: x’1C
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Char. 1F	Char. 1E	Char. 1D	Char. 1C	Char. 1B	Char. 1A	Char. 19	Char. 18

The TXACCM registers define transmitted characters in the range 00–1F as mapped (Control bit set) or not mapped (Control bit clear) as follows:

TXACCM0 bits 0–7 control characters 00–07, respectively.
TXACCM1 bits 0–7 control characters 08–0F, respectively.
TXACCM2 bits 0–7 control characters 10–17, respectively.
TXACCM3 bits 0–7 control characters 18–1F, respectively.

8.2.16 Receive Async Control Character Maps — PPP Mode Only

8.2.16.1 Receive Async Control Character Map 0 (RXACCM0)

Register Name: RXACCM0 Register Description: Receive Async Control Character Map 0 Default Value: x’00 Access: Byte Read/Write							Intel Hex Address: x’20 Motorola Hex Address: x’23
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Char. 07	Char. 06	Char. 05	Char. 04	Char. 03	Char. 02	Char. 01	Char. 00

8.2.16.2 Receive Async Control Character Map 1 (RXACCM1)

Register Name: RXACCM1 Register Description: Receive Async Control Character Map 1 Default Value: x’00 Access: Byte Read/Write						Intel Hex Address: x’21 Motorola Hex Address: x’22	
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Char. 0F	Char. 0E	Char. 0D	Char. 0C	Char. 0B	Char. 0A	Char. 09	Char. 08

8.2.16.3 Receive Async Control Character Map 2 (RXACCM2)

Register Name: RXACCM2 Register Description: Receive Async Control Character Map 2 Default Value: x’00 Access: Byte Read/Write						Intel Hex Address: x’22 Motorola Hex Address: x’21	
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Char. 17	Char. 16	Char. 15	Char. 14	Char. 13	Char. 12	Char. 11	Char. 10

8.2.16.4 Receive Async Control Character Map 3 (RXACCM3)

Register Name: TXACCM3 Register Description: Transmit Async Control Character Map 3 Default Value: x’00 Access: Byte Read/Write						Intel Hex Address: x’23 Motorola Hex Address: x’20	
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Char. 1F	Char. 1E	Char. 1D	Char. 1C	Char. 1B	Char. 1A	Char. 19	Char. 18

The RXACCM registers define received characters in the range 00–1F as mapped (Control bit set) or not mapped (Control bit clear) as follows:

RXACCM0 bits 0–7 control characters 00–07, respectively.
RXACCM1 bits 0–7 control characters 08–0F, respectively.
RXACCM2 bits 0–7 control characters 10–17, respectively.
RXACCM3 bits 0–7 control characters 18–1F, respectively.

8.3 Bit Rate and Clock Option Registers

8.3.1 Receive Baud Rate Generator Registers

8.3.1.1 Receive Baud Rate Period Register (RBPR)

Register Name: RBPR	Intel Hex Address: x'C9
Register Description: Receive BitRate Period	Motorola Hex Address: x'CB
Default Value: x'81	
Access: Byte Read/Write	

Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Receive bit rate period (divisor)							

This register contains the preload value for the receive baud rate counter. When using an internal clock option or an n-times external clock, the preload value in conjunction with the receiver clock source chosen, determines the receive bit rate. If a 1× external clock is used, a value of 01h must be loaded in the RBPR.

8.3.1.2 Receive Clock Option Register (RCOR)

Register Name: RCOR	Intel Hex Address: x'CA
Register Description: Receive Clock Option	Motorola Hex Address: x'C8
Default Value: x'00	
Access: Byte Read/Write	

Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
TLVal	0	DpllEn	Dpllmd1	Dpllmd0	ClkSel2	ClkSel1	ClkSel0

This register is used to select the DPLL mode, and the desired clock source for the receive bit rate generator.

Bit 7 TLVal – Transmit line value
This bit reflects the logical value of the transmit data pin. It is a read-only bit; writing to this bit has no effect.

Bit 6 Reserved – *must be '0'*.

Bit 5 DPLL enable
1 = DPLL is enabled
0 = DPLL is disabled

Bits 4:3 DPLL mode selects the type of data encoding used.

Dpllmd1	Dpllmd0	Encoding
0	0	NRZ
0	1	NRZI
1	0	Manchester
1	1	Reserved

Bits 2:0 These three bits select the clock source for the receive baud rate generator or DPLL.

clkSel2	clkSel1	clkSel0	Clock Source
0	0	0	Clk 0
0	0	1	Clk 1
0	1	0	Clk 2
0	1	1	Clk 3
1	0	0	Clk 4
1	0	1	Reserved
1	1	0	External clock
1	1	1	Reserved

NOTE: See the description of clock options in Section 5.5.

8.3.2 Transmit Baud Rate Generator Registers

8.3.2.1 Transmit Baud Rate Period Register (TBPR)

Register Name: TBPR
Intel Hex Address: x'C1
Register Description: Transmit Bit-Rate Period
Motorola Hex Address: x'C3
Default Value: x'81
Access: Byte Read/Write

Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Transmit bit rate period (divisor)							

This register contains the preload value for the transmit baud rate count. When using one of the internal clocks or an n-times external clock, the preload value in conjunction with the transmitter clock source chosen, determines the transmit bit rate. If a 1× external clock or the receive clock is used, a value of 01h must be loaded in the TBPR.

8.3.2.2 Transmit Clock Option Register (TCOR)

Register Name: TCOR Register Description: Transmit Clock Option Default Value: x'00 Access: Byte Read/Write						Intel Hex Address: x'C2 Motorola Hex Address: x'C0	
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
ClkSel2	ClkSel1	ClkSel0	0	Ext-1X	0	LLM	0

This register controls the transmit bit rate generator and Local Loopback mode.

Bits 7:5 These bits select the clock source for the transmit bit rate generator.

ClkSel2	ClkSel1	ClkSel0	Select
0	0	0	Clk 0
0	0	1	Clk 1
0	1	0	Clk 2
0	1	1	Clk 3
1	0	0	Clk 4
1	0	1	Reserved
1	1	0	External clock
1	1	1	Receive clock

NOTE: See the description of clock options in Section 5.5.

Bit 4 Reserved – *must be '0'*.

Bit 3 Times 1 external clock. This bit is set to '1' when the user supplies the data clock on TXCIN pin whose frequency is equal to the transmit data rate. When using the external 1× clock or the clock from the receiver's DPLL, the TBPR must be programmed to 01h.

Bit 2 Reserved – *must be '0'*.

Bit 1 Local Loopback mode
1 = enables the Local Loopback mode
0 = disables the Local Loopback mode

Bit 0 Reserved – *must be '0'*.

8.4 Channel Command and Status Registers

8.4.1 Channel Command Register (CCR)

There are two CCR command sets. Mode 1 (if bit 7 is '0') commands affect basic channel control. In Mode 2 (if bit 7 is '1'), additional commands that control timer functions are available.

8.4.1.1 CCR — Mode 1

Register Name: CCR Register Description: Channel Command, Mode 1 Default Value: x'00 Access: Byte Read/Write							Intel Hex Address: x'10 Motorola Hex Address: x'13
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
0	ClrCh	InitCh	RstAll	EnTx	DisTx	EnRx	DisRx

The various command and control bits in this register perform largely independent functions. The host can assert multiple command bits to achieve the desired effect. The CD2231 clears the register to zero after it accepts and acts on a host command. The host must verify that the contents of this register are zero prior to issuing a new command. If the RESET ALL command is issued, all other commands are ignored. All other combinations are legal, and the order of processing is as follows:

1. Clear channel
2. Initialize channel
3. Enable receive
4. Disable receive
5. Enable transmit
6. Disable transmit

Note: Processing CCR commands is a low-priority task for the internal firmware, since they seldom occur. The user musttake care when waiting for command completions at critical times, that is, during interrupt service routines.

Channel Control Commands (Bit 7 = 0)

Bit 7 Must be zero.

Bit 6 Clear channel command
When this command is issued, the CD2231 clears the data FIFOs and current transmit and receive status of the channel in the CSR. If the channel is currently transmitting a frame in synchronous protocol, the host should issue the transmit abort (special transmit command) before issuing a Clear command. The channel parameters are not affected by a Channel Clear command. The Clear Channel command causes both receive and transmit FIFOs to be cleared, the transmitter and receiver to be disabled, and all DMA Status registers (DMABSTS, A/BRBSTS and A/BTBSTS) to be cleared.

Bit 5 Initialize channel
If any change is made to the Protocol Mode Select bits in the CMR (Channel Mode register) or to the COR1 (Channel Option Register 1), the channel must be reinitialized by this command. The InitCh command causes the internal protocol-specific registers to be initialized.

Warning: If the Initialize Channel command is issued after a channel is already in operation, then a Clear Channel command must be issued prior to, or coinciding with, the Initialize Channel command. Failure to observe this requirement will result in unpredictable device behavior.

Bit 4 RESET ALL
An on-chip firmware initialization of all channels is performed. All channel and global parameters are reset to their power-on reset condition. This command is the strongest the host can issue. None of the other command bits are interpreted if the RESET ALL command is given. The host must re-initialize the CD2231 following the execution of this command just as after a hardware power-on reset. When this command is completed, the GFRCR is updated with the firmware revision code.

Bit 3 Enable transmitter
Enables the transmitter by setting TxEn bit in the Channel Status register (CSR[3]). In Asynchronous mode, this command also clears the transmit flow control options.

Bit 2 Disable transmitter
Disables the transmitter by clearing TxEn bit (CSR[3]). In Asynchronous mode, the Transmit Flow Control bits are cleared.

Bit 1 Enable receiver
Enables the receiver by setting the RxEn bit (CSR[7]). In Asynchronous mode, the Receive Flow Control bits are cleared.

Bit 0 Disable receiver
Disables the receiver by clearing the RxEn bit (CSR[7]). In Asynchronous mode, the Receive Flow Control bits are cleared.

8.4.1.2 CCR Mode 2

Register Name: CCR
Register Description: Channel Command, Mode 2
Default Value: x’00
Access: Byte Read/Write

Intel Hex Address: x’10
Motorola Hex Address: x’13

Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
1	ClrT1	ClrT2	ClrRcv	ClrTx	0	0	0

Either one or both of the timers can be cleared with a single command. Note that if the running timer value is 01h at the time this command is issued, the timer may expire and cause a timer interrupt before the command is processed.

Bit 7 Must be ‘1’.

Bit 6 Clear timer 1
General timer 1 is cleared.

Bit 5 Clear timer 2
General timer 2 is cleared.

Bit 4 Clear receiver command
This command only affects the receiver. It resets all receiver functions like a combination of clear channel, initialize channel and enable receiver commands. ClrRcv clears the receive FIFO and clears receive status in the CSR, except for the RcvEn bit. ClrRcv clears receive DMA buffer status in ARBSTS, BRBSTS, and Receive Status bits in DMABSTS. Clearing the 2231own bits in both the Receive Buffer Status registers means that DMA buffers have to be returned to the CD2231 before receive transfers begin again.

For Synchronous modes, this command puts the receiver back into Syn/Flag Hunt mode.

Bit 3	Clear transmitter command This command only affects the trasmitter and is only available on Revision C and later devices and is only effective in asynchronous protocols. It resets all transmitter functions like a combination of clear channel, initialize channel and transmit commands. ClrTx clears the transmit FIFO and clears transmit status in the CSR, except for the TxEn bit. ClrTx clears transmit DMA buffer status in ATBSTS, BTBSTS, and Transmit Status bits in DMABSTS. Clearing the 24312OWN bits in both the Transmit Buffer Status registers means that DMA buffers have to be returned to the CD2231 before transmit transfers begin again.
Bits 2:0	Reserved – *must be ‘0’.*

8.4.2 Special Transmit Command Register (STCR)

8.4.2.1 Async-HDLC/PPP Mode

Register Name: STCR — Intel Hex Address: x’11
Register Description: Special Transmit Command — Motorola Hex Address: x’12
Default Value: x’00
Access: Byte Read/Write

Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
0	AbortTx	0	0	SndSpc	Frame	Xon	Xoff

Special characters can be transmitted preemptively (ahead of any characters in the transmit FIFO) upon commands described below. When the special character is transmitted, the STCR is cleared by the device.

Bit 7	Reserved – *must be ‘0’.*
Bit 6	Abort Transmission of the two-character sequence (7D–7E) aborts the current transmit frame. All data in the FIFO following the abort is discarded. If DMA is used, the remaining data up to the EOF is discarded.
Bits 5:4	Reserved – *must be ‘0’.*
Bit 3	SndSpc – Send special character command When clear, the frame, Xon, and Xoff bits described below have no meaning. When set, the host should also set one of the following bits: frame, Xon, or Xoff.
Bit 2	Frame – Send framing error Causes the next character in the transmit stream to be sent with an incorrect Stop bit (Stop bit is ‘0’). This bit is intended as a test function. Unlike the Abort bit, this bit does not terminate the transmission.
Bit 1	Xon – Send XON Causes the transmission of an XON (cntl-Q or hex 11).

Note: The user should not use the send XON/XOFF commands if automatic in-band flow control is enabled (Asynchronous modes only) in COR5.

Bit 0 Xoff – Send XOFF
Causes the transmission of an XOFF (cntl-S or hex 13).

The command structure associated with the sndsp Control bit is:

sndsp	frame	Xon	Xoff	Action
0	X	X	X	Send Special Disabled
1	1	X	X	Send one character with FE
1	0	1	X	Send Xon
1	0	0	1	Send Ooff

NOTE: The user should not use the send XON/XOFF commands if automatic in-band flow control is enabled (Asynchronous modes only) in COR5.

8.4.2.2 SLIP/MNP®4 Mode

SLIP, MNP4, and Automatic In-Band Flow Control modes are only available on Revision B and later devices.

Register Name: STCR Register Description: Special Transmit Command Default Value: x’00 Access: Byte Read/Write							Intel Hex Address: x’11 Motorola Hex Address: x’12
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
0	AbortTx	0	0	SndSpc	Frame	0	0

Special characters can be transmitted preemptively (ahead of any characters in the transmit FIFO) upon commands described below. When the special character is transmitted, the STCR is cleared by the device.

Bit 7 Reserved – *must be ‘0’*.

Bit 6 Abort
Transmission of the two-character sequence (7D–7E) aborts the current transmit frame. All data in the FIFO following the abort is discarded. If DMA is used, the remaining data up to the EOF is discarded.

Bits 5:4 Reserved – *must be ‘0’*.

Bit 3 SndSpc – Send special character command
When clear, the frame, Xon, and Xoff bits described below have no meaning.
When set, the host should also set one of the following bits: frame, Xon, or Xoff.

Bit 2 Frame – Send framing error
Causes the next character in the transmit stream to be sent with an incorrect stop bit (stop bit is ‘0’).
This bit is intended as a test function. Unlike the Abort bit, this bit does not terminate the transmission.

Bits 1:0 Reserved – *must be ‘0’*.

8.4.2.3 Async and HDLC Modes

Register Name: STCR Register Description: Special Transmitl Command Default Value: x'00 Access: Byte Read/Write							Intel Hex Address: x'11 Motorola Hex Address: x'12
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
0	AbortTx	AppdCmp	0	SndSpc	SSPC2	SSPC1	SSPC0

The CD2231 clears the register to zero when it accepts a host CPU command.

Bit 7 Reserved – *must be '0'*.

Bit 6 Abort transmission (HDLC)
Terminates the frame currently in transmission with an abort sequence. In DMA mode, all data up to the next EOF is discarded.

Bit 5 Append complete (Asynchronous DMA mode)
This bit should be set by the host when the last addition has been made to the append buffer.

Bit 4 Reserved – *must be '0'*.

Bit 3 SndSpc – Send special character command
In Asynchronous mode, sends a user-defined special character or special-character sequence. The special character is transmitted ahead of any data remaining in the FIFO.

