SECTION 11
DATA LINK CONTROLLER MODULE (DLCMD2)

11.1 Scope

This document contains data for the data link controller digital module (DLCMD2). This module is based on the IMB DLCMD module, but has several feature enhancements including those required for the IMB3. The primary purpose of this document is to form the foundation for the functionality and features of the module. This module is designed in a modular structure and is fully compatible with the inter-module bus version 3.

The DLCMD2 is designed with sufficient flexibility to accommodate feature mixes such as byte or symbol-level message buffering.

11.2 Features

The DLCMD2 is essentially the digital portion of a Class B serial data link controller. A separate transceiver is required. The DLCMD2 will provide the following features:

- SAE J1850 compatible
- GM class 2 compatible
- 10.4 Kbytes/s VPW bit format
- Handles all network protocol functions (access, arbitration, error detection)
- Parallel 16-bit accesses
- All registers are individually addressable
- Polling and IMB3 interrupt generation with vector lookup available
- Transmit buffer first byte can be loaded without a command byte
- Message buffering on transmit and receive
- 8-bit hardware CRC generation and checking
- No on-board oscillator (uses system clock)
- DLCMD2 logic is clocked from IMB3
- No analog (transceiver) circuit
- Interface to the external transceiver
- Transmit and receive block mode supported
- Transmit and receive 4X mode supported
- Two extra 1-bits sent if lose arbitration on a byte boundary
- Transmitter underrunning indication added to status register
- IMB3 full feature support with option plug selection
- Software programmable prescaler to support two 128-MHz system clock range
- Programmable receiver input polarity
- Programmable normalization bit format
- Digitally filtered receiver
- Power conserving sleep mode with wakeup from bus activity and no loss of data
- IFR type 1, 2 and 3 supported

- Auto retry for loss of arbitration and errors
- Symbol timing control and pre-scaler register
- Symbol timing data register (SDATA)
- Write access to symbol timing parameter table through SDATA

11.3 Background

The DLCMD2 is an evolution of an earlier module found on IMB MCUs.The analog function, or transceiver, necessary to interface to the J1850 bus will be applied external to the module, and will not be built into the DLCMD2.

11.4 Applicable Documents

- SAE J1850 Class B Data Communications Network Interface

11.5 General Requirements

This module is an integrated module for inclusion on-board an IMB3 MCU. Refer to **Table E-20**.

11.6 Logic Description

The data link controller module (DLCMD2) provides access to an external serial communication multiplex bus, operating according to the SAE J1850 protocol.

This section describes the features, functions, and operation, of the DLCMD2 used as one of several nodes in a vehicle multiplex and/or diagnostic wiring network. All control, status, and message bytes (head of FIFOs only) are accessible as memory mapped registers within the MCU.

The DLCMD2 module supports the SAE J1850 protocol. It retains the maximum throughput performance for single-chip applications including full J1850 message-level buffering but does not provide an internal transceiver.

11.6.1 Block Diagram

A block diagram of the DLCMD2 is shown in **Figure 11-1**.

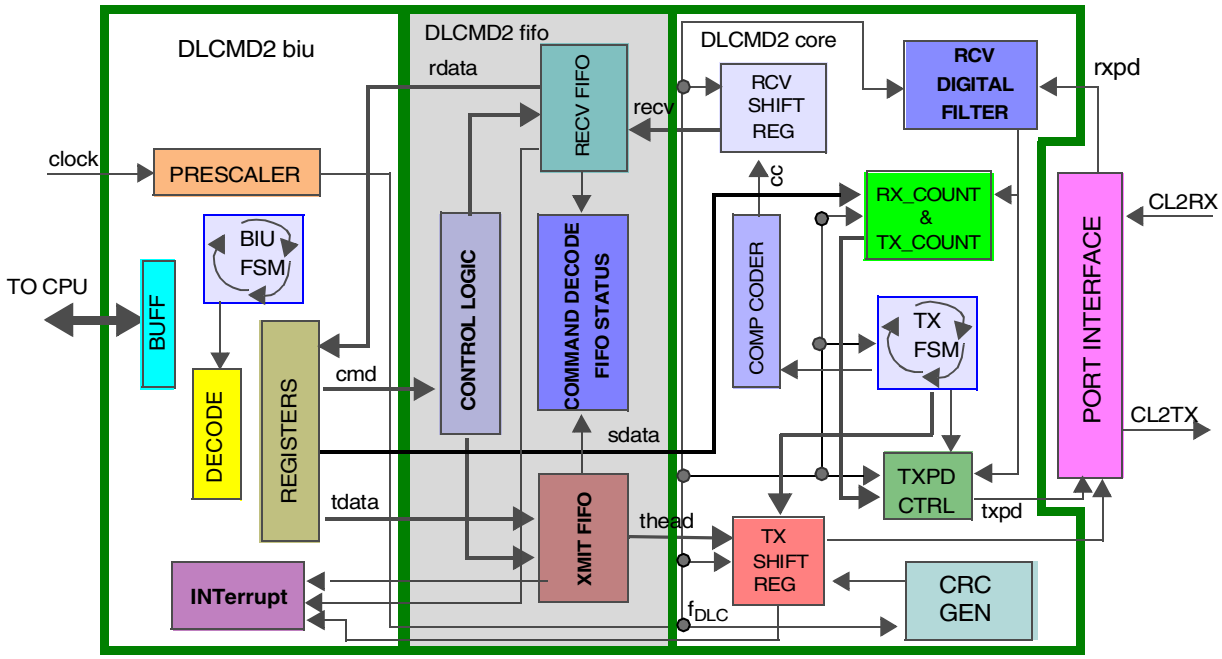

Figure 11-1 DLCMD2 Block Diagram

11.6.2 DLCMD2 Operation

The sections that follow describe the operation of the DLCMD2.

11.6.2.1 General

The DLCMD2 handles J1850 messages with minimal CPU servicing. The MCU will typically transfer complete messages into the DLCMD2 for transmission on the J1850 data link and is interrupted only when a complete message is received from the J1850 data link. Internal buffers of 20 bytes on the receive side and 11 bytes on the transmit side allow full message length operations (maximum 12 bytes normal mode, including a one-byte CRC). The DLCMD2 handles all arbitration and error detection duties internally.

The class 2 data link has been defined as the GM implementation of the SAE J1850 automotive communications protocol. The class 2 bus is a 10.4 Kbytes/s carrier sense multiple access with collision resolution (CSMA/CR) communications bus. CSMA/CR operates by arbitrating ownership of the bus on a bit-by-bit basis. The major advantage of CSMA/CR is that no message is lost in a collision. If a node determines that it needs to send a message while another message is in progress, it must wait until the link is idle. When one node begins to transmit (after bus idle), all nodes desiring to transmit are obligated to begin their transmission at the same time. The rule of arbitration is that any node that transmits a 1-bit when another node transmits a 0-bit stops transmitting on the bus immediately. This is called a zero-dominant bus. See **11.7 Signals Over-**

view for 1-bit and 0-bit definitions. All nodes are obligated to receive all bits of every message on the bus, even while transmitting. Arbitration continues to the end of a message, if necessary, resulting in a properly transmitted message even if arbitration were to continue to the end of the message. When the bus becomes idle, the node(s) which previously lost arbitration will re-transmit its message if the TxFIFO was not cleared or overwritten with another message.

11.6.2.2 Logic Section Description and Relation to Transceiver

The logic section includes the IMB3 interface, J1850 waveform generation and timing logic, buffers for data (transmit and receive), error detection code generation and checking, configuration logic, control logic, status logic, and arbitration logic.

The nondestructive contention protocol of the J1850 bus requires that there be an active and a passive state of the bus. The bus is in the active or driven state when one or more of the connected transceivers is active and passive when all transceivers are inactive. This is a logical wired OR arrangement.

The function of the transceiver is to drive the bus active in response to a signal from the DLCMD2 logic and to detect the state of the bus for the DLCMD2 logic handler. The transceiver establishes and reliably detects the state of the bus within a limited period of time. It does so in the presence of conducted and induced noise and without creating radio interference. Operation of the transceiver is constrained by available power and the need to tolerate a number of abnormal conditions.

The J1850 bus is intended to work in a relatively noisy environment. The main source of low frequency noise is ground offset between the nodes. Proper operation is assured with any combination of ground offsets up to a maximum differential of two volts at any frequency. Induced noise tends to be short duration pulses. The protocol handler includes a digital filter to remove these pulses. Additional filtering is not needed in the receiver which responds quickly and avoids stretching large amplitude pulses.

11.6.2.3 DLCMD2 Transmit/Receive Operation

A standard data exchange is composed of one data byte and (in some cases) one command byte going from CPU to DLCMD2 or one status byte and one data byte going from DLCMD2 to CPU. The use of the byte written to the DLCMD2 is specified in the command byte that accompanies it or deduced from current DLCMD2 state. The command byte also contains instructions for the DLCMD2 regarding the receive first in/first out (FIFO) buffer, transmit actions, resetting the transmitter, and sending a break signal. The status byte that the DLCMD2 sends to the CPU contains information on the status of the receive FIFO buffer, the status of the transmit FIFO buffer, the condition of the bus, and the type of accompanying data. The data accompanying the status byte can be data received off of the J1850 bus, a completion code which contains information about a received message, or nothing.

Figure 11-2 shows the DLCMD2 transmit/receive operation.

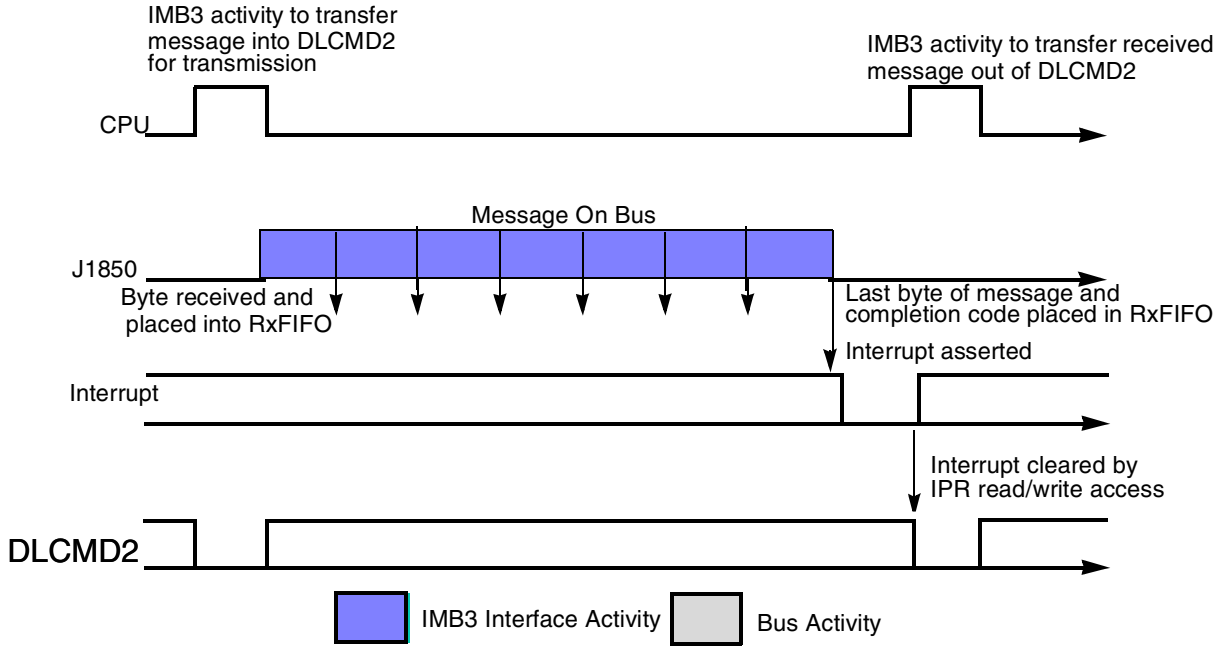

Figure 11-2 Typical Transmit/Receive Operation

All messages received from the J1850 bus will have their start bit removed and the CRC replaced with a completion code. All other bytes of the message are placed in order, most significant bit (MSB) first, in the receive FIFO buffer.

The DLCMD2 requests servicing by requesting an interrupt. Interrupts may be selected to operate in two different modes or be disabled. Typically, the DLCMD2 will only interrupt the CPU when a complete message has been received. The CPU then will service the DLCMD2 and remove the message. When the CPU desires to send a message on the J1850 bus, it will select the DLCMD2 and transfer a complete message (without start bit or CRC).

The DLCMD2 allows the CPU to continually write message bytes to be transmitted without intervening command bytes; only the last byte must be accompanied by a command byte indicating “last byte.” User code must read both status and receive data in pairs. The DLCMD2 supports aligned word writes and reads of certain locations.

The DLCMD2 can be programmed by the CPU to enter a power conserving sleep mode as soon as bus traffic stops. If interrupts are enabled, the DLCMD2 will wake-up its internal circuitry and interrupt the host when activity on the bus is sensed. The DLCMD2 will be able to correctly receive the first message that wakes it up in sleep mode.

Error conditions and transmit status, such as lost arbitration, are sent to the CPU either in the status byte or in the completion code that is placed in the receive FIFO immediately after each received message.

11.6.2.4 Message Transmission

As described in the preceding section, the DLCMD2 is loaded with a message from the CPU for transmission. The DLCMD2 will then add a start bit to the outgoing data, and contend for a message slot on the J1850 bus. The transmit buffer in the DLCMD2 is 11 bytes long, to allow complete messages to be transferred to the DLCMD2 for transmission. Information about which byte the CPU is transferring is sent to the DLCMD2 as part of the control information sent with each byte transferred. When the DLCMD2 has transmitted all of the message bytes, it will automatically append a CRC to the end of the message.

The DLCMD2 will automatically retry a transmission if it lost arbitration. The auto retry feature causes the DLCMD2 to signal to the CPU (via the status byte) that the transmit FIFO is full until the message is successfully sent. As soon as the CPU transfers the last byte of a message, the DLCMD2 will indicate that the transmit FIFO buffer (TxFIFO) is full. Once successfully sent, the DLCMD2 will signal the CPU that the TxFIFO is empty and hence ready for the next message. Each time the message loses arbitration, the completion code for that received message will indicate that the transmitter attempted transmission, and lost arbitration. As soon as a transmit slot on the bus becomes open, the DLCMD2 will automatically attempt to retransmit the message. If there were any errors during the transmission of the message the auto retry feature will cause the message to be retransmitted. The auto retry feature can be terminated by the CPU through the command byte. This causes the DLCMD2 to finish its current transmit activity, and then clear the transmit buffer. If there is no transmit activity when the auto retry is disabled and the DLCMD2 previously attempted to transmit, the DLCMD2 will immediately clear the transmit buffer.

11.6.2.5 Message Reception

Receiving information off of the J1850 bus occurs in much the same manner as sending data.

NOTE

By definition, every message a DLCMD2 sends on the data link is also received by its own receiver as if the transmission had been initiated by a different node.

The DLCMD2 will automatically remove the start bit from the message and check the CRC for errors. If a CRC error occurs, it is flagged in the message completion information (completion code) that takes the place of the CRC byte in the receive FIFO buffer (RxFIFO). The DLCMD2 will interrupt the CPU to signal that a complete message has been received or when the RxFIFO is approaching full. When the CPU starts the transfer process, the status byte from the DLCMD2 indicates the data's presence and the amount of data left in the RxFIFO. When a DLCMD2 loses arbitration to another node, it will continue to receive the remainder of the message. The completion

code will indicate that the DLCMD2 contended for a transmit slot, and lost arbitration to the received message. The CPU does not need to resend the message if the auto retry feature is enabled.

The timing of the transmit waveform is re-synchronized on each edge as received off of the bus.

Provisions have been made for immediate in-message reply to allow a path for compatibility with other J1850 implementations. In-frame response (IFR) requires a byte-by-byte interrupt mode and careful CPU attention to accomplish. IFR is described in detail in a later section.

A break/reset waveform on the bus is shaped such that it will win arbitration against any currently transmitting message. When the DLCMD2 senses that such a waveform has occurred on the bus, it will stop transmitting its current message, reset its transmitter (clear the data link controller module (DLCMD2) TxFIFO), and set a bit in the completion code and request an interrupt to indicate to the CPU that such a condition has occurred. Since the break signal always wins arbitration, any in-progress messages will simply lose arbitration, and the DLCMD2 will treat the in-progress received message as complete. If a break occurs while there are no in progress messages, a completion code indicating that a break has occurred will be placed in the RxFIFO and a CPU interrupt generated. The break/reset waveform is sent by a DLCMD2 combination in the command byte. This waveform is sent immediately upon the command byte being latched in. If there is a current transmission, it will be stopped, and the break waveform sent. A break will also take the DLCMD2 out of 4X mode.

11.6.2.6 Sleep Mode

The CPU may put the DLCMD2 in sleep mode by setting the STOP bit in the MCR. Setting this bit will tell the DLCMD2 to halt its internal clocks, immediately after any currently in progress messages are completed.

Interrupts to the host on bus activity can be disabled by configuring the DLCMD2 with the ILR register. Normal use of the sleep feature will have interrupts to the host enabled, so that the host will not miss any messages on the data link. If interrupts are disabled, and then the DLCMD2 is put to sleep, the only way to wake up the DLCMD2 is by the CPU clearing the STOP bit.

11.6.2.7 Debug Mode

This mode is entered from the reset state or from the run state by asserting or deasserting the appropriate signals. See **11.8.5 DLCMD2 DEBUG** for details.

11.6.2.8 4X Speed Mode

The DLCMD2 has the ability to transfer large amounts of data in a 4X speed mode under special conditions such as memory loading at the vehicle factory, and diagnostic responses. There is a bit in the MCR to control this feature. The 4X speed mode affects only the bit timing section of the DLCMD2, including the digital filter. A break will reset

any listening nodes out of 4X speed mode. The 4X speed mode is not for use during normal operation.

To use 4X mode there must be coordination of all nodes on the network. This mode will not work properly at the network level unless ALL nodes are transmitting in 4X mode. Certain nodes may elect not to take part in 4X communications; these nodes may listen but must not transmit.

Notification of entrance into the special 4X mode is communicated to all nodes with a regular speed message indicating the bus protocol speed switch to 4X mode. A BREAK received will automatically take the DLCMD2 out of 4X mode.

11.6.2.9 Block Mode

The DLCMD2 has the ability to receive a message of unlimited length, provided the CPU reads bytes out of the RxFIFO before it overflows.

If the TxFIFO was filled with no last byte indicated, the DLCMD2 will start sending that message in terminate auto-retry mode; the status will indicate “TxFIFO almost full” as soon as the first byte is sent, and “TxFIFO contains some data bytes” as soon as the second byte is sent. As new data is written to the TxFIFO, the status will accurately reflect the condition of the TxFIFO until a “last byte” is written. When a “last byte” command has been sent to the DLCMD2, the TxFIFO will indicate “TxFIFO full” until the transmission is finished. The DLCMD2 will send an infinite length message if properly handled by the CPU.

