# THE DATASHEET OF

# CY7B933-JXIT

www.win-source.net 0086-755-83957316

# CY7B923/CY7B933

# HOTLink® Transmitter/Receiver

## Features

- Fibre Channel-compliant
- IBM ESCON®-compliant
- DVB-ASI-compliant
- ATM-compliant
- 8B/10B-coded or 10-bit unencoded
- Standard HOTLink®: 160 to 330 Mbps
- High-speed HOTLink: 160 to 400 Mbps for high-speed applications
- Transistor-transistor logic (TTL)-synchronous I/O
- No external phase locked-loop (PLL) components
- Triple positive emitter coupled logic (PECL) 100 K serial outputs
- Dual PECL 100 K serial inputs
- Low-power: 350 mW (Tx), 650 mW (Rx)
- Compatible with fiber-optic modules, coaxial cable, and twisted pair media
- Built-in self-test (BIST)
- Single +5-V supply
- 28-pin small outline integrated circuit (SOIC)/plastic leaded chip carrier (PLCC)
- Pb-free packages available
- 0.8-µ bipolar complementary metal oxide semiconductor (BiCMOS)

## Functional Description

The CY7B923 HOTLink transmitter and CY7B933 HOTLink receiver are point-to-point communications building blocks that transfer data over high-speed serial links (fiber, coax, and twisted pair). Standard HOTLink data rates range from 160 to 330 Mbps. Higher speed HOTLink is also available for high-speed applications (160 to 400 Mbits/second). Figure 1 illustrates typical connections to host systems or controllers.

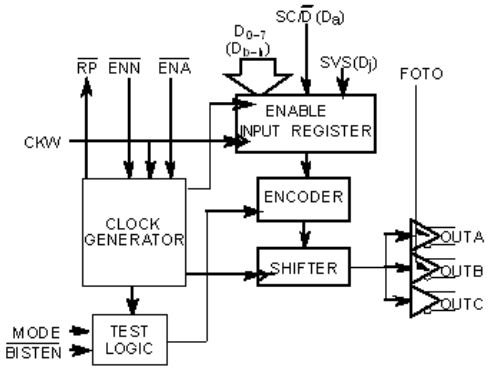
Eight bits of user data or protocol information are loaded into the HOTLink transmitter and are encoded. Serial data is shifted out of the three differential PECL serial ports at the bit rate (which is ten times the byte rate).

The HOTLink receiver accepts the serial bit stream at its differential line receiver inputs and, using a completely integrated PLL clock synchronizer, recovers the timing information necessary for data reconstruction. The bit stream is deserialized, decoded, and checked for transmission errors. Recovered bytes are presented in parallel to the receiving host along with a byte-rate clock.

The 8B/10B encoder/decoder can be disabled in systems that already encode or scramble the transmitted data. I/O signals are available to create a seamless interface with both asynchronous FIFOs (that is, CY7C42X) and clocked FIFOs (that is, CY7C44X). A BIST pattern generator and checker allows testing of the transmitter, receiver, and the connecting link as a part of a system diagnostic check.

HOTLink devices are ideal for a variety of applications where a parallel interface can be replaced with a high-speed point-to-point serial link. Applications include interconnecting workstations, servers, mass storage, and video transmission equipment.

## CY7B923 Transmitter Block Diagram



## CY7B933 Receiver Block Diagram

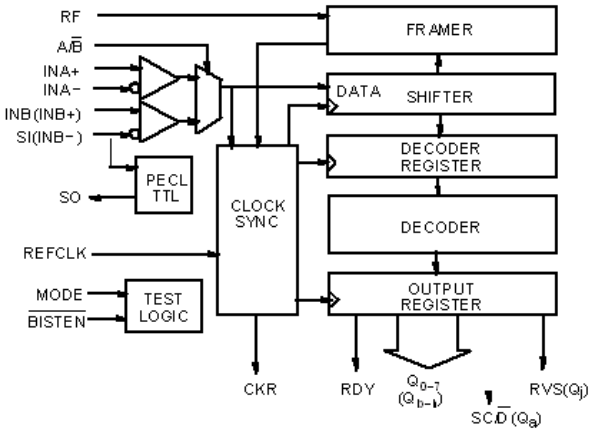
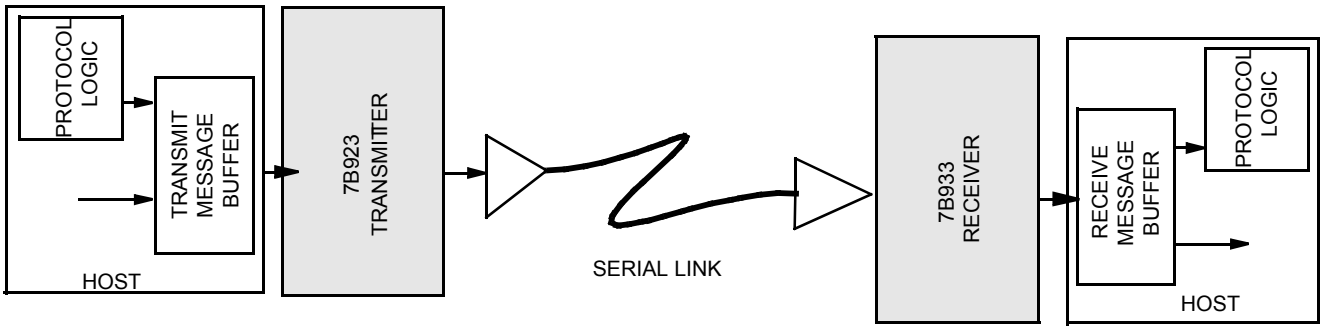



**Figure 1. HOTLink System Connections**



## Contents

**Pin Configurations** ..... **4**
**Pin Descriptions** ..... **6**
**CY7B923 HOTLink Transmitter Block Diagram Description** ..... **8**
Input Register ..... 8
Encoder ..... 8
Shifter ..... 8
OutA, OutB, OutC ..... 8
Clock Generator ..... 8
Test Logic ..... 9
**CY7B933 HOTLink Receiver Block Diagram Description** ..... **9**
Serial Data Inputs ..... 9
PECL-TTL Translator ..... 9
Clock Synchronization ..... 9
Framer ..... 9
Shifter ..... 9
Decode Register ..... 9
Decoder ..... 10
Output Register ..... 10
Test Logic ..... 10
**HOTLink CY7B923 Transmitter and CY7B933 Receiver Operation** ..... **10**
**CY7B923 HOTLink Transmitter Operating Mode Description** ..... **11**
Encoded Mode Operation ..... 12
Bypass Mode Operation ..... 13
PECL Output Functional and Connection Options ..... 13
**Transmitter Serial Data Characteristics** ..... **13**
**Transmitter Test Mode Description** ..... **13**
BIST Mode ..... 15
Test Mode ..... 16
**CY7B933 HOTLink Receiver Operating Mode Description** ..... **16**
Encoded Mode Operation ..... 16
Bypass Mode Operation ..... 17
Parallel Output Function ..... 17
**Receiver Serial Data Requirements** ..... **17**
**Receiver Test Mode Description** ..... **18**
BIST Mode ..... 18
Test Mode ..... 18
**X3.230 Codes and Notation Conventions** ..... **18**
Notation Conventions ..... 18
8B/10B Transmission Code ..... 19
Transmission Order ..... 19
Valid and Invalid Transmission Characters ..... 19
Using the Tables for Generating Transmission Characters ..... 20
Using the Tables for Checking the Validity of Received Transmission Characters ..... 20
**Valid Data Characters (SC/D = LOW)** ..... **21**
**Valid Special Character Codes and Sequences (SC/D = HIGH)[1, 2]** ..... **29**
**Maximum Ratings** ..... **30**
**Operating Range** ..... **30**
**CY7B923/CY7B933 Electrical Characteristics** ..... **30**
**Capacitance[13]** ..... **31**
**Transmitter Switching Characteristics** ..... **32**
**Receiver Switching Characteristics** ..... **33**
**Switching Waveforms** ..... **34**
**Ordering Information** ..... **37**
Ordering Code Definitions ..... 37
**Package Diagrams** ..... **38**
**Acronyms** ..... **40**
**Document Conventions** ..... **40**
Units of Measure ..... 40
**Document History Page** ..... **41**
**Sales, Solutions, and Legal Information** ..... **42**
Worldwide Sales and Design Support ..... 42
Products ..... 42
PSoC® Solutions ..... 42
Cypress Developer Community ..... 42
Technical Support ..... 42

## Pin Configurations

**Figure 2. CY7B923 Transmitter Pin Configurations**

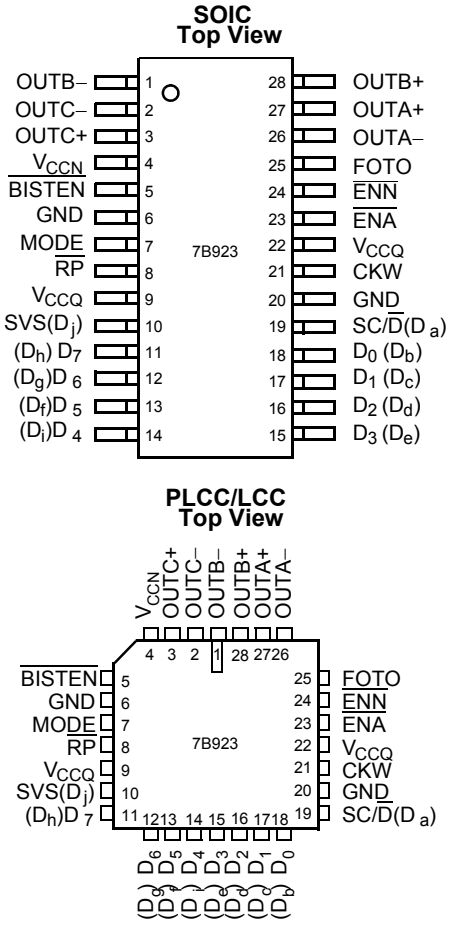
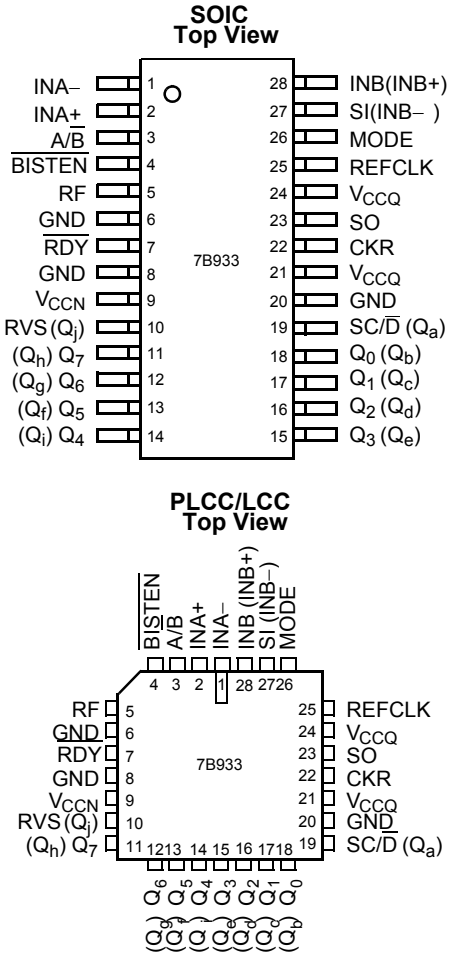