Bits 2:0 Special character select

SSPC2	SSCP1	SSPC0	Function
0	0	0	Reserved
0	0	1	Send Special Character 1
0	1	0	Send Special Character 2
0	1	1	Send Special Character 3
1	0	0	Send Special Character 4
1	0	1	Reserved
1	1	0	Reserved
1	1	1	Reserved

NOTE: The user should not use the send XON/XOFF commands if automatic in-band flow control is enabled (Asynchronous modes only) in COR5.

8.4.3 Channel Status Register (CSR)

This status register stores the current state of the channel. It can be read by the host at any time. The states of the RxEn and the TxEn bits are controlled by host CPU commands to the CCR.

8.4.3.1 CSR — HDLC Mode

Register Name: CSR Register Description: Channel Status Default Value: x’00 Access: Byte Read/Write							Intel Hex Address: x’19 Motorola Hex Address: x’1A
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
RxEn	RxFlag	RxFrame	RxMark	TxEn	TxFlag	TxFrame	TxMark

Bit 7 Receiver enable
0 = Receiver is disabled
1 = Receiver is enabled

Bit 6 Rx flag
0 = Currently not receiving flag/sync
1 = Currently receiving flag/sync

Bit 5 Rx frame
0 = Currently not receiving frame
1 = Currently receiving frame

Bit 4 Rx mark
0 = Currently not receiving continuous mark
1 = Currently receiving continuous mark

Bit 3 Transmitter enable
0 = Transmitter is disabled
1 = Transmitter is enabled

Bit 2 Tx flag
0 = Currently not transmitting flag
1 = Currently transmitting flag

Bit 1 Tx frame
0 = Currently not transmitting frame
1 = Currently transmitting frame

Bit 0 Tx mark
0 = Currently not transmitting continuous ones
1 = Currently transmitting continuous ones

8.4.3.2 CSR — Async Mode

Register Name: CSR Register Description: Channel Status Default Value: x’00 Access: Byte Read/Write							Intel Hex Address: x’19 Motorola Hex Address: x’1A
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
RxEn	RxFloff	RxFlon	0	TxEn	TxFloff	TxFlon	0

If the host determines that a flow control state is inappropriate, it can be cleared by enabling or disabling the transmitter or receiver by CCR command.

Bit 7 Receiver enable
0 = Receiver is disabled
1 = Receiver is enabled

Bit 6 Receive flow off
0 = Normal
1 = The CD2231 has requested the remote to stop transmission (Send XOFF command has been given to the channel). This bit is reset when the CD2231 has requested the remote to restart transmission, or when the receiver is enabled or disabled, or the channel is reset.

Bit 5 Receive flow on
0 = Normal
1 = The CD2231 has requested the remote to restart character transmission (Send XON command has been given to the channel). This bit is reset when the next (non-flow control) character is received, or when the receiver is enabled or disabled, or the channel is reset.

Bit 4 Unused

Bit 3 Transmitter enable
0 = Transmitter is disabled
1 = Transmitter is enabled

Bit 2 Transmit flow off
0 = Normal
1 = The CD2231 has been requested by the remote to stop transmission. This bit is reset when the CD2231 receives a request to resume transmission, or when the transmitter is enabled or disabled, or the channel is reset.

Bit 1 Transmit flow on
0 = Normal
1 = The CD2231 has been requested by the remote to resume transmission. This bit is reset once character transmission is resumed, or when the transmitter is enabled or disabled, or the channel is reset.

Bit 0 Unused

8.4.3.3 Async-HDLC/PPP Mode

Register Name: CSR
Register Description: Channel Status
Default Value: x’00
Access: Byte Read/Write

Intel Hex Address: x’19
Motorola Hex Address: x’1A

Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
RxEn	RxFloff	RFram	Ridle	TxEn	TxFloff	TFram	Tidle

Bit 7 RxEn – Receiver enabled status
When set, the receiver is enabled.
When clear, the receiver is disabled.

Bit 6 RxFloff – Receive flow off status
When set, Xoff has been transmitted as commanded in the STCR. RxFloff indicates that the remote station has been requested to stop transmission. RxFloff remains set until the host issues an STCR command to send an Xon, or when the receiver is enabled or disabled, or the channel is reset.

When clear, the remote station is not requested to stop transmission. RxFloff remains set until the host issues an STCR command to send an Xon.

Bit 5 RFram – Receive frame status
When set, a frame is being received.
When clear, no frame is being received.

Bit 4 RIdle – Receiver idle status
When set, the receiver input is idle.
When clear, the receiver input is not idle.
Notice that RFram and RIdle are mutually exclusive.

Bit 3 TxEn – Transmitter enabled status
When set, the transmitter is enabled.
When clear, the transmitter is disabled.

Bit 2 TxFloff– Transmit flow off status
This bit has no meaning unless TxIBE in COR2 is set.
When set, an Xoff has been received, and the transmitter has stopped sending data.
When clear, the transmitter is able to transmit if there are characters to send.

Bit 1 TFram – Transmit frame status
When set, a frame is being transmitted.
When clear, no frame is being transmitted.

Bit 0 TIdle – Transmitter idle status
When set, the transmitter output is idle.
When clear, the transmitter output is not idle.
Note that TFram and TIdle are mutually exclusive.

8.4.3.4 SLIP/MNP®4 Mode

SLIP, MNP4, and Automatic In-Band Flow Control modes are only available on Revision B and later devices.

Register Name: CSR Register Description: Channel Status Default Value: x'00 Access: Byte Read/Write							Intel Hex Address: x'19 Motorola Hex Address: x'1A
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
RxEn	0	RFram	Ridle	TxEn	0	TFram	Tidle

Bit 7 RxEn – Receiver enabled status
When set, the receiver is enabled.
When clear, the receiver is disabled.

Bit 6 Reserved – *must be '0'*.

Bit 5 RFram – Receive frame status
When set, a frame is being received.
When clear, no frame is being received.

Bit 4 RIdle – Receiver idle status
When set, the receiver input is idle.
When clear, the receiver input is not idle.
Note that RFram and RIdle are mutually exclusive.

Bit 3 TxEn – Transmitter enabled status
When set, the transmitter is enabled.
When clear, the transmitter is disabled.

Bit 2 Reserved – *must be '0'*.

Bit 1 TFram – Transmit frame status
When set, a frame is being transmitted.
When clear, no frame is being transmitted.

Bit 0 TIdle – Transmitter idle status
When set, the transmitter output is idle.
When clear, the transmitter output is not idle.
Note that TFram and TIdle are mutually exclusive.

8.4.4 Modem Signal Value Registers (MSVR)

8.4.4.1 Modem Signal Value Register (MSVR-RTS)

<table>
<tr><td colspan="5">Register Name: MSVR-RTS
Register Description: Modem Signal Value - RTS
Default Value: x'00
Access: Byte Read/Write</td><td colspan="3">Intel Hex Address: x'DC
Motorola Hex Address: x'DE</td></tr>
<tr><td><i>Bit 7</i></td><td><i>Bit 6</i></td><td><i>Bit 5</i></td><td><i>Bit 4</i></td><td><i>Bit 3</i></td><td><i>Bit 2</i></td><td><i>Bit 1</i></td><td><i>Bit 0</i></td></tr>
<tr><td>DSR</td><td>CD</td><td>CTS</td><td>DTRop</td><td>0</td><td>0</td><td>DTR</td><td>RTS</td></tr>
</table>

8.4.4.2 Modem Signal Value Register (MSVR-DTR)

<table>
<tr><td colspan="5">Register Name: MSVR-DTR
Register Description: Modem Signal Value - DTR
Default Value: x'00
Access: Byte Read/Write</td><td colspan="3">Intel Hex Address: x'DD
Motorola Hex Address: x'DF</td></tr>
<tr><td><i>Bit 7</i></td><td><i>Bit 6</i></td><td><i>Bit 5</i></td><td><i>Bit 4</i></td><td><i>Bit 3</i></td><td><i>Bit 2</i></td><td><i>Bit 1</i></td><td><i>Bit 0</i></td></tr>
<tr><td>DSR</td><td>CD</td><td>CTS</td><td>DTRop</td><td>0</td><td>0</td><td>DTR</td><td>RTS</td></tr>
</table>

Either of these registers is read to determine the current input levels on the input modem pins. Note that the pin definitions for these signals is negative true while the register values are positive true. Two registers are provided for control of the outputs — DTR and RTS. Writing to the MSVR-DTR register affects only the DTR pin. Writing to the MSVR-RTS register affects only the RTS pin.

Bit 7 DSR – Current state of data set ready input

Bit 6 CD – Current state of carrier detect input

Bit 5 CTS – Current state of clear to send input

Bit 4 DTR option – written by MSVR-DTR register
0 = value of DTR bit is output on TXCOUT/DTR* pin
1 = Transmit clock is output on TXCOUT/DTR* pin

Note: If the transmit clock source is a 1× clock on the TXCIN pin, this signal cannot be driven on TXCOUT/DTR*.

Bit 3 Reserved – *must be '0'*.

Bit 2 Reserved – returns '0' when read; writing has no effect

Bit 1 DTR – Current state of data terminal ready output

Bit 0 RTS – Current state of request to send output

8.5 Interrupt Registers

8.5.1 General Interrupt Registers

8.5.1.1 Local Interrupt Vector Register (LIVR)

Register Name: LIVR Register Description: Local Interrupt Vector Default Value: x'00 Access: Byte Read/Write							Intel Hex Address: x'0A Motorola Hex Address: x'09
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
X	X	X	X	X	X	IT1	IT0

The host effectively controls bits 7:2; the device provides bits 1 and 0 within an interrupt acknowledge context.

The CD2231 has one Local Interrupt Vector register per channel, each with six host-defined bits. The host can opt to embed the channel number and the protocol in use on the channel in the channel vector. The CD2231 supplies two modified bits signifying the type of interrupt service required.

Bits 7:2 User-defined. These six bits can be used as the CD2231 device ID number.

Bits 1:0 Interrupt type. These two bits indicate the group/type of interrupt occurring.

IT[1:0]	Group/Type
01	Group 1: Modem signal change interrupt/general timer interrupt
10	Group 2: Transmit data interrupt
11	Group 3: Receive data interrupt
00	Group 3: Receive exception interrupt

Note: Note that because the CD2231 provides a unique Local Interrupt Vector register for each channel, the host has the option to include the channel number within the interrupt vector.

8.5.1.2 Interrupt Enable Register (IER), Non-PPP Modes

Register Name: IER
Register Description: Interrupt Enable
Default Value: x'00
Access: Byte Read/Write

Intel Hex Address: x'12
Motorola Hex Address: x'11

Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Mdm	0	RET	0	RxD	TIMER	TxMpty	TxD

Bit 7 Modem pin change detect
Master interrupt enable for modem change detect functions. The host can select which modem pins are monitored for input change and select either or both directions of change by programming the change detect option bits in COR4 and COR5. A Group1-type interrupt (see LIVR description) is generated from this enable.

Bit 6 Reserved – *must be '0'*.

Bit 5 RET (Async)
In Asynchronous mode, this bit enables a Group 3 receive exception timeout interrupt when a receive data timeout occurs with an empty receive FIFO. This provides a mechanism for the host to manage a partially full receive buffer when receive data stops.

Bit 4 Reserved – *must be '0'*.

Bit 3 Rx data
The receive FIFO threshold has been reached in Interrupt Transfer mode, causing a Group 3 receive data interrupt. Any receive exception causes a Group 3 receive exception interrupt.

Bit 2 Timer
General timer(s) timeout
In Synchronous mode, this bit enables a Group 1 interrupt when either timer reaches zero.

Bit 1 Tx Mpty
Transmitter empty. If enabled, a Group 2 interrupt is generated when the channel is completely empty of transmit data.

Bit 0 Tx Data
Any transmit exception or transmit FIFO threshold reached in Interrupt Transfer mode. Group 2 interrupts are generated at the end of transmit DMA buffers or when the FIFO threshold is reached in Interrupt Transfer mode.

8.5.1.3 Interrupt Enable Register (IER), PPP Mode

Register Name: IER
Register Description: Interrupt Enable
Default Value: x'00
Access: Byte Read/Write

Intel Hex Address: x'12
Motorola Hex Address: x'11

Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Mdm	0	0	0	RxD	TIMER	TxMpty	TxD

Bit 7 Modem pin change detect
Master interrupt enable for modem change detect functions. The host can select which modem pins are watched for input change and select either or both directions of change by programming the change detect option bits in COR4 and COR5. A Group1 type interrupt (see LIVR description) is generated from this enable.

Bits 6:4 Reserved – *must be ‘0’*.

Bit 3 Rx data
The receive FIFO threshold has been reached in Interrupt Transfer mode, causing a Group 3 receive data interrupt. Any receive exception causes a Group 3 receive exception interrupt.

Bit 2 Timer
General timer(s) timeout
In Synchronous mode, this bit enables a Group 1 interrupt when either timer reaches zero.

Bit 1 Tx Mpty
Transmitter empty. If enabled, a Group 2 interrupt is generated when the channel is completely empty of transmit data.

Bit 0 Tx Data
Any transmit exception or transmit FIFO threshold reached in Interrupt Transfer mode. Group 2 interrupts are generated at the end of transmit DMA buffers or when the FIFO threshold is reached in Interrupt Transfer mode.

8.5.1.4 Local Interrupting Channel Register (LICR)

Register Name: LICR
Register Description: Local Interrupting Channel
Default Value: C1:C0 contain channel number
Access: Byte Read/Write

Intel Hex Address: x’25
Motorola Hex Address: x’26

Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
X	X	X	X	X	C0	X	X

These per-channel registers are initialized with each channel number. The locations are RAM registers and can be used for any purpose.

Bits 7:4 User-defined

Bits 3:2 Defines the interrupting channel number

C0	Channel Number
0	Channel 0
1	Channel 1

Bits 1:0 User-defined

8.5.1.5 Interrupt Stack Register (STK)

Register Name: STK Register Description: Interrupt Stack Default Value: x’00 Access: Byte Read only							Intel Hex Address: x’E0 Motorola Hex Address: x’E2
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
CLvl [1]	MLvl [1]	TLvl [1]	0	0	TLvl [0]	MLvl [0]	CLvl [0]

This register is a 4-bit-deep by 2-bit-wide stack that contains the internal interrupt nesting history. The stack is pushed from bits 7 and 0 toward the center during an interrupt acknowledge cycle, and popped from the center during a write to an end of interrupt register.

Bits 7, 0 CLvl [0:1]These bits provide the currently active interrupt level.

CLvl [1]	CLvl [0]	
0	0	No interrupt active; CAR provides the current channel number
0	1	Currently in a modem interrupt service, MIR provides the current channel number.
1	0	Currently in a transmit interrupt service, TIR provides the current channel number.
1	1	Currently in a receive interrupt service, RIR provides the
		current channel number.

Bits 6, 1 MLvl [0:1]These bits hold a previously active interrupt now nested.

Bits 5, 2 TLvl [0:1]These bits hold the oldest interrupt now nested two bits deep.

Bits 4:3 Reserved – always returns ‘0’ when read.

8.5.2 Receive Interrupt Registers

8.5.2.1 Receive Priority Interrupt Level Register (RPILR)

Register Name: RPILR Register Description: Receive Priority Interrupt Match Default Value: x’00 Access: Byte Read/Write							Intel Hex Address: x’E3 Motorola Hex Address: x’E1
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
User-assigned priority match value							

This register must be initialized by the host to contain the codes that are presented on the address bus by the host system to indicate which of the three CD2231 interrupt types (modem, transmit, or receive) is being acknowledged when IACKIN* is asserted. The CD2231 compares bits 0–6 in this register with A[0–6] to determine if the acknowledge level is correct. The value programmed in the MSB of the register has no effect on the IACK cycle.

RPILR must contain the code used to acknowledge receive interrupts.

Note: Bit 7 of the register is always read back as ‘0’. When each of the three Priority Interrupt Level registers is programmed with the same value, they are internally prioritized, with receive as the highest priority, followed by transmit and modem.