11.6.2.10 Error Detection

The DLCMD2 uses a digital filter and the cyclical redundancy check (CRC) byte to detect errors.

The digital filter eliminates short duration noise spikes and transition noise from the incoming waveform. It is a “hysteresis” type filter with a time constant of approximately 8 µs, depending on the IMB3 clock frequency. The step response of the filter is a step function delayed by ±8 µs. It can be described by example with a 2-MHz clock (T_{DLC} = 0.5 µs) and a 4 bit up/down counter. The counter counts up for every oscillator clock pulse when the input is in the active state and down when the input is in the passive state. The counter clamps at 0 and 15. The output is defined by **Table 11-1**.

Table 11-1 Digital Filter Output

Count	Output
0	0
1 – 14	Unchanged
15	1

This filter will cause a receive delay of 16-17 times T_{DLC} in addition to the delay in the transmitter and receiver analog interface circuitry. This delay’s only variation is due to the tolerance on the CPU’s oscillator.

In simple terms, the effect of the filter is that a low or high level on the bus is not recognized unless it is longer than about 8 µs.

The CRC byte is used by the receiver to determine if any errors have occurred during transmission. CRC generation uses the divisor polynomial: $X^8 + X^4 + X^3 + X^2 + 1$. The transmitted CRC is generated by the receiver by initially setting the remainder polynomial to all ones, serially processing the first byte and then all remaining bytes of the message, and appending the one's complement of the remainder to the end of the transmitted message. The receiver uses the same divisor polynomial to process all received message bits including the CRC but excluding the start bit. If the transmission is received correctly, at the completion of the message reception, the remainder polynomial will be: $X^7 + X^6 + X^2$ (%11000100 or $C4).

This CRC code will detect all single and 2 bit errors and all 8 bit burst errors (i.e., any number of errors within a single 8-bit span). Severe noise will normally be detected separately as a bit timing error.

11.6.2.11 Arbitration

The J1850 bus is classified as a carrier sense multiple access with collision resolution (CSMA/CR). This type of bus operates by arbitrating ownership of the network on a bit-by-bit basis. The major advantage of CSMA/CR is that no message is lost in a collision. If a node determines that it needs to send a message while another message is in progress, it must wait until the link is idle. When one node begins to transmit (after bus idle), all nodes desiring to transmit are obligated to begin their transmission at the same time. The rule of arbitration is that any node that transmits a 1-bit when another node transmits a 0-bit stops transmitting on the bus immediately. This is called a 0 dominant bus. To prevent noise from corrupting the bus, arbitration is also lost if a 1-bit is detected when a 0-bit was transmitted. All nodes are obligated to receive all bits of every message on the bus, even while transmitting. Arbitration continues to the end of a message, if necessary. If an opposite bit is detected, transmission is immediately stopped unless it occurs on the 8th bit of a byte. In this case the DLCMD2 will automatically append two extra 1-bits and then stop transmitting.

NOTE

Two extra bits must be transmitted due to the fact that the eighth bit of a byte is an active, high level on the J1850 bus. Therefore the first extra bit will be a passive, low level, and one more bit is needed in the active, high level so that after the falling edge of this bit the bus will be in the passive state.

These two extra bits will be arbitrated normally and thus will not interfere with another message.

11.6.2.12 Timebase Generation

The generation of time intervals within the DLCMD2 module takes into account the variations of MCU family, oscillator frequency, and physical interface delays that may occur. The frequency f_{IMB3} is sent to the DLCMD2 where it is further divided by "n" (set

by the user through the **Symbol Timing Control and Pre-Scaler Register (SCTL**)), such that the main DLCMD2 operating frequency (f_{DLC}) is approximately 2.00 MHz, depending on IMB3 clock frequency.)

The DLCMD2 J1850 bit timings are derived from the f_{DLC} time base and a stored table of VPW symbol values. The table of VPW symbol values is generated by the user via the **Symbol Timing Data Register (SDATA**).

NOTE

The f_{DLC} signal defines the fundamental resolution of the DLCMD2 module. All bit timings within the DLCMD2 are based upon integer multiples of the fundamental resolution.

Should a physical interface exhibit an unusually large delay, the length of the J1850 transmit symbol values stored in the DLCMD2 symbol table may be reduced pro rata to compensate.

The VPW symbol length table is determined by the user after the symbol lengths have been set via the transceiver REXT bias resistor selection. The REXT resistor values are chosen so as to minimize the radio frequency interference (RFI) from the J1850 bus by inputting a 10.4-kHz square wave into the transmitter and subsequently out on the J1850 bus. These biasing resistors will affect the length of the VPW symbols to some degree due to their effect on the corners of the bus signal that is output by the transmitter.

11.6.2.13 Receive and Transmit Message Buffers

The RxFIFO and TxFIFO are 20 and 11 bytes in length respectively, to allow buffering of a complete message.

The TxFIFO must be able to differentiate between three types of data:

1. Message data byte
2. First byte in message
3. Last byte in message

The auto retry feature recirculates the bytes of a message in the TxFIFO until the message is successfully sent, at which time the FIFO's contents are flushed. When auto retry is disabled, the FIFO will complete an "in progress" transmission, if any, and then flush the contents of the FIFO. If the node is not transmitting, the FIFO will be flushed immediately. If the auto retry feature is disabled as a message is being loaded into theDLCMD2, the DLCMD2 will try to transmit the message once and then clear the transmit FIFO.

Received bytes will be placed into the RxFIFO as soon as they are completely received off of the bus. When an EOD has occurred on the bus, a completion code will be inserted into the RxFIFO after the last received byte of the message. The CRC byte will be checked by the logic and discarded.

11.6.2.14 Bus Waveforms Generation

The DLCMD2 supports Huntzicker encoding. Each symbol generated by the DLCMD2 will be synchronized with the latest edge seen on the bus. Errors due to oscillator tolerance and ground offsets will not accumulate through the message in this manner. Synchronizing in this manner does require that the bit timing unit account for all known delays. The transmit timing will have a very narrow window due to oscillator tolerance and variation in the known delay only. The receive timing will have much wider windows due to the uncertainty in determining edge position resulting from ground offsets, oscillator tolerance, and delay time variation. In either case, transmit or receive, the timing will be specified as beginning when the DLCMD2 senses a transition, to when the DLCMD2 causes or senses the next transition.

11.6.2.15 Huntzicker Encoding

The information contained in this section describes the bit timing section of the logic on the DLCMD2. The timing of VPW (Huntzicker) waveforms requires knowledge of the fixed delays in the transceiver and the logic section. The J1850 bus is a single wire ground referenced bus. This configuration has two important consequences for the bit timing section. In order to reduce the radiated emissions of the bus, each edge must be slew rate limited, and have its corners shaped. To not adversely affect the corner shaping, the specification must not place limits that force the corners. The other consequence is due to the ground offset requirement for the bus. This requirement dictates a minimal voltage swing necessary to operate in the presence of ground offset. The combination of the two factors gives rise to an uncertainty in both when the receiver (of a receiving node) detects a given transition, and when the transmitter (through its own receiver) detects the same transition.

VPW encoding defines one edge for each symbol. A symbol is composed of a period of time (at a particular state of the bus) and the edge that follows that period. The point of reference for the time period is the trip point that the receiver uses to recognize the preceding transition on the bus. Three independent variables are used to describe the waveform generated. These are the times from the trip point to each of the following transitions threshold levels, and the time between these threshold levels. The corners of the waveform fall outside of the “slew rate” time requirement, and may bargain for time and voltage more freely.

The following symbol limits are consistent with $T_{t,max}$ = 16 µs and an oscillator tolerance of 2%. T_{nom} is the nominal symbol time with no oscillator error and the receiver detecting the transition at $T_{t,max/2}$. T_{r1} to T_{r2} is the required acceptance range while T_{r1typ} to T_{r2typ} is a typical acceptance range with a 2% guard band plus a small margin. T_{r1typ} to T_{r2typ} in **Table 11-2** represent the receiver windows. T_{x1} and T_{x2} in **Table 11-3** represent transmitter windows consistent with a 2% oscillator tolerance and 3 µs for all other variations in the transmit path.

Table 11-2 Receive Windows

Symbol [1, 2, 3]	T_{rnom}	T_{rmin}	T_{rmax}	T_{r1}	T_{r2}	T_{r1typ}	T_{r2typ}	Units
Short 1/0	64	53	75	37	91	34	96	µs
Long 1/0	128	116	141	100	157	96	163	µs
SOF/EOD	200	186	214	170	230	163	239	µs
EOF	280	265	—	249	—	239	320[4]	µs

1. All waveforms less than 8µs will be filtered out by the digital filter, and will not be seen as an error.
2. Break is an active symbol that will be transmitted as at least 239µs in length.
3. All window times include digital as well as analog signal delays.
4. All transmitters are armed as soon as they detect the EOF. The end of the guard band on the EOF serves as a arming point for all transmitters. This is the point that all nodes must have recognized an EOF.

Table 11-3 Transmit Windows

Symbol [1, 2, 3]	T_{xnom}	T_{xmin}	T_{xmax}	Units
Short 1/0	64	60	68	µs
Long 1/0	128	122	134	µs
SOF/EOD	200	193	207	µs
EOF	280	271	289	µs

The symbol waveforms seen on the bus have two important characteristics:

Each transition of the transmitted bus signal, as initiated by CL2TX (LOTI), is slew rate limited and has its corners rounded (wave shaped) so that the nominal rise or fall time is about 16 µs to reduce the radiated emissions of the bus. This wave shaping is disabled when 4X mode is enabled.

The received bus signal needs only a minimal voltage swing around the receiver's nominal trip point voltage for proper detection. The point of reference for the time period is the trip point voltage (V_t) a receiver uses to recognize a transition on the bus and produce the CL2RX (LITO) signal.

The CL2RX signal is digitally filtered with an approximate 8 µs delay at the 10.4 kHz bus rate (2 µs in 4X mode). Since a high or low level input to the filter must last longer than the filter delay time in order to appear at the filter output, noise pulses shorter than this are eliminated.

When a single node is transmitting, the symbol time period between successive transitions is controlled completely by the transmitter's transmit symbol timing logic. When two or more nodes are contending for the bus the start point for an active to passive state transition is determined by the node with the slowest clock rate and the start point for an inactive to active state transition is determined by the node with the fastest clock rate, assuming that both nodes are transmitting the same symbol. The symbol width as controlled by the transmitting node's CL2TX signal, can range from T_{xmin} to T_{xmax}. The receiver's acceptance time window range (T_{rmin} to T_{rmax}) is much broader to allow all widths to be classified into defined symbols.

The time windows are not affected by multiple nodes trying to transmit at the same time during arbitration. This is because one node effectively dominates each transition (the first node to leave the passive state or the last node to leave the active state). Although the fastest or slowest node dominates a particular transition, the arbitration scheme assures that the highest priority message always wins.

J1850 bus transmitter output and input signal waveforms are shown in **Figure 11-3**.

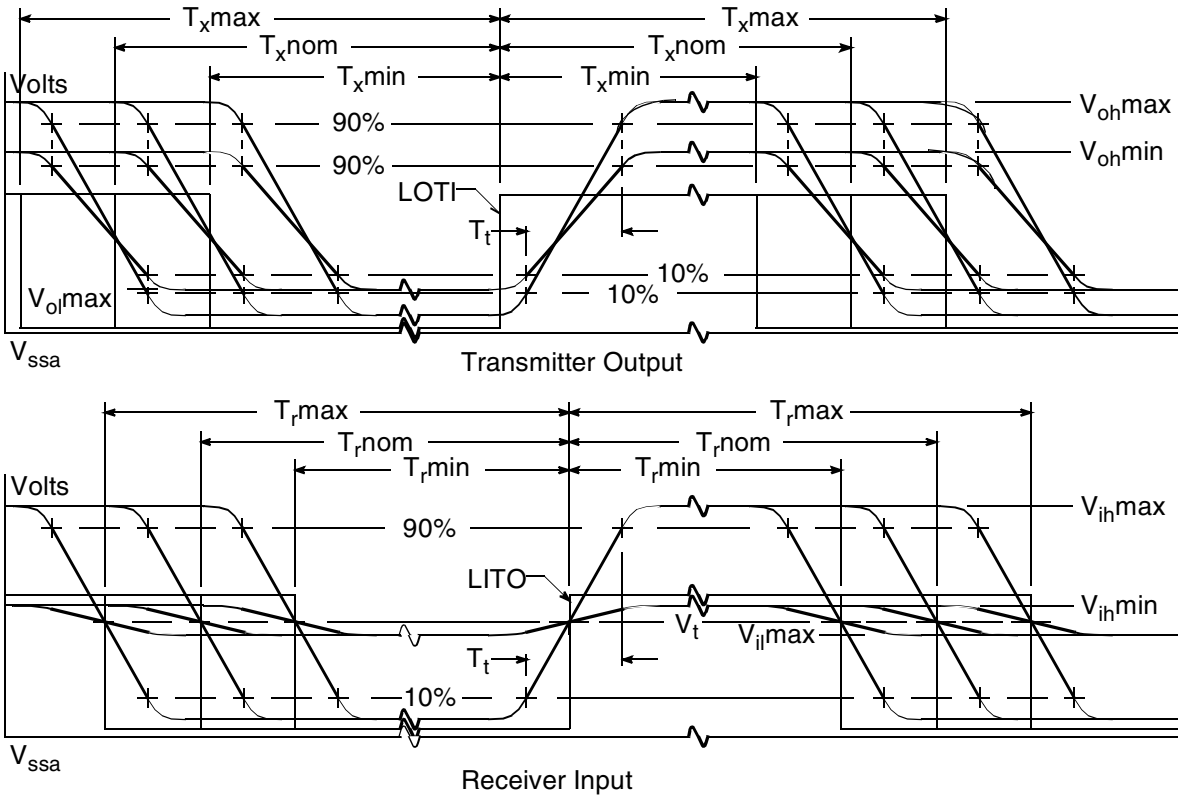

Figure 11-3 VPW Signal Waveforms

NOTES:
1. T_{xmax} is maximum symbol transmission time.
2. T_{xnom} is nominal symbol transmission time.
3. T_{xmin} is minimum symbol transmission time.
4. T_{rmax} is maximum symbol receive window time.
5. T_{rnom} is nominal symbol receive window time.
6. T_{rmin} is minimum symbol receive window time.

11.7 Signals Overview

This section provides an overview of DLCMD2 signals.

11.7.1 J1850 Bus Waveforms

The DLCMD2 module must be able to generate and recognize the set of Huntzicker waveforms described in the following sections. See **Figure 11-4**. Additionally:

- Each symbol is represented by the time between two consecutive transitions
- There is one transition per symbol and one symbol per transition
- There are both active and passive symbols that are used alternately
- A longer active symbol will dominate a shorter one
- A shorter passive symbol will dominate a longer one

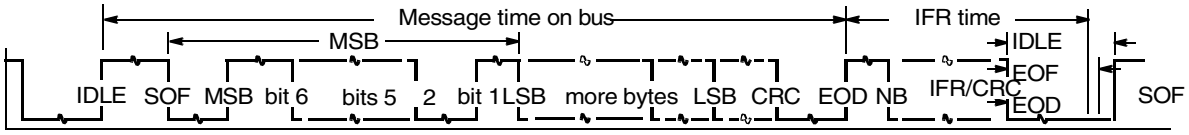

Figure 11-4 Huntzicker Waveform Message

The following sections show the nominal time duration, in microseconds (µs), of the VPW message symbols generated by the DLCMD2 as they appear on the J1850 bus when operating at the normal bus speed. When the DLCMD2 is operating at the high bus speed all 4X symbol times are one fourth that shown, except for “Break”, which will be transmitted the same length in 1X or 4X mode.

11.7.1.1 Start of Frame (SOF)

This active symbol appears at the start of every message when a transmitter drives the bus high to start a message.

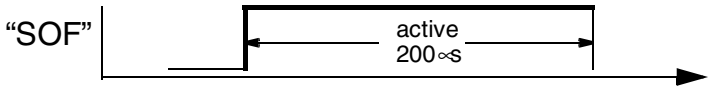

Figure 11-5 Start of Frame Symbol

11.7.1.2 Data Bits

Each data bit is represented by the time between two consecutive transitions. There are both passive and active bit states that are used alternately. The “0” bit is the dominant bit in arbitration.

11.7.1.3 “0” bit

The two dominant “0” bit waveforms are:

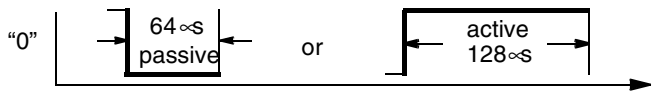

Figure 11-6 Passive “0” and Active “0”

11.7.1.4 “1” Bit

The two “1” bit waveforms are:

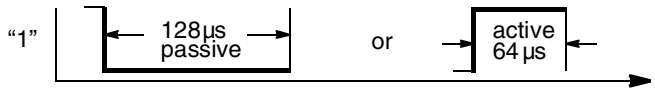

Figure 11-7 Passive “1” and Active “1”

11.7.1.5 End of Data (EOD)

This passive symbol appears after the first CRC byte only in the “request in-frame data” message. It ends when the responding transmitter sends its normalization/format bit prior to the start of the first in-frame response byte. If no node responds, this passive symbol will stretch into an “end of frame” symbol.

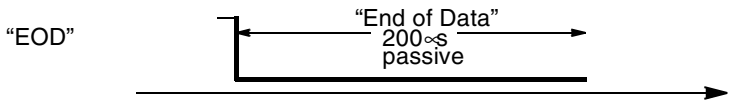

Figure 11-8 End of Data Symbol

11.7.1.6 Normalization Bit

The normalization bit symbol’s duration is the same as an active “1” or “0” bit time. The format of this bit, whether it is a “1” or “0”,” can be selected by the normalization bit format select (NBFS) bit in the **Symbol Timing Control and Pre-Scaler Register (SCTL**). J1850 protocol encourages the use of a “0” when the in-frame response (IFR) ends with a CRC byte and a “1” when the IFR does not end with a CRC byte.

11.7.1.7 End of Frame (EOF)

This passive symbol appears at the end of every message. It is at least 280 µs long. If the bus remains passive until 320 µs, the bus is idle and a transmitter may begin transmitting. If a transmitter desiring bus access detects a rising edge on the bus

between 280 µs and 320 µs (due to clock mismatch between nodes) it may join in and arbitrate.

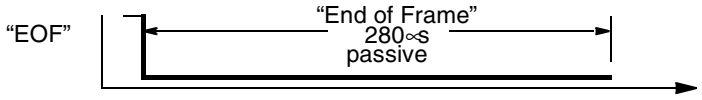

Figure 11-9 End of Frame Symbol

11.7.1.8 Break

The active "Break" signal causes any other transmitting module to stop transmitting immediately because it loses arbitration. It is at least 239 µs long.