**Figure 3. CY7B933 Receiver Pin Configurations**



## Pin Descriptions

**Table 1. CY7B923 HOTLink Transmitter**

| Name | I/O | Description |
|---|---|---|
| $D_{0-7}$ ($D_{b-h}$) | TTL In | Parallel data input. Data is clocked into the Transmitter on the rising edge of CKW if ENA is LOW (or on the next rising CKW with ENN LOW). If ENA and ENN are HIGH, a Null character (K28.5) is sent. When MODE is HIGH, $D_{0, 1, ...7}$ become $D_{b, c,...h}$, respectively. |
| SC/D ($D_a$) | TTL In | Special character/data select. A HIGH on SC/D when CKW rises causes the transmitter to encode the pattern on $D_{0-7}$ as a control code (Special Character), while a LOW causes the data to be coded using the 8B/10B data alphabet. When MODE is HIGH, SC/D ($D_a$) acts as $D_a$ input. SC/D has the same timing as $D_{0-7}$. |
| SVS ($D_j$) | TTL In | Send violation symbol. If SVS is HIGH when CKW rises, a Violation symbol is encoded and sent while the data on the parallel inputs is ignored. If SVS is LOW, the state of $D_{0-7}$ and SC/D determines the code sent. In normal or test mode, this pin overrides the BIST generator and forces the transmission of a Violation code. When MODE is HIGH (placing the transmitter in unencoded mode), SVS ($D_j$) acts as the $D_j$ input. SVS has the same timing as $D_{0-7}$. |
| ENA | TTL In | Enable parallel data. If ENA is LOW on the rising edge of CKW, the data is loaded, encoded, and sent. If ENA and ENN are HIGH, the data inputs are ignored and the Transmitter will insert a Null character (K28.5) to fill the space between user data. ENA may be held HIGH/LOW continuously or it may be pulsed with each data byte to be sent. If ENA is being used for data control, ENN will normally be strapped HIGH, but can be used for BIST function control. |
| ENN | TTL In | Enable next parallel data. If ENN is LOW, the data appearing on $D_{0-7}$ at the next rising edge of CKW is loaded, encoded, and sent. If ENA and ENN are HIGH, the data appearing on $D_{0-7}$ at the next rising edge of CKW will be ignored and the Transmitter will insert a Null character to fill the space between user data. ENN may be held HIGH/LOW continuously or it may be pulsed with each data byte sent. If ENN is being used for data control, ENA will normally be strapped HIGH, but can be used for BIST function control. |
| CKW | TTL In | Clock write. CKW is both the clock frequency reference for the multiplying PLL that generates the high-speed transmit clock, and the byte rate write signal that synchronizes the parallel data input. CKW must be connected to a crystal controlled time base that runs within the specified frequency range of the Transmitter and Receiver. |
| FOTO | TTL In | Fiber optic transmitter off. FOTO determines the function of two of the three PECL transmitter output pairs. If FOTO is LOW, the data encoded by the Transmitter will appear at the outputs continuously. If FOTO is HIGH, OUTA± and OUTB± are forced to their "logic zero" state (OUT+ = LOW and OUT– = HIGH), causing a fiber-optic transmit module to extinguish its light output. OUTC is unaffected by the level on FOTO, and can be used as a loop-back signal source for board-level diagnostic testing. |
| OUTA±<br>OUTB±<br>OUTC± | PECL Out | Differential serial data outputs. These PECL 100 K outputs (+5 V referenced) are capable of driving terminated transmission lines or commercial fiber optic transmitter modules. Unused pairs of outputs can be left open, or wired to $V_{CC}$ to reduce power, if the output is not required. OUTA± and OUTB± are controlled by the level on FOTO, and will remain at their "logical zero" states when FOTO is asserted. OUTC± is unaffected by the level on FOTO. (OUTA+ and OUTB+ are used as a differential test clock input while in Test mode, that is, MODE = UNCONNECTED or forced to $V_{CC}/2$). |
| MODE | Three-Level In | Encoder mode select. The level on MODE determines the encoding method to be used. When wired to GND, MODE selects 8B/10B encoding. When wired to $V_{CC}$, data inputs bypass the encoder and the bit pattern on $D_{a-j}$ goes directly to the shifter. When left floating (internal resistors hold the input at $V_{CC}/2$) the internal bit-clock generator is disabled and OUTA+/OUTB+ become the differential bit clock to be used for factory test. In typical applications MODE is wired to $V_{CC}$ or GND. |
| BISTEN | TTL In | BIST enable. When BISTEN is LOW and ENA and ENN are HIGH, the transmitter sends an alternating 1–0 pattern (D10.2 or D21.5). When either ENA or ENN is set LOW and BISTEN is LOW, the transmitter begins a repeating test sequence that allows the Transmitter and Receiver to work together to test the function of the entire link. In normal use this input is held HIGH or wired to $V_{CC}$. The BIST generator is a free-running pattern generator that need not be initialized, but if required, the BIST sequence can be initialized by momentarily asserting SVS while BISTEN is LOW. BISTEN has the same timing as $D_{0-7}$. |
| RP | TTL Out | Read pulse. RP is a 60% LOW duty-cycle byte-rate pulse train suitable for the read pulse in CY7C42X FIFOs. The frequency on RP is the same as CKW when enabled by ENA, and duty cycle is independent of the CKW duty cycle. Pulse widths are set by logic internal to the transmitter. In BIST mode, RP will remain HIGH for all but the last byte of a test loop. RP will pulse LOW one byte time per BIST loop. |
| $V_{CCN}$ | | Power for output drivers. |
| $V_{CCQ}$ | | Power for internal circuitry. |
| GND | | Ground. |

**Table 2. CY7B933 HOTLink Receiver**

| Name | I/O | Description |
|---|---|---|
| $Q_{0-7}$ ($Q_{b-h}$) | TTL Out | $Q_{0-7}$ parallel data output. $Q_{0-7}$ contain the most recently received data. These outputs change synchronously with CKR. When MODE is HIGH, $Q_{0,1,...7}$ become $Q_{b,c,...h}$, respectively. |
| SC/$\overline{D}$ ($Q_a$) | TTL Out | Special character/data select. SC/$\overline{D}$ indicates the context of received data. HIGH indicates a Control (Special Character) code, LOW indicates a Data character. When MODE is HIGH (placing the receiver in Unencoded mode), SC/$\overline{D}$ acts as the $Q_a$ output. SC/$\overline{D}$ has the same timing as $Q_{0-7}$. |
| RVS ($Q_j$) | TTL Out | Received violation symbol. A HIGH on RVS indicates that a code rule violation has been detected in the received data stream. A LOW shows that no error has been detected. In BIST mode, a LOW on RVS indicates correct operation of the Transmitter, Receiver, and link on a byte-by-byte basis. When MODE is HIGH (placing the receiver in Unencoded mode), RVS acts as the $Q_j$ output. RVS has the same timing as $Q_{0-7}$. |
| $\overline{RDY}$ | TTL Out | Data output ready. A LOW pulse on $\overline{RDY}$ indicates that new data has been received and is ready to be delivered. A missing pulse on $\overline{RDY}$ shows that the received data is the Null character (normally inserted by the transmitter as a pad between data inputs). In BIST mode $\overline{RDY}$ will remain LOW for all but the last byte of a test loop and will pulse HIGH one byte time per BIST loop. |
| CKR | TTL Out | Clock read. This byte rate clock output is phase and frequency aligned to the incoming serial data stream. $\overline{RDY}$, $Q_{0-7}$, SC/$\overline{D}$, and RVS all switch synchronously with the rising edge of this output. |
| A/$\overline{B}$ | PECL in | Serial data input select. This PECL 100K (+5V referenced) input selects INA or INB as the active data input. If A/$\overline{B}$ is HIGH, INA is connected to the shifter and signals connected to INA will be decoded. If A/$\overline{B}$ is LOW INB is selected. |
| INA± | Diff In | Serial data input A. The differential signal at the receiver end of the communication link may be connected to the differential input pairs INA± or INB±. Either the INA pair or the INB pair can be used as the main data input and the other can be used as a loopback channel or as an alternative data input selected by the state of A/$\overline{B}$. One input of an intentionally unused differential-pair (INA± or INB±) should be terminated to $V_{CC}$ through a 1–5-KΩ resistor to assure that no data transitions are accidentally created. |
| INB (INB+) | PECL in (Diff In) | Serial data input B. This pin is either a single-ended PECL data receiver (INB) or half of the INB differential pair. If SO is wired to $V_{CC}$, then INB± can be used as differential line receiver interchangeably with INA±. If SO is normally connected and loaded, INB becomes a single-ended PECL 100K (+5 V referenced) serial data input. INB is used as the test clock while in Test mode. |
| SI (INB–) | PECL in (Diff In) | Status input. This pin is either a single-ended PECL status monitor input (SI) or half of the INB differential pair. If SO is wired to $V_{CC}$, then INB± can be used as differential line receiver interchangeably with INA±. If SO is normally connected and loaded, SI becomes a single-ended PECL 100K (+5V referenced) status monitor input, which is translated into a TTL-level signal at the SO pin. |
| SO | TTL Out | Status out. SO is the TTL-translated output of SI. It is typically used to translate the carrier detect output from a fiber-optic receiver connected to SI. When this pin is normally connected and loaded (without any external pull-up resistor), SO will assume the same logical level as SI and INB will become a single-ended PECL serial data input. If the status monitor translation is not desired, then SO may be wired to $V_{CC}$ and the INB± pair may be used as a differential serial data input. |
| RF | TTL In | Reframe enable. RF controls the Framer logic in the Receiver. When RF is held HIGH, each SYNC (K28.5) symbol detected in the shifter will frame the data that follows. If it is HIGH for 2,048 consecutive bytes, the internal framer switches to double-byte mode. When RF is held LOW, the reframing logic is disabled. The incoming data stream is then continuously deserialized and decoded using byte boundaries set by the internal byte counter. Bit errors in the data stream will not cause alias SYNC characters to reframe the data erroneously. |
| REFCLK | TTL In | Reference clock. REFCLK is the clock frequency reference for the clock/data synchronizing PLL. REFCLK sets the approximate center frequency for the internal PLL to track the incoming bit stream. REFCLK must be connected to a crystal-controlled time base that runs within the frequency limits of the Tx/Rx pair, and the frequency must be the same as the transmitter CKW frequency (within CKW ± 0.1%). |
| MODE | Three-Level In | Decoder mode select. The level on the MODE pin determines the decoding method to be used. When wired to GND, MODE selects 8B/10B decoding. When wired to $V_{CC}$, registered shifter contents bypass the decoder and are sent to $Q_{a-j}$ directly. When left floating (internal resistors hold the MODE pin at $V_{CC}/2$) the internal bit clock generator is disabled and INB becomes the bit rate test clock to be used for factory test. In typical applications, MODE is wired to $V_{CC}$ or GND. |

**Table 2. CY7B933 HOTLink Receiver** (continued)

| Name | I/O | Description |
|---|---|---|
| $\overline{BISTEN}$ | TTL In | Built-in self-test enable. When $\overline{BISTEN}$ is LOW the Receiver awaits a D0.0 (sent once per BIST loop) character and begins a continuous test sequence that tests the functionality of the Transmitter, the Receiver, and the link connecting them. In BIST mode the status of the test can be monitored with $\overline{RDY}$ and RVS outputs. In normal use BISTEN is held HIGH or wired to $V_{CC}$. BISTEN has the same timing as $Q_{0–7}$. |
| $V_{CCN}$ | | Power for output drivers. |
| $V_{CCQ}$ | | Power for internal circuitry. |
| GND | | Ground. |

## CY7B923 HOTLink Transmitter Block Diagram Description

### Input Register

The input register holds the data to be processed by the HOTLink transmitter and allows the input timing to be made consistent with standard FIFOs. The input register is clocked by CKW and loaded with information on the $D_{0-7}$, SC/$\overline{D}$, and SVS pins. Two enable inputs ($\overline{ENA}$ and $\overline{ENN}$) allow the user to choose when data is loaded in the register. Asserting Enable, active LOW ($\overline{ENA}$) causes the inputs to be loaded in the register on the rising edge of CKW. If $\overline{ENN}$ (Enable Next, active LOW) is asserted when CKW rises, the data present on the inputs on the next rising edge of CKW are loaded into the Input register. If neither $\overline{ENA}$ nor $\overline{ENN}$ are asserted LOW on the rising edge of CKW, then a SYNC (K28.5) character is sent. These two inputs allow proper timing and function for compatibility with either asynchronous FIFOs or clocked FIFOs without external logic, as shown in Figure 6.

In BIST mode, the input register becomes the signature pattern generator by logically converting the parallel input register into a linear feedback shift register (LFSR). When enabled, this LFSR generates a 511-byte sequence that includes all data and special character codes, including the explicit violation symbols. This pattern provides a predictable but pseudo-random sequence that can be matched to an identical LFSR in the receiver.

### Encoder

The encoder transforms the input data held by the input register into a form more suitable for transmission on a serial interface link. The code used is specified by ANSI X3.230 (Fibre Channel) and the IBM ESCON channel (see the table Valid Data Characters (SC/D = LOW) on page 21). The eight $D_{0–7}$ data inputs are converted to either a data symbol or a special character, depending upon the state of the SC/$\overline{D}$ input. If SC/$\overline{D}$ is HIGH, the data inputs represent a control code and are encoded using the special character code table. If SC/$\overline{D}$ is LOW, the data inputs are converted using the data code table. If a byte time passes with the inputs disabled, the encoder outputs a special character comma K28.5 (or SYNC) that maintains link synchronization. SVS input forces the transmission of a specified violation symbol to allow the user to check the error handling system logic in the controller or for proprietary applications.

The 8B/10B coding function of the encoder can be bypassed for systems that include an external coder or scrambler function as part of the controller. This bypass is controlled by setting the MODE select pin HIGH. When in bypass mode, $D_{a-j}$ (note that the bit order is specified in the fibre channel 8B/10B code) become the ten inputs to the shifter, with $D_a$ being the first bit to be shifted out.

### Shifter

The shifter accepts parallel data from the encoder after each byte time and shifts it to the serial interface output buffers using a PLL multiplied bit clock that runs at 10 times the byte clock rate. Timing for the parallel transfer is controlled by the counter included in the clock generator and is not affected by signal levels or timing at the input pins.

### OutA, OutB, OutC

The serial interface PECL output buffers (ECL100K referenced to +5 V) are the drivers for the serial media. They are all connected to the shifter and contain the same serial data. Two of the output pairs (OUTA± and OUTB±) are controllable by the FOTO input and can be disabled by the system controller to force a logical zero (that is, “light off”) at the outputs. The third output pair (OUTC±) is not affected by FOTO and supplies a continuous data stream suitable for loopback testing of the subsystem.

OUTA± and OUTB± responds to FOTO input changes within a few bit times. However, since FOTO is not synchronized with the transmitter data stream, the outputs will be forced off or turned on at arbitrary points in a transmitted byte. This function is intended to augment an external laser safety controller and as an aid for Receiver PLL testing.

In wire-based systems, control of the outputs may not be required, and FOTO can be strapped LOW. The three outputs are intended to add system and architectural flexibility by offering identical serial bit streams with separate interfaces for redundant connections or for multiple destinations. Unneeded outputs can be wired to VCC to disable and power down the unused output circuitry.

### Clock Generator

The clock generator is an embedded PLL that takes a byte-rate reference clock (CKW) and multiplies it by 10 to create a bit rate clock for driving the serial shifter. The byte rate reference comes from CKW, the rising edge of which clocks data into the input register. This clock must be a crystal referenced pulse stream that has a frequency between the minimum and maximum specified for the HOTLink transmitter/receiver pair. Signals controlled by this block form the bit clock and the timing signals that control internal data transfers between the input register and the shifter.