8.5.2.2 Receive Interrupt Register (RIR)

Register Name: RIR
Register Description: Receive Interrupt
Default Value: x’00
Access: Byte Read only

Intel Hex Address: x’EF
Motorola Hex Address: x’ED

Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Ren	Ract	Reoi	0	Rvct [1]	Rvct [0]	0	Rcn [0]

Bit 7 Ren
Receive enable is set by the CD2231 to initiate a receive interrupt request sequence. It is cleared during a valid receive interrupt acknowledge cycle.

Bit 6 Ract
Receive active is set automatically when Ren is set, and the Fair Share logic allows the assertion of a receive interrupt request. It is cleared when the host CPU writes to the Receive End of Interrupt register.

Bit 5 Reoi
Receive end of interrupt is set automatically when the host CPU writes to the Receive End of Interrupt register while in a receive interrupt routine.

Ren	Ract	Reoi	Sequence of Events
0	0	0	Idle
1	0	0	Receive interrupt requested, but not asserted
1	1	0	Receive interrupt asserted
0	1	0	Receive interrupt acknowledged
0	0	1	Receive interrupt service routine completed

Bit 4 Reserved – always returns ‘0’ when read.

Bits 3:2 Rvct [1:0]
Receive vector bits are set by the CD2231 to provide the lower two bits of the vector supplied to the host CPU during an interrupt acknowledge cycle. Receive good data vector is decoded as follows: Rvct [1] = 1, and Rvct [0] = 1. Receive exception vector is decoded as follows: Rvct [1] = 0, and Rvct [0] = 0.

Bit 1 Reserved – always returns ‘0’ when read.

Bit 0 Rcn [0]
Receive channel number is set by the CD2231 to indicate the channel requiring receive interrupt service.

8.5.2.3 Receive Interrupt Status Register (RISR

Register Name: RISR Register Description: Receive Interrupt Status Default Value: x'00 Access: Word Read Only							Intel Hex Address: x'8A Motorola Hex Address: x'88
Bit 15	*Bit 14*	*Bit 13*	*Bit 12*	*Bit 11*	*Bit 10*	*Bit 9*	*Bit 8*
RISR High							
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
RISR Low							

This register reports the status of the channel during the receive interrupt service. It is a 16-bit register, with the lower byte displaying current receive character-oriented status while the upper byte displays current DMA interrupt status. The upper byte is not used if DMA mode is not active.

RISRl — HDLC Mode

Register Name: RISRl Register Description: Receive Interrupt Status — Low Default Value: x'00 Access: Byte Read only							Intel Hex Address: x'8A Motorola Hex Address: x'89
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
0	EOF	RxAbt	CRC	OE	ResInd	0	ClrDct

If RxData in IER is set, these interrupts are enabled.

Bit 7 Reserved, always returns ‘0’ when read.

Bit 6 Receiving a data frame is essentially complete.

Bit 5 Received abort sequence terminating the frame.

Bit 4 CRC error on current frame.

Bit 3 Overrun error – indicates that new data has arrived, but the CD2231 FIFO or holding registers are full. The new data is lost, and the overrun indication is flagged on the last character received before the overrun occurred. In HDLC and Bisync modes, the remainder of a frame, following an overrun, is discarded.

Bit 2 Residual indication – indicates that the last character of the frame was a partial character.

Bit 1 Reserved, always returns ‘0’ when read.

Bit 0 Clear detect – indicates an X.21 data transfer phase clear signal has been detected. This is defined as two consecutive all-zero receive characters with the CTS* pin high. Clear Detect mode is enabled by COR1.

During an interrupt service routine, the host can use this register to provide a timer value as detailed in the Receive End of Interrupt register. The host can load only one of the two timers in the interrupt service routine.

RISRl — Asynchronous Mode

Register Name: RISRl Register Description: Receive Interrupt Status — Low Default Value: x'00 Access: Byte Read only							Intel Hex Address: x'8A Motorola Hex Address: x'89
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Timeout	SCdet2	SCdet1	SCdet0	OE	PE	FE	Break

If RxData in IER is set, these interrupts are enabled.

Bit 7 Timeout – indicates that the receive FIFO is empty, and no data has been received within the receive timeout period. There is no data character associated with this status, and no other status bits are valid if the Time-Out bit is set.

Bits 6:4 Special character detect

SCdet[2:0]	Status
000	None detected
001	Special Character 1 matched
010	Special Character 2 matched
011	Special Character 3 matched (only if ESCDE is enabled in COR3)
100	Special Character 4 matched (only if ESCDE is enabled in COR3)
111	Character is within the inclusive range of the characters in the Special Character Range low and high registers (only if RngDE is enabled in COR3). Special character match can be enabled for error characters by COR7.

Bit 3 Overrun error – indicates that new data has arrived, but the CD2231 FIFO or holding registers are full. The new data is lost and the overrun indication is flagged on the last character received before the overrun occurred.

Bit 2 Parity error – indicates that a parity error has occurred.

Bit 1 Framing error – indicates that a bad Stop bit is detected.

Bit 0 Break – indicates that a break is detected.

RISRl — Async-HDLC / PPP / MNP®4 Mode

Register Name: RISRl Register Description: Receive Interrupt Status — Low Default Value: x'00 Access: Byte Read only							Intel Hex Address: x'8A Motorola Hex Address: x'89
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
0	EOF	RxAbt	CRC	OE	FE	0	Break

If RxData in IER is set, these interrupts are enabled.

Bit 7 Reserved – always returns '0' when read.

Bit 6	EOF – End of frame The EOF bit indicates that a valid end of frame (7E) character has been received, and the 7E was not preceded by a 7D.
Bit 5	RxAbt – Receive abort The rxabt bit indicates that an abort sequence (7D–7E) has been received.
Bit 4	CRC – Receive CRC error (The terms CRC and FCS are used interchangeably in this document.) The CRC bit indicates that a frame with a valid end of frame has been received, but the FCS was not correct. CRC is set only if EOF is set.
Bit 3	OE – Overrun error The OE bit indicates that the receiver buffer and FIFO have been overrun. At least one new character has been received, but lost since there was no room available in the receiver buffer and/or FIFO.
Bit 2	FE – Framing error The FE bit indicates that a character has been received with an incorrect Stop bit. The stop bit was ‘0’; it should have been ‘1’.
Bit 1	Reserved – always returns ‘0’ when read.
Bit 0	Break – Break detection The Break bit indicates that a break has been received. A break is a continuous sequence of at least ten ‘0’ bits.

Note: 0E, FE, and break are cumulative over the entire packet in PPP mode. This means that the respective error occurred somewhere in the packet, but did not cause an immediate interrupt.

The following table defines the encoding of RxAbt and FE for an aborted receive frame:

RxAbt	FE	Error
0	0	None
0	1	Not used
1	0	Received abort sequence: x’7D, x’7E
1	1	Framing error caused a frame abort

SLIP Mode

Register Name: RISRl Register Description: Receive Interrupt Status — Low Default Value: x’00 Access: Byte Read only							Intel Hex Address: x’8A Motorola Hex Address: x’89
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
0	EOF	RxAbt	0	OE	FE	0	Break

If RxData in IER is set, these interrupts are enabled.

Bit 7	Reserved – always returns ‘0’ when read.
Bit 6	EOF – End of frame The EOF bit indicates that a valid end of frame (7E) character has been received, and the 7E was not preceded by a 7D.

Bit 5 RxAbt – Receive abort
The rxabt bit indicates that an abort sequence (7D–7E) has been received.

Bit 4 Unused; returns ‘0’ when read.

Bit 3 OE – Overrun error
The OE bit indicates that the receiver buffer and FIFO have been overrun. At least one new character has been received, but lost since there was no room available in the receiver buffer and/or FIFO.

Bit 2 FE – Framing error
The FE bit indicates that a character has been received with an incorrect Stop bit. The Stop bit was ‘0’; it should have been ‘1’.

Bit 1 Reserved – always returns ‘0’ when read.

Bit 0 Break – Break detection
The Break bit indicates that a break has been received. A break is a continuous sequence of at least ten ‘0’ bits.

Note: 0E, FE, and break are cumulative over the entire packet in PPP mode. This means that the respective error occurred somewhere in the packet, but did not cause an immediate interrupt.

8.5.2.4 Receive Interrupt Status Register high (RISRh)

Register Name: RISRh — Intel Hex Address: x’8B
Register Description: Receive Interrupt Status — High — Motorola Hex Address: x’88
Default Value: x’00
Access: Byte Read only

Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Berr	EOF	EOB	0	BA/BB	0	0	0

This register is used in DMA mode only.

Bit 7 Bus error (written by CD2231)
0 = No bus error
1 = Bus error was detected on the last transfer
The actual address at which the error occurred is available in the Receive Current Buffer Address register. In response to a bus error status, the host has two possible options:
1. Retry from the next position in the buffer.
2. Terminate this buffer by setting TermBuff bit in REOIR, and move onto the next.

Bit 6 Reception of a data frame is complete (Sync DMA mode only).

Bit 5 The end of a receive buffer has been reached. Used only for DMA-supported transmission. The end of one of the host-supplied receive buffers has been reached.

Bit 4 Unused; returns ‘0’ when read.

Bit 3 Status during buffer A or buffer B data transfer
0 = Buffer A
1 = Buffer B

Bits 2:0 Unused; returns ‘0’ when read.

8.5.2.5 Receive FIFO Output Count (RFOC)

Register Name: RFOC
Register Description: Receive FIFO Output Count
Default Value: x’00
Access: Byte Read only

Intel Hex Address: x’33
Motorola Hex Address: x’30

Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
0	0	0	RxCt4	RxCt3	RxCt2	RxCt1	RxCt0

Bits 7:5 Reserved – always returns ‘0’ when read.

Bits 4:0 Receive data count
If the receive channel is interrupt driven, a non-zero value in this bit field is the number of data characters available for transfer within the current receive interrupt.

8.5.2.6 Receive Data Register (RDR)

Register Name: RDR
Register Description: Receive Data
Default Value: x’00
Access: Byte Read only

Intel Hex Address: x’F8
Motorola Hex Address: x’F8

Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
D7	D6	D5	D4	D3	D2	D1	D0

This Virtual register accesses the receive data FIFO of a channel interrupting for receive data transfer. This register address is used for all channels to transfer receive FIFO data to the host, if programmed in Interrupt Transfer mode. Data must be read as bytes, and follows the rules listed in Section 8.3 for the positioning of valid data on the bus. If the BYTESWAP pin is high, data is valid on A/D[7:0], if BYTESWAP is low, data is valid on A/D[15:8]. This is true because the RDR is on an even address.

8.5.2.7 Receive End of Interrupt Register (REOIR)

REOIR — Asynchronous and HDLC Modes

Register Name: REOIR
Register Description: Receive End of Interrupt
Default Value: x’00
Access: Byte Write Only

Intel Hex Address: x’87
Motorola Hex Address: x’84

Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
TermBuff	DiscExc	SetTm2	SetTm1	NoTrans	Gap2	Gap1	Gap0

The CD2231 interprets values written to this register at the completion of all receive interrupts.

Bit 7 Terminate current DMA buffer
If this bit is set, the current receive buffer is terminated and data transfer is switched

to the other buffer. This bit should only be set in response to an async exception interrupt. If the buffer is terminated in response to an exception character (that is, parity error) interrupt and the discard exception character bit is not set, the exception character is written at the start of the next buffer.

Before writing the terminate buffer command to REOIR, a new buffer descriptor can be written to the current buffer.

Bit 6 Discard exception character (DMA mode only)
When this bit is set in response to an async exception interrupt, the exception character is not transferred to memory.

Bit 5 Set general timer 2 in Synchronous modes
0 = do not set general timer
1 = load the value, to general timer 2, provided in RISRl.

Bit 4 Set general timer 1 in Synchronous modes
0 = do not set general timer 1
1 = load the value, to the high byte of general timer 1, provided in RISRl.

At the end of an interrupt service routine, the user can set a timer by setting a timer value in the Receive Interrupt Status register. When the timer reaches ‘0’, the CD2231 generates a modem/timer group interrupt to the host.

Bit 3 No transfer of data
This bit must be set by the host, if no data is transferred from the receive FIFO during a receive interrupt.

Bits 2:0 Gap2, Gap1, Gap0
Size of the optional gaps to be left in DMA buffer, starting at the current location, before resuming data transfer. The CD2231 moves forward its buffer address pointer to the selected number of bytes. It does not write to any location ‘in the gap’. If the gap is large enough to complete, or extend beyond the end of the current buffer, it is completed, and the gap continued in the other receive buffer. If the discard exception character is not selected, the character where the exception occurred is written to the buffer following the gap.

REOIR — Async-HDLC / PPP / SLIP / MNP$^{®}$4 Mode

Register Name: REOIR | Intel Hex Address: x’87
Register Description: Receive End of Interrupt | Motorola Hex Address: x’84
Default Value: x’00
Access: Byte Write Only

Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
TermBuff	DiscExc	SetTm2	SetTm1	NoTrans	0	0	0

The CD2231 interprets values written to this register at the completion of all receive interrupts.

Bit 7 Terminate current DMA buffer
If this bit is set, the current receive buffer is terminated and data transfer is switched to the other buffer. This bit should only be set in response to an async exception interrupt. If the buffer is terminated in response to an exception character (that is, parity error) interrupt and the discard exception character bit is not set, the exception character is written at the start of the next buffer.

Before writing the terminate buffer command to REOIR, a new buffer descriptor can be written to the current buffer.

Bit 6 Discard exception character (DMA mode only)
When this bit is set in response to an async exception interrupt, the exception character is not transferred to memory.

Bit 5 Set general timer 2 in Synchronous modes
0 = do not set general timer
1 = load the value, to general timer 2, provided in RISRl.

Bit 4 Set general timer 1 in Synchronous modes
0 = do not set general timer 1
1 = load the value, to the high byte of general timer 1, provided in RISRl.

At the end of an interrupt service routine, the user can set a timer by setting a timer value in the Receive Interrupt Status register. When the timer reaches '0', the CD2231 generates a modem/timer group interrupt to the host.

Bit 3 No transfer of data
This bit must be set by the host, if no data is transferred from the receive FIFO during a receive interrupt.

Bits 2:0 Not used; must be zero.

8.5.3 Transmit Interrupt Registers

8.5.3.1 Transmit Priority Interrupt Level Register (TPILR)

Register Name: TPILR Register Description: Transmit Priority Interrupt Match Default Value: x'00 Access: Byte Read/Write							Intel Hex Address: x'E2 Motorola Hex Address: x'E0
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
User-assigned priority match value							

This register must be initialized by the host to contain the codes that are presented on the address bus by the host system to indicate which of the three CD2231 interrupt types (modem, transmit, or receive) is being acknowledged when IACKIN* is asserted. The CD2231 compares bits 0–6 in this register with A[0–6] to determine if the acknowledge level is correct. The value programmed in the MSB of the register has no effect on the IACK cycle.

The TPILR must contain the code used to acknowledge transmit interrupts.

Note: Bit 7 of this register is always read back as '0'. When each of the three Priority Interrupt Level registers are programmed with the same value, they are internally prioritized, with receive as the highest priority, followed by transmit and modem.

8.5.3.2 Transmit Interrupt Register (TIR)

Register Name: TIR Register Description: Transmit Interrupt Default Value: None, value varies Access: Byte Read only							Intel Hex Address: x'EE Motorola Hex Address: x'EC
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Ten	Tact	Teoi	0	Tvct [1]	Tvct [0]	0	Tcn [0]

Bit 7 Ten
Transmit enable is set by the CD2231 to initiate a transmit interrupt request sequence. It is cleared during a valid transmit interrupt acknowledge cycle.

Bit 6 Tact
Transmit active is set automatically when Ten is set, and the Fair Share logic allows the assertion of a transmit interrupt request. It is cleared when the host CPU writes to the Transmit End of Interrupt register.

Bit 5 Teoi
Transmit end of interrupt is set automatically when the host CPU writes to the Transmit End of Interrupt register while in a transmit interrupt routine.