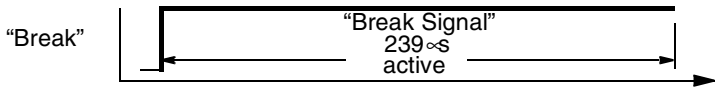

Figure 11-10 Break Symbol Controller Module (DLCMD2)

11.7.2 General Symbol Transmission

The J1850 transmitter will drive the bus to active state and expect that the external RC network will pull the bus back down to the passive state, which is relative since there may be a difference of base ground potential between J1850 nodes in the vehicle. The transmitter is responding to feedback from the receiver in order to know precisely when to switch the transmitter on or off. There is a set of basic transmit timing windows for transmitted symbols within the logic section of the DLCMD2 but if the receiver detects the state of the bus as changing early, the transmitter will also change to that level unless it had not intended to transmit that symbol whereby arbitration is lost and transmission will cease immediately. Thus, all J1850 devices on the J1850 bus synchronize to each other's clock and ground mismatches. Remember that there is only one train of symbols appearing on the bus. The individual symbols are pulled high and released low by various transmitters but the end result is one waveform. It just may be seen differently by the devices due to clock, ground, and power supply variations.

11.7.3 General Symbol Reception

An external transceiver passes unfiltered bus status information to the DLCMD2's Rx pin. Internal to the DLCMD2, the digital 1 or 0 is clocked through a digital delay filter for 16 ticks of its internal frequency clock (a delay of 8 µs at normal (2-MHz f_{DLC}) speed before the filter output changes state. High and low levels on the J1850 bus are timed in the logic section and compared to a set of received symbol threshold windows. Every received high or low level is translated into one of the symbols in the above sections or is flagged as a bit timing error.

The bus is “idle” when the output of the DLCMD2’s digital filter has been in the passive state for 320 µs.

11.7.4 Support For External Transceiver

As shown in **Figure 11-11**, the DLCMD2 will be designed to use an external (IC) transceiver. Along with the CL2TX and CL2RX signals, a 4XEN signal will be provided at the periphery of the DLCMD2 to disable transceiver waveshaping during DLCMD2 4X mode.

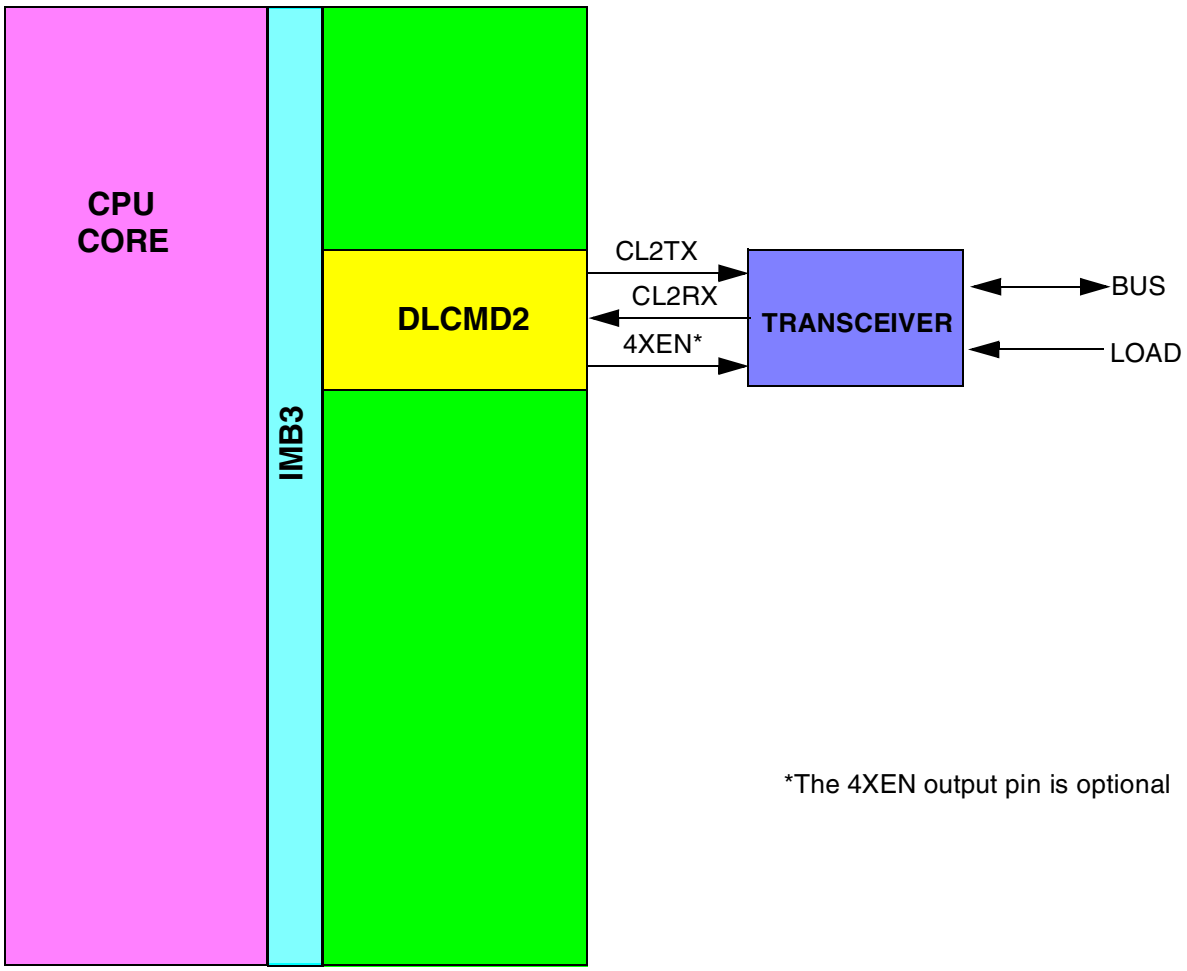

Figure 11-11 Support For External Transceiver

11.8 Operating Modes

This section describes DLCMD2 operating modes. The DLCMD2 has five main modes of operation which interact with the power supplies, pins, and the rest of the MCU. Refer to **Figure 11-12**.

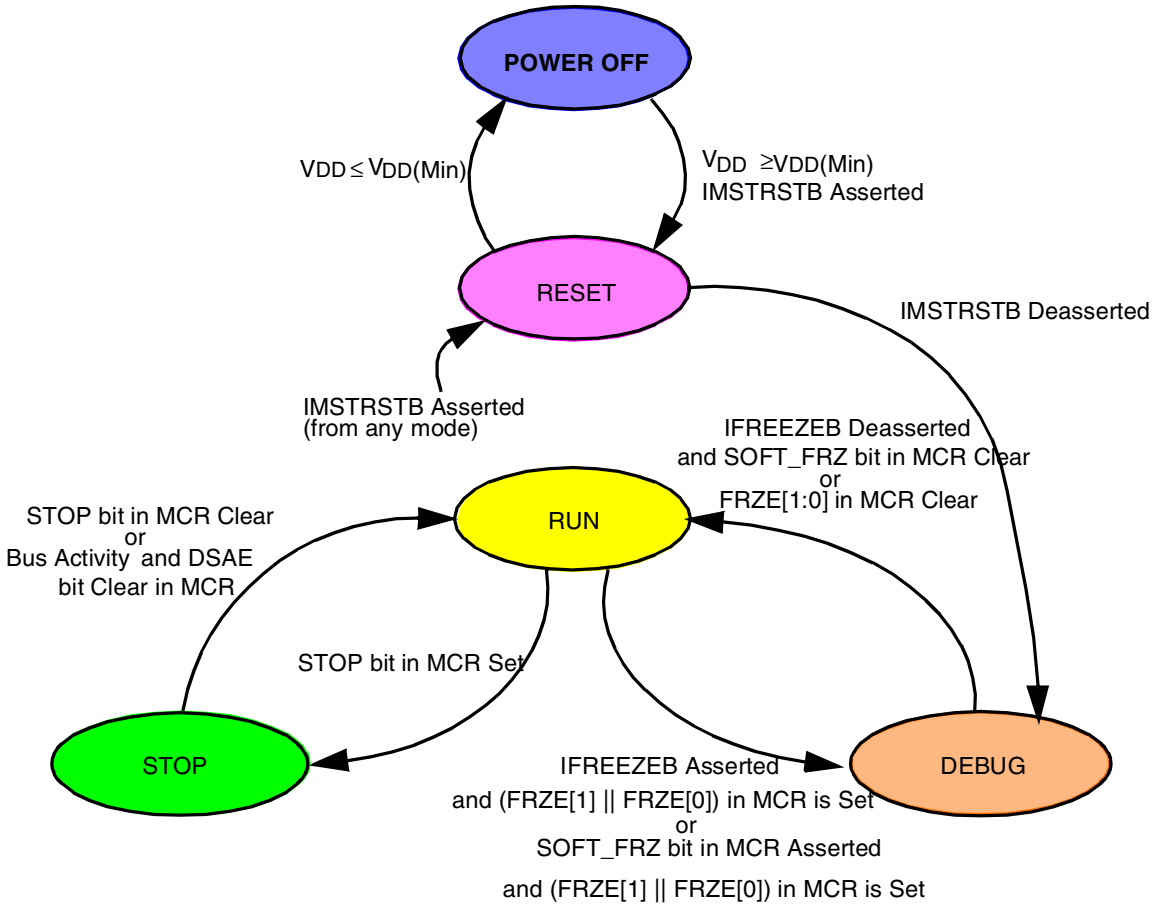

Figure 11-12 DLCMD2 Operation Modes

11.8.1 Power Off

This mode is entered from the reset mode whenever the DLCMD2 supply voltage V_{DD} drops below its minimum specified value for the DLCMD2 to guarantee operation. This implies that the DLCMD2 must be placed in the reset mode before being powered down. In this mode, the pin input and output specifications are not guaranteed.

11.8.2 Reset

This mode is entered from the power off mode whenever the DLCMD2 supply voltage V_{DD} rises above its minimum specified value and IMSTRSTB is asserted. This implies that IMSTRSTB must be asserted while powering up the DLCMD2 or an unknown

state will be entered and correct operation cannot be guaranteed. It is also entered from any other mode on the falling edge of CLOCK after IMSRSTB is asserted.

In this mode V_{DD} is supplied to the internal circuits, which are held in their reset state and the internal DLCMD2 system clock is running. Registers will assume their reset condition. Outputs are held in their programmed reset state, inputs and network activity are ignored.

11.8.3 Run

This mode is entered from the debug mode after all MCU reset sources are no longer asserted. It is entered from the DLCMD2 STOP mode whenever a message is successfully received or the CPU has accessed the DLCMD2 and negated the STOP bit (if previously set).

It is entered from the DLCMD2 LPSTOP mode whenever network activity is sensed although messages will not be received properly until the clocks have stabilized and the CPU is also in the run mode.

In this mode, normal network operation takes place. The user should ensure that all DLCMD2 transmissions have ceased before exiting this mode.

11.8.4 DLCMD2 STOP and LPSTOP

11.8.4.1 DLCMD2 STOP mode

This mode is automatically entered from the run mode whenever the CPU executes a STOP instruction. The IMB3 clocks continue to run, but the CPU clock is stopped.

In this mode, the DLCMD2 internal clocks continue to run and the module will await a valid network message. If a valid network message is successfully received, a CPU interrupt request will be generated (if interrupts are enabled).

Controller module (DLCMD2) DLCMD2 power is only conserved in this mode if the STOP bit in the MCR is set, stopping the DLCMD2 clocks.

11.8.4.2 DLCMD2 LPSTOP mode

This power conserving mode is automatically entered from the run mode whenever the CPU executes a LPSTOP instruction.

In this mode, the DLCMD2 internal clocks are stopped and the module will await any J1850 activity (including noise). If network activity is sensed, then a CPU interrupt request will be generated, restarting both the IMB and DLCMD2 internal clocks.

11.8.5 DLCMD2 DEBUG

This is a special debug mode entered by asserting the SOFT_FRZ bit in the MCR register, or by asserting IMB3 IFREEZEB line. For both, activating the DLCMD2 debug mode is qualified by the FRZE[1:0] in MCR register.

Upon exiting the reset state the SOFT_FRZ bit and FRZE[1:0] bits are set in the MCR register. Hence, reset mode is always followed by the debug mode.

Once this mode is set, the following occurs:

- The pre-scaler divider is stopped, thus halting all related activities.
- Any activity on the J1850 bus will be ignored. The DLCMD2 ignores the CL2RX input pin and drives CL2TX to the passive state.
- The CPU can read and write into most of DLCMD2 registers except for otherwise noted in the register description section.
- The NOT_RDY and FREEZ_ACK bits in MCR register are set.
- After asserting the debug mode configuration bits, the user must wait for the FREEZ_ACK bit to be set in MCR register, before executing any other action to the DLCMD2; otherwise the DLCMD2 may operate in an unpredictable way.

Exiting the debug mode is done in one of the following ways:

- Both, IMB3 freeze and SOFT_FRZ bits are negated.
- CPU negates the FRZE bit.
- Once debug mode is exited, the DLCMD2 is ready to transmit/receive normally on the J1850 bus.

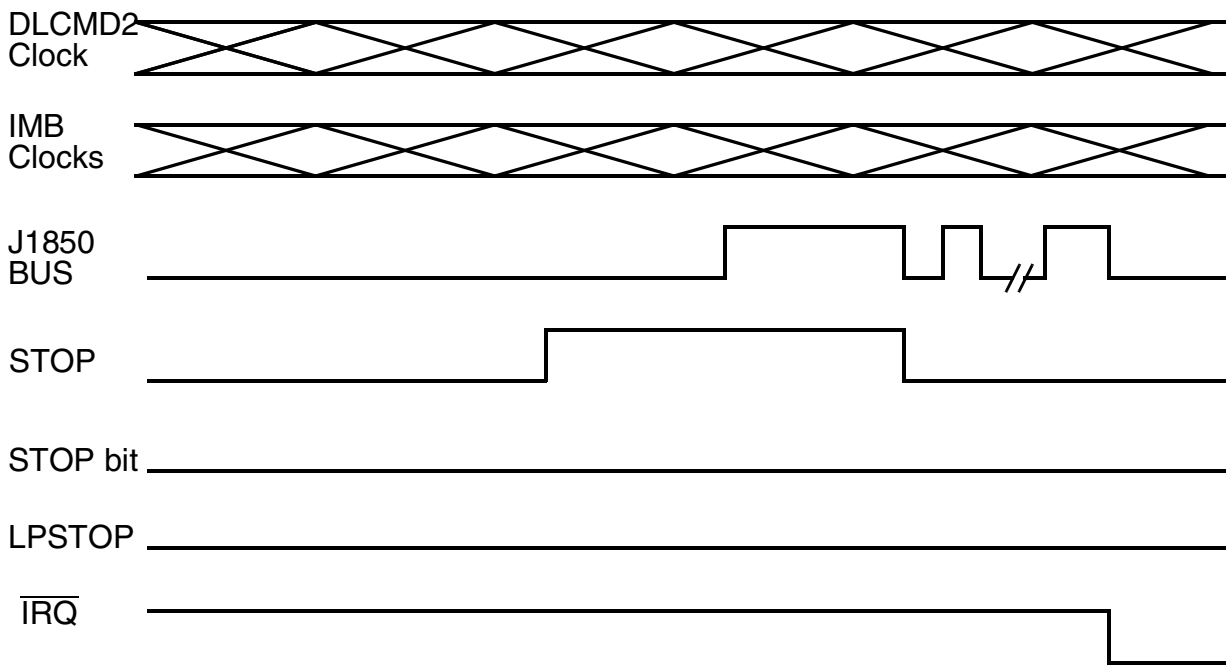

Figure 11-13 STOP Power Mode (No STOP Bit Set)

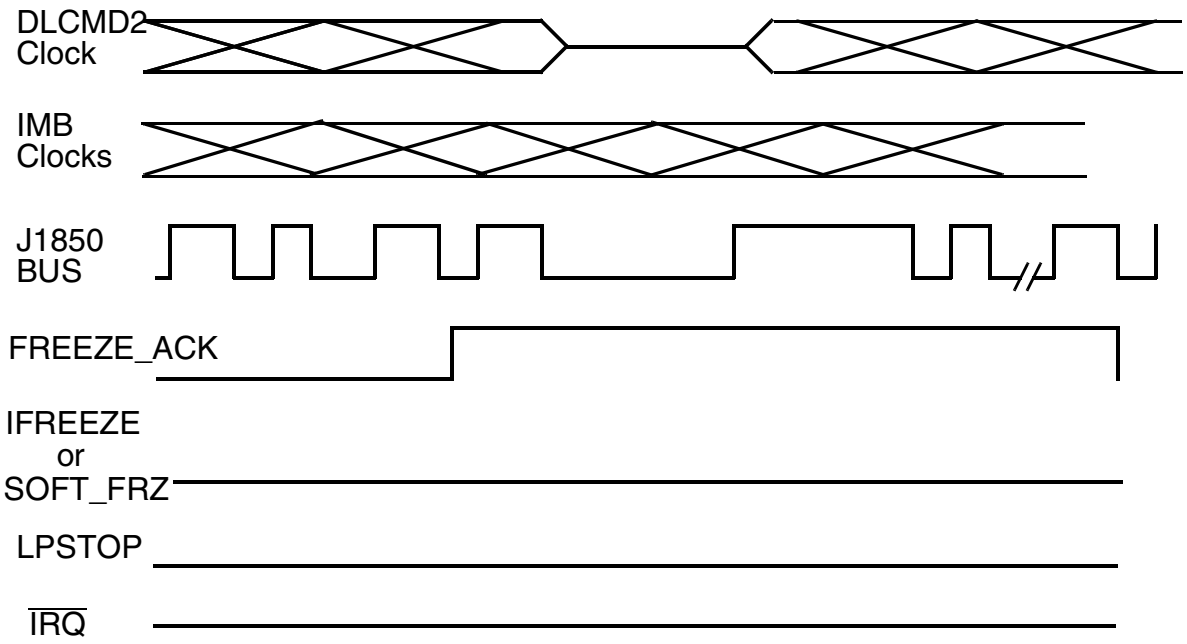

Figure 11-14 DEBUG Power Mode (IFEEZEB or SOFT_FRZ)