The read pulse ($\overline{RP}$) is derived from the feedback counter used in the PLL multiplier. It is a byte-rate pulse stream with the proper phase and pulse widths to allow transfer of data from an asynchronous FIFO. Pulse width is independent of CKW duty cycle, since proper phase and duty cycle is maintained by the PLL. The $\overline{RP}$ pulse stream ensures correct data transfers between asynchronous FIFOs and the transmitter input latch with no external logic.

### Test Logic

Test logic includes the initialization and control for the BIST generator, the multiplexer for test mode clock distribution, and control logic to properly select the data encoding. Test logic is discussed in detail in CY7B923 HOTLink Transmitter Operating Mode Description on page 11.

## CY7B933 HOTLink Receiver Block Diagram Description

### Serial Data Inputs

Two pairs of differential line receivers are the inputs for the serial data stream. INA± or INB± can be selected with the A/B input. INA± is selected with A/B HIGH and INB± is selected with A/B LOW. The threshold of A/B is compatible with the ECL 100K signals from PECL fiber optic interface modules. TTL logic elements can be used to select the A or B inputs by adding a resistor pull-up to the TTL driver connected to A/B. The differential threshold of INA± and INB± will accommodate wire interconnect with filtering losses or transmission line attenuation greater than 20 db ($V_{DIF} \geq 50$ mv) or can be directly connected to fiber optic interface modules (any ECL logic family, not limited to ECL 100K). The common mode tolerance will accommodate a wide range of signal termination voltages. The highest HIGH input that can be tolerated is $V_{IN} = V_{CC}$, and the lowest LOW input that can be interpreted correctly is $V_{IN}$ = GND+2.0V.

### PECL-TTL Translator

The function of the INB(INB+) input and the SI(INB–) input is defined by the connections on the SO output pin. If the PECL/TTL translator function is not required, the SO output is wired to VCC. A sensor circuit detects this connection and causes the inputs to become INB± (a differential line-receiver serial-data input). If the PECL/TTL translator function is required, the SO output is connected to its normal TTL load (typically one or more TTL inputs, but no pull-up resistor) and the INB+ input becomes single-ended ECL 100K, serial data input (INB) and the INB– input becomes single-ended, ECL 100K status input (SI).

This positive-referenced PECL-to-TTL translator is provided to eliminate external logic between an PECL fiber-optic interface module “carrier detect” output and the TTL input in the control logic. The input threshold is compatible with ECL 100K levels (+5-V referenced). It can also be used as part of the link status indication logic for wire connected systems.

### Clock Synchronization

The clock synchronization function is performed by an embedded PLL that tracks the frequency of the incoming bit stream and aligns the phase of its internal bit rate clock to the serial data transitions. This block contains the logic to transfer the data from the shifter to the decode register once every byte. The counter that controls this transfer is initialized by the framer logic. CKR is a buffered output derived from the bit counter used to control the decode register and the output register transfers.

Clock output logic is designed so that when reframing causes the counter sequence to be interrupted, the period and pulse width of CKR is never less than normal. Reframing may stretch the period of CKR by up to 90%, and either CKR Pulse Width HIGH or Pulse Width LOW may be stretched, depending on when reframe occurs.

The REFCLK input provides a byte-rate reference frequency to improve PLL acquisition time and limit unlocked frequency excursions of the CKR when no data is present at the serial inputs. The frequency of REFCLK is required to be within ±0.1% of the frequency of the clock that drives the transmitter CKW pin.

### Framer

Framer logic checks the incoming bit stream for the pattern that defines the byte boundaries. This combinatorial logic filter looks for the X3.230 symbol defined as a Special Character Comma (K28.5). When it is found, the free-running bit counter in the Clock Synchronization block is synchronously reset to its initial state, thus framing the data correctly on the correct byte boundaries.

Random errors that occur in the serial data can corrupt some data patterns into a bit pattern identical to a K28.5, and thus cause an erroneous data-framing error. The RF input prevents this by inhibiting reframing during times when normal message data is present. When RF is held LOW, the HOTLink receiver will deserialize the incoming data without trying to reframe the data to incoming patterns. When RF rises, $\overline{RDY}$ will be inhibited until a K28.5 has been detected, after which $\overline{RDY}$ will resume its normal function. While RF is HIGH, it is possible that an error could cause misframing, after which all data will be corrupted. Likewise, a K28.7 followed by D11.x, D20.x, or an SVS (C0.7) followed by D11.x will create alias K28.5 characters and cause erroneous framing. These sequences must be avoided while RF is HIGH.

If RF remains HIGH for greater than 2048 bytes, the framer converts to double-byte framing, requiring two K28.5 characters aligned on the same byte boundary within 5 bytes in order to reframe. Double-byte framing greatly reduces the possibility of erroneously reframing to an aliased K28.5 character.

### Shifter

The shifter accepts serial inputs from the serial data inputs one bit at a time, as clocked by the clock synchronization logic. Data is transferred to the framer on each bit, and to the decode register once per byte.

### Decode Register

The decode register accepts data from the shifter once per byte as determined by the logic in the clock synchronization block. It is presented to the decoder and held until it is transferred to the output latch.

### Decoder

Parallel data is transformed from ANSI-specified X3.230 8B/10B codes back to 'raw data' in the decoder. This block uses the standard decoder patterns shown in Valid Data Characters (SC/D = LOW) on page 21 and Valid Special Character Codes and Sequences (SC/D = HIGH)[1, 2] on page 29. Data patterns are signaled by a LOW on the SC/$\overline{D}$ output and special character patterns are signaled by a HIGH on the SC/$\overline{D}$ output. Unused patterns or disparity errors are signaled as errors by a HIGH on the RVS output and by specific special character codes.

### Output Register

The output register holds the recovered data ($Q_{0-7}$, SC/$\overline{D}$, and RVS) and aligns it with the recovered byte clock (CKR). This synchronization insures proper timing to match a FIFO interface or other logic that requires glitch-free and specified output behavior. Outputs change synchronously with the rising edge of CKR.

In BIST mode, this register becomes the signature pattern generator and checker by logically converting itself into a linear feedback shift register (LFSR) pattern generator. When enabled, this LFSR generates a 511-byte sequence that includes all data and special character codes, including the explicit violation symbols. This pattern provides a predictable but pseudo-random sequence that can be matched to an identical LFSR in the Transmitter. When synchronized, it checks each byte in the Decoder with each byte generated by the LFSR and shows errors at RVS. Patterns generated by the LFSR are compared after being buffered to the output pins and then fed back to the comparators, allowing test of the entire receive function.

In BIST mode, the LFSR is initialized by the first occurrence of the transmitter BIST loop start code D0.0 (D0.0 is sent only once per BIST loop). Once the BIST loop has been started, RVS will be HIGH for pattern mismatches between the received sequence and the internally generated sequence. Code rule violations or running disparity errors that occur as part of the BIST loop will not cause an error indication. $\overline{RDY}$ will pulse HIGH once per BIST loop and can be used to check test pattern progress. The receiver BIST generator can be reinitialized by leaving and re-entering BIST mode.

### Test Logic

Test logic includes the initialization and control for the Built-In Self-Test (BIST) generator, the multiplexer for Test mode clock distribution, and control logic for the decoder. Test logic is discussed in more detail in the CY7B933 HOTLink Receiver Operating Mode Description.

## HOTLink CY7B923 Transmitter and CY7B933 Receiver Operation

The CY7B923 Transmitter operating with the CY7B933 Receiver form a general purpose data communications subsystem capable of transporting user data at up to 33 Mbytes per second (40 Mbytes per second for –400 devices) over several types of serial interface media. Figure 10 on page 34 illustrates the flow of data through the HOTLink CY7B923 transmitter pipeline. Data is latched into the transmitter on the rising edge of CKW when enabled by $\overline{ENA}$ or $\overline{ENN}$. $\overline{RP}$ is asserted LOW with a 60% LOW/40% HIGH duty cycle when $\overline{ENA}$ is LOW. $\overline{RP}$ may be used as a read strobe for accessing data stored in a FIFO. The parallel data flows through the encoder and is then shifted out of the OUTx± PECL drivers. The bit-rate clock is generated internally from a multiply-by-ten PLL clock generator. The latency through the transmitter is approximately $21t_B$ – 10 ns over the operating range. A more complete description is found in the section CY7B923 HOTLink Transmitter Operating Mode Description.

Figure 5 illustrates the data flow through the HOTLink CY7B933 receiver pipeline. Serial data is sampled by the receiver on the INx± inputs. The receiver PLL locks onto the serial bit stream and generates an internal bit rate clock. The bit stream is deserialized, decoded and then presented at the parallel output pins. A byte rate clock (bit clock ÷ 10) synchronous with the parallel data is presented at the CKR pin. The $\overline{RDY}$ pin will be asserted to LOW to indicate that data or control characters are present on the outputs. $\overline{RDY}$ will not be asserted LOW in a field of K28.5s except for any single K28.5 or the last one in a continuous series of K28.5's. The latency through the receiver is approximately $24t_B$ + 10 ns over the operating range. A more complete description of the receiver is in the section CY7B933 HOTLink Receiver Operating Mode Description.

The HOTLink receiver has a built-in byte framer that synchronizes the Receiver pipeline with incoming SYNC (K28.5) characters. Figure 5 illustrates the HOTLink CY7B933 Receiver framing operation. The Framer is enabled when the RF pin is asserted HIGH. RF is latched into the receiver on the falling edge of CKR. The framer looks for K28.5 characters embedded in the serial data stream. When a K28.5 is found, the framer sets the parallel byte boundary for subsequent data to the K28.5 boundary. While the framer is enabled, the $\overline{RDY}$ pin indicates the status of the framing operation.

When the RF pin is asserted HIGH, $\overline{RDY}$ leaves it normal mode of operation and is asserted HIGH while the framer searches the data stream for a K28.5 character. After the framer has synchronized to a K28.5 character, the Receiver will assert the $\overline{RDY}$ pin LOW when the K28.5 character is present at the parallel output. The $\overline{RDY}$ pin will then resume its normal operation as dictated by the MODE and $\overline{BISTEN}$ pins.

The normal operation of the $\overline{RDY}$ pin in encoded mode is to signal when parallel data is present at the output pins by pulsing LOW with a 60% LOW/40% HIGH duty cycle. $\overline{RDY}$ does not pulse LOW in a field of K28.5 characters; however, $\overline{RDY}$ does pulse LOW for the last K28.5 character in the field or for any single K28.5. In unencoded mode, the normal operation of the $\overline{RDY}$ pin is to signal when any K28.5 is at the parallel output pins.

**Figure 4. CY7B933 Receiver Data Pipeline in Encoded Mode**

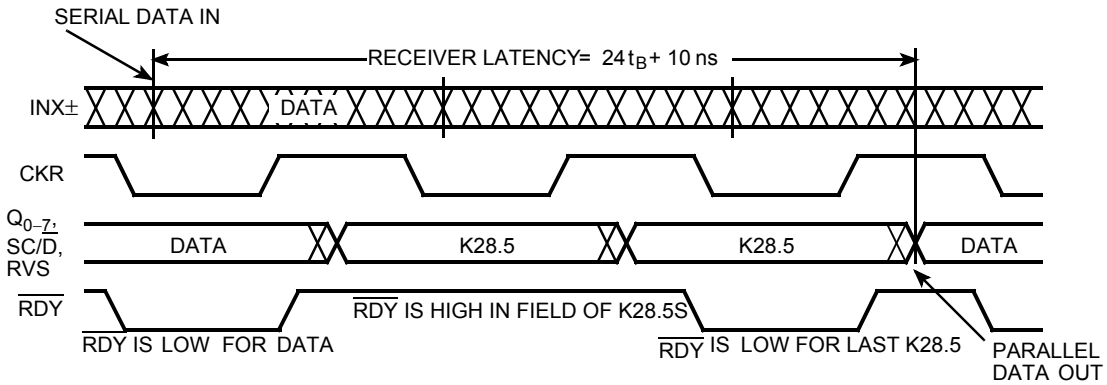
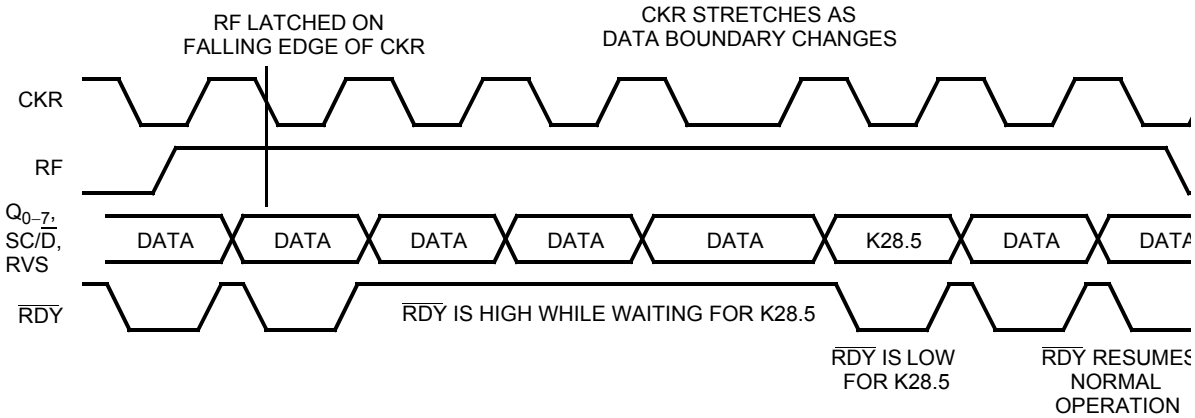



**Figure 5. CY7B933 Framing Operation in Encoded Mode**




The transmitter and receiver parallel interface timing and functionality can be made to match the timing and functionality of either an asynchronous FIFO or a clocked FIFO by appropriately connecting signals (see Figure 6 on page 12). Proper operation of the FIFO interface depends upon various FIFO-specific access and response specifications.