Ten	Tact	Teoi	Sequence of Events
0	0	0	Idle
1	0	0	Transmit interrupt requested, but not asserted
1	1	0	Transmit interrupt asserted
0	1	0	Transmit interrupt acknowledged
0	0	1	Transmit interrupt service routine completed

Bit 4 Reserved – always returns ‘0’ when read.

Bits 3:2 Tvct [1:0]
Transmit Vector bits are set by the CD2231 to provide the lower two bits of the vector supplied to the host CPU during an interrupt acknowledge cycle. Transmit vector is decoded as follows: Tvct [1] = 1, and Tvct [0] = 0.

Bit 1 Reserved – always returns ‘0’ when read.

Bit 0 Tcn [0]
Transmit channel number is set by the CD2231 to indicate the channel requiring transmit interrupt service.

8.5.3.3 Transmit Interrupt Status Register (TISR)

Register Name: TISR Register Description: Transmit Interrupt Status Default Value: x'00 Access: Byte Read only							Intel Hex Address: x'89 Motorola Hex Address: x'8A
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Berr	EOF	EOB	UE	BA/BB	0	TxEmpty	TxDat

When the host receives a transmit interrupt, the following status is provided in this register:

Bit 7 Berr – Bus error (written by the CD2231)
0 = no bus error
1 = bus error was detected on the last transfer

Bit 6 EOF – Transmit end of frame indication in the DMA mode
This interrupt occurs when the final data character of a transmit frame is transferred to the transmit FIFO.

Bit 5 EOB – Transmit end of buffer indication in the DMA mode

Bit 4 Transmit underrun error (HDLC only), otherwise zero (Async, PPP, SLIP, and MNP4).

Bit 3 BA/BB – Applicable buffer for the register interrupt
0 = Transmit Buffer A
1 = Transmit Buffer B

Bit 2 Reserved – always returns '0' when read.

Bit 1 TxEmpty – Transmitter empty
All characters have been completely transmitted, and the serial output is idle.

Bit 0 TxDat – The number of characters in the FIFO is below the threshold.

8.5.3.4 Transmit FIFO Transfer Count (TFTC)

Register Name: TFTC Register Description: Transmit FIFO Transfer Count Default Value: x'00 Access: Byte Read only							Intel Hex Address: x'83 Motorola Hex Address: x'80
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
0	0	0	TxCt4	TxCt3	TxCt2	TxCt1	TxCt0

Bits 7:5 Reserved – always returns '0' when read.

Bits 4:0 Transmit data count
If the Transmit channel is interrupt driven, a non-zero value is a request for data. These bits give the number of spaces available in the transmit FIFO.

8.5.3.5 Transmit Data Register (TDR)

Register Name: TDR Register Description: Transmit Data Default Value: x'00 Access: Byte Write Only							Intel Hex Address: x'F8 Motorola Hex Address: x'F8
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
D7	D6	D5	D4	D3	D2	D1	D0

This register accesses the transmit data FIFO of a channel, interrupting for transmit data transfer. This register address is used for all channels to transfer transmit FIFO data to the host, if programmed in Interrupt Transfer mode. Data must be written as bytes, and follows the rules listed in Section 7.4 for positioning valid data on the bus. If the BYTESWAP pin is high, data must be valid on A/D[0–7]; if BYTESWAP is low, data must be valid on A/D[8–15] because the TDR is on an even address.

8.5.3.6 Transmit End of Interrupt Register (TEOIR)

Register Name: TEOIR Register Description: Transmit End of Interrupt Default Value: x'00 Access: Byte Write only							Intel Hex Address: x'86 Motorola Hex Address: x'85
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
TermBuff	EOF	SetTm2	SetTm1	Notrans	0	0	0

The Transmit End of Interrupt register must be written to by the corresponding host interrupt service routine to signal to the CD2231 that the current interrupt service is concluded. This must be the last access to the CD2231 during an interrupt service routine. Writing to this register generates an internal end of interrupt signal which pops the CD2231 interrupt context stack.

Depending on the circumstances of an individual interrupt service, the host can be required to pass a parameter to the CD2231 through these registers.

Bit 7 1 = Terminate buffer in DMA mode forces the current buffer to be discarded.

Note: If current interrupt is a transmit end-of-buffer interrupt, setting this bit at the end of the service routine causes the next buffer to be terminated also.

Bit 6 End of frame in Synchronous modes using interrupt-driven data transfer
0 = this data transfer does not complete the frame/block.
1 = this data transfer does complete the frame/block.

Bit 5 Set General Timer 2 in Synchronous modes
0 = do not set General Timer 2.
1 = load the value, provided in TISR, to General Timer 2.

Bit 4 Set General Timer 1 in Synchronous modes
0 = do not set General Timer 1.
1 = load the value, provided in TISR, to the high byte of General Timer 1.

At the end of an interrupt service routine, the user can set a timer by setting a timer value in the Transmit Interrupt Status register. When the timer reaches ‘0’, the CD2231 generates a modem/timer group interrupt to the host.

Bit 3 No transfer of data
This bit must be set by the host, if no data is transferred to the transmit FIFO during a data transfer interrupt.

Bits 2:0 Reserved – *must be ‘0’*.

8.5.4 Modem Interrupt Registers

8.5.4.1 Modem Priority Interrupt Level Register (MPILR)

Register Name: MPILR	Intel Hex Address: x’E1
Register Description: Modem Priority Interrupt Match	Motorola Hex Address: x’E3
Default Value: x’00	
Access: Byte Read/Write	

Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
User-assigned priority match value							

This register must be initialized by the host to contain the codes that are presented on the address bus by the host system to indicate which of the three CD2231 interrupt types (modem, transmit, or receive) is being acknowledged when IACKIN* is asserted. The CD2231 compares bits 0–6 in this register with A[6:0] to determine if the acknowledge level is correct. The value programmed in the MSB of this register has no effect on the IACK cycle.

The MPILR must contain the code used to acknowledge modem/timer interrupts.

Note: Bit 7 of this register is always read back as ‘0’. When each of the three Priority Interrupt Level registers is programmed with the same value, they are internally prioritized, with receive as the highest priority, followed by transmit and modem.

8.5.4.2 Modem Interrupt Register (MIR)

Register Name: MIR	Intel Hex Address: x’ED
Register Description: Modem Interrupt	Motorola Hex Address: x’EF
Default Value: x’00	
Access: Byte Read only	

Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Men	Mact	Meo	0	Mvct [1]	Mvct [0]	0	Mcn [0]

Bit 7 Mer
Modem enable is set by the CD2231 to initiate a modem interrupt request sequence. It is cleared during a valid modem interrupt acknowledge cycle.

Bit 6 Mact
Modem active is set automatically when Men is set, and the Fair Share logic allows

the assertion of a modem interrupt request. It is cleared when the host CPU writes to the Modem End of Interrupt register.

Bit 5 Meo
Modem end of interrupt is set automatically when the host CPU writes to the Modem End of Interrupt register while in a modem interrupt routine.

Men	Mact	Meo	Sequence of Events
0	0	0	Idle
1	0	0	Modem interrupt requested, but not asserted
1	1	0	Modem interrupt asserted
0	1	0	Modem interrupt acknowledged
0	0	1	Modem interrupt service routine completed

Bit 4 Reserved – always returns ‘0’ when read.

Bits 3:2 Mvct [1:0]
Modem Vector bits are set by the CD2231 to provide the lower two bits of the vector supplied to the host CPU during an interrupt acknowledge cycle. Modem vector is decoded as follows: Mvct [1] = 0, and Mvct [0] = 1.

Bit 1 Reserved – always returns ‘0’ when read.

Bit 0 Mcn [0]
Modem channel number is set by the CD2231 to indicate the channel requiring modem interrupt service.

8.5.4.3 Modem (/Timer) Interrupt Status Register (MISR)

Register Name: MISR
Register Description: Modem Interrupt Status
Default Value: x’00
Access: Byte Read/Write

Intel Hex Address: x’88
Motorola Hex Address: x’8B

Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
DSRChg	CDChg	CTSChg	0	0	0	Timer2	Timer1

When the host receives a modem interrupt, the following status is provided in this register:

Bit 7 DSR changed
A logic ‘1’ indicates that a change has been detected on the DSR* input. The change detect is programmed in COR4 and COR5.

Bit 6 CD changed
A logic ‘1’ indicates that a change has been detected on the CD* input. The change detect is programmed in COR4 and COR5.

Bit 5 CTS changed
A logic ‘1’ indicates that a change has been detected on the CTS* input. The change detect is programmed in COR4 and COR5.

Bits 4:2 Unused; returns ‘0’ when read

Bit 1 General Timer 2 timed-out (count reaches zero before being reset or disabled).

Bit 0 General Timer 1 timed-out (count reaches zero before being reset or disabled).

During an interrupt service routine, the host can use this register to provide a binary timer value to one of the timers (Sync modes only), as detailed in the Modem End of Interrupt register. The host can only load one of the two timers in each interrupt service routine.

8.5.4.4 Modem End of Interrupt Register (MEOIR)

Register Name: MEOIR Register Description: Modem End of Interrupt Default Value: x'00 Access: Byte Write only							Intel Hex Address: x'85 Motorola Hex Address: x'86
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
0	0	SetTm2	SetTm1	0	0	0	0

Bits 7:6 Reserved – *must be '0'*.

Bit 5 Set General Timer 2 in Synchronous modes
0 = do not set General Timer 2.
1 = load the value, provided in MISR, to General Timer 2.

Bit 4 Set General Timer 1 in Synchronous modes
0 = do not set General Timer 1.
1 = load the value, provided in MISR, to the high byte of General Timer 1.

At the end of an interrupt service routine, the user can set the timer by setting a timer value in the Modem Interrupt Status register. When the timer reaches '0', the CD2231 generates a modem/timer group interrupt to the host.

Bits 3:0 Reserved – *must be '0'*.

8.6 DMA Registers

8.6.1 DMA Mode Register (DMR)

Register Name: DMR Register Description: DMA Mode Default Value: x'00 Access: Byte Write only							Intel Hex Address: x'F4 Motorola Hex Address: x'F6
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
EnSync	0	0	0	ByteDMA	0	0	0

This register is write only. No misoperation occurs if the register is read, but the read value is not consistent.

Bit 7 Internal DTACK synchronization enable. If external synchronization of DTACK* with BUSCLK is not provided, an internal synchronization can be enabled by setting this bit (Revision D and later).

Bits 6:4 Reserved – *must be '0'*.

Bit 3 Byte DMA
0 = The CD2231 attempts to perform 16-bit data transfers whenever possible, and 8-bit data transfers only when necessary (when only one byte is available or there are odd address boundaries).
1 = The CD2231 always performs 8-bit DMA transfers, the position of the data on the bus still follows the normal rules relating to the BYTESWAP pin.

Bits 2:0 Reserved – *must be '0'*.

8.6.2 Bus Error Retry Count (BERCNT)

Register Name: BERCNT Register Description: Bus Error Retry Count Default Value: x'00 Access: Byte Read/Write							Intel Hex Address: x'8D Motorola Hex Address: x'8E
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Binary value							

When this register is programmed to zero, any bus error causes a receive/transmit interrupt to be generated and DMA operations suspended to the buffer in error, until the interrupt is processed by the host CPU.

When this register contains a non-zero value and when a bus error occurs, the CD2231 retries the same DMA operation and decrements the register value by one. When the value reaches zero, the next bus error causes an interrupt, at that time a new count can be loaded by the host CPU.

8.6.3 DMA Buffer Status (DMABSTS)

Register Name: DMABSTS Register Description: DMA Buffer Status Default Value: x'00 Access: Byte Read only							Intel Hex Address: x'1A Motorola Hex Address: x'19
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
TDAlign	RstApd	CrtBuf	Append	Ntbuf	Tbusy	Nrbuf	Rbusy

When CD2231 requires an external buffer for DMA transfer, it checks Ntbuf/Nrbuf bits to decide which buffer to use. Once the CD2231 starts using the buffer, it toggles Ntbuf/Nrbuf bits, and sets Tbusy/Rbusy bits. Ntbuf and Nrbuf bits are set to Buffer A at system initialization.

Bit 7 This status bit is used internally to manage data alignment in the transmit FIFO.

Bit 6 Reset Append mode is set after the terminate append buffer command in STCR has been recognized, and is cleared after the remaining data has been flushed from the buffer.

Bit 5 Current transmit buffer is used internally to mark the actual buffer in use.

Bit 4 Append (only Buffer A can be used as an append buffer)
Transmit append buffer usage indicator

0 = Append buffer is not in use.
1 = Append buffer is in use.

Bit 3 Ntbuf
Next transmit buffer
0 = Buffer A is the next transmit buffer.
1 = Buffer B is the next transmit buffer.

This bit is toggled when transmission starts from a buffer, that is, when data is first read from Buffer A, the bit is set to indicate that Buffer B is next.

Bit 2 Tbusy
Current transmit buffer is in use
0 = No buffer is in use.
1 = Current transmit buffer is in use.

Bit 1 Nrbuf
Next receive buffer
0 = Buffer A is the next receive buffer.
1 = Buffer B is the next receive buffer.

This bit is toggled when receive data is first written to a buffer, that is, when data is first written to Buffer A, the bit is set to indicate Buffer B is next.

Bit 0 Rbusy
Current receive buffer is in use
0 = No buffer is in use.
1 = Current receive buffer is in use.

8.6.4 DMA Receive Registers

8.6.4.1 A Receive Buffer Address Register — Lower (ARBADRL)

<table>
<tr><td colspan="6">Register Name: ARBADRL
Register Description: Receive Buffer ‘A’ 32-bit Address, lower word
Default Value: x’0000
Access: Word Read/Write</td><td colspan="2">Intel Hex Address: x’40
Motorola Hex Address: x’42</td></tr>
<tr><td>*Bit 15*</td><td>*Bit 14*</td><td>*Bit 13*</td><td>*Bit 12*</td><td>*Bit 11*</td><td>*Bit 10*</td><td>*Bit 9*</td><td>*Bit 8*</td></tr>
<tr><td colspan="8">Binary address value, 32-bit address, bits 15:8</td></tr>
<tr><td>*Bit 7*</td><td>*Bit 6*</td><td>*Bit 5*</td><td>*Bit 4*</td><td>*Bit 3*</td><td>*Bit 2*</td><td>*Bit 1*</td><td>*Bit 0*</td></tr>
<tr><td colspan="8">Binary address value, 32-bit address, bits 7:0</td></tr>
</table>

8.6.4.2 A Receive Buffer Address Register — Upper (ARBADRU)

Register Name: ARBADRU	Intel Hex Address: x’42
Register Description: Receive Buffer ‘A’ 32-bit Address, upper word	Motorola Hex Address: x’40
Default Value: x’0000	
Access: Word Read/Write	

Bit 15	*Bit 14*	*Bit 13*	*Bit 12*	*Bit 11*	*Bit 10*	*Bit 9*	*Bit 8*
Binary address value, 32-bit address, bits 23:16							
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Binary address value, 32-bit address, bits 31:24							

8.6.4.3 B Receive Buffer Address Register — Lower (BRBADRL)

Register Name: BRBADRL	Intel Hex Address: x’44
Register Description: Receive Buffer ‘B’ 32-bit Address, lower word	Motorola Hex Address: x’46
Default Value: x’0000	
Access: Word Read/Write	

Bit 15	*Bit 14*	*Bit 13*	*Bit 12*	*Bit 11*	*Bit 10*	*Bit 9*	*Bit 8*
Binary address value, 32-bit address, bits 15:8							
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Binary address value, 32-bit address, bits 7:0							

8.6.4.4 B Receive Buffer Address Register — Upper (BRBADRU)

Register Name: BRBADRU	Intel Hex Address: x’46
Register Description: Receive Buffer ‘B’ 32-bit Address, upper word	Motorola Hex Address: x’44
Default Value: x’0000	
Access: Word Read/Write	

Bit 15	*Bit 14*	*Bit 13*	*Bit 12*	*Bit 11*	*Bit 10*	*Bit 9*	*Bit 8*
Binary address value, 32-bit address, bits 23:16							
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Binary address value, 32-bit address, bits 31:24							

These registers contain the start addresses of two external buffers that are used by the CD2231 to store the next two receive data blocks. They are written to by the host and copied internally to control the data transfer to the memory.