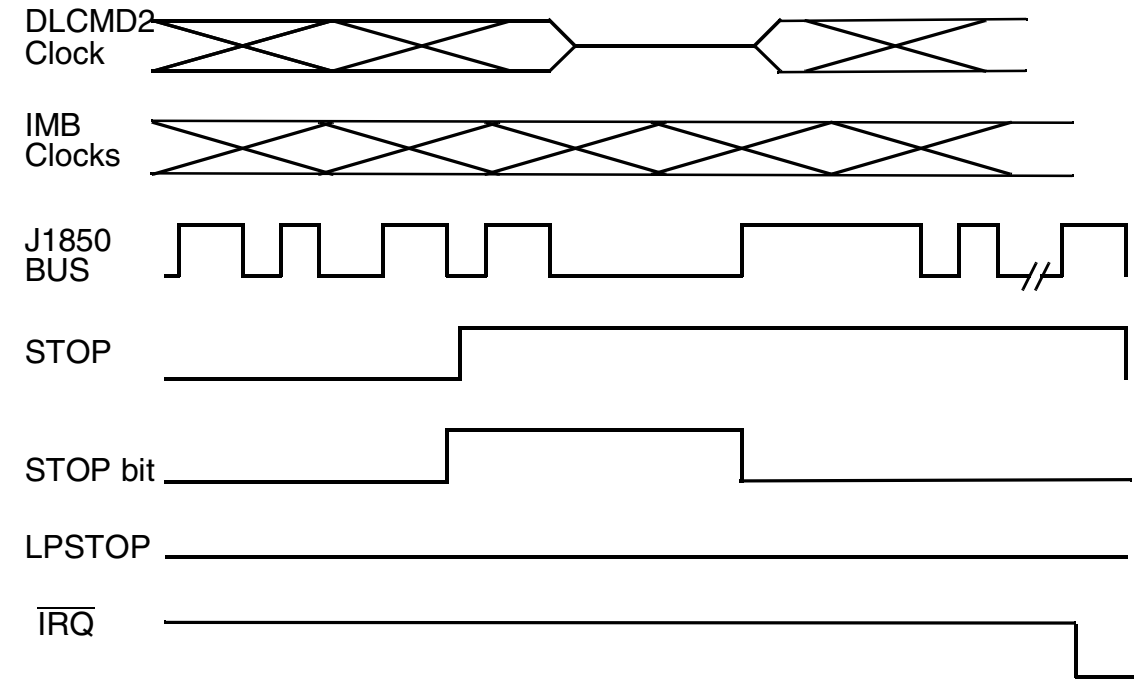

Figure 11-15 STOP Power Mode (STOP Bit Set)

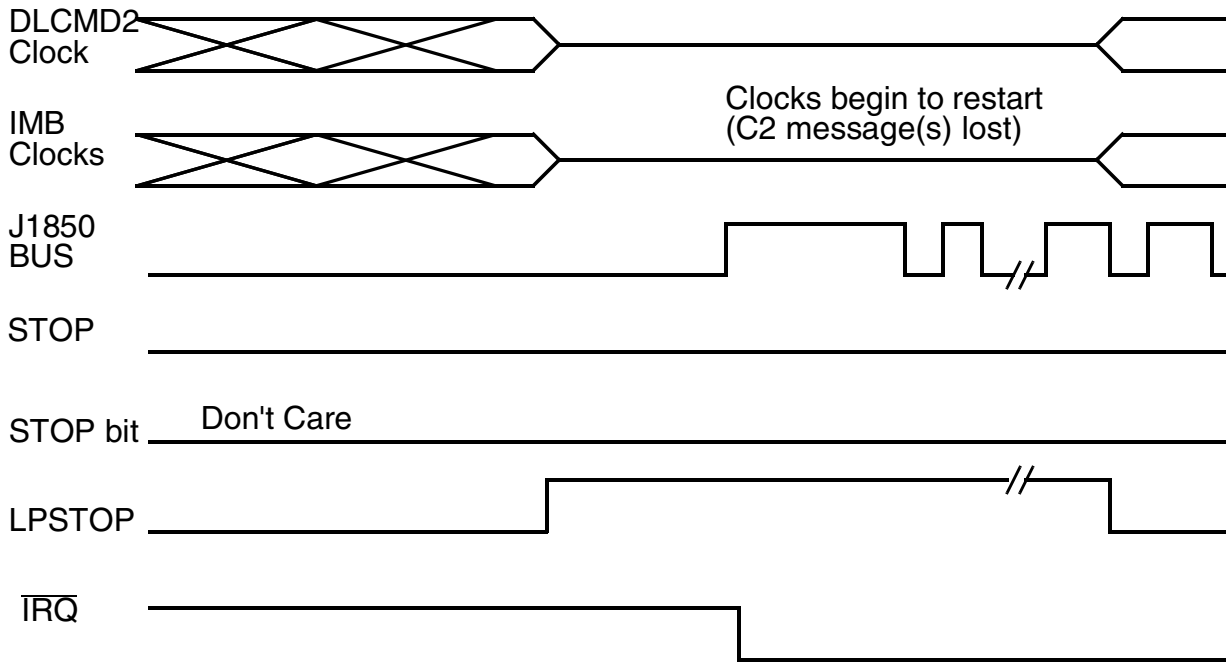

Figure 11-16 LPSTOP Power Mode

11.9 CPU Interface

This sections covers DLCMD2 interfaces to the CPU.

11.9.1 Interface Requirements

This section defines the interface protocol used.

The logical operations done in a DLCMD2 module service routine are as follows:

1. Read status byte from DLCMD2 module
2. Read received data from DLCMD2 module if required
3. Write command byte to DLCMD2 module if required
4. Write transmit data to DLCMD2 module

These four bytes are a complete exchange of information. However, these bytes are not all necessary or required during a CPU/DLCMD2 data link controller module (DLCMD2) module transfer. All possible unique CPU/DLCMD2 module transfers are shown in **Table 11-4**. All CPU/DLCMD2 module transfers will be made up of combinations of one or more of these building blocks. Pop and push refer to automatic (without command byte) flushing and loading of data bytes out of and into the DLCMD2 module.

A transmit data byte need not immediately follow the command. The transmit data byte must be the next byte transferred to the DLCMD2 but could be sent any time later. The command will not be acted upon until this next byte is sent to the DLCMD2.

Table 11-4 CPU/DLCMD2 Transfers

Acceptable Read/Write Combinations	Auto Pop?	Auto Push?	First Byte Flag Set?	Word Read/ Write Allowed?
1. Read status byte from DLCMD2[1]	No	No	No	No
2. Write command byte to DLCMD2	—	No	No	Yes
Write transmit or dummy data to DLCMD2 module	—	No	Maybe	Yes
If no data accompanies, then command byte causes no action	—	—	—	—
3. Read status byte from DLCMD2	No	—	—	Yes
Read received data byte from DLCMD2	Yes	—	—	Yes
Read status byte from DLCMD2	No	—	—	Yes
Read received data byte from DLCMD2	Yes	—	—	—
(Until entire message received)	—	—	—	—
4. Write transmit data byte to DLCMD2	—	Yes[2]	Yes	No
Write transmit data byte to DLCMD2	—	Yes	No	No
Write command byte to DLCMD2 (load as last byte)	—	No	No	Yes
Write transmit data to DLCMD2 (last byte)	—	No	No	Yes

NOTES:
1. Minimal transfer.
2. If TxFIFO empty.

In numbers 2-4, word reads and writes may be done to read a status byte and a received data byte, or write a command byte and a transmit data byte with one instruction. If number 2 was for loading a single byte for transmit the command byte would specify load as first and last byte (no auto push, no first byte flag set). Number 3 would be for reading an entire message from the DLCMD2 module when there is no transmit data to be sent to the DLCMD2 module. Number 4 is a quick way to load an entire message into the DLCMD2 module when there is no data to read from the DLCMD2 module.

Auto pop is the default for reading from the RxFIFO. This means that a read of a data byte from the RxFIFO causes the current byte in the FIFO to be automatically flushed. Auto pop can not be disabled. Auto push is the automatic loading of a data byte into the TxFIFO via a write to the DLCMD2 module. This is the default when there was no preceding command byte. There is no auto push if a command byte is sent before the data byte; and the command byte must specify what to do with the following data byte.

The action(s) called out in the command byte will be acted upon the moment the transmit (or dummy) data byte is written into the DLCMD2 module.

11.9.2 Reset Operation

When master reset is asserted, the DLCMD2 module will be held in an off state. System power should be up and stable when master reset is negated.

After toggling the reset line, the CPU writes command and any configuration bits into the DLCMD2 module to initialize it.

11.10 Operational Information

The following sections will be included in the DLCMD2 application document and are mostly redundant information to previous sections.

11.10.1 Initialization

After power up and/or reset the DLCMD2 should be configured via writes to the MCR, ILR,IVR, SDATA, and SCTL registers.

NOTE

Interrupts are disabled at power up and reset and must be enabled if a polled method of servicing the DLCMD2 is not used.

- Step 1 — Initialize MCR
  Begin initialization of the configuration bits by writing the SUPV bit in the MCR. This will determine what types of accesses are to be allowed (user and supervisor or just supervisor) to DLCMD2 registers for the rest of the initialization process and during normal operation. Care should be taken not to clear the SOFT_FRZ or FRZ[1:0] bits while writing the SUPV bit. This precaution will ensure the DLCMD2 remains in the debug mode until initialization is complete.
- Step 2 — Initialize ILR and IVR registers if interrupts employed
- Step 3 — Initialize SCTL and SDATA registers
- Step 4 — Enable DLCMD2 by exiting DEBUG mode

11.10.2 Transmitting a Message

The DLCMD2 is loaded with a message from the CPU for transmission. The DLCMD2 will then add a SOF bit to the outgoing data, and contend for a message slot on the J1850 bus. The TxFIFO in the DLCMD2 is 11 bytes long, to allow complete (except block) messages to be transferred to the DLCMD2 by the CPU for transmission. Information about which byte the CPU is transferring is sent to the DLCMD2 as part of the control information sent with each byte transferred and is optional with the DLCMD2 except for the last byte. When the DLCMD2 has transmitted all of the message bytes, it will automatically append a CRC to the end of the outgoing message.

The DLCMD2 will automatically retry a transmission if it lost arbitration or any errors were detected. The auto retry feature causes the DLCMD2 to indicate that the TxFIFO is full until the message is successfully sent except where no last byte was indicated. As soon as the CPU transfers a “last byte” of a message to the DLCMD2, or fills the eleventh position of the TxFIFO, the DLCMD2 will indicate that the TxFIFO is full. In the case of filling the TxFIFO with no “last byte” indicated (block mode) the status will say “TxFIFO almost full” after a byte has been sent so that the Data Link Controller Module (DLCMD2) next byte of the block message can be loaded by the CPU. Once successfully sent, the DLCMD2 will indicate that the TxFIFO is empty and hence ready for the next message. Each time the message loses arbitration, the completion code

for that received message will indicate to the transmitter that it attempted transmission, and lost arbitration. The auto retry feature can be terminated by the CPU through the command byte. This causes the DLCMD2 to finish its current transmit activity, and then clear the TxFIFO. If there had not been any transmit activity when the auto retry is disabled, the DLCMD2 will attempt to transmit the message once and then clear the TxFIFO.

A break waveform on the bus is shaped such that it will win arbitration against any currently transmitting message. When the DLCMD2 senses that such a waveform has occurred on the bus, it will stop transmitting its current message, reset its transmitter (clear the TxFIFO), and set a bit in the completion code and assert an interrupt to indicate to the CPU that such a condition has occurred. Since the break signal wins arbitration all of the time, any in progress messages will have simply lost arbitration, and the DLCMD2 will treat the in progress received message as complete. If a break occurs while there are no in progress messages, a completion code indicating that a break has occurred will be placed in the RxFIFO and a CPU interrupt generated. The break/reset waveform is sent by a DLCMD2 by setting a bit combination in the command byte. This waveform is sent immediately upon the command byte being latched in. If there is a current transmission, it will be stopped, and the break waveform sent. Break automatically takes all nodes on the network out of 4X speed mode.

NOTE

An SOF from a node in regular mode will be seen as a break by any nodes in 4X mode.

11.10.3 Receiving a Message

The status byte that the CPU reads from the DLCMD2 contains information on the status of the RxFIFO, the status of the TxFIFO, the condition of the bus, and the type of accompanying data byte. The data accompanying the status byte can be data received off of the J1850 busa completion code which contains information about a received message, or nothing.

All messages received off of the J1850 bus will have their start bit removed and error detection code replaced with a completion code. All other bytes of the message are placed in order, MSB first, in the RxFIFO.

Receiving information off of the J1850 bus occurs in much the same manner as sending data.

NOTE

By definition, every message a DLCMD2 sends on the data link is also received by its own receiver as if the transmission had been initiated by a different node.

The DLCMD2 will automatically remove the SOF bit from the message, and check the CRC for errors. If a CRC error occurs, it is flagged in the message completion infor-

mation (completion code) that takes the place of the CRC byte in the RxFIFO. The DLCMD2 will interrupt the CPU to signal that a complete message has been received, or when the RxFIFO is approaching full. When the CPU starts the transfer process, the status byte from the DLCMD2 indicates the data's presence, and something about the amount of data left in the RxFIFO. Conditions necessary for interrupting the CPU are selectable with a write to the interrupt level register (ILR). When a DLCMD2 loses arbitration to another node, it will continue to receive the remainder of the message. The completion code will indicate that the DLCMD2 contended for a transmit slot, and lost arbitration to the received message. The CPU does not need to resend the message if the auto retry feature is enabled.

11.10.4 Receiving a Message in Block Mode

Although not a part of the SAE J1850 protocol, the DLCMD2 does allow for a special "block mode" of operation for the receiver. As far as the DLCMD2 is concerned, a block mode message is simply a long J1850 frame that contains an indefinite number of data bytes. All of the other features of the frame remain the same, including the SOF, CRC, and EOD symbols.

Data link controller module (DLCMD2) class 2 convention requires that another node wishing to send a block mode transmission must first inform all other nodes on the network that this is about to happen. This is usually accomplished by sending a special predefined message.

Block mode may be combined with 4X mode.

11.10.5 Transmitting a Message in Block Mode

If the TxFIFO was filled with no last byte indicated, the DLCMD2 will start sending that message in terminate auto-retry mode; the status will indicate "TxFIFO almost full" as soon as the first byte is sent, and "TxFIFO contains some data bytes" as soon as the second byte is sent. As new data is written to the TxFIFO, the status will accurately reflect the condition of the TxFIFO until a "last byte" is written. When a "last byte" command has been sent to the DLCMD2, the TxFIFO will indicate "TxFIFO full" until the transmission is finished. The DLCMD2 will send an infinite length message if properly handled by the CPU.

Block mode transmit may be combined with 4X mode.

11.10.6 Receiving a Message in 4X Mode

Although not a part of the SAE J1850 protocol, the DLCMD2 does allow for a special "4x mode" of operation for the receiver. 4X mode is entered by the programmer setting the 4X bit in the MCR register. This bit is cleared automatically by a BREAK symbol reception, or may be manually cleared by the programmer.

11.10.7 Transmitting a Message in 4x Mode

If the programmer sets the 4X mode bit in the MCR register, the DLCMD2 will use a different set of VPW timing numbers to set the widths of the J1850 bus symbols when

transmitting. This bit is cleared automatically by a BREAK symbol reception, or it may be cleared manually by the programmer.

11.11 Register Descriptions

The following section provides register descriptions for the DLCMD2. In the following paragraphs, the top line lists the bit number in the register. The second line contains the mnemonic for the bit. The values shown under the mnemonic are the values of those register bits after IMSTRSTB is asserted. If ISYSRSTB affects the bits differently than IMSTRSTB, this fact is discussed in the paragraphs following the chart.

The DLCMD2 registers hold, in some cases, bit locations marked as “reserved”. These bits will always read as logic ‘0’ and writes to these bits are ignored.

The register decode map is fixed and begins at the first address of the module base address. **Table 11-5** shows the registers associated with the DLCMD2 module and their relative offset from the base address. Four of the registers are in supervisor-only data space and the remainder are in assignable data space.

Table 11-5 DLCMD2 Memory Map

Access	Offset	15	0	R/W	Reset
S	0xYF F600	Module Configuration Register (MCR)		R/W	0x6780
S	0xYF F602	Reserved		—	—
S	0xYF F604	Interrupt Pending Register (IPR)		R/W	0x0000
S	0xYF F606/ 0XYF F607	Interupt Level Register (ILR)	Interrupt Vector Register (IVR)	R/W, RO (bit2~bit0)	0x000f(IRQ_PLUG=0) 0x000f(IRQ_PLUG=1)
S/U	0xYF F608	Symbol Timing Control and Pre-scaler Register (SCTL)		R/W	0x0000
S/U	0xYF F60A	Symbol Timing Data Register (SDATA)		WO	0x00xx
S/U	0xYF F60C	Command Register (CMD)	Transmit Data Register (TDATA)	R/W-WO for (TDATA)	0x00xx
S/U	0xYF F60E/ 0XYF F60D	Status Register (STAT)	Receive Data Register (RDATA)	RO	0x00xx

11.11.1 Module Configuration Register (MCR)

MCR— Module Configuration Register **0xYF F600**

MSB 15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	LSB 0
STOP	FRZ		DSAE	4XMD	SOFT_FRZ	NOT_RDY	FREEZ_ACK	SUPV	0	0	STOP_ACK	IARB			
RESET:															
0			0	0	1	1	1	1	0	0	0	0	0	0	0

= Unimplemented

Table 11-6 MCR Bit Descriptions

Bit(s)	Name	Description
15	STOP	Stop system clock. The STOP bit, if asserted, will stop the system clock within the DLCMD2 module except for the IMB bus interface (BIU). The module's BIU must continue to operate to allow the CPU access to the module's registers (except for LPSTOP). The system clock is stopped on the low phase of ICLOCK. Once the bus becomes idle, the system clock will remain stopped until the STOP bit is negated by the CPU, or a reset occurs, or an edge from the J1850 bus passes through the digital filter (if DSAE not set).
14:13	FRZ	Freeze bit. There exist two bits in the DLCMD2 module control register to determine the action to be taken when the freeze signal of the IMB is asserted. **Table 11-7** defines the freeze bit field
12	DSAE	Disable STOP mode automatic exit. When asserted, the DSAE bit will prevent J1850 activity from causing the DLCMD2 to exit STOP mode, and will prevent INTACL2-type interrupts from being asserted. (When DSAE is negated, the DLCMD2 will automatically restart its internal clocks from STOP mode upon sensing any J1850 activity, and INTACL2-type interrupts will be allowed (although they must still be explicitly enabled by INTACL2E).) The CPU must negate the STOP bit to exit DLCMD2 STOP mode and restart the DLCMD2 clocks, since J1850 activity will not take the DLCMD2 out of STOP mode. When cleared, any J1850 activity that passes through the digital filter will take the DLCMD2 out of STOP mode and clear the STOP bit (if set).
11	4XMD	4X mode. When the 4XMD bit is asserted, the DLCMD2 will use 4X mode bit timings rather than the normal J1850 symbol timings. Note that normal waveshaping by the analog transceiver (whether on-chip or off-chip) must be disabled for the DLCMD2/transceiver combination to transmit properly in 4X mode. This bit is automatically reset upon receipt of a BREAK symbol.
10	SOFT_ FRZ	Software freeze. Assertion of this bit has the same effect as the assertion of the IFREEZEB signal on the IMB3, as described in the description of IFREEZEB/SOFT_FRZ mode and FRZE bits. However, it does not require that the IFREEZBE signal be asserted in order to enter debug mode. This bit is initialized to 1 (debug mode).The CPU clears it after initializing the control registers. No transmissions or receptions are performed by the DLCMD2 before this bit is cleared. For detailed description of the debug mode, refer to **11.8.5 DLCMD2 DEBUG**. 0 = No DLCMD2 internal request to enter Debug Mode. 1 = Enter Debug mode if (FRZE[1] || FRZE[0])
9	NOT_ RDY	Not ready. This bit indicates that the DLCMD2 is either in STOP or in DEBUG state. This bit is read-only. Whenever one of these two modes is asserted, this bit is set once the DLCMD2 has entered the corresponding mode. It is negated once the DLCMD2 has exited these modes. For more details refer to **11.8.4 DLCMD2 STOP and LPSTOP** and LPSTOP and **11.8.5 DLCMD2 DEBUG**.
8	FREEZ_ ACK	DLCMD2 disabled. DLCMD2 is in debug mode and its pre-scaler is stopped. This bit is read-only. When the DLCMD2 enters debug mode it sets the FREEZ_ACK bit. The CPU can poll this bit to know if the DLCMD2 entered debug mode. If debug mode is negated then this bit is negated once the DLCMD2's pre-scaler is running. 0 = Debug mode is negated and the pre-scaler is running again 1 = The DLCMD2's pre-scaler is stopped
7	SUPV	Supervisor/user data space. The SUPV bit affects the SCTL/SDATA, CMD/TDATA and STAT/RDATA registers, the only registers in the DLCMD2 currently defined as supervisor/unrestricted access. If SUPV is asserted (all DLCMD2 registers are to be treated as supervisor only) bit 2 of the function code (FC2) must be asserted during module address decoding to allow the supervisor/unrestricted access registers to respond to accesses in supervisor data space. If the SUPV bit is cleared (SCTL/SDATA, CMD/TDATA and STAT/RDATA accesses are to be treated as unrestricted) FC2 is ignored during module address decoding, allowing supervisor/unrestricted access registers to respond to accesses in supervisor data or user data space.
6:5	—	Reserved