The HOTLink transmitter and receiver serial interface provides a seamless interface to various types of media. A minimal number of external components are needed to properly terminate transmission lines and provide PECL loads. For proper power supply decoupling, a single 0.01 µF for each device is all that is required to bypass the $V_{CC}$ and GND pins. Figure 7 on page 14 illustrates a HOTLink transmitter and receiver interface to fiber-optic and copper media. More information on interfacing HOTLink to various media can be found in the *HOTLink Design Considerations* application note.

## CY7B923 HOTLink Transmitter Operating Mode Description

In normal operation, the transmitter can operate in either of two modes. The encoded mode allows a user to send and receive eight-bit data and control information without first converting it to transmission characters. The bypass mode is used for systems in which the encoding and decoding is performed in an external protocol controller.

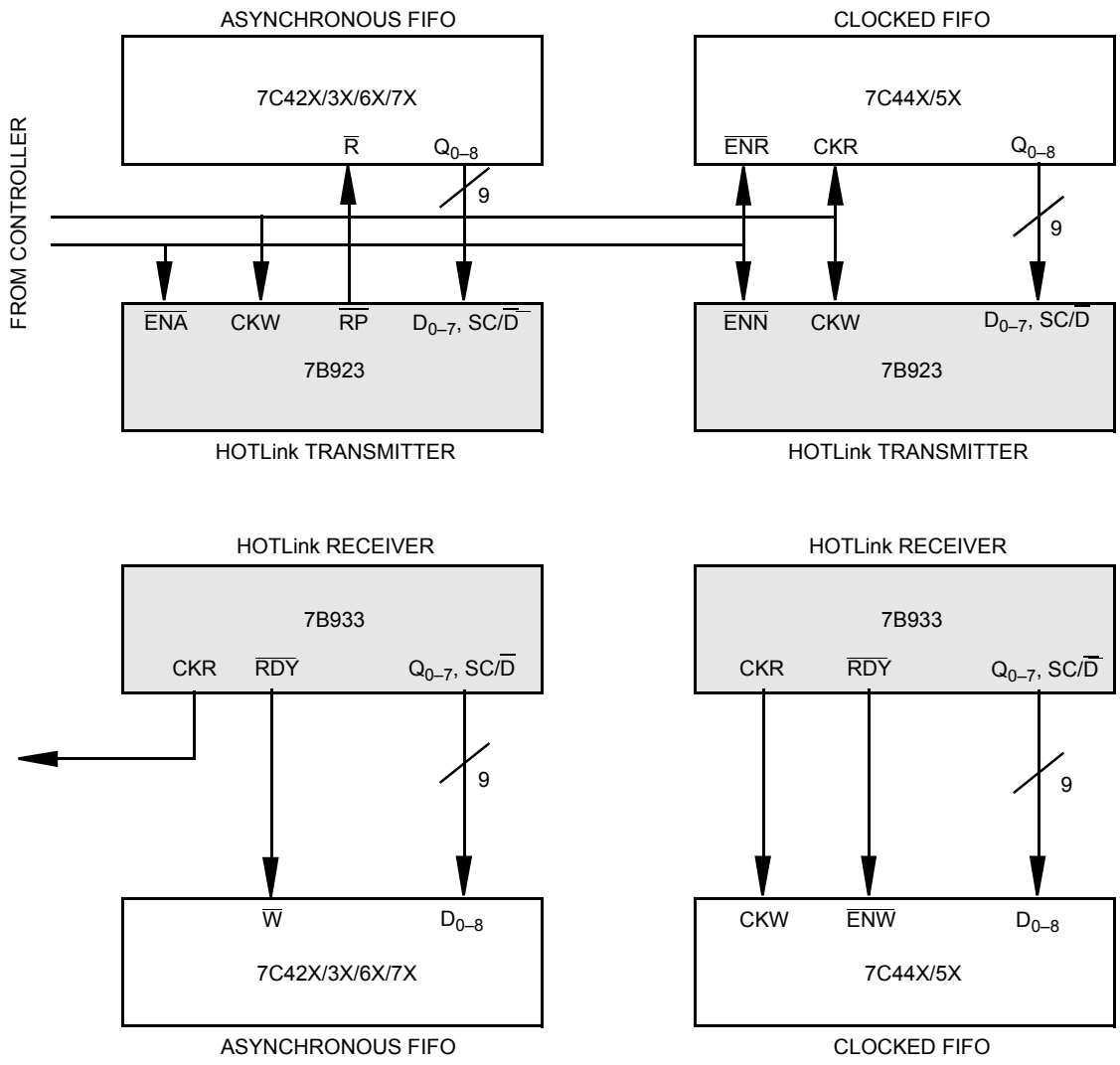
In either mode, data is loaded into the Input register of the Transmitter on the rising edge of CKW. The input timing and functional response of the Transmitter input can be made to match the timing and functionality of either an asynchronous FIFO or a clocked FIFO by an appropriate connection of input signals (see Figure 6 on page 12). Proper operation of the FIFO interface depends upon various FIFO-specific access and response specifications.

## Encoded Mode Operation

In the encoded mode, the input data is interpreted as eight bits of data ($D_0$–$D_7$), a context control bit (SC/$\overline{D}$), and a system diagnostic input bit (SVS). If the context of the data is to be normal message data, the SC/$\overline{D}$ input should be LOW, and the data should be encoded using the valid data character set described in Valid Data Characters (SC/D = LOW) on page 21. If the context of the data is to be control or protocol information, the SC/$\overline{D}$ input is HIGH, and the data is encoded using the valid special character set described in Valid Special Character Codes and Sequences (SC/D = HIGH)[1, 2] on page 29. Special characters include all protocol characters necessary to encode packets for Fibre Channel, ESCON, proprietary systems, and diagnostic purposes.

The diagnostic characters and sequences available as special characters include those for Fibre Channel link testing, as well as codes to be used for testing system response to link errors and timing. A Violation symbol can be explicitly sent as part of a user data packet (i.e., send C0.7; $D_{7-0}$ = 111 00000 and SC/$\overline{D}$ = 1), or it can be sent in response to an external system using the SVS input. This allows the system diagnostic logic to evaluate the errors in an unambiguous manner, and does not require any modification to the transmission interface to force transmission errors for testing purposes.

**Figure 6. Seamless FIFO Interface**



### Bypass Mode Operation

In the bypass mode, the input data is interpreted as 10 bits ($D_{b–h}$), SC/$\overline{D}$ ($D_a$), and SVS ($D_j$) of pre-encoded transmission data to be serialized and sent over the link. This data can use any encoding method suitable to the designer. The only restrictions upon the data encoding method is that it contain suitable transition density for the receiver PLL data synchronizer (one per 10-bit byte), and that it be compatible with the transmission media.

Data loaded into the Input register on the rising edge of CKW is loaded into the shifter on the subsequent rising edges of CKW. It will then be shifted to the outputs one bit at a time using the internal clock generated by the clock generator. The first bit of the transmission character ($D_a$) appears at the output (OUTA±, OUTB±, and OUTC±) after the next CKW edge.

While in either the encoded mode or bypass mode, if a CKW edge arrives when the inputs are not enabled ($\overline{ENA}$ and $\overline{ENN}$ both HIGH), the encoder inserts a pad character K28.5 (for example, C5.0) to maintain proper link synchronization (in the bypass mode the proper sense of running disparity cannot be guaranteed for the first pad character, but is correct for all pad characters that follow). This automatic insertion of pad characters can be inhibited by insuring that the transmitter is always enabled (that is, $\overline{ENA}$ or $\overline{ENN}$ is hard-wired LOW).

### PECL Output Functional and Connection Options

The three pairs of PECL outputs all contain the same information and are intended for use in systems with multiple connections. Each output pair may be connected to a different serial media, each of which may be a different length, link type, or interface technology. For systems that do not require all three output pairs, the unused pairs should be wired to $V_{CC}$ to minimize the power dissipated by the output circuit, and to minimize unwanted noise generation. An internal voltage comparator detects when an output differential pair is wired to $V_{CC}$, causing the current source for that pair to be disabled. This results in a power savings of around 5 mA for each unused pair.

In systems that require the outputs to be shut off during some periods when link transmission is prohibited (for example, for laser safety functions), the FOTO input can be asserted. While it is possible to insure that the output state of the PECL drivers is LOW (that is, light is off) by sending all 0s in bypass mode, it is often inconvenient to insert this level of control into the data transmission channel, and it is impossible in encoded mode. FOTO is provided to simplify and augment this control function (typically found in laser-based transmission systems). FOTO will force OUTA+ and OUTB+ to go LOW, OUTA– and OUTB– to go HIGH, while allowing OUTC± to continue to function normally (OUTC is typically used as a diagnostic feedback and cannot be disabled). This separation of function allows various system configurations without undue load on the control function or data channel logic.

## Transmitter Serial Data Characteristics

The CY7B923 HOTLink transmitter serial output conforms to the requirements of the Fibre Channel specification. The serial data output is controlled by an internal PLL that multiplies the frequency of CKW by 10 to maintain the proper bit clock frequency. The jitter characteristics (including both PLL and logic components) are as follows:

- Deterministic Jitter ($D_j$) < 35 ps (peak-peak). Typically measured while sending a continuous K28.5 (C5.0).
- Random Jitter ($R_j$) < 175 ps (peak-peak). Typically measured while sending a continuous K28.7 (C7.0).

## Transmitter Test Mode Description

The CY7B923 transmitter offers two types of test mode operation, BIST mode and Test mode. In a normal system application, the BIST mode can be used to check the functionality of the transmitter, the receiver, and the link connecting them. This mode is available with minimal impact on user system logic, and can be used as part of the normal system diagnostics. Typical connections and timing are shown in Figure 8 on page 15.