8.6.4.5 A Buffer Receive Byte Count (ARBCNT)

Register Name: ARBCNT Register Description: Receive Buffer ‘A’ Byte Count Default Value: x’0000 Access: Word Read/Write							Intel Hex Address: x’48 Motorola Hex Address: x’4A
Bit 15	*Bit 14*	*Bit 13*	*Bit 12*	*Bit 11*	*Bit 10*	*Bit 9*	*Bit 8*
Binary Count Value, 16-bit count, bits 15:8							
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Binary Count Value, 16-bit count, bits 7:0							

8.6.4.6 B Buffer Receive Byte Count (BRBCNT)

Register Name: BRBCNT Register Description: Receive Buffer ‘B’ Byte Count Default Value: x’0000 Access: Word Read/Write							Intel Hex Address: x’4A Motorola Hex Address: x’48
Bit 15	*Bit 14*	*Bit 13*	*Bit 12*	*Bit 11*	*Bit 10*	*Bit 9*	*Bit 8*
Binary Count Value, 16-bit Count, bits 15:8							
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Binary Count Value, 16-bit count, bits 7:0							

These registers contain the number of bytes stored in the external data buffers by the CD2231. The count is updated after a block of data is moved to memory and the buffer is terminated. As initially written by the host, the register contains the number of bytes the buffer can hold.

8.6.4.7 A Receive Buffer Status (ARBSTS)

Register Name: ARBSTS Register Description: Receive Buffer ‘A’ Status Default Value: x’00 Access: Byte Read/Write							Intel Hex Address: x’4C Motorola Hex Address: x’4F
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Berr	EOF	EOB	0	0	0	0	2231own

8.6.4.8 B Receive Buffer Status (BRBSTS)

Register Name: BRBSTS Register Description: Receive Buffer 'B' Status Default Value: x'00 Access: Byte Read/Write							Intel Hex Address: x'4D Motorola Hex Address: x'4E
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Berr	EOF	EOB	0	0	0	0	2231own

These registers contain the current status of associated receive buffers and enable the buffers to be passed between the host and CD2231.

Bit 7 Bus error (set by the CD2231 and cleared by the host CPU)
0 = No bus error
1 = Bus error occurred on the last transfer; the suspect address is available in RCBADR.

Bit 6 End of frame (set by the CD2231 and cleared by the host CPU)
0 = This buffer does not terminate a frame.
1 = This buffer terminates a frame.

Bit 5 Buffer complete (set by the CD2231 and cleared by the host CPU)
0 = Buffer not complete.
1 = Buffer complete.

Bits 4:1 Reserved – *must be '0'*.

Bit 0 Ownership of the transfer buffer (set by the host CPU and cleared by the CD2231)
0 = Buffer not free to be used by CD2231.
1 = Buffer free to be used by CD2231.

When the Buffer Complete bit is set by the CD2231, the buffer is free for the host to process. (RBCNT information is updated to the number of bytes available in the buffer, and a new buffer can be allocated.)

8.6.4.9 Receive Current Buffer Address Register — Lower (RCBADRL)

Register Name: RCBADRL Register Description: Current Receive Buffer Address, lower word Default Value: x'0000 Access: Word Read Only							Intel Hex Address: x'3C Motorola Hex Address: x'3E
Bit 15	*Bit 14*	*Bit 13*	*Bit 12*	*Bit 11*	*Bit 10*	*Bit 9*	*Bit 8*
Binary address value, 32-bit address, bits 15:8							
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Binary address value, 32-bit address, bits 7:0							

8.6.4.10 Receive Current Buffer Address Register — Upper (RCBADRU)

Register Name: RCBADRU Register Description: Current Receive Buffer Address, upper word Default Value: x’0000 Access: Word Read Only							Intel Hex Address: x’3E Motorola Hex Address: x’3C
Bit 15	*Bit 14*	*Bit 13*	*Bit 12*	*Bit 11*	*Bit 10*	*Bit 9*	*Bit 8*
Binary address value, 32-bit address, bits 31:24							
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Binary address value, 32-bit address, bits 23:16							

These registers contain the address of the current DMA buffer being used for receive data, updated at the end of receive data transfers. These registers are for the private use of the CD2231 to manage DMA transfers. In Asynchronous mode, the host can read this register during a receive exception interrupt to determine how much data is in the buffer. The address is the location of the next character to be transferred to the buffer. The host needs this information to process newly arrived data in the buffer if used in Append mode, and a data timeout has occurred. The address is also needed if an exception has occurred, and a gap is to be left in the DMA (see the description of the Gap bits in Section 8.5.2.7 on page 131) for the insertion of status information by the host. For a bus error during receive data transfer, this register provides the start address of the transfer causing the bus error.

8.6.5 DMA Transmit Registers

8.6.5.1 A Transmit Buffer Address Register — Lower (ATBADRL)

Register Name: ATBADRL Register Description: Transmit Buffer ‘A’ 32-bit Address, lower word Default Value: x’0000 Access: Word Read/Write							Intel Hex Address: x’50 Motorola Hex Address: x’52
Bit 15	*Bit 14*	*Bit 13*	*Bit 12*	*Bit 11*	*Bit 10*	*Bit 9*	*Bit 8*
Binary address value, 32-bit address, bits 15:8							
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Binary address value, 32-bit address, bits 7:0							

8.6.5.2 A Transmit Buffer Address Register — Upper (ATBADRU)

Register Name: ATBADRU Register Description: Transmit Buffer ‘A’ 32-bit Address, upper word Default Value: x’0000 Access: Word Read/Write							Intel Hex Address: x’52 Motorola Hex Address: x’50
Bit 15	*Bit 14*	*Bit 13*	*Bit 12*	*Bit 11*	*Bit 10*	*Bit 9*	*Bit 8*
Binary address value, 32-bit address, bits 23:16							
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Binary address value, 32-bit address, bits 31:24							

8.6.5.3 B Transmit Buffer Address Register — Lower (BTBADRL)

Register Name: BTBADRL Register Description: Transmit Buffer ‘B’ 32-bit Address, lower word Default Value: x’0000 Access: Word Read/Write							Intel Hex Address: x’54 Motorola Hex Address: x’56
Bit 15	*Bit 14*	*Bit 13*	*Bit 12*	*Bit 11*	*Bit 10*	*Bit 9*	*Bit 8*
Binary address value, 32-bit address, bits 15:8							
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Binary address value, 32-bit address, bits 7:0							

8.6.5.4 B Transmit Buffer Address Register — Upper (BTBADRU)

Register Name: BTBADRU Register Description: Transmit Buffer ‘B’ 32-bit Address, upper word Default Value: x’0000 Access: Word Read/Write							Intel Hex Address: x’56 Motorola Hex Address: x’54
Bit 15	*Bit 14*	*Bit 13*	*Bit 12*	*Bit 11*	*Bit 10*	*Bit 9*	*Bit 8*
Binary address value, 32-bit address, bits 23:16							
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Binary address value, 32-bit address, bits 31:24							

These registers contain the start addresses of two external buffers that are used by the CD2231 to transmit the next data blocks. They are written to by the host and copied internally to control the data transfer from the memory to the CD2231 FIFO.

8.6.5.5 A Buffer Transmit Byte Count (ATBCNT)

<table>
<tr><td colspan="6">Register Name: ATBCNT
Register Description: Transmit Buffer ‘A’ Byte Count
Default Value: x’0000
Access: Word Read/Write</td><td colspan="2">Intel Hex Address: x’58
Motorola Hex Address: x’5A</td></tr>
<tr><td><i>Bit 15</i></td><td><i>Bit 14</i></td><td><i>Bit 13</i></td><td><i>Bit 12</i></td><td><i>Bit 11</i></td><td><i>Bit 10</i></td><td><i>Bit 9</i></td><td><i>Bit 8</i></td></tr>
<tr><td colspan="8">Binary count value, 16-bit count, bits 15:8</td></tr>
<tr><td><i>Bit 7</i></td><td><i>Bit 6</i></td><td><i>Bit 5</i></td><td><i>Bit 4</i></td><td><i>Bit 3</i></td><td><i>Bit 2</i></td><td><i>Bit 1</i></td><td><i>Bit 0</i></td></tr>
<tr><td colspan="8">Binary count value, 16-bit count, bits 7:0</td></tr>
</table>

8.6.5.6 B Buffer Transmit Byte Count (BTBCNT)

<table>
<tr><td colspan="6">Register Name: BTBCNT
Register Description: Transmit Buffer ‘B’ Byte Count
Default Value: x’0000
Access: Word Read/Write</td><td colspan="2">Intel Hex Address: x’5A
Motorola Hex Address: x’58</td></tr>
<tr><td><i>Bit 15</i></td><td><i>Bit 14</i></td><td><i>Bit 13</i></td><td><i>Bit 12</i></td><td><i>Bit 11</i></td><td><i>Bit 10</i></td><td><i>Bit 9</i></td><td><i>Bit 8</i></td></tr>
<tr><td colspan="8">Binary count value, 16-bit count, bits 15:8</td></tr>
<tr><td><i>Bit 7</i></td><td><i>Bit 6</i></td><td><i>Bit 5</i></td><td><i>Bit 4</i></td><td><i>Bit 3</i></td><td><i>Bit 2</i></td><td><i>Bit 1</i></td><td><i>Bit 0</i></td></tr>
<tr><td colspan="8">Binary count value, 16-bit count bits 7:0</td></tr>
</table>

These registers contain the count of the bytes in the buffers to be transmitted.

8.6.5.7 A Transmit Buffer Status (ATBSTS) — Async-HDLC/PPP Mode

<table>
<tr><td colspan="6">Register Name: ATBSTS
Register Description: Transmit Buffer ‘A’ Status
Default Value: x’00
Access: Byte Read/Write</td><td colspan="2">Intel Hex Address: x’5C
Motorola Hex Address: x’5F</td></tr>
<tr><td><i>Bit 7</i></td><td><i>Bit 6</i></td><td><i>Bit 5</i></td><td><i>Bit 4</i></td><td><i>Bit 3</i></td><td><i>Bit 2</i></td><td><i>Bit 1</i></td><td><i>Bit 0</i></td></tr>
<tr><td>Berr</td><td>EOF</td><td>EOB</td><td>0</td><td>0</td><td>map32</td><td>INTR</td><td>2231own</td></tr>
</table>

Setting the Append bit allows data to be added to the buffer after transmission has begun. In this mode, the host sets ATADR and ATCNT as normal, but when new data is appended to the buffer, the ATBCNT/BTBCNT (Transmit Buffer Count) can be updated. When the A buffer is used in Append mode, the CD2231 does not set the EOB bit. When the host has completed use of the buffer, it must issue the append complete command through STCR. The CD2231, upon transmitting the last characters from the buffer, sets EOB, thus allowing the host to allocate a new transmit buffer.

B Transmit Buffer Status (BTBSTS) — Async-HDLC/PPP Mode

Register Name: BTBSTS Register Description: Transmit Buffer 'B' Status Default Value: x'00 Access: Byte Read/Write							Intel Hex Address: x'5D Motorola Hex Address: x'5E
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Berr	EOF	EOB	0	0	map32	INTR	2231own

Bit 7	Berr – Bus error (set by the CD2231, and cleared by the host) 0 = No bus error 1 = Bus error was detected on the last transfer
Bit 6	EOF – End of frame (set and cleared by the host) 0 = This buffer is not the last in frame/block. 1 = This buffer is the last in frame/block.
Bit 5	EOB – The end of a transmit buffer (set by the CD2231, and cleared by the host). The end of a host supplied transmit buffer has been reached.
Bits 4:3	Reserved – *must be '0'*.
Bit 2	map32 – Map all transmit characters from 00–1F (set and cleared by the host) 0 = Use the normal TXACCM map. 1 = Map all characters in the range from 00–1F.
Bit 1	INTR – Interrupt 0 = No interrupt required after the buffer is transmitted. 1 = Interrupt required after the buffer is transmitted.
Bit 0	2231own – Ownership of the transmit buffer (set by the host and cleared by the CD2231) 0 = Buffer is owned by the host, and not ready for use by the CD2231. 1 = Buffer is owned by the CD2231, and is ready for use by the CD2231.

To start transmission of a buffer, the host must set the ATBADR/BTBADR (Transmit Buffer Address) and ATBCNT/BTBCNT (Transmit Buffer Count) registers, and then set the 2231own bit. If the CD2231 is to generate and send the CRC for the frame, the CRC bit in COR1 must be set. If the buffer contains the end of a frame, the EOF bit must also be set. When the buffer has been sent, the EOB bit is set by the CD2231, and 2231own is reset, allowing a new buffer to be allocated.

8.6.5.8 A Transmit Buffer Status (ATBSTS) — SLIP/MNP®4 Mode

Register Name: ATBSTS Register Description: Transmit Buffer 'A' Status Default Value: x'00 Access: Byte Read/Write							Intel Hex Address: x'5C Motorola Hex Address: x'5F
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Berr	EOF	EOB	0	0	0	INTR	2231own

8.6.5.9 B Transmit Buffer Status (BTBSTS) — SLIP/MNP®4 Mode

Register Name: BTBSTS Register Description: Transmit Buffer 'B' Status Default Value: x'00 Access: Byte Read/Write							Intel Hex Address: x'5D Motorola Hex Address: x'5E
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Berr	EOF	EOB	0	0	0	INTR	2231own

Bit 7 Berr – Bus error (set by the CD2231, and cleared by the host)
0 = No bus error.
1 = Bus error was detected on the last transfer.

Bit 6 EOF – End of Frame (set and cleared by the host)
0 = This buffer is not the last in frame/block.
1 = This buffer is the last in frame/block.

Bit 5 EOB – The end of a transmit buffer has been reached
(set by the CD2231, and cleared by the host)
The end of a host supplied transmit buffer has been reached.

Bits 4:2 Reserved – *must be '0'*.

Bit 1 INTR – Interrupt
0 = No interrupt required after the buffer is transmitted.
1 = Interrupt required after the buffer is transmitted.

Bit 0 2231own – Ownership of the transmit buffer (set by the host and cleared by the CD2231)
0 = Buffer is owned by the host, and not ready for use by the CD2231.
1 = Buffer is owned by the CD2231, and is ready for use by the CD2231.

8.6.5.10 A Transmit Buffer Status (ATBSTS) — HDLC Mode

Register Name: ATBSTS Register Description: Transmit Buffer 'A' Status Default Value: x'00 Access: Byte Read/Write							Intel Hex Address: x'5C Motorola Hex Address: x'5F
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Berr	EOF	EOB	UE	0	0	INTR	2231own

8.6.5.11 B Transmit Buffer Status (BTBSTS) — HDLC Mode

Register Name: BTBSTS Register Description: Transmit Buffer 'B' Status Default Value: x'00 Access: Byte Read/Write							Intel Hex Address: x'5D Motorola Hex Address: x'5E
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Berr	EOF	EOB	UE	0	0	INTR	2231own

This register contains the status of the associated transmit buffer, and it enables successive buffers to be passed between the host and the CD2231. Status bits within the register are defined as:

Bit 7 Bus error (set by the CD2231 and cleared by the host CPU)
0 = No bus error.
1 = Bus error occurred on the last transfer; the suspect address is available in TCBADR.

Bit 6 End of frame (set and cleared by host CPU)
0 = This buffer is not the last in frame/block.
1 = This buffer is the last in frame/block.

Bit 5 The end of a transmit buffer has been reached. This bit is used only for DMA supported transfer. The end of one of the host supplied transmit buffers has been reached. This bit is set by the CD2231 and cleared by the host CPU.