Table 11-6 MCR Bit Descriptions (Continued)

Bit(s)	Name	Description
4	STOP_ ACK	STOP acknowledge. DLCMD2 is in STOP mode and its main clocks are stopped, (see **11.8.4 DLCMD2 STOP and LPSTOP**) This bit is read-only. When the DLCMD2 enters STOP mode and shuts its clocks off, it sets the STOP_ACK bit. The CPU can poll this bit to know if the DLCMD2 entered stop mode (e.g., stopped its clocks). If stop mode bit is negated, then this bit is negated once the DLCMD2's clocks are running. 0 = STOP mode is negated and the DLCMD2's clocks are running again 1 = The DLCMD2 enters STOP Mode and its clocks are stopped
3:0	IARB	Interrupt arbitration. This four-bit encoded field contains the interrupt arbitration number of this particular DLCMD2 module with respect to all other subsystems and peripherals that may generate interrupts. The interrupt arbitration ID is used to arbitrate the IMB3 (relevant only when IRQ_PLUG = 0) when two or module/peripherals have an interrupt on the same priority level pending simultaneously. This field is initialized to the non-arbitrating state, $0 during reset. If no arbitration takes place during the IACK cycle, the spurious interrupt vector is generated after a time-out by the SIM module, alerting the system that an interrupt arbitration ID has not been initialized. The IARB field should be initialized by the system software to a value between 0xF (highest priority) and 0x1 (lowest priority).

Table 11-7 Freeze Bit Field Description

FRZ1	FRZ0	Result
0	0	Ignore FREEZE
0	1	Freeze on DLCMD2 internal f_{DLC} high state. (CTWO is in high state when IFREEZEB is asserted, or f_{DLC} enters its high state sometime after IFREEZEB is asserted.)
1	0	Freeze on receipt of next bit. (DLCMD2 internal RIT signal is asserted IFREEZEB is asserted, or RBIT becomes asserted sometime after IFREEZEB is asserted.)
1	1	Freeze immediately

11.11.2 Interrupt Pending Register (IPR)

IPR — Interrupt Pending Register **0xYF F604**

MSB 15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	LSB 0
0	0	0	0	0	0	0	0	0	0	0	ipr_4	ipr_3	ipr_2	ipr_1	ipr_0
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

= Unimplemented

Table 11-8 IPR Bit Descriptions

Bit(s)	Name	Description
15:5	—	Reserved
4:0	IPR[4:0]	Interrupt pending. The interrupt pending register (IPR) indicates that an interrupt service request has been made by the module interrupt logic. An interrupt request to the IMB3 is generated whenever the interrupt asserting condition is met. A bit in the IP register (indicating the application logic interrupt request is detected) can be set in any order to generate the IMB3 interrupt request. See **Figure 11-17** for the interrupt request logic path. Once set, the IP bits remain set until the IP bit is cleared by software, or reset. To clear an IP bit, the bit must be first read as a 1 and then the bit must be written to a 0. IP bits which are 0 when the IP register is read are unaffected by the write operation. Also, if the IP logic detects another application logic interrupt request after the IP bit was read as a 1 and before a 0 is written to clear it, the IP bit cannot be cleared until the IP bit is again read as a 1 and then written to a 0. Bits in this register are set by the application logic request and cannot be written to a one by software (writing 1 to the IP register have no effect). Only writes of 0 are valid, when permitted, to clear the IP bit(s). IP_0 : R1STBYTE interrupt pending setsIP[0]. IP_1 : RCCODE interrupt pending sets IP[1]. IP_2 : R13BYTE interrupt pending sets IP[2]. IP_3 : THLFMTY interrupt pending sets IP[3]. IP_4 : ACL2 interrupt pending sets IP[4].

11.11.3 Interrupt Level Register (ILR)

ILR — Interrupt Level Register **0xYF F606**

MSB 15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	LSB 0
INTMODE	INTACL2E	RESERVED			ILR			INTERRUPT VECTOR REGISTER (IVR)							
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

= Unimplemented

Table 11-9 ILR Bit Descriptions

Bit(s)	Name	Description
15	INTMODE	Interrupt mode. When the INTMODE bit is asserted, the DLCMD2 will assert an interrupt when a single byte is received into an empty RxFIFO. When cleared, only standard interrupts are enabled.
14	INTACL2E	Interrupt any bus activity enable. When the INTACL2E bit is set, the DLCMD2 will assert an interrupt when any network activity (including noise) is detected. This bit must be set for the DLCMD2 to wake up the processor from LPSTOP. Although the detected activity may be only noise, neither IMB3 nor module clocks can run in LPSTOP, therefore the DLCMD2 cannot distinguish noise from valid network activity, and must wake up the processor to restart the clocks necessary for J1850 message reception. Normally, this bit would be set by the CPU just before going into LPSTOP, and is cleared once the ACL2 interrupt condition is latched.
13:11	—	Reserved

Table 11-9 ILR Bit Descriptions (Continued)

Bit(s)	Name	Description
10:8	ILR	Interrupt request level. The interrupt request level field contains the priority level of the DLCMD2 interrupts for the CPU. Level seven for this field indicates a nonmaskable interrupt, while level zero indicates that interrupts have been disabled. The interrupt request level field is initialized to zero during reset which prevents the module from generating an interrupt until this register has been initialized. Note: level zero corresponds to IRQ[0] which is not recognized at the system level, hence the interrupts are treated as disabled. The interrupt request level field, therefore acts as master enable for the interrupts.
7:0	IVR	Interrupt vector register. The interrupt vector register holds the offset into the exception vector table, and is what is driven by the DLCMD2 in response to an IMB IACK cycle. IVR[7:3] are programmable by the user. IVR are read-only bits and encoded from the highest priority of any currently active interrupt sources per **Table 11-34**. All reads of the IVR register will return $0F, the defined state for an uninitialized interrupt vector, until a write access to this register occurs. Therefore IVR[2:0] would not be updated from DLCMD2's internal interrupt status until IVR is written once by the software when IRQ_PLUG =0..

Table 11-10 IVR Encoding

ACL2[1]	THLFMTY[2]	R13BYTE[3]	RCCODE[4]	R1STBYTE[5]	IVR2	IVR1	IVR0
A[6]	x	x	x	x	1	0	1
N[7]	A	x	x	x	1	0	0
N	N	A	x	x	0	1	1
N	N	N	A	x	0	1	0
N	N	N	N	A	0	0	1
N	N	N	N	N	0	0	0

NOTES:
1. ACL2 = Any J1850 activity, DSAE bit in MCR register is cleared and INTACL2E in ILR is set.
2. THLFMTY = TxFIFO half empty.
3. R13BYTE = RxFIFO has 13 bytes and no completion code.
4. RCCODE = RxFIFO has a complete message.
5. R1STBYTE = RxFIFO has one data byte and nothing else.
6. A = Assert interrupt.
7. N =No interrupt for this event.

11.11.4 Symbol Timing Control and Pre-Scaler Register (SCTL)

SCTL — Symbol Timing Control and Pre-Scaler Register **0xYF F608**

MSB 15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	LSB 0
0	NBFS	RXPOL	0	0	0	LCK	SEL	0	0	PS5	PS4	PS3	PS2	PS1	PS0
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

= Unimplemented

Table 11-11 SCTL Bit Descriptions

Bit(s)	Name	Description
15	—	Reserved
14	NBFS	Normalization bit (NB) format. This bit controls the format of normalization bit (NB). SAE J1850 strongly encourages the use of an active long, ‘0’, for in-frame responses containing CRC and active short, ‘1’, for in-frame response without CRC. Once the LCK is set, the writes to NBFS are disabled. 0 = NBFS, NB that is received or transmitted is a ‘1’ when the response part of an in-frame response (IFR) ends with a CRC byte. NB that is received or transmitted is a ‘0’ when the response part of an in-frame response (IFR) does not end with a CRC byte. 1 = NBFS, NB that is received or transmitted is a ‘0’ when the response part of an in-frame response (IFR) ends with a CRC byte. NB that is received or transmitted is a ‘1’ when the response part of an in-frame response (IFR) does not end with a CRC byte
13	RXPOL	Receive pin polarity. The receive pin polarity bit is used to select the polarity of incoming signal on the receive pin. Some external analog transceivers invert the receive signal from the J1850 bus before feeding back to the digital receive pin. Once the LCK is set, the writes to RXPOL are disabled. 0 = RXPOL, select normal/true polarity; true non-inverted signal from J1850 bus, (i.e., the external transceiver does not invert the receive signal). 1 = RXPOL, select inverted polarity, where external transceiver inverts the receive signal
12:10	—	Reserved
9	LCK	Write lock on symbol timing parameter table. LCK disables writes to the symbol timing parameter table through SDATA. Once LCK is set, only reset will clear the bit (unless in test mode). 0 = LCK, enables the writes to the symbol timing parameter table through SDATA register. 1 = LCK, disables the writes to SEL, NBFS, RXPOL, PS5 -PS0 in SCTL register and the symbol timing parameter table through SDATA register. Once LCK is set, only reset will clear this bit (unless in test mode).
8	SEL	Select normal or 4X bit-timing parameter table. There are two sets of parameters used for DLMCD2 bit-timing. These two sets of parameters are accessible through the SDATA register. There are twelve parameters for normal bit-timing (four for receiving and eight for transmitting) and twelve parameters for 4X bit-timing (four for receiving and eight for transmitting). These bits may be written only when LCK is cleared. Once the LCK is set, the writes to SEL are disabled and SEL is cleared. 0 = SEL, allows access (write) to the normal mode parameters. 1 = SEL, allows access (write) to the 4X mode parameters
7:6	—	Reserved
5:0	PS[5:0]	Pre-scaler rate select. Set the system clock divisor necessary to achieve the DLCMD2 internal bit-rate clock. The frequency should be as close to 2 MHz as possible. These bits may be written only when LCK is cleared. Once the LCK is set, the writes to PS5-PS0 are disabled. The value programmed into PS5-PS0 bits is dependent on the chosen IMB3 system clock frequency per **Table 11-12**. Note: The PS5-PS0 is always loaded with “desired divisor” -1 and comes out of reset programmed for a divided by one clock rate (PS5-PS0 = 0x00)

Table 11-12 DLCMD2 Pre-Scaler Rate Selection

IMB3 Bus Clock Frequency	PS5-PS0	Division	f_{DLC}
f_{CLOCK} = 2.000 MHz	0x00	1	2.000 MHz
f_{CLOCK} = 15.000 MHz	0x06	7	2.143 MHz
f_{CLOCK} = 16.000 MHz	0x07	8	2.0 MHz
f_{CLOCK} = 25.000 MHz	0x0B	12	2.083 MHz
f_{CLOCK} = 26.000 MHz	0x0C	13	2.0 MHz
f_{CLOCK} = 40.000 MHz	0x13	20	2.0 MHz
f_{CLOCK} = 45.000 MHz	0x15	22	2.045 MHz
f_{CLOCK} = 66.000 MHz	0x20	33	2.0 MHz
f_{CLOCK} = 128.000 MHz	0X3F	64	2.0 MHz

11.11.5 Symbol Timing Data Register (SDATA)

SDATA — Symbol Timing Data Register **0xYF F60A**

MSB 15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	LSB 0
0	0	0	0	0	S10	S9	S8	S7	S6	S5	S4	S3	S2	S1	S0
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

= Unimplemented

Table 11-13 SDATA Bit Descriptions

Bit(s)	Name	Description
15:11	—	Reserved
10:0	S[10:0]	Symbol timing data. The bit-timing of J1850 symbols is written into the twenty-four entries of the parameter table through the SDATA register. An internal pointer along with the SEL bit is used to select which parameter is accessed (write) through the SDATA register. This pointer is incremented when SDATA is written and cleared when the SCTL register is accessed. Refer to **Figure 11-17** for a block diagram and **Table 11-14** for the parameter table. The parameter table lists the cycle counts for four receive symbols and eight transmit symbols in normal and 4X modes. The user must calculate the cycle counts for each symbol based on the desired value, round trip delay, digital filter delay, and bus frequency. The calculated cycle counts must be entered into the parameter table through the SDATA. To calculate the cycle count for receive symbols, multiply f_{DLC} by the desired symbol time (in µs) and round to the nearest integer. To calculate the cycle count for transmit symbols, subtract the round trip delay for the transceiver from the desired symbol time (in µs), multiply by f_{DLC}, round to the nearest integer, and subtract the cycle count of the digital filter (16 cycles in normal mode, four cycles in 4X mode). As an example, let the transceiver round trip delay be 16 µs, the desired symbol be 64 µs, and fDLC be 2.083 MHz. The cycle count would be: cycle count = ((64 µs - 16 µs) x 2.083 cycles / µs) - 16 cycles = 84 cycles After the cycle count has been computed, it must be converted into binary format and entered into the respective parameter through the SDATA.

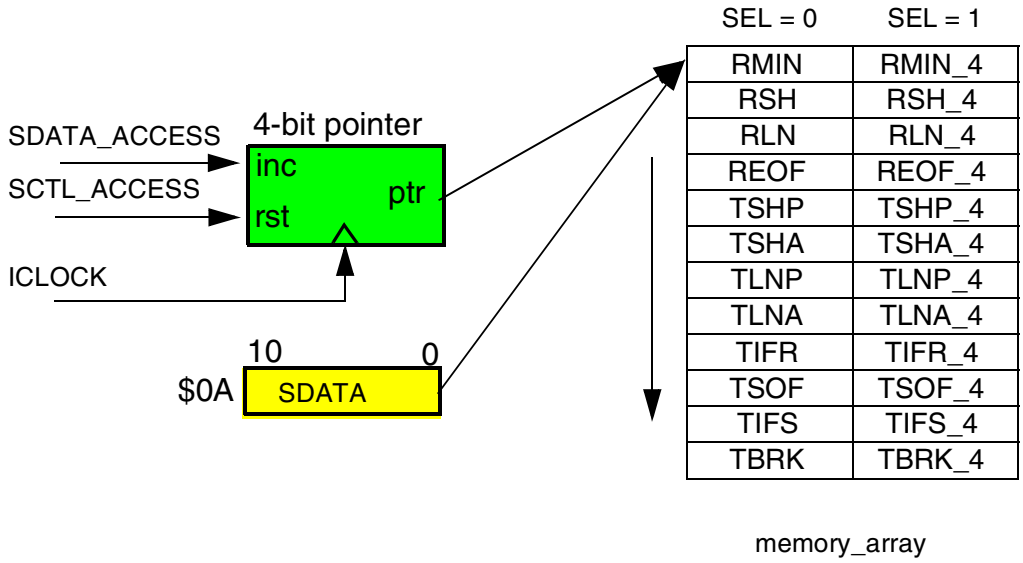

Figure 11-17 DLCMD2 SDATA Block Diagram

Table 11-14 Timing Parameter Table

Programming Sequence	Symbol Description	Symbol (SEL = 0)	Symbol Timing (µS)	Symbol (SEL = 1)	Symbol Timing 4X (µs)
1	Minimum receive symbol time	RMIN	34	RMIN_4	8
2	Maximum receive short pulse time	RSH	96	RSH_4	24
3	Maximum receive long pulse time	RLN	163	RLN_4	41
4	Minimum end-of-frame time	REOF	239	REOF_4	60
5	Target transmit short passive symbol time	TSHP	64	TSHP_4	16
6	Target transmit short active symbol time	TSHA	64	TSHA_4	16
7	Target transmit long passive symbol time	TLNP	128	TLNP_4	32
8	Target transmit long active symbol time	TLNA	128	TLNA_4	32
9	Target end-of-data time before IFR begins	TIFR	200	TIFR_4	50
10	Target transmit start-of-frame symbol time	TSOF	200	TSOF_4	50
11	Target inter-frame separation time before DLCMD2 is ready for new message	TIFS	320	TIFS_4	80
12	Target transmit break symbol time	TBRK	800	TBRK_4	800

11.11.6 Command Register (CMD)

Command and configuration of the DLCMD2 module is accomplished through a command byte that accompanies every data byte sent from the CPU to the DLCMD2 module.

NOTE

The command byte is the first byte transferred to the DLCMD2 module, and must be followed by a byte (data or dummy) which causes the command to be acted on (this can be done in an aligned word write or two separate byte writes).

CMD — Command Register **0xYF F60C**

MSB 15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	LSB 0
CMD								Transmit Data Register (TDATA)							
RESET:															
0	0	0	0	0	0	0	0								

Table 11-15 CMD Bit Descriptions

Bit(s)	Name	Description
15:8	CMD	Command byte. The command byte can be broken in three fields. Bits 7:5 describe general commands. Bits 4:3 describe the type and destination of the accompanying byte associated with the command. Bit 1:0 control the receive FIFO. Refer to **Table 11-16** for general command descriptions. Refer to **Table 11-17** for type and destination of accompanying byte descriptions. Refer to **Table 11-19** for RxFIFO command descriptions.
7:0	TDATA	Transmit data register. See **11.11.7 Transmit Data Register (TDATA)**.

Table 11-16 General Commands

CMD 7	CMD6	CMD5	Description
0	0	0	Do nothing — DLCMD2 will not perform any actions defined in this field.
0	0	1	Reserved — Results in “do nothing.”
0	1	0	Send Break signal — When this command is latched in, the DLCMD2 will immediately send a break signal on the bus, regardless of its current transmit or receive status.