**Figure 7. HOTLink Connection Diagram**

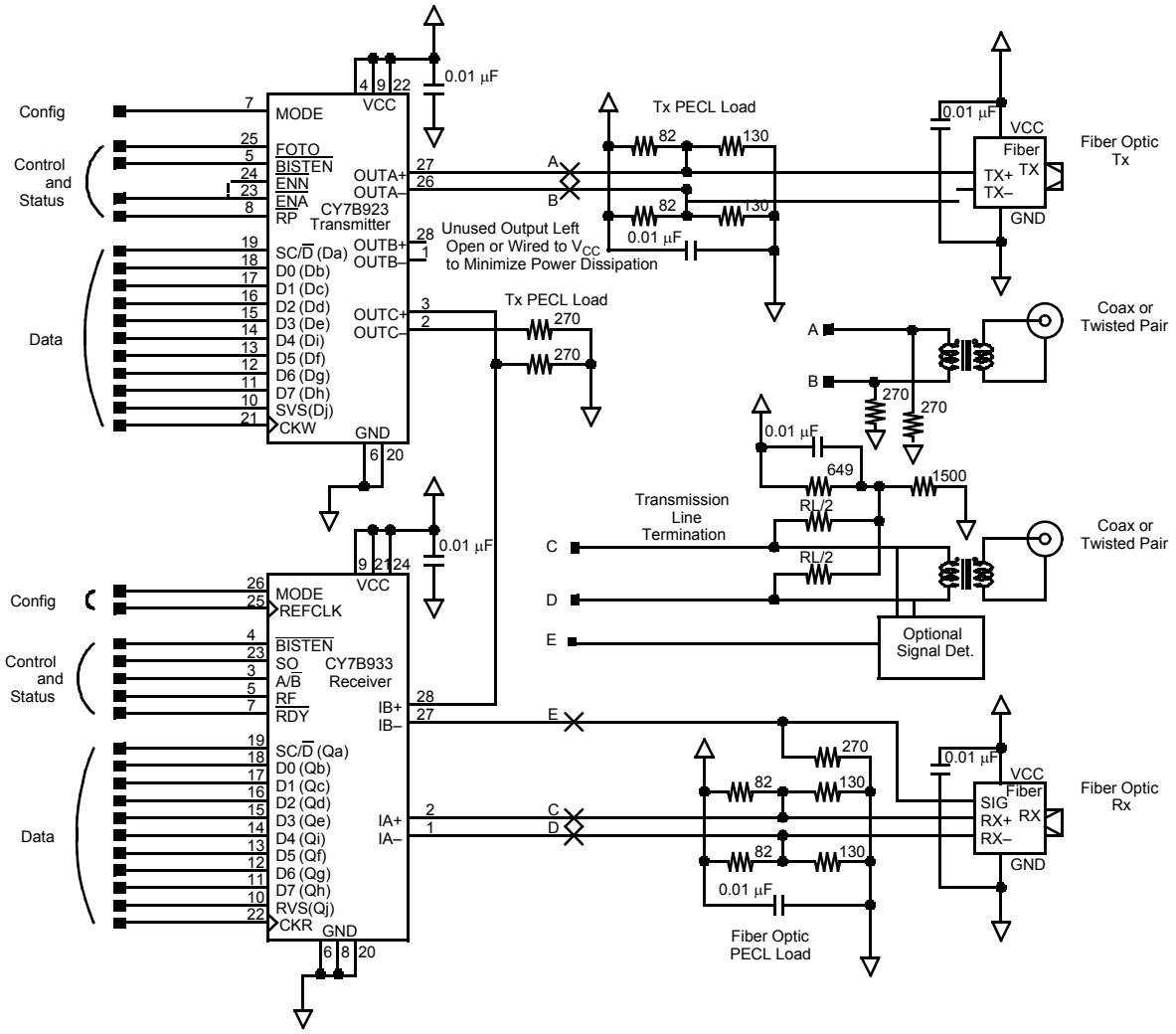
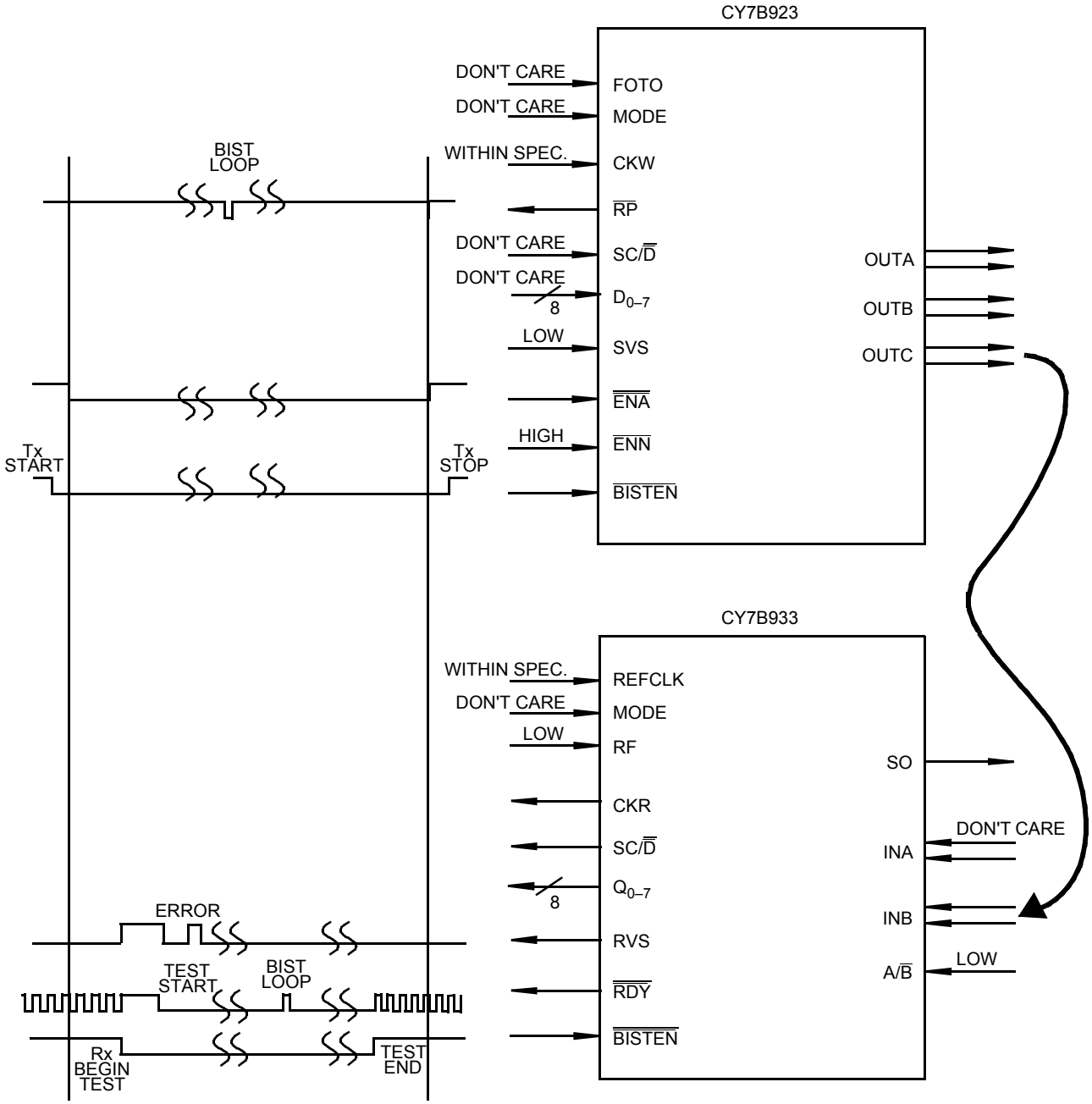


**Figure 8. BIST Illustration**




## BIST Mode

The BIST mode functions as follows:

1. Set $\overline{\text{BISTEN}}$ LOW to begin test pattern generation. The transmitter begins sending bit rate ...1010...
2. Set either $\overline{\text{ENA}}$ or $\overline{\text{ENN}}$ LOW to begin pattern sequence generation (use of the Enable pin not being used for normal FIFO or system interface can minimize logic delays between the controller and transmitter).
3. Allow the transmitter to run through several BIST loops or until the receiver test is complete. $\overline{\text{RP}}$ will pulse LOW once per BIST loop, and can be used by an external counter to monitor the number of test pattern loops.
4. When testing is completed, set $\overline{\text{BISTEN}}$ HIGH and $\overline{\text{ENA}}$ and $\overline{\text{ENN}}$ HIGH and resume normal function.

**Note**: It may be advisable to send violation characters to test the RVS output in the receiver. This can be done by explicitly sending a violation with the SVS input, or allowing the transmitter BIST loop to run while the receiver runs in normal mode. The BIST loop includes deliberate violation symbols and will adequately test the RVS function.

BIST mode is intended to check the entire function of the transmitter (except the transmitter input pins and the bypass function in the Encoder), the serial link, and the receiver. It augments normal factory ATE testing and provides the designer with a rigorous test mechanism to check the link transmission system without requiring any significant system overhead.

While in bypass mode, the BIST logic will function in the same way as in the encoded mode. MODE = HIGH and $\overline{BISTEN}$ = LOW causes the transmitter to switch to encoded mode and begin sending the BIST pattern, as if MODE = LOW. When $\overline{BISTEN}$ returns to HIGH, the Transmitter resumes normal Bypass operation. In Test mode the BIST function works as in the Normal mode. For more information on BIST, consult the "HOTLink Built-In Self-Test" application note.

### Test Mode

The MODE input pin selects between three transmitter functional modes. When wired to $V_{CC}$, the $D_{(a-j)}$ inputs bypass the encoder and load directly from the input register into the shifter. When wired to GND, the inputs $D_{0-7}$, SVS, and SC/$\overline{D}$ are encoded using the Fibre Channel 8B/10B codes and sequences (shown at the end of this datasheet). Since the transmitter is usually hardwired to encoded or bypass mode and not switched between them, a third function is provided for the MODE pin. The test mode is selected by floating the MODE pin (internal resistors hold the MODE pin at $V_{CC}/2$). Test mode is used for factory or incoming device tests.

The test mode causes the transmitter to function in its encoded mode, but with OutA+/OutB+ (used as a differential test clock input) as the bit rate clock input instead of the internal PLL-generated bit clock. In this mode, inputs are clocked by CKW and transfers between the Input register and Shifter are timed by the internal counters. The bit-clock and CKW must maintain a fixed phase and divide-by-ten ratio. The phase and pulse width of $\overline{RP}$ are controlled by phases of the bit counter (PLL feedback counter) as in normal mode. Input and output patterns can be synchronized with internal logic by observing the state of $\overline{RP}$ or the device can be initialized to match an ATE test pattern using the following technique:

1. With the MODE pin either HIGH or LOW, stop CKW and bit-clock.
2. Force the MODE pin to MID (open or $V_{CC}/2$) while the clocks are stopped.
3. Start the bit-clock and let it run for at least two cycles.
4. Start the CKW clock at the bit-clock/10 rate.

Test mode is intended to allow logical, DC, and AC testing of the transmitter without requiring that the tester check output data patterns at the bit rate, or accommodate the PLL lock, tracking, and frequency range characteristics that are required when the HOTLink part operates in its normal mode. To use OutA+/OutB+ as the test clock input, the FOTO input is held HIGH while in Test mode. This forces the two outputs to go to a 'PECL LOW' which can be ignored while the test system creates a differential input signal at some higher voltage.

## CY7B933 HOTLink Receiver Operating Mode Description

In normal user operation, the receiver can operate in either of two modes. The encoded mode allows a user system to send and receive eight-bit data and control information without first converting it to transmission characters. The bypass mode is used for systems in which the encoding and decoding is performed by an external protocol controller.

In either mode, serial data is received at one of the differential line receiver inputs and routed to the shifter and the clock synchronization. The PLL in the clock synchronizer aligns the internally generated bit rate clock with the incoming data stream and clocks the data into the shifter. At the end of a byte time (ten bit times), the data accumulated in the shifter is transferred to the decode register.

To properly align the incoming bit stream to the intended byte boundaries, the bit counter in the clock synchronizer must be initialized. The framer logic block checks the incoming bit stream for the unique pattern that defines the byte boundaries. This combinatorial logic filter looks for the X3.230 symbol defined as 'Special Character Comma' (K28.5). After K28.5 is found, the free running bit counter in the clock synchronizer block is synchronously reset to its initial state, thus 'framing' the data to the correct byte boundaries.

Since noise-induced errors can cause the incoming data to be corrupted, and because many combinations of error and legal data can create an alias K28.5, an option is included to disable resynchronization of the bit counter. The framer will be inhibited when the RF input is held LOW. When RF rises, $\overline{RDY}$ will be inhibited until a K28.5 has been detected, and $\overline{RDY}$ will resume its normal function. Data will continue to flow through the Receiver while $\overline{RDY}$ is inhibited.

### Encoded Mode Operation

In encoded mode the serial input data is decoded into eight bits of data ($Q_0$–$Q_7$), a context control bit (SC/$\overline{D}$), and a system diagnostic output bit (RVS). If the pattern in the decode register is found in the Valid Data Characters table, the context of the data is decoded as normal message data and the SC/$\overline{D}$ output will be LOW. If the incoming bit pattern is found in the Valid Special Character Codes and Sequences table, it is interpreted as 'control' or 'protocol information', and the SC/$\overline{D}$ output will be HIGH. Special characters include all protocol characters defined for use in packets for Fibre Channel, ESCON, and other proprietary and diagnostic purposes.

The violation symbol that can be explicitly sent as part of a user data packet (that is, transmitter sending C0.7; $D_{7-0}$ = 111 00000 and SC/$\overline{D}$ = 1; or SVS = 1) will be decoded and indicated in exactly the same way as a noise-induced error in the transmission link. This function will allow system diagnostics to evaluate the error in an unambiguous manner, and will not require any modification to the receiver data interface for error-testing purposes.

### Bypass Mode Operation

In bypass mode the serial input data is not decoded, and is transferred directly from the decode register to the output register's 10 bits ($Q_{(a-j)}$). It is assumed that the data has been preencoded prior to transmission, and will be decoded in subsequent logic external to HOTLink. This data can use any encoding method suitable to the designer. The only restrictions upon the data encoding method is that it contain suitable transition density for the Receiver PLL data synchronizer (one per 10-bit byte) and that it be compatible with the transmission media.

The framer function in bypass mode is identical to encoded mode, so a K28.5 pattern can still be used to reframe the serial bit stream.

### Parallel Output Function

The 10 outputs ($Q_{0-7}$, SC/$\overline{D}$, and RVS) all transition simultaneously, and are aligned with $\overline{RDY}$ and CKR with timing allowances to interface directly with either an asynchronous FIFO or a clocked FIFO. Typical FIFO connections are shown in Figure 6 on page 12.

Data outputs can be clocked into the system using either the rising or falling edge of CKR, or the rising or falling edge of $\overline{RDY}$. If CKR is used, $\overline{RDY}$ can be used as an enable for the receiving logic. A LOW pulse on $\overline{RDY}$ shows that new data has been received and is ready to be delivered. The signal on $\overline{RDY}$ is a 60%-LOW duty cycle byte-rate pulse train suitable for the write pulse in asynchronous FIFOs such as the CY7C42X, or the enable write input on Clocked FIFOs such as the CY7C44X. HIGH on $\overline{RDY}$ shows that the received data appearing at the outputs is the null character (normally inserted by the transmitter as a pad between data inputs) and should be ignored.

When the transmitter is disabled it will continuously send pad characters (K28.5). To assure that the receive FIFO will not be overfilled with these dummy bytes, the $\overline{RDY}$ pulse output is inhibited during fill strings. Data at the $Q_{0-7}$ outputs will reflect the correct received data, but will not appear to change, since a string of K28.5s all are decoded as $Q_{7-0}$ =000 00101 and SC/D = 1 (C5.0). When new data appears (not K28.5), the $\overline{RDY}$ output will resume normal function. The "last" K28.5 will be accompanied by a normal $\overline{RDY}$ pulse.

Fill characters are defined as any K28.5 followed by another K28.5. All fill characters will not cause $\overline{RDY}$ to pulse. Any K28.5 followed by any other character (including violation or illegal characters) will be interpreted as usable data and will cause $\overline{RDY}$ to pulse.

As noted above, $\overline{RDY}$ can also be used as an indication of correct framing of received data. While the receiver is awaiting receipt of a K28.5 with RF HIGH, the $\overline{RDY}$ outputs will be inhibited. When $\overline{RDY}$ resumes, the received data will be properly framed and will be decoded correctly. In Bypass mode with RF HIGH, $\overline{RDY}$ will pulse once for each K28.5 received. For more information on the $\overline{RDY}$ pin, consult the "HOTLink CY7B933 $\overline{RDY}$ Pin Description" application note.