Bit 4 Underrun – Transmit underrun occurred as the buffer was not available, and it applies to this buffer (only in HDLC mode).

Bit 3 Reserved – *must be '0'*.

Bit 2 Reserved – *must be '0'*.

Bit 1 Interrupt
0 = No interrupt required after the buffer is sent.
1 = Interrupt required after the buffer is sent.

Bit 0 Ownership of the transfer buffer (set by the host CPU and cleared by the CD2231)
0 = Buffer not ready to be used by CD2231.
1 = Buffer is ready for CD2231 to transmit.

8.6.5.12 A Transmit Buffer Status (ATBSTS) — Async Mode

Register Name: ATBSTS Register Description: Transmit Buffer 'A' Status Default Value: x'00 Access: Byte Read/Write							Intel Hex Address: x'5C Motorola Hex Address: x'5F
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Berr	EOF	EOB	0	Append	0	INTR	2231own

8.6.5.13 B Transmit Buffer Status (BTBSTS) — Async Mode

Register Name: BTBSTS Register Description: Transmit Buffer 'B' Status Default Value: x'00 Access: Byte Read/Write						Intel Hex Address: x'5D Motorola Hex Address: x'5E	
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Berr	EOF	EOB	0	Append	0	INTR	2231own

This register contains the status of the associated transmit buffer, and it enables successive buffers to be passed between the host and the CD2231. Status bits within the register are defined as:

Bit 7 Bus error (set by the CD2231 and cleared by the host CPU)
0 = No bus error.
1 = Bus error occurred on the last transfer; the suspect address is available in TCBADR.

Bit 6 End of frame (set and cleared by host CPU)
0 = This buffer is not the last in frame/block.
1 = This buffer is the last in frame/block.

Bit 5 The end of a transmit buffer has been reached. This bit is used only for DMA supported transfer. The end of one of the host supplied transmit buffers has been reached. This bit is set by the CD2231 and cleared by the host CPU.

Bit 4 Reserved – *must be '0'*.

Bit 3 Append (Asynchronous mode only; set and cleared by the host CPU)
0 = No data is appended to the buffer.
1 = Data can be appended to buffer after tx started.

Bit 2 Reserved – *must be '0'*.

Bit 1 Interrupt
0 = No interrupt required after the buffer is sent.
1 = Interrupt required after the buffer is sent.

Bit 0 Ownership of the transfer buffer (set by the host CPU and cleared by the CD2231)
0 = Buffer not ready to be used by CD2231.
1 = Buffer is ready for CD2231 to transmit.

8.6.5.14 Transmit Current Buffer Address Register — Lower (TCBADRL)

Register Name: TCBADRL	Intel Hex Address: x’38
Register Description: Current Transmit Buffer Address, lower word	Motorola Hex Address: x’3A
Default Value: x’0000	
Access: Word Read Only	

Bit 15	*Bit 14*	*Bit 13*	*Bit 12*	*Bit 11*	*Bit 10*	*Bit 9*	*Bit 8*
Binary address value, 32-bit address, bits 15:8							
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Binary address value, 32-bit address, bits 7:0							

8.6.5.15 Transmit Current Buffer Address Register — Upper (TCBADRU)

Register Name: TCBADRU	Intel Hex Address: x’3A
Register Description: Current Transmit Buffer Address, upper word	Motorola Hex Address: x’38
Default Value: x’0000	
Access: Word Read Only	

Bit 15	*Bit 14*	*Bit 13*	*Bit 12*	*Bit 11*	*Bit 10*	*Bit 9*	*Bit 8*
Binary address value, 32-bit address, bits 31:24							
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Binary address value, 32-bit address, bits 23:16							

These registers contain the address of the current DMA buffer being used for transmit data; this address is updated at the end of transmit data transfers. In the case of a bus error during transmit data transfer, this register contains the start address of the transfer causing the bus error.

8.7 Timer Registers

8.7.1 Timer Period Register (TPR)

Register Name: TPR	Intel Hex Address: x’D8
Register Description: Timer Period	Motorola Hex Address: x’DA
Default Value: x’FF	
Access: Byte Read/Write	

Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Binary value							

This register provides the initialization value for the timer prescaler that is itself clocked by a prescaled clock equal to system clock ÷ 2048. The timer prescaler establishes the clock for the various on-chip timers (including RTPR, TTR, and the general timers available to the host in the Synchronous modes). The minimum value loaded in this register to maintain accuracy in the timer is 0A hex.

8.7.2 Receive Timeout Period Register (RTPR) — Async Mode Only

Register Name: RTPR Register Description: Receive Timeout Period, 16-bit Default Value: x’0000 Access: Word Read/Write							Intel Hex Address: x’26 Motorola Hex Address: x’24
Bit 15	*Bit 14*	*Bit 13*	*Bit 12*	*Bit 11*	*Bit 10*	*Bit 9*	*Bit 8*
Binary value, bits 15:8							
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Binary value, bits 7:0							

8.7.2.1 Receive Timeout Period Register low (RTPRl) — Async Mode Only

Register Name: RTPRl Register Description: Receive Timeout Period, Low Byte Default Value: x’00 Access: Byte Read/Write, ASYNC Mode Only							Intel Hex Address: x’26 Motorola Hex Address: x’25
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Binary value							

8.7.2.2 Receive Timeout Period Register high (RTPRh) — Async Mode Only

Register Name: RTPRh Register Description: Receive Timeout Period, High Byte Default Value: x’00 Access: Byte Read/Write, ASYNC Mode Only							Intel Hex Address: x’27 Motorola Hex Address: x’24
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Binary value							

Receive Timeout Period register (16 bits)

This value sets the receive data timeout period. As each character is moved to the receive FIFO or the last data is transferred from the FIFO to the host, the receive timer (an internal timer) is reloaded with the Receive Timeout Period register. The receive timer is decremented on each ‘tick’ of the prescaler counter, whose period is controlled by TPR. If the receive timer reaches zero, it causes a receive data interrupt.

8.7.3 General Timer 1 (GT1) Sync Modes Only

Register Name: GT1 | Intel Hex Address: x’28
Register Description: General Timer 1 | Motorola Hex Address: x’2A
Default Value: x’00
Access: Word Read/Write

Bit 15	*Bit 14*	*Bit 13*	*Bit 12*	*Bit 11*	*Bit 10*	*Bit 9*	*Bit 8*
Binary value, bits 15:8							
Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Binary value, bits 7:0							

8.7.3.1 General Timer 1 low (GT1l) Sync Modes Only

Register Name: GT1l | Intel Hex Address: x’28
Register Description: General Timer 1, low byte | Motorola Hex Address: x’2B
Default Value: x’00
Access: Byte Read/Write

Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Binary value							

8.7.3.2 General Timer 1 high (GT1h) Sync Modes Only

Register Name: GT1h | Intel Hex Address: x’29
Register Description: General Timer 1, high byte | Motorola Hex Address: x’2A
Default Value: x’00
Access: Byte Read/Write

Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Binary value							

8.7.4 General Timer 2 (GT2) Sync Modes Only

Register Name: GT2 | Intel Hex Address: x’2A
Register Description: General Timer 2 | Motorola Hex Address: x’29
Default Value: x’00
Access: Byte Read/Write

Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
Binary value							

This 16-bit timer can be started by the user whenever it is inactive by writing a 16-bit timeout value to the register. When non-zero, it is decremented on each prescaler clock ‘tick’. When it reaches zero, a modem/timer group interrupt is generated to the host. The timer can be disabled by the Channel Command register. In addition, during a receive or transmit interrupt, the user can reload a running timer (high byte only) by providing a reload value in the Interrupt Status register and a reload timer command in the End of Interrupt register for the interrupt being serviced. Only one general timer can be restarted this way in a single-interrupt routine.

This 8-bit timer can be started by the user whenever it is inactive by writing an 8-bit timeout value to the Timer register. When non-zero, it is decremented on each prescaler clock ‘tick’. When it reaches zero, a modem/timer group interrupt is generated to the host. The timer can be disabled by the Channel Command register if the timer’s current value is greater than one. In addition, during a receive or transmit interrupt, the user can reload a running timer by providing a reload value in the Interrupt Status register and a reload timer command in the End of Interrupt register for the interrupt being serviced. Only one general timer can be restarted this way in a single-interrupt routine.

8.7.5 Transmit Timer Register (TTR) Async Modes Only

Register Name: TTR	Intel Hex Address: x’2A
Register Description: Transmit Timer	Motorola Hex Address: x’29
Default Value: x’00	
Access: Byte Read only	

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Binary value							

This Asynchronous mode timer is managed by the CD2231 to implement embedded transmit delays when that option is used by the host (see description of Channel Option Register 2). This register should not be modified by the host under any circumstances.

9.0 Electrical Specifications

Note: Verify with your local sales office that you have the latest datasheet before finalizing a design.

9.1 Absolute Maximum Ratings

Operating ambient temperature (T_A) .... 0°C to 70°C

Storage temperature .... –65°C to 150°C

All voltages with respect to ground .... –0.5 V to V_{CC} +0.5 V (volts)

Supply voltage (V_{CC}) .... +7.0 V

Power dissipation .... 0.25 W (watt)

Note: Stresses above those listed under Absolute Maximum Ratings can cause permanent damage to the device. These are stress ratings only and functional operation of the device at these or any conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods can affect device reliability.

9.2 DC Electrical Characteristics

(@ V_{CC} = 5 V ± 5%, T_A = 0°C to 70°C)

Symbol	Parameter	MIN	MAX	Units	Test Conditions
V_{IL}	Input low voltage	–0.5	0.8	V	
V_{IH}	Input high voltage (all pins except CLK, RESET*, and BGIN*)	2.0		V_{CC}	V
V_{IH}	Input high voltage for CLK, RESET*, and BGIN*	2.7	V_{CC}	V	
V_{OL}	Output low voltage	0.4		V	I_{OL} = 2.4 mA
V_{OH}	Output high voltage	2.4		V	I_{OH} = –400 µA
I_{IL}	Input leakage current	–10	10	µA	$0 < V_{IN} < V_{CC}$
I_{LL}	Data bus tristate leakage current	–10	10	µA	$0 < V_{OUT} < V_{CC}$
I_{OC}	Open-drain output leakage	–10	10	µA	$0 < V_{OUT} < V_{CC}$
I_{CC}	Power supply current		50	mA	CLK = 35 MHz
C_{IN}	Input capacitance		10	pF	
C_{OUT}	Output capacitance		10	pF	

NOTE: The maximum CLK of 35 MHz applies only to Revision D and later devices; revisions prior to D remain specified at 33 MHz, maximum. All values in the following tables apply to the 35-MHz specification.

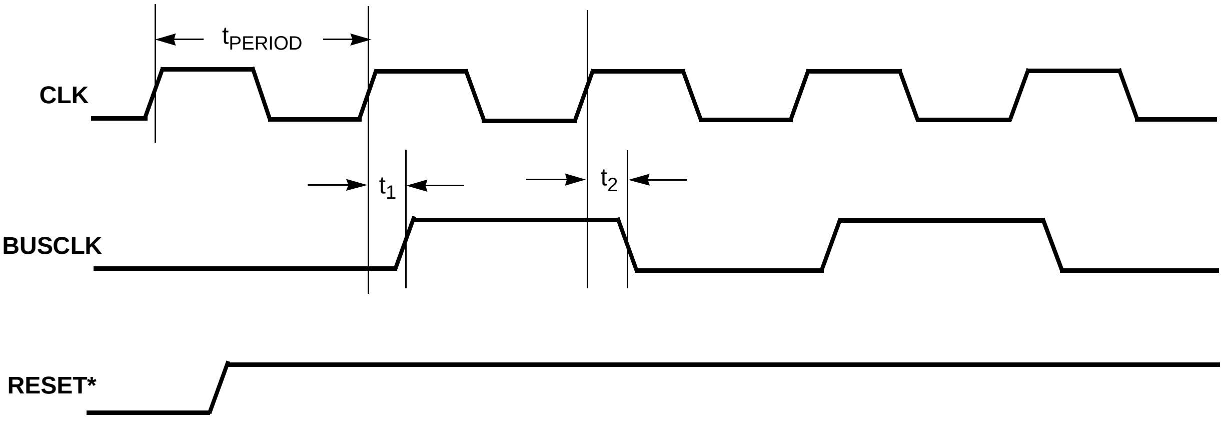
9.3 AC Electrical Characteristics

Symbol	Parameter	MIN	MAX
(Sheet 1 of 2)			
t_{PERIOD}	Period of CLK input (35 MHz maximum)	28.57	
t_1	CLK high to BUSCLK high		20
t_2	CLK high to BUSCLK low		20
Bus Arbitration			
t_{11}	CLK high to BGACK* tristate		25
t_{12}	BGIN* low to address valid[1]		40
t_{13}	Address hold after CLK high	0	
t_{14}	CLK high to address tristate		25
t_{15}	CLK high to ADLD* low		25
t_{16}	CLK high to ADLD* high		20
t_{17}	Address setup to ADLD* high	15	
t_{18}	CLK high to AEN*/DATEN*/DATDIR* high		25
t_{19}	CLK high to AEN*/DATEN*/DATDIR* tristate		25
t_{20}	CLK high to AEN*/DATEN*/DATDIR* low		25
DMA Read			
t_{21}	Data setup to CLK high	10	
t_{22}	Data hold after CLK high	15	
t_{23}	CLK high to address valid		30
t_{24}	CLK low to AS* low		25
t_{25}	CLK high to AS* high		20
t_{26}	CLK low to DS* low		25
t_{27}	CLK high to DS* high		20
t_{28}	DTACK* low setup to CLK high	10	
t_{29}	DTACK* high setup to CLK high (to avoid false termination)	50	
DMA Write			
t_{31}	CLK high to data valid		40
t_{32}	Data hold after CLK high	0	
t_{33}	CLK low to DS* low		25
t_{34}	CLK high to DS* high		20
t_{35}	DTACK* low setup to CLK high	10	
t_{36}	DTACK* high setup to CLK high (to avoid false termination)	50	
Host Read/Write			
t_{41}	DS* and CS* low setup to CLK high	7	

Symbol	Parameter	MIN	MAX
(Sheet 2 of 2)			
t_{42}	Reserved		
t_{43}	Reserved		
t_{44}	R/W* setup to CLK high	5	
t_{45}	CLK high to data valid		25
t_{46}	Data setup time to CLK high	6	
t_{47}	Data hold time after CLK high	15	
t_{48}	Address setup time to CLK high	5	
t_{49}	Address hold time after CLK high	15	
t_{50}	CLK high to DTACK* low (read cycle)		25
t_{51}	CLK high to DTACK* low (write cycle)		25
t_{52}	(CS* and DS*) low to DATEN*/DATDIR* low		28
t_{53}	DS* high to DATEN*/DATDIR* tristate		25
t_{54}	DS* high to data bus tristate		25
t_{55}	DS* high to DTACK* high-impedance		25
Interrupt Acknowledge			
t_{61}	CLK high to IACKIN*, DS* setup	20	
t_{63}	CLK high to data valid		35
t_{64}	Address setup to IACKIN* low	0	
t_{65}	Address hold after IACKIN* high	0	
t_{66}	CLK high to DTACK* low		25
t_{67}	(IACKIN* and DS*) low and BUSCLK high to DATEN* and DATDIR* low		40

1.This timing assumes the following conditions BGACK* high, DTACK* high, DS* high, and BUSCLK high.

Figure 21. CLK / BUSCLK / RESET* TIming Relationship



During RESET* active period, BUSCLK is held low. BUSCLK will transition high and begin running at one/half CLK frequency on the first rising edge of CLK after RESET* is released.