Table 11-16 General Commands (Continued)

CMD 7	CMD6	CMD5	Description
0	1	1	Send on EOD with CRC (Transmit Single or Multiple Byte IFR with CRC) 1. Must be accompanied by a data byte that is flagged as “1st byte” or “1st and last” byte. The remaining IFR bytes may not be sent to the DLCMD2 with the “Send on EOD” combination set. 2. This command causes the DLCMD2 to go into Terminate Auto Retry (TAR) mode automatically. It also causes a reset of the transmitter and TxFIFO before the response byte is loaded. If the DLCMD2 is transmitting and loses arbitration, an IFR can be loaded in response to the message winning arbitration. 3. If this command is latched in while the bus is idle or between EOD and EOF, the IFR command/data bytes will be ignored. If valid command/data bytes follow the invalid IFR command/data bytes, the DLCMD2 will attempt to transmit the bytes as a normal message once. 4. If a message is in progress and the EOD symbol has not been detected when this command is latched in, the DLCMD2 will send the IFR if there were no receiver errors detected. If receiver errors were detected, the IFR will be lost and the DLCMD2 will not be in TAR mode. 5. If the remaining IFR bytes, if any are required, are not placed into the TxFIFO before they are needed, then an underrun will occur. This will cause the CRC to be inverted in the IFR. The second byte must be loaded before the falling edge of the Normalization Bit. The next IFR byte must be loaded before the last bit of the current byte is transmitted. Since there is only one responder transmitting an IFR, the inverted CRC will cause the DLCMD2 to corrupt its own message. Other nodes will receive this as a CRC error. 6. If this DLCMD2 is the one transmitting, send on EOD will cause a bit timing or incomplete byte error and a reset of the TxFIFO. Both the original message and the IFR will be lost. 7. Cannot be sent in response to an IFR without CRC.
1	0	0	Terminate Auto Retry (TAR) 1. If this command is latched in while there is not a complete message in the TxFIFO, then when that message is completely loaded the DLCMD2 will attempt to transmit that message only once. After this transmit attempt, the TxFIFO will be empty. 2. If this command is latched in while the DLC is transmitting, then the DLCMD2 will finish its transmit activity (successful transmission, or lose arbitration) and then clear the TxFIFO. 3. If a complete message is in the TxFIFO, but not yet transmitting when this command is latched in, and: a. The message has just lost arbitration and is waiting for the next slot for retransmission, the DLCMD2 will attempt transmission one more time. b. The message has not tried to transmit yet (loaded while a message was on the bus). The DLCMD2 will try to transmit once, and then reset the TxFIFO. c. If the TxFIFO is full, and there is no last byte indicated (message of more than 11 bytes), then an automatic TAR is executed.
1	0	1	Send on EOD without CRC (Transmit Single or Multiple Byte IFR w/o CRC) 1. Same as 0 1 1 but a CRC will not be transmitted. 2. This response cannot be sent after a previous IFR without CRC.

Table 11-16 General Commands (Continued)

CMD 7	CMD6	CMD5	Description
1	1	0	Send on EOD with auto re-try (Transmit Single Byte IFR with auto re-try) 1. If a loss of arbitration occurs when the DLCMD2 attempts to transmit and after the IFR byte winning arbitration completes transmission, the DLCMD2 will again attempt to transmit. The DLCMD2 will continue transmission attempts until an error is detected on the bus. 2. The TxFIFO will not be empty until the message is successfully sent.
1	1	1	Abort Transmission Now (Reset Transmitter) 1. This command causes the transmitter, including the TxFIFO to be reset immediately. If a message is in progress, it will be terminated immediately. 2. The first byte of a new message may accompany this command in the same 2-byte host-DLCMD2 transfer.

Table 11-17 Type and Destination of Accompanying Byte Commands

CMD 4	CMD 3	CMD 2	Description
0	0	0	Do Not Load as Transmit Data 1. The DLCMD2 will not perform any actions defined in this field.
0	0	1	Load as Transmit Data 1. If the TxFIFO is indicating full (complete message already in TxFIFO), then the byte is not loaded and is lost.
0	1	0	Reserved
0	1	1	Load as Last Byte of Transmit Data 1. If there is not a “1st byte” at the head of the TxFIFO, the byte is not loaded and is lost unless a block transfer is in progress. 2. If the TxFIFO is empty, the byte is not loaded and is lost. 3. If the TxFIFO is indicating full (complete message already in TxFIFO), then the byte is not loaded and is lost.
1	0	0	Reserved
1	0	1	Load as First Byte of Message 1. If there is already a first byte in the TxFIFO, the byte is loaded, and will cause a transmit underrun error during transmission of this message. The first “first byte” and subsequent bytes will transmit correctly until the second “first byte” is encountered and this byte is not transmitted and is lost. 2. If the status of the TxFIFO is full (complete message already loaded, or partial message < 11 bytes), then the data byte is not loaded, and is lost.
1	1	0	Reserved
1	1	1	Load as First and Last Byte of Message 1. If the TxFIFO is full, then the byte is not loaded and is lost.

11.11.7 Transmit Data Register (TDATA)

TDATA — Transmit Data Register **0xYF F60C**

MSB 15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	LSB 0
CMD								Transmit Data Register (TDATA)							
RESET:								0	0	0	0	0	0	0	0

Table 11-18 TDATA Bit Descriptions

Bit(s)	Name	Description
15:8	CMD	Command byte. See **11.11.6 Command Register (CMD)**.
7:0	TDATA	Transmit data register. This register is used to load data into the TxFIFO. Data is automatically pushed into the TxFIFO on every write to the TDATA register. See **Figure 11-18**.

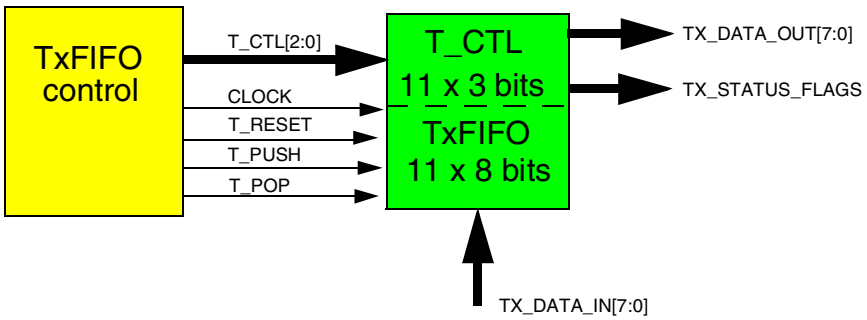

Figure 11-18 TxFIFO Block Diagram

Table 11-19 RxFIFO Commands

CMD 1	CMD 0	Description
0	0	Do Nothing
0	1	Reserved (Do nothing)
1	0	Flush Byte 1. If there is nothing in the RxFIFO, then no action is taken even if a byte arrives before the DLCMD2 access is complete. 2. If there are any bytes in the RxFIFO, then the first byte (at the head of the RxFIFO) is flushed. 3. Flush byte commands will not queue up.
1	1	Flush Current Message[1] 1. If there is nothing in the RxFIFO when this command is latched in, the next message received will be flushed except for the completion code. Only the completion code will cause an interrupt. 2. If there is a partial message in the RxFIFO (still being received off of the data link), all remaining bytes of the message will be flushed, except for the completion code, when it is formed. Only the completion code will cause an interrupt. 3. If there is a complete message in the RxFIFO (message bytes and completion code), all bytes of the message will be flushed except for the completion code. 4. If there is completion code at the head of the RxFIFO, no action is taken when this command is latched in.

NOTE:
1. Flush message commands cannot be queued up. Only one dump command is ever active. A Flush mes-sage command can not be stopped after being issued. In cases 2 and 3, the RxFIFO status may be invalid during a flush message operation. It could take a few þs (or even longer if there is no completion code in the RxFIFO yet) for the status to correctly indicate the condition of the RxFIFO. In cases 1 and 2, the RxFIFO status invalid time varies with how much of the message is yet to be received, (i.e., the completion code must “leak down” to the head of the RxFIFO before the RxFIFO is “valid” again). The RxFIFO is not necessarily empty after a flush message command.

11.11.8 TxFIFO Command Load Sequences

The following table shows various load sequences for the CMD and TDATA registers. The table indicates whether the sequence is valid or invalid and how the DLCMD2 will handle each case.

Table 11-20 Command Load Sequences

Load Sequence	DLCMD2 operation
1) Write “Load as First Byte” and data byte as 16-bit write. 2) Write “Load as Transmit Data Byte” and data byte as 16-bit write. 3) Write “Load as Last Byte” and data byte as 16-bit write.	All three command and data bytes are pushed into the TxFIFO. DLCMD2 transmits message correctly. “Load as Last Byte” command and last data byte can be written with two separate 8-bit writes.
1) Write data bytes as 8-bit writes. 2) Write “Load as Last Byte” and last data byte as 16-bit write.	First data byte treated as “first byte.” “Load as Last Byte” and last data byte is pushed into the TxFIFO. DLCMD2 transmits message correctly. “Load as Last Byte” command and last data byte can be written with two separate 8-bit writes.
1) Write “Send on EOD” as 8-bit write. 2) Write data bytes as 8-bit writes. 3) Write “Load as Last Byte” and last byte as 16-bit write.	No IFR transmission occurs. “Send on EOD” command will be ignored and first data byte will be treated as “first byte.” If EOF is received, the DLCMD2 will transmit the bytes as a normal message.
1) Write “Send on EOD” as 8-bit write. 2) Write “Load as First Byte” and data byte as 16-bit write. 3) Write data bytes as 8-bit writes. 4) Write “Load as Last Byte” and data byte as 16-bit write.	No IFR transmission occurs. “Send on EOD” command will be ignored and first data byte will be treated as “first byte.” If EOF is received, the DLCMD2 will transmit the bytes as a normal message.
1) Write “Send on EOD,” “Load as First Byte,” and data byte as 16-bit write. 2) Write data bytes as 8-bit writes. 3) Write “Load as Last Byte” and last data byte as 16-bit write.	Command and data word is pushed into the TxFIFO. Data bytes are pushed into the TxFIFO. Command and data byte is pushed into the TxFIFO. DLCMD2 transmits an IFR correctly.
1) Write “Load as First Byte” as 8-bit write. 2) Write “Send on EOD” as 8-bit write. 3) Write data bytes as 8-bit writes. 4) Write “Load as Last Byte” and data byte as 16-bit write.	No IFR transmission occurs. “Send on EOD” command will be ignored and first data byte will be treated as “first byte.” If EOF is received, the DLCMD2 will transmit the bytes as a normal message.
1) Write “Send on EOD,” “Load as First Byte,” and data byte as 16-bit write. 2) Write “Send on EOD,” “Load as transmit data byte,” and data byte as 16-bit write. 3) Write “Send on EOD,” Load as Last Byte,” and data byte as 16-bit write.	No IFR transmission occurs. Data bytes other than the first byte of an IFR cannot be written with “Send on EOD.” All data bytes are ignored.

NOTE

Any load sequence other than sequences 1-5 will not result in DLCMD2 transmission activity.

11.11.9 Status Register (STAT)

Each byte of data that the DLCMD2 module has received from the J1850 bus will be transferred to the CPU along with a status byte, which relays information on the condition of the DLCMD2 module and the data that has been received.

NOTE

Commands that are in progress may not be reflected in the status byte until sufficient time has elapsed for that command to be performed. The status must be read with each received data byte on the DLCMD2 module, either in an aligned word read, or two separate byte reads.

STAT — Status Register **0xYF F60E**

MSB 15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	LSB 0
STAT								Receive Data Register (RDATA)							
RESET:															
0	0	0	0	0	0	0	0								

Table 11-21 STAT Bit Descriptions

Bit(s)	Name	Description
15:8	STAT	Status byte. The status byte can be broken in five fields. Bits 7:5 describe the status of the RxFIFO. Bits 4 indicates whether the bus is idle. Bit 3 indicates whether the bus is shorted to ground. Bit 2 indicates whether the TxFIFO is underrunning. Bit 1:0 describes the status of the TxFIFO. Refer to **Table 11-22** for RxFIFO status descriptions. Refer to **Table 11-23** for data link idle status descriptions. Refer to **Table 11-24** for transmitter shorted status descriptions. Refer to **Table 11-25** for TxFIFO underrunning status descriptions. Refer to **Table 11-26** for TxFIFO status descriptions
7:0	RDATA	Receive data register. See **11.11.10 Receive Data Register (RDATA)**.

Table 11-22 RxFIFO Status

CMD 7	CMD 6	CMD 5	Description
0	0	0	RxFIFO Invalid or Empty. 1. Read of accompanying data byte is not valid. 2. RxFIFO invalid (flush message command in progress).
0	0	1	RxFIFO Contains More Than One Byte. 1. RxFIFO has between 2-12 bytes of data at the DLCMD2 access and no completion code.
0	1	0	RxFIFO Contains a Completion Code. 1. Indicates the presence of a completion code in the RxFIFO (not at head of RxFIFO) at the DLCMD2 access. There is no other data also in the RsFIFO.
0	1	1	Receive Data Byte in Postion 13 and No Completion Code in RxFIFO. 1. Position 13 in the RxFIFO is filled and no completion code is present.
1	0	0	RxFIFO Contains Exactly One Byte. 1. RxFIFO contains exactly one data byte (not a completion code).

Table 11-22 RxFIFO Status (Continued)

CMD 7	CMD 6	CMD 5	Description
1	0	1	Completion Code at Head of RxFIFO, More Bytes Available 1. RxFIFO has a completion code at the head, additional bytes from an in-progress message and no other completion codes.
1	1	0	Completion Code at Head of RxFIFO, Another Complete Message Available. 1. RxFIFO has a completion code at the head and another completed message (completion code presetn) in the RxFIFO.
1	1	1	Completion Code on at Head of FIFO. 1. RxFIFO has a completion code at the head and it is the only byte in the RxFIFO.

Table 11-23 Data Link Idle Status

STAT 4	Description
0	Data Link is Busy 1. A high level on the bus has been detected for more than 8 µs, or an EOF symbol is bing timed by the receive logic. Once bit 4 is a zero because of message activity it will not become one until an EOF symbol has been received. Noise on an idle line could cause the idle bit to change state as the idle bit is unfiltered. The order of events after a transmission is: EOD —> comp. code pushed —> interupt —> EOF (idle).
1	Data Link is Idle. 1. This is the point that a transmitter may begin accessing the bus.

Table 11-24 Transmitter Shorted Status

STAT 3	Description
0	Data Link is Not Shorted to Ground
1	Data Link is Shorted to Ground[1] 1. When a hardware fault prevents the active level from being sensed after the transmitter has driven the bus for 60 µs the “data link shorted” bit is set. The only way to clear this error and try again is to reset the transmitter with the DLCMD2 Command byte or a master reset of the system reset pin.

NOTE:

1. If the problem is a momentary short on the data link, it is appropriate to try reloading and transmitting the message again. If the problem is a defective line receiver then a retry could corrupt another mes-sage. The CPU must be able to handle this condition and execute a graceful recovery routine. A short to 12V will be sensed as a “data link is busy” condition. If the short to high is longer than the minimum time for a BREAK signal a completion code indicating a BREAK will be pushed into the RxFIFO after the short goes away.

Table 11-25 TxFIFO Underrunning Status

STAT 2	Description
0	The DLCMD2 TxFIFO in Not Underrunning
1	The DLCMD2 TxFIFO is Underrunning (TxFIFO is Half Empty, No Last Byte Indicated) 1. Cleared automatically following a read of the status register. 2. Actual TxFIFO underrun event (if it occurs) will be indicated in the completion code.

Table 11-26 TxFIFO Status

STAT 1	STAT 0	Description
0	0	TxFIFO is Empty
0	1	TxFIFO Contains Some Data Bytes. 1. TxFIFO contains between one and nine data bytes.
1	0	TxFIFO Almost Full. 1. TxFIFO contains 10 bytes (one byte left empty).
1	1	TxFIFO Full. 1. TxFIFO contains a valid “last byte” or it contains 11 bytes and no “last byte” (see Block Mode section).

11.11.10 Receive Data Register (RDATA)

RDATA— Receive Data Register 0xYF F60D

MSB 15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	LSB 0
Status Byte Register (STAT)								RDATA							
RESET:															
0	0	0	0	0	0	0	0								

Table 11-27 RDATA Bit Descriptions

Bit(s)	Name	Description
15:8	STAT	Status byte. See **11.11.9 Status Register (STAT)**.
7:0	RDATA	Receive data register. The receive data register is used to read data from the RxFIFO. Data is popped from the RxFIFO on every read of the RDATA register. See **Figure 11-19**.

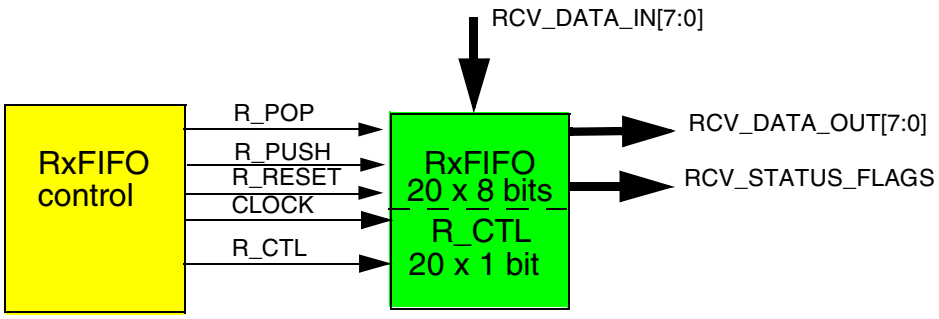

Figure 11-19 RxFIFO Block Diagram

11.11.11 Completion Code

When the message that is being received is complete, that is, after EOD has elapsed since the last transition on the bus, the DLCMD2 will prepare a completion code, which is a one byte descriptor with information about the message. If an error (CRC error, incomplete byte, bit timing error, underrun, and loss of arbitration) has occurred, the DLCMD2 will push a completion code into the RxFIFO after EOD has elapsed. If a BREAK symbol has been received, the DLCMD2 will push a completion code into the RxFIFO as soon as the bus is passive. Refer to **Figure 11-20**. This byte will be queued and transferred to the host in the same manner that a data byte is. The status byte will flag its presence.

NOTE

Any activity on the bus (i.e., noise pulses greater than 8 µs, messages, truncated messages, bad messages, etc.) will cause a completion code to be pushed into the RxFIFO after the bus is idle for an EOD length of time. In the case of noise, the completion code will not be pushed until the noise train is over for at least an EOD period of time and there will be only one completion code for all of the noise.