Code rule violations and reception errors will be indicated as follows:

| | RVS | SC/$\overline{D}$ | Qouts | Name |
|---|---|---|---|---|
| 1. Good Data code received with good running disparity (RD) | 0 | 0 | 00-FFD | 0.0-31.7 |
| 2. Good Special Character code received with good RD | 0 | 1 | 00-0BC | 0.0-11.0 |
| 3. K28.7 immediately following K28.1 (ESCON Connect_SOF) | 0 | 1 | 27 | C7.1 |
| 4. K28.7 immediately following K28.5 (ESCON Passive_SOF) | 0 | 1 | 47 | C7.2 |
| 5. Unassigned code received | 1 | 1 | E0 | C0.7 |
| 6. -K28.5+ received when RD was + | 1 | 1 | E1 | C1.7 |
| 7. +K28.5– received when RD was – | 1 | 1 | E2 | C2.7 |
| 8. Good code received with wrong RD | 1 | 1 | E4 | C4.7 |

## Receiver Serial Data Requirements

The CY7B933 HOTLink Receiver serial input capability conforms to the requirements of the Fibre Channel specification. The serial data input is tracked by an internal PLL that is used to recover the clock phase and to extract the data from the serial bit stream. Jitter tolerance characteristics (including both PLL and logic component requirements) are shown below:

- Deterministic Jitter Tolerance (Dj) > 40% of $t_B$. Typically measured while receiving data carried by a bandwidth-limited channel (e.g., a coaxial transmission line) while maintaining a Bit Error Rate (BER) < 10–12.
- Random Jitter Tolerance (Rj) > 90% of $t_B$. Typically measured while receiving data carried by a random-noise-limited channel (e.g., a fiber-optic transmission system with low light levels) while maintaining a Bit Error Rate (BER) < 10–12.
- Total Jitter Tolerance > 90% of $t_B$. Total of Dj + Rj.
- PLL-acquisition time < 500-bit times from worst-case phase or frequency change in the serial input data stream, to receiving data within BER objective of 10–12. Stable power supplies within specifications, stable REFCLK input frequency and normal data framing protocols are assumed.
  **Note** Acquisition time is measured from worst-case phase or frequency change to zero phase and frequency error. As a result of the receiver's wide jitter tolerance, valid data appears at the receiver's outputs a few byte times after a worst-case phase change.

## Receiver Test Mode Description

The CY7B933 receiver offers two types of test mode operation, BIST mode and Test mode. In a normal system application, the Built-In Self-Test (BIST) mode can be used to check the functionality of the transmitter, the receiver and the link connecting them. This mode is available with minimal impact on user system logic, and can be used as part of the normal system diagnostics. Typical connections and timing are shown in Figure 8 on page 15.

### BIST Mode

The BIST mode function is as follows:

1. Set $\overline{\text{BISTEN}}$ LOW to enable self-test generation and await $\overline{\text{RDY}}$ LOW indicating that the initialization code has been received.
2. Monitor RVS and check for any byte time with the pin HIGH to detect pattern mismatches. $\overline{\text{RDY}}$ will pulse HIGH once per BIST loop, and can be used by an external counter to monitor test pattern progress. $Q_{0-7}$ and SC/$\overline{\text{D}}$ will show the expected pattern and may be useful for debug purposes.
3. When testing is completed, set $\overline{\text{BISTEN}}$ HIGH and resume normal function.

**Note** A specific test of the RVS output may be required to assure an adequate test. To perform this test, it is only necessary to have the transmitter send violation (SVS = HIGH) for a few bytes before beginning the BIST test sequence. Alternatively, the receiver could enter BIST mode after the transmitter has begun sending BIST loop data, or be removed before the transmitter finishes sending BIST loops, each of which contain several deliberate violations and should cause RVS to pulse HIGH.

BIST mode is intended to check the entire function of the Transmitter, serial link, and receiver. It augments normal factory ATE testing and provides the user system with a rigorous test mechanism to check the link transmission system, without requiring any significant system overhead.

When in bypass mode, the BIST logic will function in the same way as in the encoded mode. MODE = HIGH and $\overline{\text{BISTEN}}$ = LOW causes the receiver to switch to encoded mode and begin checking the decoded received data of the BIST pattern, as if MODE = LOW. When $\overline{\text{BISTEN}}$ returns to HIGH, the receiver resumes normal bypass operation. In test mode the BIST function works as in the normal mode.

### Test Mode

The MODE input pin selects between three receiver functional modes. When wired to VCC, the shifter contents bypass the decoder and go directly from the decoder latch to the $Q_{a-j}$ inputs of the output latch. When wired to GND, the outputs are decoded using the 8B/10B codes shown at the end of this datasheet and become $Q_{0-7}$, RVS, and SC/$\overline{\text{D}}$. The third function is test mode, used for factory or incoming device test. This mode can be selected by leaving the MODE pin open (internal circuitry forces the open pin to $V_{CC}/2$).

Test mode causes the Receiver to function in its Encoded mode, but with INB (INB+) as the bit rate Test clock instead of the Internal PLL generated bit clock. In this mode, transfers between the Shifter, Decoder register and Output register are controlled by their normal logic, but with an external bit rate clock instead of the PLL (the recovered bit clock). Internal logic and test pattern inputs can be synchronized by sending a SYNC pattern and allowing the Framer to align the logic to the bit stream. The flow is as follows:

1. Assert Test mode for several test clock cycles to establish normal counter sequence.
2. Assert RF to enable reframing.
3. Input a repeating sequence of bits representing K28.5 (Sync).
4. $\overline{\text{RDY}}$ falling shows the byte boundary established by the K28.5 input pattern.
5. Proceed with pattern, voltage and timing tests as is convenient for the test program and tester to be used.

(While in Test mode and in BIST mode with RF HIGH, the $Q_{0-7}$, RVS, and SC/$\overline{\text{D}}$ outputs reflect various internal logic states and not the received data.)

Test mode is intended to allow logical, DC, and AC testing of the Receiver without requiring that the tester generate input data at the bit rate or accommodate the PLL lock, tracking and frequency range characteristics that are required when the part operates in its normal mode.

## X3.230 Codes and Notation Conventions

Information to be transmitted over a serial link is encoded eight bits at a time into a 10-bit Transmission Character and then sent serially, bit by bit. Information received over a serial link is collected ten bits at a time, and those Transmission Characters that are used for data (Data Characters) are decoded into the correct eight-bit codes. The 10-bit Transmission Code supports all 256 8-bit combinations. Some of the remaining Transmission Characters (Special Characters) are used for functions other than data transmission.

The primary rationale for use of a Transmission Code is to improve the transmission characteristics of a serial link. The encoding defined by the Transmission Code ensures that sufficient transitions are present in the serial bit stream to make clock recovery possible at the Receiver. Such encoding also greatly increases the likelihood of detecting any single or multiple bit errors that may occur during transmission and reception of information. In addition, some Special Characters of the Transmission Code selected by Fibre Channel Standard consist of a distinct and easily recognizable bit pattern (the Special Character Comma) that assists a Receiver in achieving word alignment on the incoming bit stream.

### Notation Conventions

The documentation for the 8B/10B Transmission Code uses letter notation for the bits in an 8-bit byte. Fibre Channel Standard notation uses a bit notation of A, B, C, D, E, F, G, H for the 8-bit byte for the raw 8-bit data, and the letters a, b, c, d, e, i, f, g, h, j for encoded 10-bit data. There is a correspondence between bit A and bit a, B and b, C and c, D and d, E and e, F and f, G and g, and H and h. Bits i and j are derived, respectively, from (A,B,C,D,E) and (F,G,H).

The bit labeled A in the description of the 8B/10B Transmission Code corresponds to bit 0 in the numbering scheme of the FC-2 specification, B corresponds to bit 1, as shown below.

```
FC-2 bit designation—76543  21 0
HOTLink D/Q designation—76543210
8B/10B bit designation—HGFEDCBA
```

To clarify this correspondence, the following example shows the conversion from an FC-2 Valid Data Byte to a Transmission Character (using 8B/10B Transmission Code notation)

```
FC-2  45
Bits: 76543210
      01000101
```

Converted to 8B/10B notation (note carefully that the order of bits is reversed):

```
Data Byte NameD5.2
Bits: ABCDEFGH
      10100010
```

Translated to a transmission Character in the 8B/10B Transmission Code:

```
Bits: abcdeifghj
      1010010101
```

Each valid Transmission Character of the 8B/10B Transmission Code has been given a name using the following convention: cxx.y, where c is used to show whether the Transmission Character is a Data Character (c is set to D, and the SC/$\overline{D}$ pin is LOW) or a Special Character (c is set to K, and the SC/$\overline{D}$ pin is HIGH). When c is set to D, xx is the decimal value of the binary number composed of the bits E, D, C, B, and A in that order, and the y is the decimal value of the binary number composed of the bits H, G, and F in that order. When c is set to K, xx and y are derived by comparing the encoded bit patterns of the Special Character to those patterns derived from encoded Valid Data bytes and selecting the names of the patterns most similar to the encoded bit patterns of the Special Character.

Under the above conventions, the Transmission Character used for the examples above, is referred to by the name D5.2. The Special Character K29.7 is so named because the first six bits (abcdei) of this character make up a bit pattern similar to that resulting from the encoding of the unencoded 11101 pattern (29), and because the second four bits (fghj) make up a bit pattern similar to that resulting from the encoding of the unencoded 111 pattern (7).

**Note**: This definition of the 10-bit Transmission Code is based on (and is in basic agreement with) the following references, which describe the same 10-bit transmission code.

A.X. Widmer and P.A. Franaszek. “A DC-Balanced, Partitioned-Block, 8B/10B Transmission Code” *IBM Journal of Research and Development*, 27, No. 5: 440-451 (September, 1983).

U.S. Patent 4, 486, 739. Peter A. Franaszek and Albert X. Widmer. “Byte-Oriented DC Balanced (0.4) 8B/10B Partitioned Block Transmission Code” (December 4, 1984).

Fibre Channel Physical and Signaling Interface (dpANS X3.230-199X ANSI FC-PH Standard).

IBM Enterprise Systems Architecture/390 ESCON I/O Interface (document number SA22-7202).

## 8B/10B Transmission Code

The following information describes how the tables shall be used for both generating valid Transmission Characters (encoding) and checking the validity of received Transmission Characters (decoding). It also specifies the ordering rules to be followed when transmitting the bits within a character and the characters within the higher-level constructs specified by the standard.

## Transmission Order

Within the definition of the 8B/10B Transmission Code, the bit positions of the Transmission Characters are labeled a, b, c, d, e, i, f, g, h, j. Bit “a” shall be transmitted first followed by bits b, c, d, e, i, f, g, h, and j in that order. (Note that bit i shall be transmitted between bit e and bit f, rather than in alphabetical order.)

## Valid and Invalid Transmission Characters

The following tables define the valid Data Characters and valid Special Characters (K characters), respectively. The tables are used for both generating valid Transmission Characters (encoding) and checking the validity of received Transmission Characters (decoding). In the tables, each Valid-Data-byte or Special-Character-code entry has two columns that represent two (not necessarily different) Transmission Characters. The two columns correspond to the current value of the running disparity (“Current RD–” or “Current RD+”). Running disparity is a binary parameter with either the value negative (–) or the value positive (+).

After powering on, the Transmitter may assume either a positive or negative value for its initial running disparity. Upon transmission of any Transmission Character, the transmitter will select the proper version of the Transmission Character based on the current running disparity value, and the Transmitter shall calculate a new value for its running disparity based on the contents of the transmitted character. Special Character codes C1.7 and C2.7 can be used to force the transmission of a specific Special Character with a specific running disparity as required for some special sequences in X3.230.

After powering on, the Receiver may assume either a positive or negative value for its initial running disparity. Upon reception of any Transmission Character, the Receiver shall decide whether the Transmission Character is valid or invalid according to the following rules and tables and shall calculate a new value for its Running Disparity based on the contents of the received character.

The following rules for running disparity shall be used to calculate the new running-disparity value for Transmission Characters that have been transmitted (Transmitter’s running disparity) and that have been received (Receiver’s running disparity).

Running disparity for a Transmission Character shall be calculated from sub-blocks, where the first six bits (abcdei) form one sub-block and the second four bits (fghj) form the other sub-block. Running disparity at the beginning of the 6-bit sub-block is the running disparity at the end of the previous Transmission Character. Running disparity at the beginning of the 4-bit sub-block is the running disparity at the end of the 6-bit sub-block. Running disparity at the end of the Transmission Character is the running disparity at the end of the 4-bit sub-block.

Running disparity for the sub-blocks shall be calculated as follows:

1. Running disparity at the end of any sub-block is positive if the sub-block contains more ones than zeros. It is also positive at the end of the 6-bit sub-block if the 6-bit sub-block is 000111, and it is positive at the end of the 4-bit sub-block if the 4-bit sub-block is 0011.
2. Running disparity at the end of any sub-block is negative if the sub-block contains more zeros than ones. It is also negative at the end of the 6-bit sub-block if the 6-bit sub-block is 111000, and it is negative at the end of the 4-bit sub-block if the 4-bit sub-block is 1100.
3. Otherwise, running disparity at the end of the sub-block is the same as at the beginning of the sub-block.

### Using the Tables for Generating Transmission Characters

The appropriate entry in the table shall be found for the valid data byte or the special character byte for which a transmission character is to be generated (encoded). The current value of the transmitter's running disparity shall be used to select the transmission character from its corresponding column. For each transmission character transmitted, a new value of the running disparity shall be calculated. This new value shall be used as the transmitter's current running disparity for the next valid data byte or special character byte to be encoded and transmitted. Table 3 shows naming notations and examples of valid transmission characters.