Figure 22. Slave Read Cycle Timing

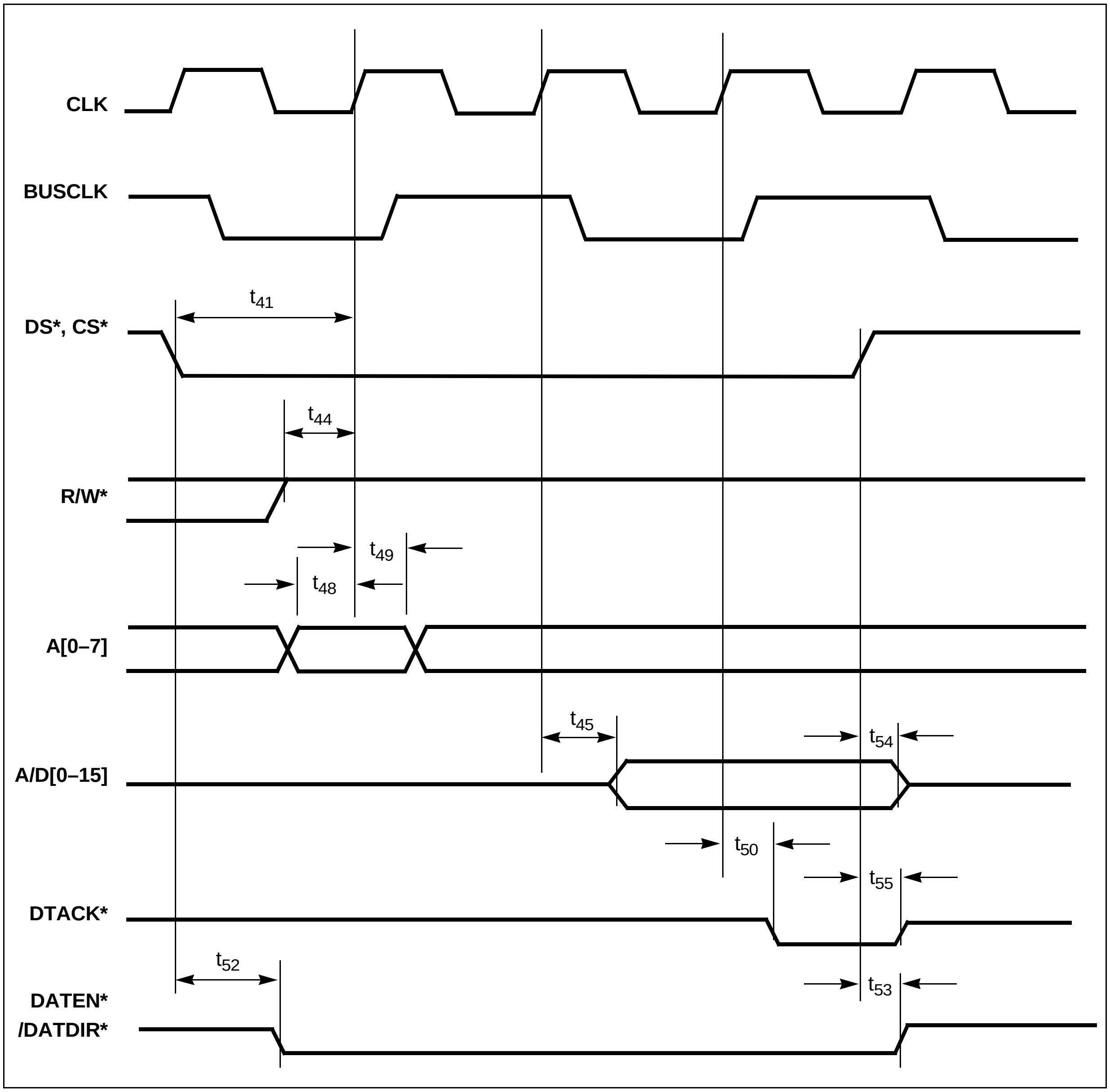


Figure 23. Slave Write Cycle Timing

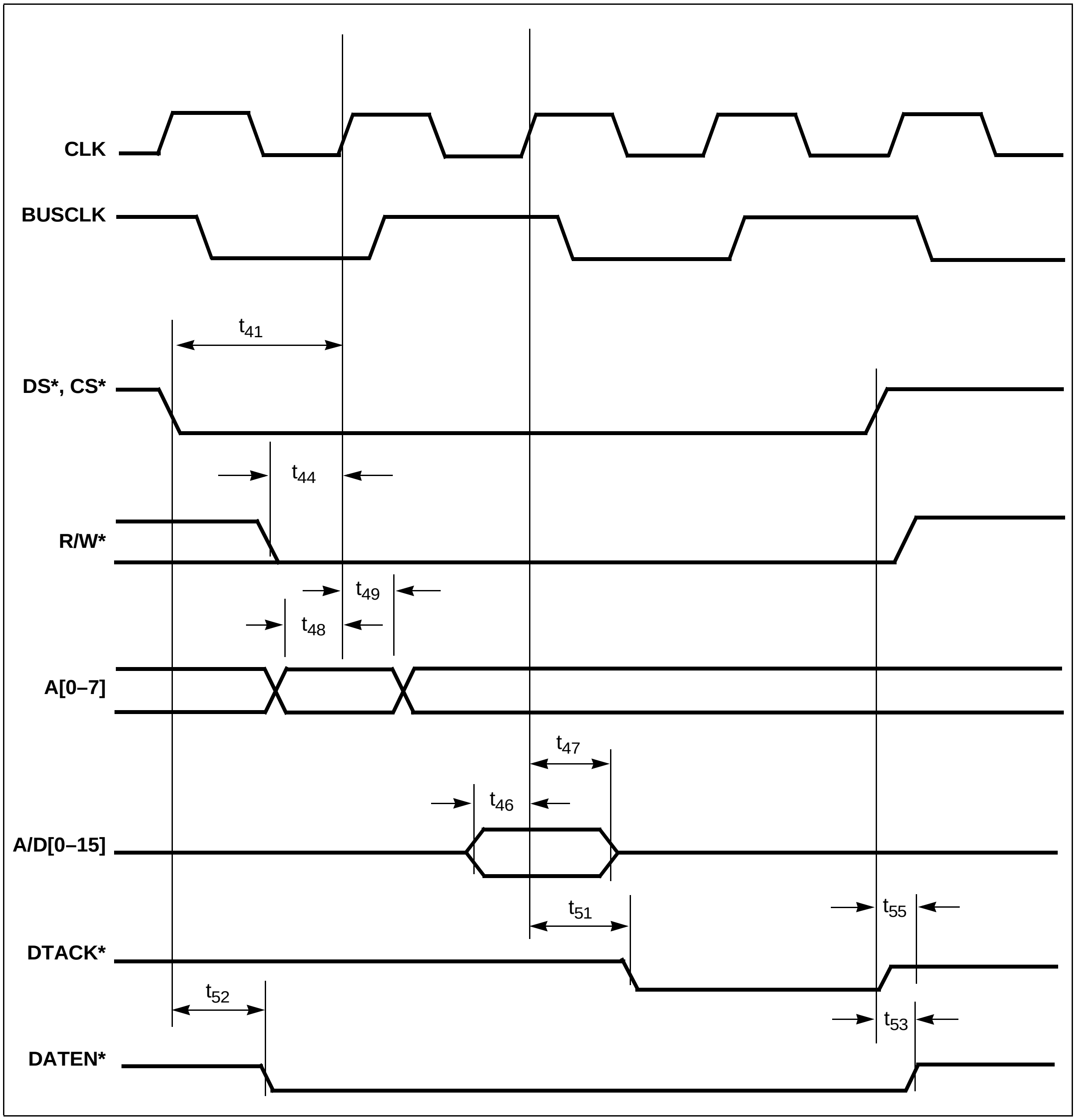


Figure 24. Interrupt Acknowledge Cycle Timing

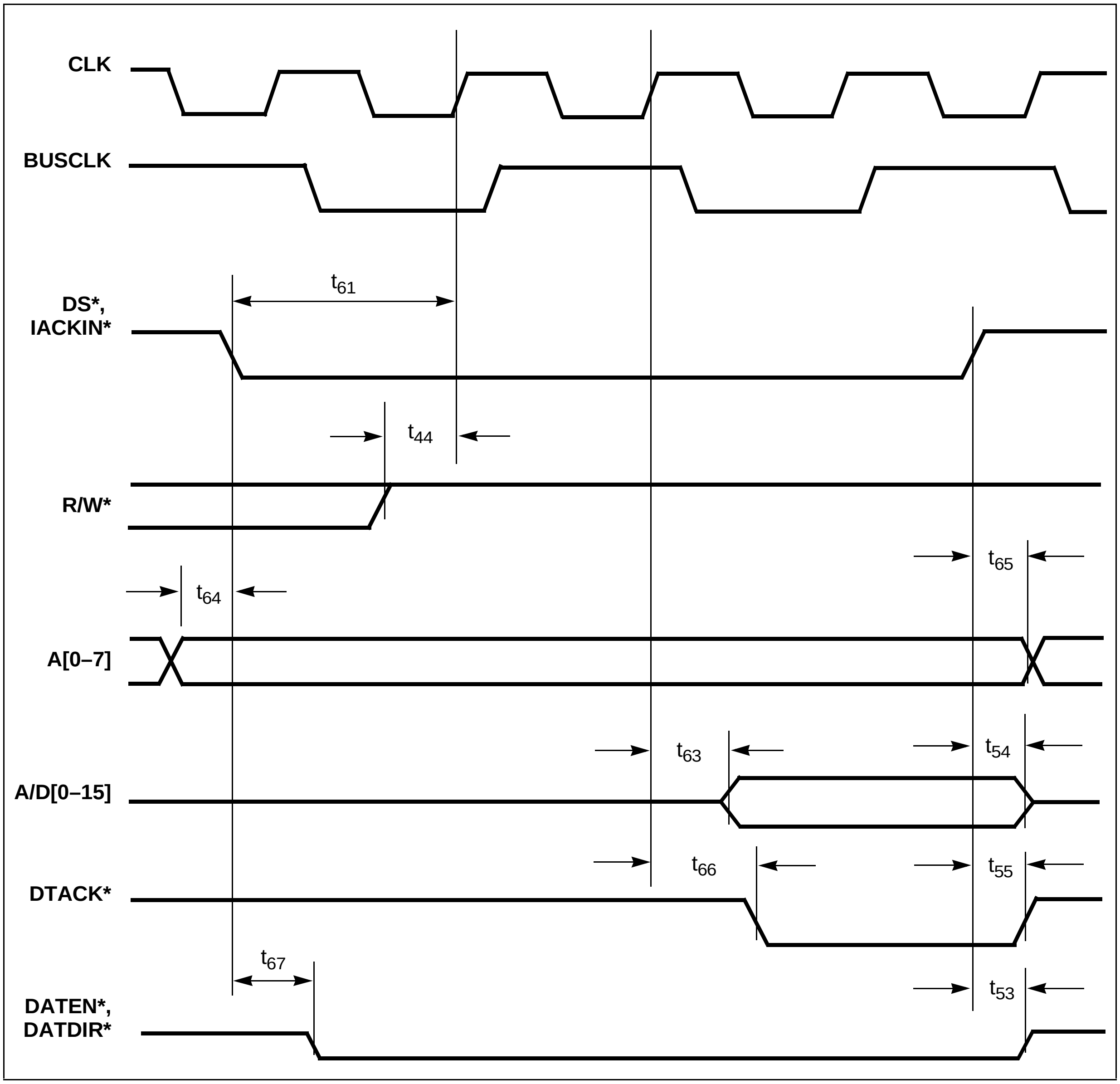
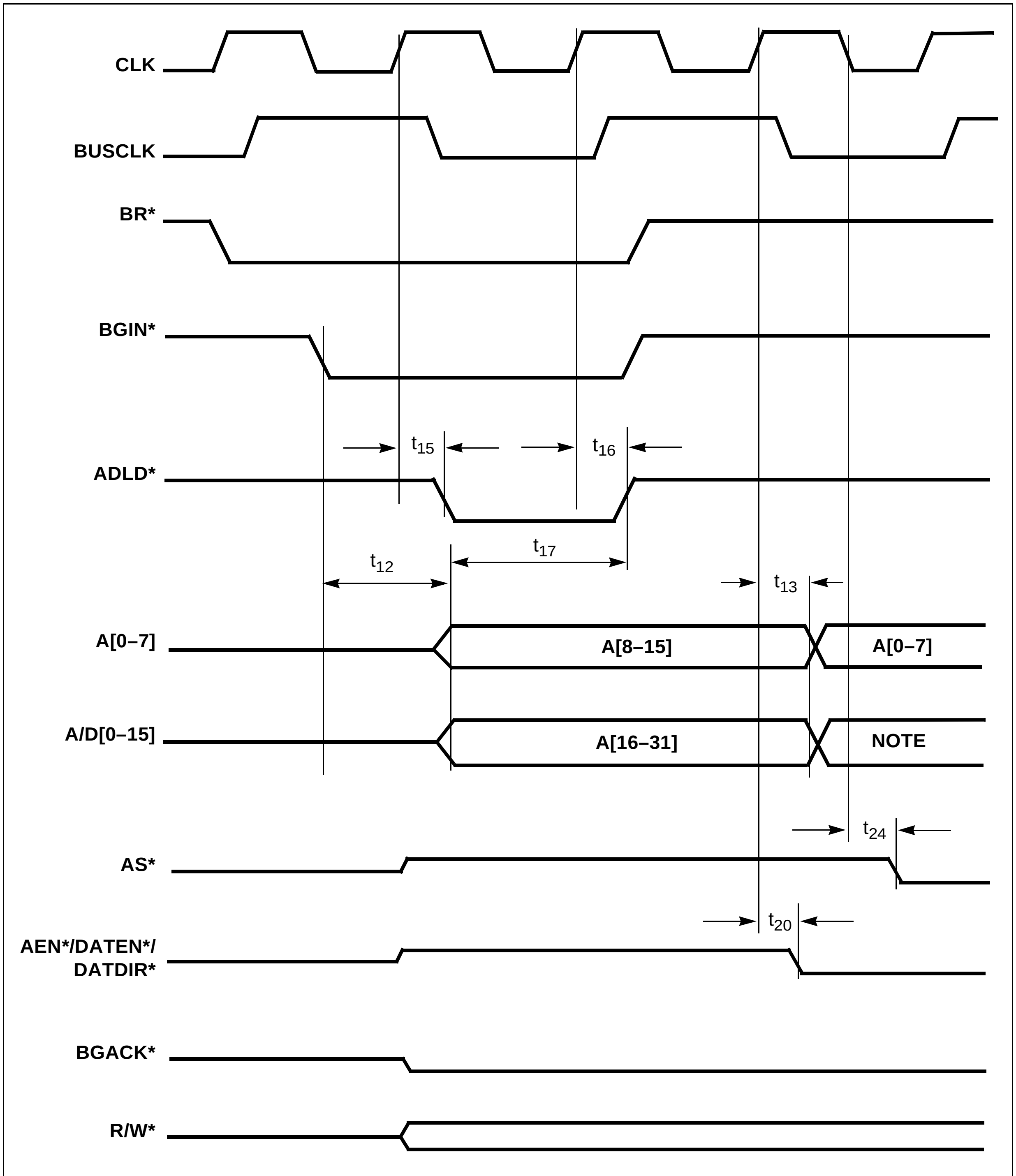


Figure 25. Bus Arbitration Cycle Timing



In DMA Read cycle, these pins will be tristate;
in DMA Write cycle, these pins will be D[0:15].