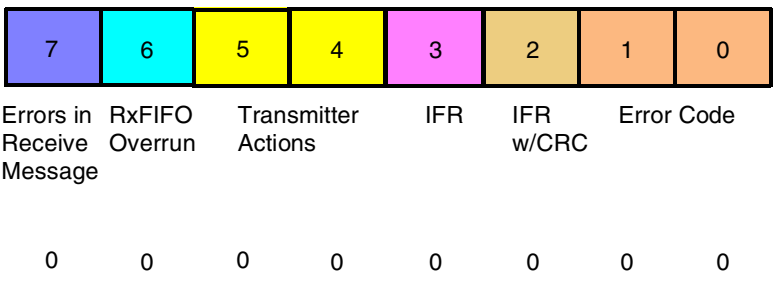

Figure 11-20 Completion Code Byte Bit Definitions

Refer to **Table 11-28** for receive error descriptions. Refer to **Table 11-29** for RxFIFO overrun descriptions. Refer to **Table 11-30** for transmitter action descriptions. Refer to

Table 11-25 for TxFIFO underrunning status descriptions. Refer to Table 11-30 for IFR bit descriptions. Refer to Table 11-31 for IFR with/without a CRC bit descriptions. Refer to Table 11-32 for error code descriptions.

Table 11-28 Receive Error Status

Bit 7	Description
0	No Error Detected
1	Error(s) Detected 1. When set, certain errors occurred in the reception of this message (see description for bits 1 and 0). If not set, then bits 1 and 0 are also 0.

Table 11-29 RxFIFO Overrun Status

Bit 6	Description
0	No Overrun
1	Overrun 1. When set, a receiver RxFIFO overrun has occurred. It will be set as soon as the 19th data byte has been placed in the FIFO (20th position is always completion code). The host has not read out the DLCMD2 regularly enough to prevent the incoming message bytes from being lost. Previously received data bytes remain in the RxFIFO. The host must make a decision to disregard the partial message and perhaps transmit a request for retransmission of the message. If this bit is set no other bits in the completion code may be taken as valid.

Table 11-30 Transmitter Action Status

Bit 5	Bit 4	Description
0	0	Transmitter Not Involved 1. The transmitter did not attempt to contend against this message.
0	1	Transmitter Underrun 1. This message was not completed (not set if transmitter lost arbitration).
1	0	Transmitter Lost Arbitration 1.Transmitter contended against this message unsuccessfully or a RxFIFO overrun occurred (arbitration was not necessarily lost nor transmission stopped).
1	1	Transmitter Successful 1. Transmitter contended for this slot successfully (message transmitted successfully). Message originated from this node.

Table 11-31 IFR Bit Status

Bit 3	Description
0	Message not an IFR.
1	IFR 1. The preceding bytes that this completion code is associated with was an in-frame response.

Table 11-32 IFR With/Without a CRC Bit Status

Bit 2	Description
0	IFR Without CRC. 1. This in-frame response does not contain a CRC.
1	IFR With CRC. 1. The preceding byte(s) that this completion code is associated with was an in-frame response with a CRC.

Table 11-33 Error Code Status[1, 2, 3]

Bit 1	Bit 0	Description
0	0	CRC Error. 1. A CRC error was detected in the reception of this message or a receiver overrun occurred.
0	1	Incomplete Byte Received. 1. An incorrect mod 8 count was detected in the reception of this message. A complete byte was not received.
1	0	Bit Timing Error. 1. A bit timing error ocurred in this message. Will occur with a mismatch of internal clock divide relative to other nodes. The receiver immediately places this completion code after the reception of this error. May happen if a break occurs after the bus has been low for only 8 to 34 µs. In this case there will be two completion codes, the first for the bit error, and the second for the break. See Table 11-28 through Table 11-32 for othr bit timing errors.
1	1	Break Symbol Received. 1. A break was detected on the bus, or the bus was shorted high for more than 239 µs and has recovered. a. If the receive buffer has 19 bytes in it and a break is received, the completion code will indicate on overflow but no break. b. If the receive buffer has 20 bytes (full) and a break is received, thre will be no new completion code pushed and no indication of the break.

NOTE:

1. These error codes are used in conjunction with the error flag to report detected errors. Errors are reported based on the order of precedence. Only the highest precedence error be reported.
2. The following list is in precedence order, highest first:
   1. Bread condition occurred.
   2. Bit timing error detected.
   3. Incomplete byte (incorrect mod 8 count).
   4. CRC error detected.
   5. If any of these error conditions is noted, any pending EOD response message (in-frame response) will not be sent.

11.11.12 Bus Errors

The following figures describe various bit timing/ invalid symbol errors.

NOTE

TXP signals shown are what is expected if the error had not occurred and not necessarily what is on the bus during these error conditions.

Figure 11-21 shows an active symbol received during SOF transmission that is greater than the 8þs digital filter and less than the minimum SOF symbol time. This symbol will be flagged as an invalid symbol and the corresponding completion code will be $82.

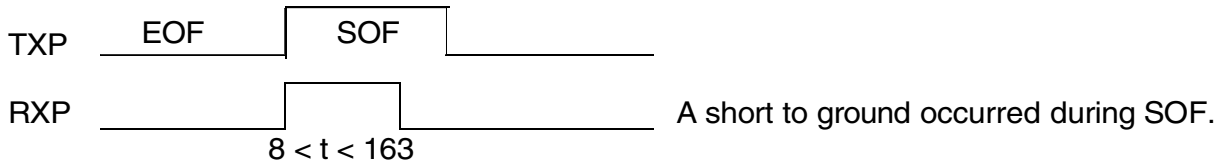

Figure 11-21 SOF Symbol Too Short

Figure 11-22 shows bit timing errors on short bits. This can be caused by noise, a bus short to ground (or VDD for passive bits), clock mismatches, etc. The corresponding completion code will be $82. If for the second case, the bus remains high for more than 239þs and then goes low, a completion code of $83 will follow.

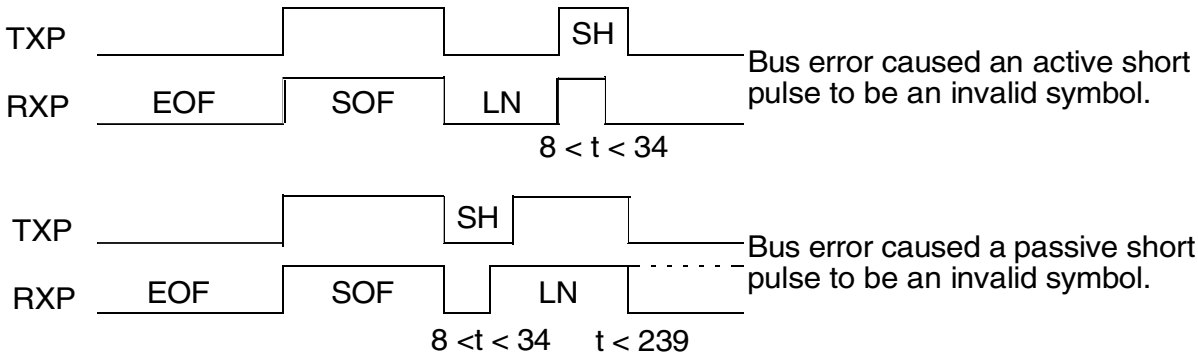

Figure 11-22 Figure 1-23 Short Bit Too Short

Figure 11-23 shows bit timing errors on long bits. This can be caused by noise, a bus short to V_{DD} (or ground for passive bits), clock mismatches, etc. For an active bit, if the bus goes low between 163 and 239 µs, the corresponding completion code will be $82.

If the bus goes low after the minimum BREAK time, a completion code of $83 will follow. For a passive bit, if the bus goes high between163 and 239 µs, the completion code will be $82. If the bus remains high for more than 239 µs and then goes low, a completion code of $83 will follow.

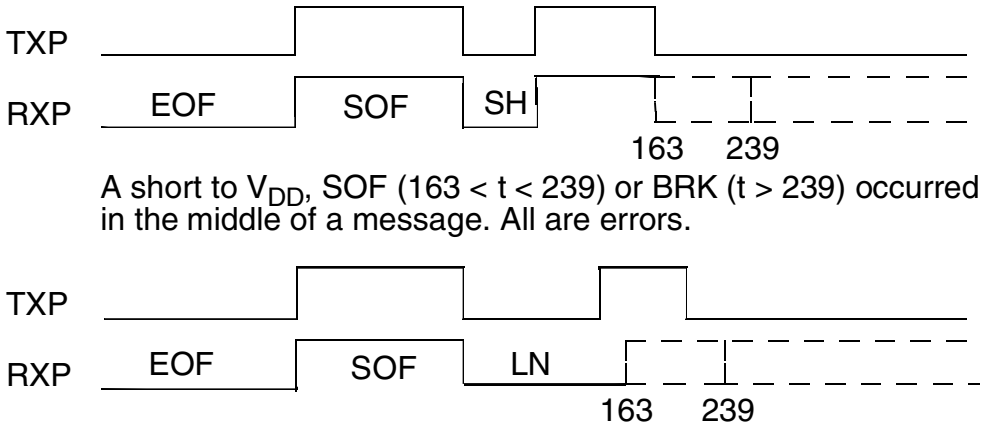

Figure 11-23 Long Bit Too Long

Figure 11-24 shows bit timing errors on an EOD and EOF. This can be caused by noise, a bus short to V_{DD}, clock mismatches, etc. In these cases, the completion code will be $82.

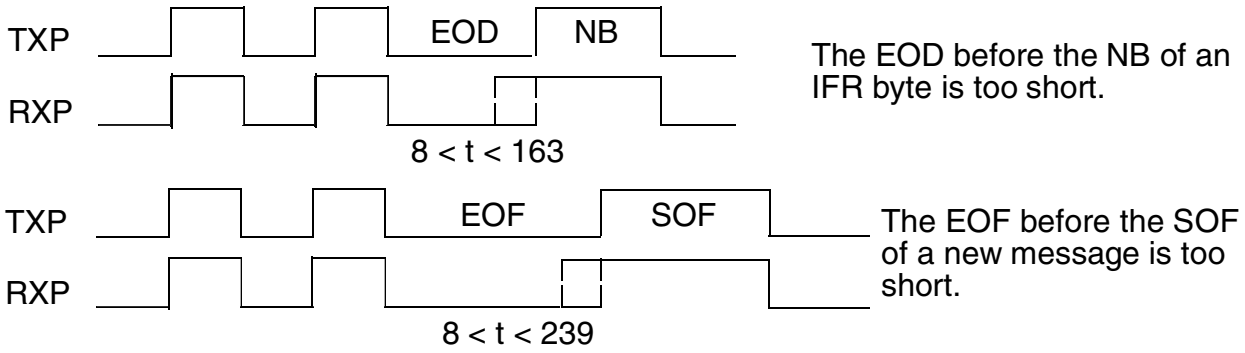

Figure 11-24 EOD and EOF Too Short

Figure 11-25 shows a bit timing error on a normalization bit. This can be caused by noise, a bus short to ground, clock mismatches, etc. The corresponding completion code will be $82.

Figure 11-25 Normalization Bit Too Short

Figure 11-26 shows a bit timing error on a normalization bit. This can be caused by noise, a bus short to V_{DD}, clock mismatches, etc. The corresponding completion code will be $82. If the bus remains high for more than 239 µs and then goes low, a completion code of $83 will follow.

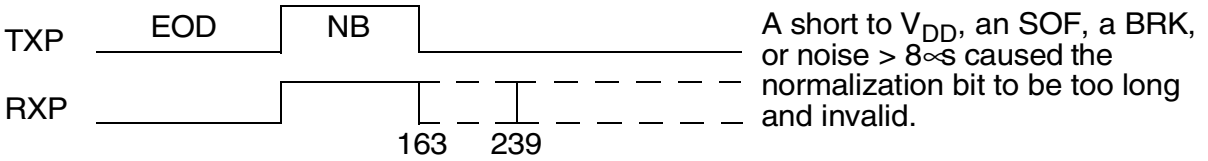

Figure 11-26 Normalization Bit Too Long

Other errors include CRC, incomplete byte, and BREAK. A CRC error occurs when the received message is corrupted by noise, delays, clock mismatches, etc. An incomplete byte error occurs when a received message ends on a non-byte boundary. This error also occurs when two extra 1's are transmitted after arbitration is lost on the last bit of a byte. Reception of a BREAK symbol is considered an error. This error occurs when the bus is held high for more than 239 µs and then goes low.

11.12 Mask Programmable Bus Error (BERR) Functionality

This DLCMD2 supports the BERR behavior according to IMB3 specification.

11.12.1 BERR_PLUG = 0

The DLCMD2 never asserts BERR signal in the IMB3.

11.12.2 BERR_PLUG = 1

The DLCMD2 will terminate the bus cycle with BERR in the following cases:

- Access to reserved 16-bit register within the DLCMD2's memory map.
- User access to supervisor-only registers when other registers in the DLCMD2's memory map are user-accessible (MCR SUPV bit is not set).
- Access to test register (TCR) when not in test mode.
- Writes to read-only registers.

11.13 Interrupt

This section describes DLCMD2 interrupt operation.

11.13.1 DLCMD2 Interrupts

In the default mode, interrupts may be requested due to one or more of four conditions becoming true. An optionally configured mode adds one more condition which will generate an interrupt. Interrupts may also be disabled.

If conditions which generate an interrupt occur while the DLCMD2 is being accessed, they will not be requested until the access is complete. Assuming interrupts are enabled, the default set of conditions that will cause the DLCMD2 module to request an interrupt are:

1. Completion code is placed in RxFIFO, DLCMD2 not accessed.

   a. An EOD has been received on the J1850 bus.
   b. bAdditional byte is received when the RxFIFO is full. This condition will cause the completion code to be pushed (in position 20 of the RxFIFO). The received byte and the rest of the message are ignored.
   c. DLCMD2 receives break. The reception of the break symbol will cause the completion code to be pushed into RxFIFO. If the break is received in the middle of DLCMD2 transmitting, it will stop transmitting, reset the transmitter, and clear the TxFIFO.
   d. DLCMD2 detects a bit-timing error. The bit-timing error will cause the completion code to be pushed into RxFIFO.

2. The RxFIFO has 12 bytes in it and the 13th byte is received, DLCMD2 not accessed. Completion code may or may not be present in RxFIFO. This differs from the 13th byte status indication which occurs only if there is no completion code in the RxFIFO.

   a. The status byte will only reflect this interrupt if there are no completion codes in the RxFIFO
   b. This interrupt generally means that the DLCMD2 is being neglected by the host.

3. A transmit operation is in progress, and there is no last byte to the message in the buffer and the buffer becomes half empty (six bytes left to transmit, five bytes in TxFIFO and the sixth byte is popped off to the transmit shift register), DLCMD2 not accessed.
4. The DLCMD2 is waking up on the rising edge of data link activity when it was previously in sleep mode.

   a. Any J1850 bus edge will wake up the DLCMD2. Will get a bit error indication if the DLCMD2 does not see at least 34 µs of the SOF.

By setting bit 15 in the ILR register in **11.11.3 Interrupt Level Register (ILR)**, one more condition capable of generating an interrupt is added:

5. A byte has been received into an empty RxFIFO, DLCMD2 not accessed.

Table 11-34 shows DLCMD2 interrupt operations. Refer to **Table 11-10** for IVR[2:0] Encoding.

Table 11-34 Interrupt Operations

Interrupt	Conditions Necessary to Enable/Re-Enable Interrupts	Conditions for Interrupt Assertion	Conditions to Clear this Interrupt
Wake-up	Enter low power mode	DLCMD2 module is in sleep mode. Positive going edge on bus > V_{ih} is sensed on bus.	Read IPR with bit 3 set and write IPR[3] “0” to clear
Transmitter half full (6 bytes)	1. “Load a byte into TxFIFO” with a fifth byte position filled, — OR — 2. Load in data bytes so that the sixth position in TxFIFO is occupied.	A transmit operation is in progress, no last byte indicated to the message in TxFIFO (*block mode*), TxFIFO becomes half empty, and transmit interrupt is enabled (*last byte not pushed into TxFIFO*)	Read IPR with bit 3 set and write IPR[3] “0” to clear
13th byte received	The only way to insure that this interruptis re-enabled is to complete empty out the Rx FIFO: 1. If 13th byte is occupeid and 14th isn’t a “flush byte” command will re-enable. — OR — 2. If 13th byte is occouped and 14th isn’t, and there is no completion code at the head of the RsFIFO, a “flush message” command will re-enable. — OR — 3. 13th byte becomes unoccupied.	RxFIFO receives 13th byte, (*Completion may or may not be present in RxFIFO, refer to* **11.13.1**)	Read IPR with bit 2 set and write IPR[2] “0” to clear.
EOD sensed on bus	Always enabled.	A completion code is placed onto the RxFIFO.	Read IPR with bit 1 set and write IPR[1] “0” to clear.
First byte received in an empty RxFIFO	1. If the first positon is filled, and the second position is empty, and a “flush byte” command is issued. — OR — 2. The first position becomes empty.	RxFIFO empty, first byte received, INTMODE in ILR must be set, and flush interrupt enabled (flush message except completion code, completion code will cauase EOD interrupt)	Read IPR with bit 0 set and write IPR[0] “0” to clear.

The DLCMD2 module is capable of generating one interrupt level on the IMB3. This level is programmed into the priority level bits in the interrupt level register (ILR[2:0]). This value determines which interrupt signal (IRQB7-1) is driven onto the bus during an interrupt request. IRQ0 is ignored by the CPU and so a zero priority level essentially disables all interrupts.

When an interrupt is requested, the CPU initiates an IACK cycle. The module decodes the IACK cycle and compares the CPU recognized level to the level that the module is currently requesting. If a match occurs, then arbitration begins. During arbitration, the arbitration number (refer to IARB[3:0] — Interrupt Arbitration ID (Bit 3 ~ Bit 0) in **11.11.1 Module Configuration Register (MCR)**) is driven in bit serial form, alternating between the IARB0 and IARB1 lines. The most significant bit of the arbitration number is driven first. Since the bus is a wire-AND one, a “low” level wins any contentions. Thus, the arbitration number is verified on a bit-by-bit basis. If contention is detected, (driving high and detecting low), then the module has lost the arbitration and immediately stop driving its arbitration number.

If the module wins the arbitration, it drives the eight bit interrupt vector stored in IVR register onto the data bus. The lower 3 bits are read-only bits and won’t be updated by the DLCMD2 until write-once to IVR occurs. These three bits represent the source of the interrupt as described in **Table 11-34**. The upper 5 bits are user programmable.

This interrupt generation scheme is shown in **Figure 11-27**.