### Using the Tables for Checking the Validity of Received Transmission Characters

The column corresponding to the current value of the receiver's running disparity shall be searched for the received transmission character. If the received transmission character is found in the proper column, then the transmission character is valid and the associated data byte or special character code is determined (decoded). If the received transmission character is not found in that column, then the transmission character is invalid. This is called a code violation. Independent of the transmission character's validity, the received transmission character shall be used to calculate a new value of running disparity. The new value shall be used as the receiver's current running disparity for the next received transmission character.

**Table 3. Valid Transmission Characters**

| Data | | | |
|---|---|---|---|
| Byte Name | $D_{IN}$ or $Q_{OUT}$ | | Hex Value |
| | 765 | 43210 | |
| D0.0 | 000 | 00000 | 00 |
| D1.0 | 000 | 00001 | 01 |
| D2.0 | 000 | 00010 | 02 |
| . . | . . | . . | . . |
| D5.2 | 010 | 000101 | 45 |
| . . | . . | . . | . . |
| D30.7 | 111 | 11110 | FE |
| D31.7 | 111 | 11111 | FF |

Detection of a code violation does not necessarily show that the transmission character in which the code violation was detected is in error. Code violations may result from a prior error that altered the running disparity of the bit stream which did not result in a detectable error at the transmission character in which the error occurred. Table 4 shows an example of this behavior.

**Table 4. Code Violations Resulting from Prior Errors**

| Description | RD | Character | RD | Character | RD | Character | RD |
|---|---|---|---|---|---|---|---|
| Transmitted data character | – | D21.1 | – | D10.2 | – | D23.5 | + |
| Transmitted bit stream | – | 101010 1001 | – | 010101 0101 | – | 111010 1010 | + |
| Bit stream after error | – | 101010 1011 | + | 010101 0101 | + | 111010 1010 | + |
| Decoded data character | – | D21.0 | + | D10.2 | + | Code Violation | + |

## Valid Data Characters (SC/$\overline{D}$ = LOW)

| Data Byte Name | Bits | | Current RD– | | Current RD+ | |
|---|---|---|---|---|---|---|
| | HGF | EDCBA | abcdei | fghj | abcdei | fghj |
| D0.0 | 000 | 00000 | 100111 | 0100 | 011000 | 1011 |
| D1.0 | 000 | 00001 | 011101 | 0100 | 100010 | 1011 |
| D2.0 | 000 | 00010 | 101101 | 0100 | 010010 | 1011 |
| D3.0 | 000 | 00011 | 110001 | 1011 | 110001 | 0100 |
| D4.0 | 000 | 00100 | 110101 | 0100 | 001010 | 1011 |
| D5.0 | 000 | 00101 | 101001 | 1011 | 101001 | 0100 |
| D6.0 | 000 | 00110 | 011001 | 1011 | 011001 | 0100 |
| D7.0 | 000 | 00111 | 111000 | 1011 | 000111 | 0100 |
| D8.0 | 000 | 01000 | 111001 | 0100 | 000110 | 1011 |
| D9.0 | 000 | 01001 | 100101 | 1011 | 100101 | 0100 |
| D10.0 | 000 | 01010 | 010101 | 1011 | 010101 | 0100 |
| D11.0 | 000 | 01011 | 110100 | 1011 | 110100 | 0100 |
| D12.0 | 000 | 01100 | 001101 | 1011 | 001101 | 0100 |
| D13.0 | 000 | 01101 | 101100 | 1011 | 101100 | 0100 |
| D14.0 | 000 | 01110 | 011100 | 1011 | 011100 | 0100 |
| D15.0 | 000 | 01111 | 010111 | 0100 | 101000 | 1011 |
| D16.0 | 000 | 10000 | 011011 | 0100 | 100100 | 1011 |
| D17.0 | 000 | 10001 | 100011 | 1011 | 100011 | 0100 |
| D18.0 | 000 | 10010 | 010011 | 1011 | 010011 | 0100 |
| D19.0 | 000 | 10011 | 110010 | 1011 | 110010 | 0100 |
| D20.0 | 000 | 10100 | 001011 | 1011 | 001011 | 0100 |
| D21.0 | 000 | 10101 | 101010 | 1011 | 101010 | 0100 |
| D22.0 | 000 | 10110 | 011010 | 1011 | 011010 | 0100 |
| D23.0 | 000 | 10111 | 111010 | 0100 | 000101 | 1011 |
| D24.0 | 000 | 11000 | 110011 | 0100 | 001100 | 1011 |
| D25.0 | 000 | 11001 | 100110 | 1011 | 100110 | 0100 |
| D26.0 | 000 | 11010 | 010110 | 1011 | 010110 | 0100 |
| D27.0 | 000 | 11011 | 110110 | 0100 | 001001 | 1011 |
| D28.0 | 000 | 11100 | 001110 | 1011 | 001110 | 0100 |
| D29.0 | 000 | 11101 | 101110 | 0100 | 010001 | 1011 |
| D30.0 | 000 | 11110 | 011110 | 0100 | 100001 | 1011 |
| D31.0 | 000 | 11111 | 101011 | 0100 | 010100 | 1011 |

## Valid Data Characters (SC/$\overline{D}$ = LOW) (continued)

| Data Byte Name | Bits | | Current RD– | | Current RD+ | |
|---|---|---|---|---|---|---|
| | HGF | EDCBA | abcdei | fghj | abcdei | fghj |
| D0.1 | 001 | 00000 | 100111 | 1001 | 011000 | 1001 |
| D1.1 | 001 | 00001 | 011101 | 1001 | 100010 | 1001 |
| D2.1 | 001 | 00010 | 101101 | 1001 | 010010 | 1001 |
| D3.1 | 001 | 00011 | 110001 | 1001 | 110001 | 1001 |
| D4.1 | 001 | 00100 | 110101 | 1001 | 001010 | 1001 |
| D5.1 | 001 | 00101 | 101001 | 1001 | 101001 | 1001 |
| D6.1 | 001 | 00110 | 011001 | 1001 | 011001 | 1001 |
| D7.1 | 001 | 00111 | 111000 | 1001 | 000111 | 1001 |
| D8.1 | 001 | 01000 | 111001 | 1001 | 000110 | 1001 |
| D9.1 | 001 | 01001 | 100101 | 1001 | 100101 | 1001 |
| D10.1 | 001 | 01010 | 010101 | 1001 | 010101 | 1001 |
| D11.1 | 001 | 01011 | 110100 | 1001 | 110100 | 1001 |
| D12.1 | 001 | 01100 | 001101 | 1001 | 001101 | 1001 |
| D13.1 | 001 | 01101 | 101100 | 1001 | 101100 | 1001 |
| D14.1 | 001 | 01110 | 011100 | 1001 | 011100 | 1001 |
| D15.1 | 001 | 01111 | 010111 | 1001 | 101000 | 1001 |
| D16.1 | 001 | 10000 | 011011 | 1001 | 100100 | 1001 |
| D17.1 | 001 | 10001 | 100011 | 1001 | 100011 | 1001 |
| D18.1 | 001 | 10010 | 010011 | 1001 | 010011 | 1001 |
| D19.1 | 001 | 10011 | 110010 | 1001 | 110010 | 1001 |
| D20.1 | 001 | 10100 | 001011 | 1001 | 001011 | 1001 |
| D21.1 | 001 | 10101 | 101010 | 1001 | 101010 | 1001 |
| D22.1 | 001 | 10110 | 011010 | 1001 | 011010 | 1001 |
| D23.1 | 001 | 10111 | 111010 | 1001 | 000101 | 1001 |
| D24.1 | 001 | 11000 | 110011 | 1001 | 001100 | 1001 |
| D25.1 | 001 | 11001 | 100110 | 1001 | 100110 | 1001 |
| D26.1 | 001 | 11010 | 010110 | 1001 | 010110 | 1001 |
| D27.1 | 001 | 11011 | 110110 | 1001 | 001001 | 1001 |
| D28.1 | 001 | 11100 | 001110 | 1001 | 001110 | 1001 |
| D29.1 | 001 | 11101 | 101110 | 1001 | 010001 | 1001 |
| D30.1 | 001 | 11110 | 011110 | 1001 | 100001 | 1001 |
| D31.1 | 001 | 11111 | 101011 | 1001 | 010100 | 1001 |
| D0.2 | 010 | 00000 | 100111 | 0101 | 011000 | 0101 |

## Valid Data Characters (SC/$\overline{D}$ = LOW) (continued)

| Data Byte Name | Bits | | Current RD– | | Current RD+ | |
|---|---|---|---|---|---|---|
| | HGF | EDCBA | abcdei | fghj | abcdei | fghj |
| D1.2 | 010 | 00001 | 011101 | 0101 | 100010 | 0101 |
| D2.2 | 010 | 00010 | 101101 | 0101 | 010010 | 0101 |
| D3.2 | 010 | 00011 | 110001 | 0101 | 110001 | 0101 |
| D4.2 | 010 | 00100 | 110101 | 0101 | 001010 | 0101 |
| D5.2 | 010 | 00101 | 101001 | 0101 | 101001 | 0101 |
| D6.2 | 010 | 00110 | 011001 | 0101 | 011001 | 0101 |
| D7.2 | 010 | 00111 | 111000 | 0101 | 000111 | 0101 |
| D8.2 | 010 | 01000 | 111001 | 0101 | 000110 | 0101 |
| D9.2 | 010 | 01001 | 100101 | 0101 | 100101 | 0101 |
| D10.2 | 010 | 01010 | 010101 | 0101 | 010101 | 0101 |
| D11.2 | 010 | 01011 | 110100 | 0101 | 110100 | 0101 |
| D12.2 | 010 | 01100 | 001101 | 0101 | 001101 | 0101 |
| D13.2 | 010 | 01101 | 101100 | 0101 | 101100 | 0101 |
| D14.2 | 010 | 01110 | 011100 | 0101 | 011100 | 0101 |
| D15.2 | 010 | 01111 | 010111 | 0101 | 101000 | 0101 |
| D16.2 | 010 | 10000 | 011011 | 0101 | 100100 | 0101 |
| D17.2 | 010 | 10001 | 100011 | 0101 | 100011 | 0101 |
| D18.2 | 010 | 10010 | 010011 | 0101 | 010011 | 0101 |
| D19.2 | 010 | 10011 | 110010 | 0101 | 110010 | 0101 |
| D20.2 | 010 | 10100 | 001011 | 0101 | 001011 | 0101 |
| D21.2 | 010 | 10101 | 101010 | 0101 | 101010 | 0101 |
| D22.2 | 010 | 10110 | 011010 | 0101 | 011010 | 0101 |
| D23.2 | 010 | 10111 | 111010 | 0101 | 000101 | 0101 |
| D24.2 | 010 | 11000 | 110011 | 0101 | 001100 | 0101 |
| D25.2 | 010 | 11001 | 100110 | 0101 | 100110 | 0101 |
| D26.2 | 010 | 11010 | 010110 | 0101 | 010110 | 0101 |
| D27.2 | 010 | 11011 | 110110 | 0101 | 001001 | 0101 |
| D28.2 | 010 | 11100 | 001110 | 0101 | 001110 | 0101 |
| D29.2 | 010 | 11101 | 101110 | 0101 | 010001 | 0101 |
| D30.2 | 010 | 11110 | 011110 | 0101 | 100001 | 0101 |
| D31.2 | 010 | 11111 | 101011 | 0101 | 010100 | 0101 |
| D0.3 | 011 | 00000 | 100111 | 0011 | 011000 | 1100 |
| D1.3 | 011 | 00001 | 011101 | 0011 | 100010 | 1100 |

## Valid Data Characters (SC/$\overline{D}$ = LOW) (continued)

| Data Byte Name | Bits | | Current RD– | | Current RD+ | |
|---|---|---|---|---|---|---|
| | HGF | EDCBA | abcdei | fghj | abcdei | fghj |
| D2.3 | 011 | 00010 | 101101 | 0011 | 010010 | 1100 |
| D3.3 | 011 | 00011 | 110001 | 1100 | 110001 | 0011 |
| D4.3 | 011 | 00100 | 110101 | 0011 | 001010 | 1100 |
| D5.3 | 011 | 00101 | 101001 | 1100 | 101001 | 0011 |
| D6.3 | 011 | 00110 | 011001 | 1100 | 011001 | 0011 |
| D7.3 | 011 | 00111 | 111000 | 1100 | 000111 | 0011 |
| D8.3 | 011 | 01000 | 111001 | 0011 | 000110 | 1100 |
| D9.3 | 011 | 01001 | 100101 | 1100 | 100101 | 0011 |
| D10.3 | 011 | 01010 | 010101 | 1100 | 010101 | 0011 |
| D11.3 | 011 | 01011 | 110100 | 1100 | 110100 | 0011 |
| D12.3 | 011 | 01100 | 001101 | 1100 | 001101 | 0011 |
| D13.3 | 011 | 01101 | 101100 | 1100 | 101100 | 0011 |
| D14.3 | 011 | 01110 | 011100 | 1100 | 011100 | 0011 |
| D15.3 | 011 | 01111 | 010111 | 0011 | 101000 | 1100 |
| D16.3 | 011 | 10000 | 011011 | 0011 | 100100 | 1100 |
| D17.3 | 011 | 10001 | 100011 | 1100 | 100011 | 0011 |
| D18.3 | 011 | 10010 | 010011 | 1100 | 010011 | 0011 |
| D19.3 | 011 | 10011 | 110010 | 1100 | 110010 | 0011 |
| D20.3 | 011 | 10100 | 001011 | 1100 | 001011 | 0011 |
| D21.3 | 011 | 10101 | 101010 | 1100 | 101010 | 0011 |
| D22.3 | 011 | 10110 | 011010 | 1100 | 011010 | 0011 |
| D23.3 | 011 | 10111 | 111010 | 0011 | 000101 | 1100 |
| D24.3 | 011 | 11000 | 110011 | 0011 | 001100 | 1100 |
| D25.3 | 011 | 11001 | 100110 | 1100 | 100110 | 0011 |
| D26.3 | 011 | 11010 | 010110 | 1100 | 010110 | 0011 |
| D27.3 | 011 | 11011 | 110110 | 0011 | 001001 | 1100 |
| D28.3 | 011 | 11100 | 001110 | 1100 | 001110 | 0011 |
| D29.3 | 011 | 11101 | 101110 | 0011 | 010001 | 1100 |
| D30.3 | 011 | 11110 | 011110 | 0011 | 100001 | 1100 |
| D31.3 | 011 | 11111 | 101011 | 0011 | 010100 | 1100 |
| D0.4 | 100 | 00000 | 100111 | 0010 | 011000 | 1101 |
| D1.4 | 100 | 00001 | 011101 | 0010 | 100010 | 1101 |
| D2.4 | 100 | 00010 | 101101 | 0010 | 010010 | 1101 |