Figure 26. Bus Release Timing

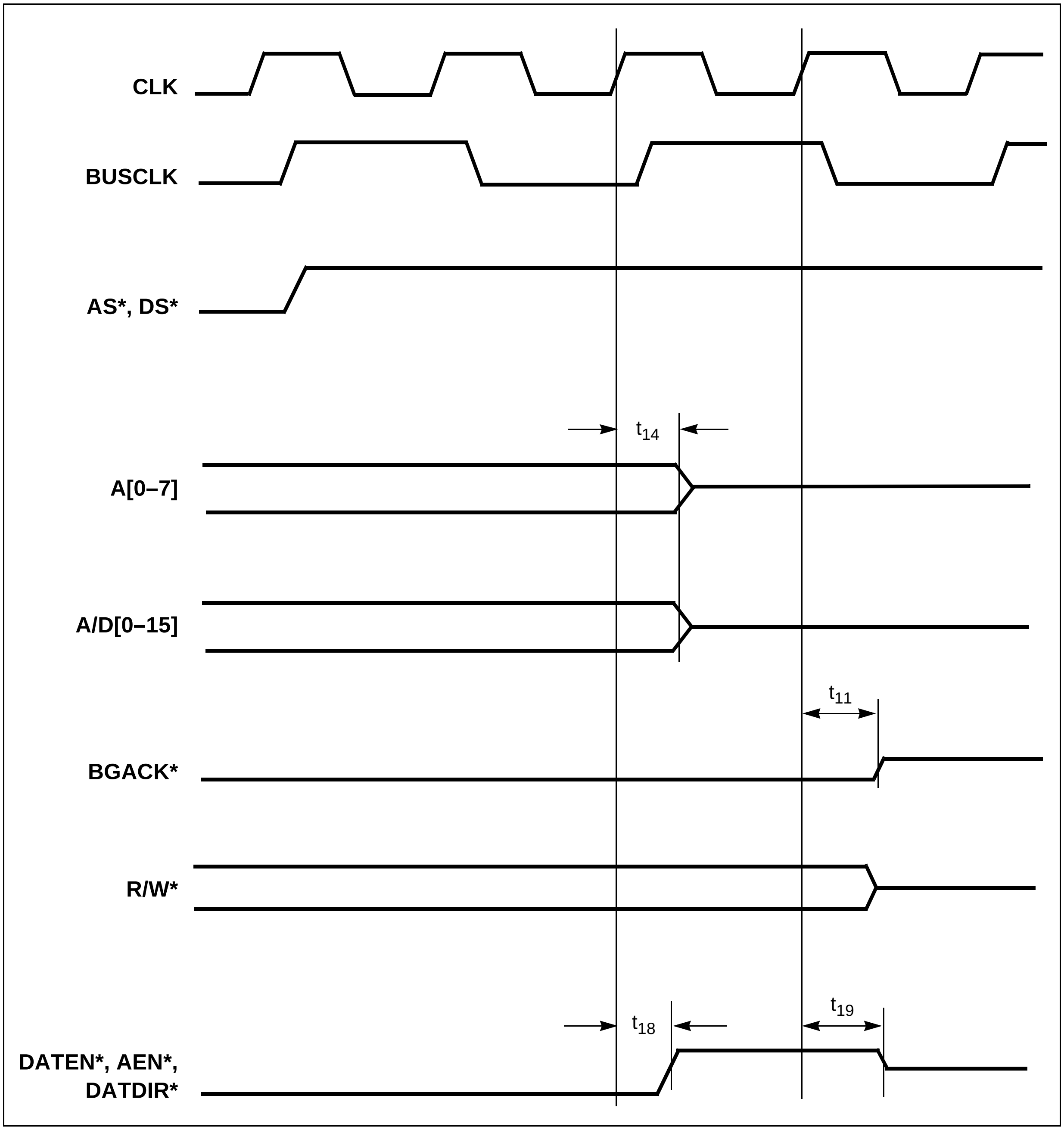
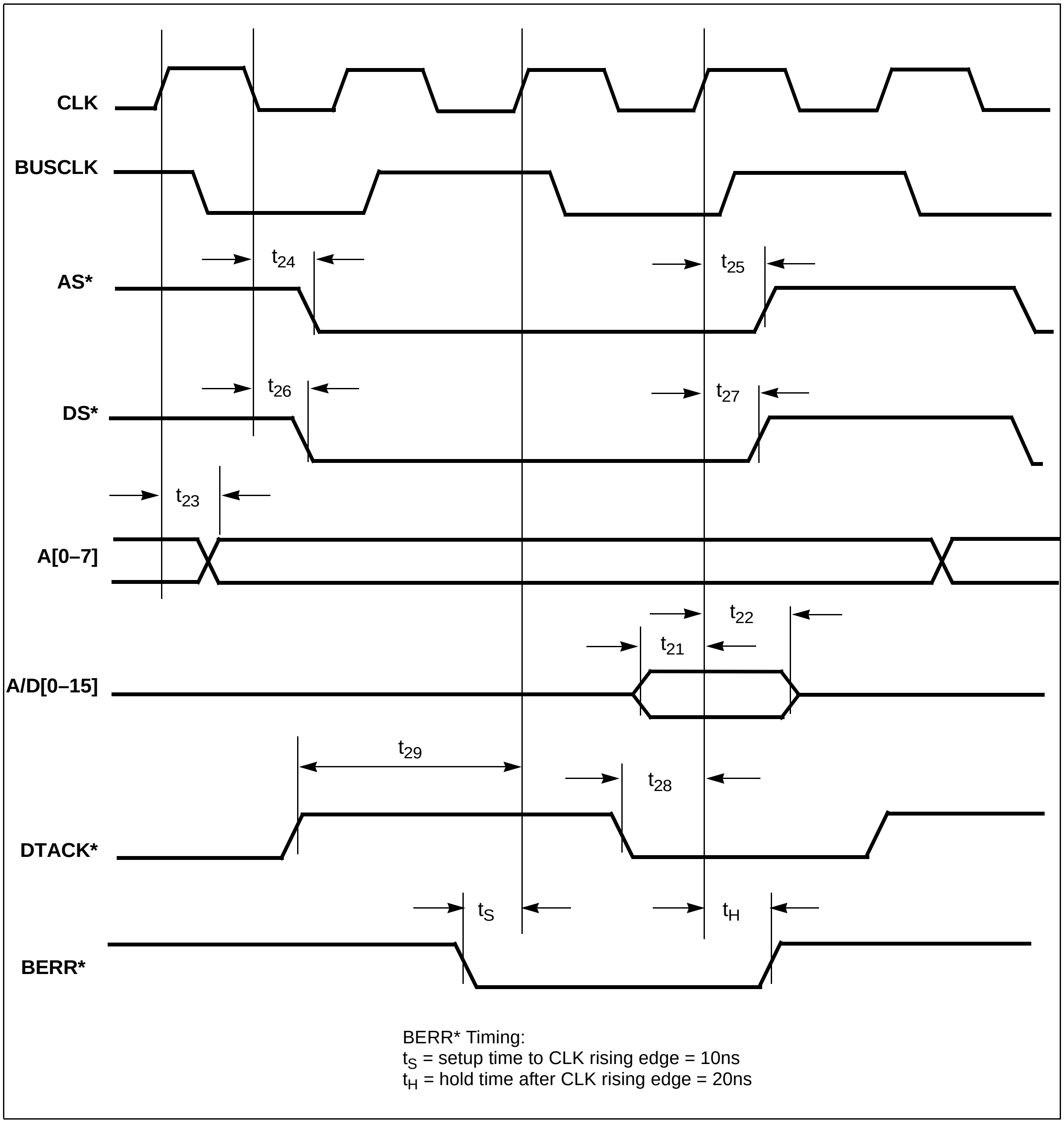
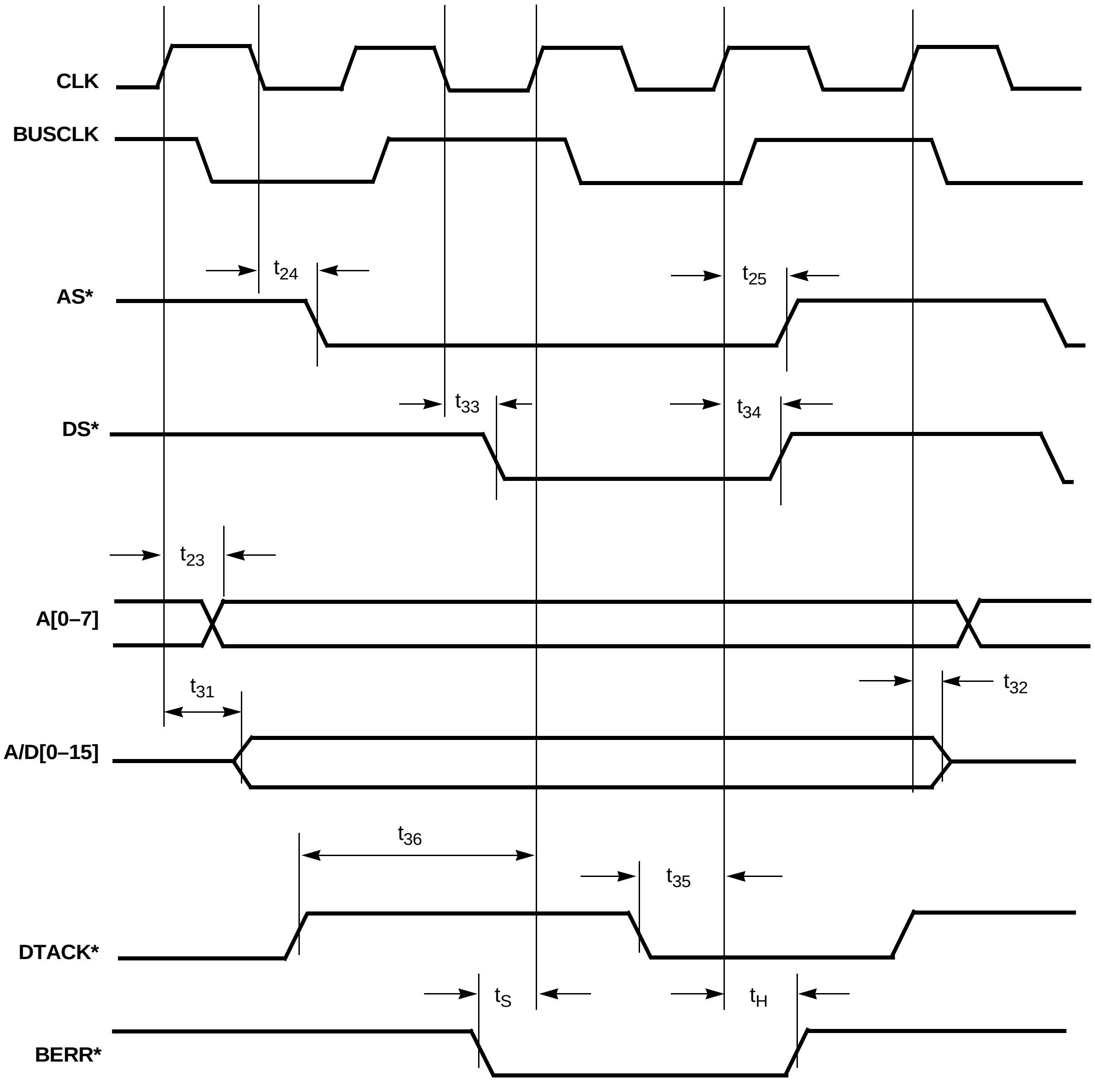


Figure 27. DMA Read Cycle Timing



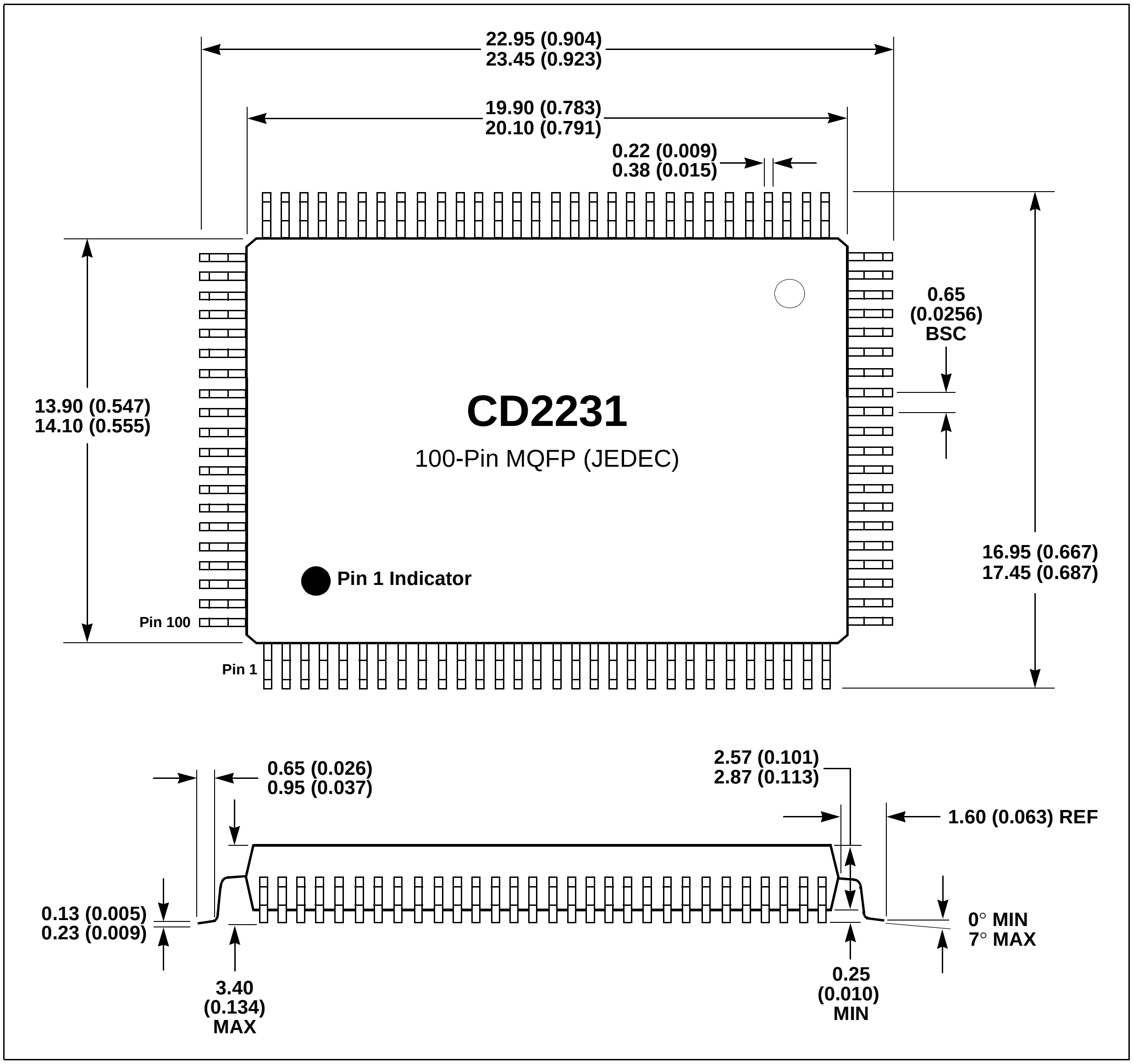
BERR* Timing:
t_S = setup time to CLK rising edge = 10ns
t_H = hold time after CLK rising edge = 20ns

Figure 28. DMA Write Cycle Timing



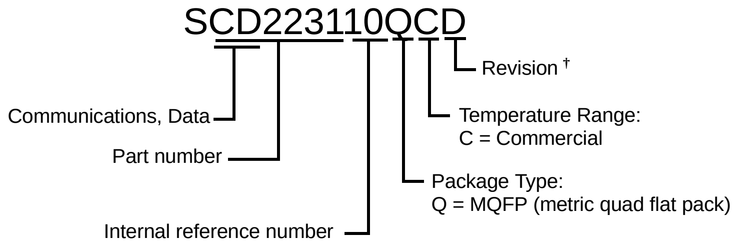
BERR* Timing:
t_S = setup time to CLK rising edge = 10ns
t_H = hold time after CLK rising edge = 20ns

10.0 Package Specifications



NOTES:
1. Dimensions are in millimeters (inches), and controlling dimension is millimeter.
2. Before beginning any new design with this device, please contact Intel for the latest package information.

11.0 Ordering Information Example



† Contact Intel Corporation for up-to-date information on revisions.

Index

Bit Index