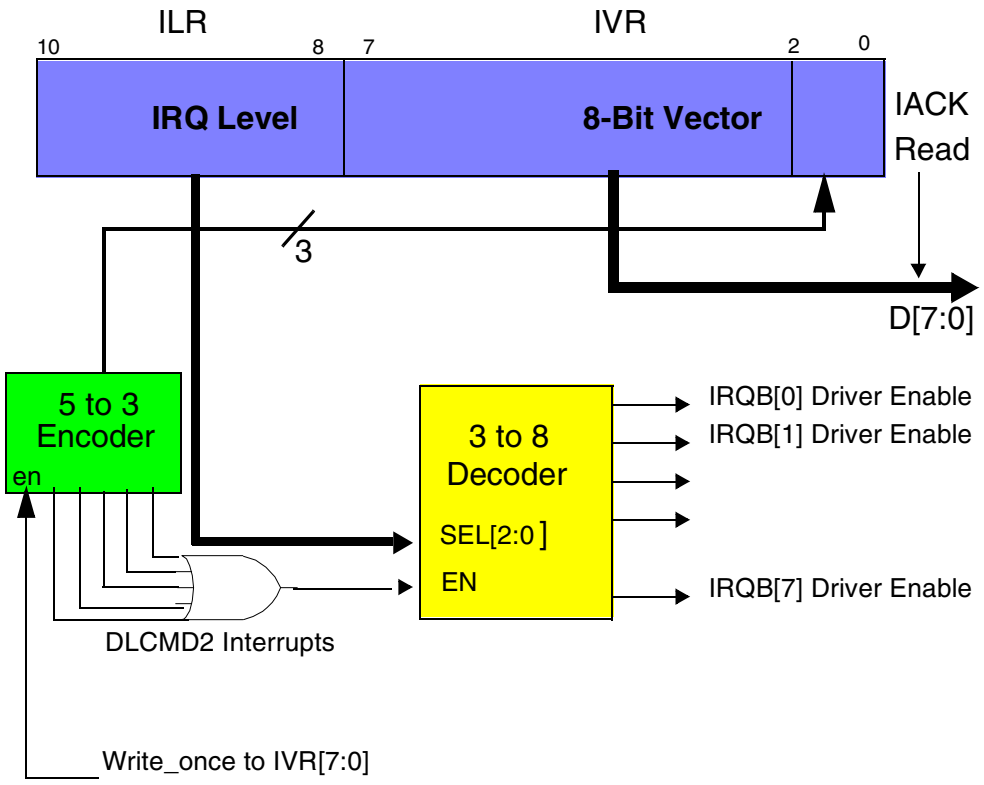

Figure 11-27 DLCMD2 Interrupt Vector Generation (IRQ_PLUG = 0)

11.14 In-Frame Response

This section describes how the DLCMD2 uses an in-frame response (IFR) in message transmission.

11.14.1 IFR Operation

The IFR may be sent by loading up a message for transmission with a “send on EOD” bit combination set in the command byte accompanying the first byte. The setting of the EOD bit combination will automatically next byte as part of, or the complete, response.

A message on the bus is determined to be ready for a response when an “end of data” (EOD) waveform has occurred on the bus. If a reply message initiates immediately following the EOD, then this will be an IFR. If no response is sent when EOD occurs, then an EOF will appear on the bus, signifying normal end of message without response. The EOF waveform is an extension of the EOD waveform. The reply will start with an active “1” or “0”, called the normalization bit, and may also have a CRC transmitted with it. The DLCMD2 is capable of arbitrating against and receiving all types of IFRs. Whether or not the received IFR contains a CRC or not is signalled through the use of the NBFS bit in the SCTL, see **11.11.4 Symbol Timing Control and Pre-Scaler Register (SCTL)**. IFRs are arbitrated just like any other message if more than one node wants to send a response. Single byte IFR from multiple responders (type 2) will retransmit if arbitration is lost. If arbitration is lost on the last bit of a type 2 IFR byte, two extra “1” bits will not be sent onto the bus since the IFR will retransmit. Upon beginning transmission of a single byte IFR from a single responder (type 1) or multiple byte IFR from a single responder (type 3), the DLCMD2 will enter TAR mode and not retransmit the IFR if arbitration is lost. If arbitration is lost on the last bit of a type 3 IFR byte, two extra “1” bits will be sent onto the bus.

This response message is sent only if:

1. There were no errors of any type in the original message requiring response. The only exception to this is that if an IFR is loaded after the transmitter lost arbitration to another message, the IFR will be sent at EOD.
2. IFR was loaded with a command byte that indicated either: “first byte” or “first and last” byte of a message.
3. This IFR was loaded into the DLCMD2 after the start of the message was detected, and t_{resp} before the EOD was recognized.

When the DLCMD2 receives the command byte and data byte signaling an IFR, it will start transmitting the response after EOD even if there is no last byte in the TxFIFO. The DLCMD2 will start the response by automatically transmitting an active phase “0” or a “1” depending on how the NBFS bit is programmed.

It is left to the host to remember that any messages that were in the transmit buffer at the time the IFR was loaded have been flushed, and must be reloaded by the host for transmission. This is true even if the IFR was not successfully loaded or transmitted.

Completion code bit 3 will inform the host of whether or not an IFR has occurred. If an IFR has occurred, the RxFIFO will contain the initial message, with a completion code, and the response, with its completion code. Each completion code identifies the message associated with it as a normal message, or a response. The only way that the CPU can tell that such a sequence has occurred on the bus is by reading the completion code.

If any errors occur during the reception of an IFR:

1. A completion code indicating an error occurred during reception is pushed into the RxFIFO.
2. Anything left of the IFR is not received off the bus.

If bit 3 of the completion code is set, then bit 2 will indicate whether the response had a CRC field or not. If bit 2 is set, then the response had a CRC field. This is important for the CPU to determine the meaning of the response.

A DLCMD2 may send an IFR to an IFR with CRC but no response may follow an IFR without CRC.

For the system to enter a mode that allows an IFR, some coordination of nodes is required. The node sending an IFR must select the appropriate interrupt configuration (interrupt on first byte), and be ready to transfer a response into the DLCMD2, signalling through the command byte that this is an IFR.

11.14.2 IFR Abort Conditions

Table 11-35 shows conditions in which IFRs are aborted.he IFR was l

Table 11-35 IFR Aborted Conditions

IFR Transmission Aborted If	Actions Taken
Host underran transmitter (no “last byte” indicated in the TxFIFO at EOD, and host did not supply a last byte in time for transmission).	DLCMD2 inverts the CRC field on the truncated IFR.
IFR loses contention to another IFR.	DLCMD2 stops transmitting multiple byte IFRs, and flushes the TxFIFO immediately. DLCMD2 can be configured to auto-retry or stop transmission of single byte IFRs upon loss of arbitration.
Error detected in the received message.	DLCMD2 resets the transmitter and TxFIFO, no IFR is sent.
Error in IFR during transmission.	IFR is stopped, transmitter and TxFIFO are reset.
IFR not loaded correctly. 1. Bus idle when IFR was loaded (loaded before any message is on the bus). 2. IFR loaded less than t_{resp} before EOD is sensed on bus (loaded too late). 3. IFR not loaded with “first byte” indicated. 4. IFR bit set in command byte, but no data is loaded at all.	DLCMD2 resets the transmitter and TxFIFO, no IFR is sent.

11.14.3 IFR Types

The DLCMD2 supports types 1, 2 and 3 IFRs with arbitration and auto-retransmission. See **Figure 11-28** for a description of each type of IFR. The following sections describe how the DLCMD2 can be configured to transmit the three types of IFRs.

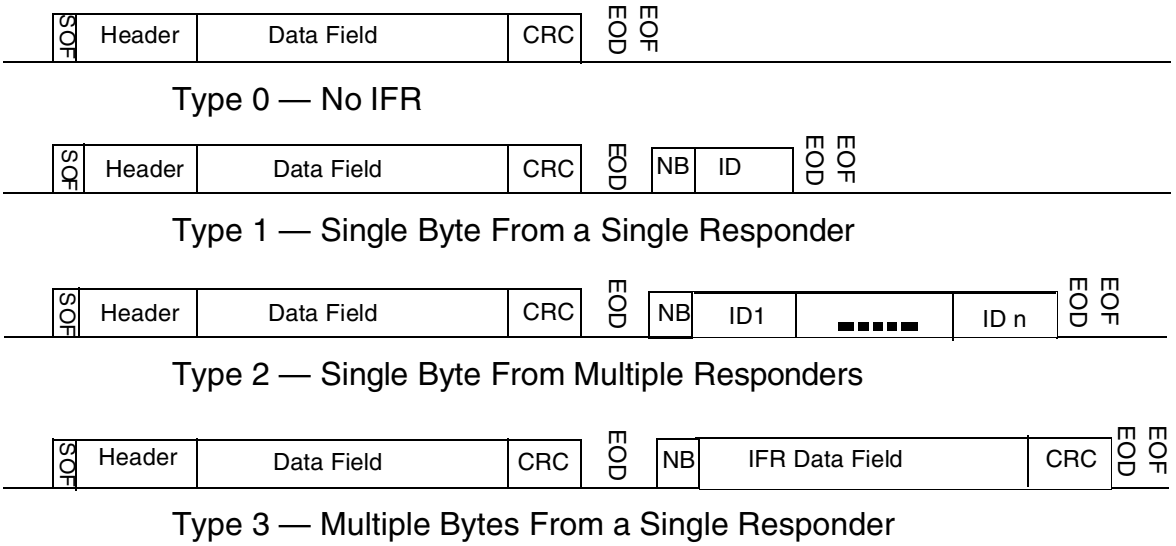

Figure 11-28 Types of IFR

11.14.3.1 Type 1 IFR

A type 1 IFR is a single byte IFR without CRC transmitted by a single responder. If arbitration is lost, the DLCMD2 will not attempt to re-transmit the IFR byte in the TxFIFO. The TxFIFO will be cleared and the IFR byte will be lost.

The DLCMD2 can be configured to transmit a type 1 IFR by writing an IFR byte into the TDATA register and a command byte of $BC indicating “send on EOD without CRC” and “load as first and last byte.”

11.14.3.2 1.Type 2 IFR

A type 2 IFR results from multiple responders transmitting a single byte IFR onto the bus. Since only one IFR will win arbitration, the other responders will re-transmit their IFRs as soon as the IFR winning arbitration finishes. The IFR byte in the TxFIFO will not be cleared until transmission is successful or an error is detected on the bus.

The DLCMD2 can be configured to transmit a type 2 IFR by writing an IFR byte into the TDATA register and a command byte of $DC indicating “send on EOD with auto re-try” and “load as first and last byte.”

11.14.3.3 Type 3 IFR

A type 3 IFR is a multiple byte IFR with or without CRC transmitted by a single responder. If arbitration is lost, the DLCMD2 will not attempt to re-transmit the IFR bytes in the TxFIFO. The TxFIFO will be cleared and the IFR bytes will be lost.

The DLCMD2 can be configured to transmit a type 3 IFR with CRC by writing the first IFR byte into the TDATA register and a command byte of $74 indicating “send on EOD with CRC” and “load as first byte.”

The DLCMD2 can also be configured to transmit a type 3 IFR without CRC by writing the first byte into the TDATA register and a command byte of $B4 indicating “send on EOD without CRC” and “load as first byte.”troller Module (DLCMD2)

11.15 System Overview

Typical usage of the DLCMD2 in a microcomputer system is shown in **Figure 11-29**.

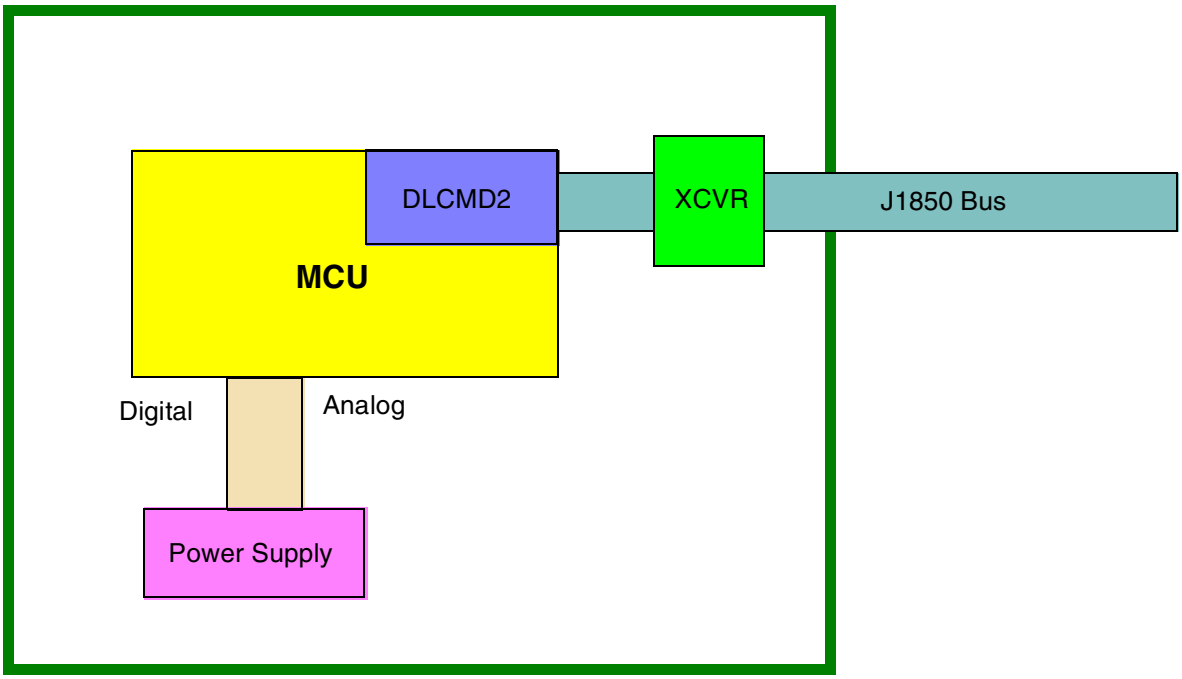

Figure 11-29 J1850 Node

11.16 Data Link Controller Module (DLCMD2) — Internal

11.16.1 Inter-Module Bus 3 (IMB3)

11.16.2 Bus Signals

The DLCMD2 shall have an internal interface compatible with the IMB3. The address and data bus along with their associated control and handshake lines are used to transfer data between the IMB3 and the DLCMD2. **Table 11-36** lists the IMB3 signals used in the DLCMD2.

Table 11-36 IMB3 Signals

Signal Name	Description	Input / Output	Assert / Negate
ICLOCK	System Clock	I	—
IMSTRSTB	Master Reset	I	CLOCK HIGH / CLOCK LOW
ISYSRSTB	System Reset	I	CLOCK HIGH / CLOCK LOW
IFREEZEB	Freeze	I	CLOCK LOW
ITSTMODB	Test Mode	I	CLOCK LOW
IMODMAP	Module Memory Mapping	I	B4
IFC[2:0]	Function Codes	I	B4 / B3 + B3A
ICYSB	Cycle Start	I	B4 / B3 + B3A
IWRITEB	Read/$\overline{\text{Write}}$	I	B4 / B3 + B3A
IADDR[23:0]	Address Bus	I	B4 / B3 + B3A
IASB	Address Strobe	I	B1 / B4
IAACKB	Address Knowledge	O	B1 / B4
ISIZ[1:0]	Size Codes	I	B4 / B3 + B3A
IDSB	Data Strobe	I	B2 / B4 * B1
IILBS[1:0]	interrupt Level Byte Select	I	CLOCK HIGH
IIMB3TSTB	Test Enable	I	CLOCK HIGH
ICLKDIS	Clock Disable	I	CLOCK LOW
ICLKE	Supports IMB 2X Bus Clock 2x bus clock is not used in this module. This signal is terminated in the BIU with a std clk rcd.	I	—
ICLK2XE	Supports IMB 2X Bus Clock 2x bus clock is not used in this module. This signal is terminated in the BIU with a std clk rcd.	I	—
IDATA[15:0]	Data Bus	I/O	B2(Write) B3(Read / Write B4(Read) / B4* + B1 + B3 following the assertion of BTACK
IDTACKB	Data Transfer Acknowledge	O	B2 + B3 / B4* + B1
IBERRB	Bus Error	I/O	B2 + B3 / B4* + B1
IIRQB[7:0]	Interupt Request Level	O	CLOCK LOW / CLOCK HIGH
IIARB[1:0]	Interrupt Arbitration	I/O	Bit0:CLOCK LOW / CLOCK HIGH Bit1:CLOCK HIGH / CLOCK LOW

11.17 Module I/O Signals

11.17.1 Signal Descriptions

This section provides information on external signal functions.

11.17.2 External Connections

Figure 11-30 shows DLCMD2 external connections.

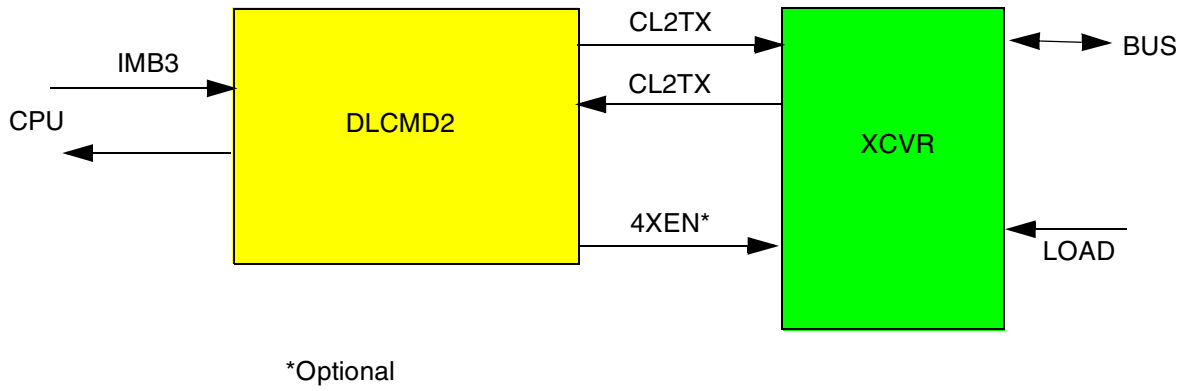

Figure 11-30 DLCMD2 External Connections

11.17.3 Signal Functions

Table 11-37 summarizes DLCMD2 external signals.

Table 11-37 Signal Names

Module Signal Name (External)	Input / Output	Description
CL2TX	O	DLCMD2 digital output to transceiver
CL2RX	I	DLCMD2 digital input from transceiver
4XEN[1]	O	4X transmit enable output to transceiver

NOTE:
1. May be provided by the DLCMD2, CPU, or other IC discrete. The DLCMD2 module will provide this signal to the periphery of the module. This way, an on-chip transceiver may be more easilty accommodated.

11.17.3.1 CL2TX

This pin is a logic level output. A logic “0” output drives the BUS output to 0 VDC (external pull down resistor to ground) and a logic “1” output produces a high voltage at the bus output. An internal 200 KΩ to ground in the XCVR guarantees a logic “0” input when this pin is open circuit (the bus output is tri-stated).

11.17.3.2 CL2RX

This is a CMOS compatible input used to get receiver data to the microprocessor. When the voltage on BUS is over 3.5 ± .2 VDC, this pin shall be a logic “1.” When the voltage on BUS is under 3.5 ± .2 VDC, this pin shall be a logic “0.” There is a minimum of.1 VDC of hysteresis between the bus high and low (and vice versa) transition points.

11.17.3.3 4XEN

This is an optional pin used to select whether waveshaping for the J1850 output is enabled or disabled. A logic “0” shall enable waveshaping and a logic “1” shall disable

waveshaping. An internal 200 KΩ to ground in the XCVR guarantees a logic “0” input when this pin is open circuit.