## Valid Data Characters (SC/$\overline{D}$ = LOW) (continued)

| Data Byte Name | Bits | | Current RD– | | Current RD+ | |
|---|---|---|---|---|---|---|
| | HGF | EDCBA | abcdei | fghj | abcdei | fghj |
| D3.4 | 100 | 00011 | 110001 | 1101 | 110001 | 0010 |
| D4.4 | 100 | 00100 | 110101 | 0010 | 001010 | 1101 |
| D5.4 | 100 | 00101 | 101001 | 1101 | 101001 | 0010 |
| D6.4 | 100 | 00110 | 011001 | 1101 | 011001 | 0010 |
| D7.4 | 100 | 00111 | 111000 | 1101 | 000111 | 0010 |
| D8.4 | 100 | 01000 | 111001 | 0010 | 000110 | 1101 |
| D9.4 | 100 | 01001 | 100101 | 1101 | 100101 | 0010 |
| D10.4 | 100 | 01010 | 010101 | 1101 | 010101 | 0010 |
| D11.4 | 100 | 01011 | 110100 | 1101 | 110100 | 0010 |
| D12.4 | 100 | 01100 | 001101 | 1101 | 001101 | 0010 |
| D13.4 | 100 | 01101 | 101100 | 1101 | 101100 | 0010 |
| D14.4 | 100 | 01110 | 011100 | 1101 | 011100 | 0010 |
| D15.4 | 100 | 01111 | 010111 | 0010 | 101000 | 1101 |
| D16.4 | 100 | 10000 | 011011 | 0010 | 100100 | 1101 |
| D17.4 | 100 | 10001 | 100011 | 1101 | 100011 | 0010 |
| D18.4 | 100 | 10010 | 010011 | 1101 | 010011 | 0010 |
| D19.4 | 100 | 10011 | 110010 | 1101 | 110010 | 0010 |
| D20.4 | 100 | 10100 | 001011 | 1101 | 001011 | 0010 |
| D21.4 | 100 | 10101 | 101010 | 1101 | 101010 | 0010 |
| D22.4 | 100 | 10110 | 011010 | 1101 | 011010 | 0010 |
| D23.4 | 100 | 10111 | 111010 | 0010 | 000101 | 1101 |
| D24.4 | 100 | 11000 | 110011 | 0010 | 001100 | 1101 |
| D25.4 | 100 | 11001 | 100110 | 1101 | 100110 | 0010 |
| D26.4 | 100 | 11010 | 010110 | 1101 | 010110 | 0010 |
| D27.4 | 100 | 11011 | 110110 | 0010 | 001001 | 1101 |
| D28.4 | 100 | 11100 | 001110 | 1101 | 001110 | 0010 |
| D29.4 | 100 | 11101 | 101110 | 0010 | 010001 | 1101 |
| D30.4 | 100 | 11110 | 011110 | 0010 | 100001 | 1101 |
| D31.4 | 100 | 11111 | 101011 | 0010 | 010100 | 1101 |
| D0.5 | 101 | 00000 | 100111 | 1010 | 011000 | 1010 |
| D1.5 | 101 | 00001 | 011101 | 1010 | 100010 | 1010 |
| D2.5 | 101 | 00010 | 101101 | 1010 | 010010 | 1010 |
| D3.5 | 101 | 00011 | 110001 | 1010 | 110001 | 1010 |

## Valid Data Characters (SC/$\overline{D}$ = LOW) (continued)

| Data Byte Name | Bits | | Current RD– | | Current RD+ | |
|---|---|---|---|---|---|---|
| | HGF | EDCBA | abcdei | fghj | abcdei | fghj |
| D4.5 | 101 | 00100 | 110101 | 1010 | 001010 | 1010 |
| D5.5 | 101 | 00101 | 101001 | 1010 | 101001 | 1010 |
| D6.5 | 101 | 00110 | 011001 | 1010 | 011001 | 1010 |
| D7.5 | 101 | 00111 | 111000 | 1010 | 000111 | 1010 |
| D8.5 | 101 | 01000 | 111001 | 1010 | 000110 | 1010 |
| D9.5 | 101 | 01001 | 100101 | 1010 | 100101 | 1010 |
| D10.5 | 101 | 01010 | 010101 | 1010 | 010101 | 1010 |
| D11.5 | 101 | 01011 | 110100 | 1010 | 110100 | 1010 |
| D12.5 | 101 | 01100 | 001101 | 1010 | 001101 | 1010 |
| D13.5 | 101 | 01101 | 101100 | 1010 | 101100 | 1010 |
| D14.5 | 101 | 01110 | 011100 | 1010 | 011100 | 1010 |
| D15.5 | 101 | 01111 | 010111 | 1010 | 101000 | 1010 |
| D16.5 | 101 | 10000 | 011011 | 1010 | 100100 | 1010 |
| D17.5 | 101 | 10001 | 100011 | 1010 | 100011 | 1010 |
| D18.5 | 101 | 10010 | 010011 | 1010 | 010011 | 1010 |
| D19.5 | 101 | 10011 | 110010 | 1010 | 110010 | 1010 |
| D20.5 | 101 | 10100 | 001011 | 1010 | 001011 | 1010 |
| D21.5 | 101 | 10101 | 101010 | 1010 | 101010 | 1010 |
| D22.5 | 101 | 10110 | 011010 | 1010 | 011010 | 1010 |
| D23.5 | 101 | 10111 | 111010 | 1010 | 000101 | 1010 |
| D24.5 | 101 | 11000 | 110011 | 1010 | 001100 | 1010 |
| D25.5 | 101 | 11001 | 100110 | 1010 | 100110 | 1010 |
| D26.5 | 101 | 11010 | 010110 | 1010 | 010110 | 1010 |
| D27.5 | 101 | 11011 | 110110 | 1010 | 001001 | 1010 |
| D28.5 | 101 | 11100 | 001110 | 1010 | 001110 | 1010 |
| D29.5 | 101 | 11101 | 101110 | 1010 | 010001 | 1010 |
| D30.5 | 101 | 11110 | 011110 | 1010 | 100001 | 1010 |
| D31.5 | 101 | 11111 | 101011 | 1010 | 010100 | 1010 |
| D0.6 | 110 | 00000 | 100111 | 0110 | 011000 | 0110 |
| D1.6 | 110 | 00001 | 011101 | 0110 | 100010 | 0110 |
| D2.6 | 110 | 00010 | 101101 | 0110 | 010010 | 0110 |
| D3.6 | 110 | 00011 | 110001 | 0110 | 110001 | 0110 |
| D4.6 | 110 | 00100 | 110101 | 0110 | 001010 | 0110 |

## Valid Data Characters (SC/$\overline{D}$ = LOW) (continued)

| Data Byte Name | Bits | | Current RD– | | Current RD+ | |
|---|---|---|---|---|---|---|
| | HGF | EDCBA | abcdei | fghj | abcdei | fghj |
| D5.6 | 110 | 00101 | 101001 | 0110 | 101001 | 0110 |
| D6.6 | 110 | 00110 | 011001 | 0110 | 011001 | 0110 |
| D7.6 | 110 | 00111 | 111000 | 0110 | 000111 | 0110 |
| D8.6 | 110 | 01000 | 111001 | 0110 | 000110 | 0110 |
| D9.6 | 110 | 01001 | 100101 | 0110 | 100101 | 0110 |
| D10.6 | 110 | 01010 | 010101 | 0110 | 010101 | 0110 |
| D11.6 | 110 | 01011 | 110100 | 0110 | 110100 | 0110 |
| D12.6 | 110 | 01100 | 001101 | 0110 | 001101 | 0110 |
| D13.6 | 110 | 01101 | 101100 | 0110 | 101100 | 0110 |
| D14.6 | 110 | 01110 | 011100 | 0110 | 011100 | 0110 |
| D15.6 | 110 | 01111 | 010111 | 0110 | 101000 | 0110 |
| D16.6 | 110 | 10000 | 011011 | 0110 | 100100 | 0110 |
| D17.6 | 110 | 10001 | 100011 | 0110 | 100011 | 0110 |
| D18.6 | 110 | 10010 | 010011 | 0110 | 010011 | 0110 |
| D19.6 | 110 | 10011 | 110010 | 0110 | 110010 | 0110 |
| D20.6 | 110 | 10100 | 001011 | 0110 | 001011 | 0110 |
| D21.6 | 110 | 10101 | 101010 | 0110 | 101010 | 0110 |
| D22.6 | 110 | 10110 | 011010 | 0110 | 011010 | 0110 |
| D23.6 | 110 | 10111 | 111010 | 0110 | 000101 | 0110 |
| D24.6 | 110 | 11000 | 110011 | 0110 | 001100 | 0110 |
| D25.6 | 110 | 11001 | 100110 | 0110 | 100110 | 0110 |
| D26.6 | 110 | 11010 | 010110 | 0110 | 010110 | 0110 |
| D27.6 | 110 | 11011 | 110110 | 0110 | 001001 | 0110 |
| D28.6 | 110 | 11100 | 001110 | 0110 | 001110 | 0110 |
| D29.6 | 110 | 11101 | 101110 | 0110 | 010001 | 0110 |
| D30.6 | 110 | 11110 | 011110 | 0110 | 100001 | 0110 |
| D31.6 | 110 | 11111 | 101011 | 0110 | 010100 | 0110 |
| D0.7 | 111 | 00000 | 100111 | 0001 | 011000 | 1110 |
| D1.7 | 111 | 00001 | 011101 | 0001 | 100010 | 1110 |
| D2.7 | 111 | 00010 | 101101 | 0001 | 010010 | 1110 |
| D3.7 | 111 | 00011 | 110001 | 1110 | 110001 | 0001 |
| D4.7 | 111 | 00100 | 110101 | 0001 | 001010 | 1110 |
| D5.7 | 111 | 00101 | 101001 | 1110 | 101001 | 0001 |

## Valid Data Characters (SC/$\overline{D}$ = LOW) (continued)

| Data Byte Name | Bits | | Current RD– | | Current RD+ | |
|---|---|---|---|---|---|---|
| | HGF | EDCBA | abcdei | fghj | abcdei | fghj |
| D6.7 | 111 | 00110 | 011001 | 1110 | 011001 | 0001 |
| D7.7 | 111 | 00111 | 111000 | 1110 | 000111 | 0001 |
| D8.7 | 111 | 01000 | 111001 | 0001 | 000110 | 1110 |
| D9.7 | 111 | 01001 | 100101 | 1110 | 100101 | 0001 |
| D10.7 | 111 | 01010 | 010101 | 1110 | 010101 | 0001 |
| D11.7 | 111 | 01011 | 110100 | 1110 | 110100 | 1000 |
| D12.7 | 111 | 01100 | 001101 | 1110 | 001101 | 0001 |
| D13.7 | 111 | 01101 | 101100 | 1110 | 101100 | 1000 |
| D14.7 | 111 | 01110 | 011100 | 1110 | 011100 | 1000 |
| D15.7 | 111 | 01111 | 010111 | 0001 | 101000 | 1110 |
| D16.7 | 111 | 10000 | 011011 | 0001 | 100100 | 1110 |
| D17.7 | 111 | 10001 | 100011 | 0111 | 100011 | 0001 |
| D18.7 | 111 | 10010 | 010011 | 0111 | 010011 | 0001 |
| D19.7 | 111 | 10011 | 110010 | 1110 | 110010 | 0001 |
| D20.7 | 111 | 10100 | 001011 | 0111 | 001011 | 0001 |
| D21.7 | 111 | 10101 | 101010 | 1110 | 101010 | 0001 |
| D22.7 | 111 | 10110 | 011010 | 1110 | 011010 | 0001 |
| D23.7 | 111 | 10111 | 111010 | 0001 | 000101 | 1110 |
| D24.7 | 111 | 11000 | 110011 | 0001 | 001100 | 1110 |
| D25.7 | 111 | 11001 | 100110 | 1110 | 100110 | 0001 |
| D26.7 | 111 | 11010 | 010110 | 1110 | 010110 | 0001 |
| D27.7 | 111 | 11011 | 110110 | 0001 | 001001 | 1110 |
| D28.7 | 111 | 11100 | 001110 | 1110 | 001110 | 0001 |
| D29.7 | 111 | 11101 | 101110 | 0001 | 010001 | 1110 |
| D30.7 | 111 | 11110 | 011110 | 0001 | 100001 | 1110 |
| D31.7 | 111 | 11111 | 101011 | 0001 | 010100 | 1110 |

## Valid Special Character Codes and Sequences (SC/$\overline{D}$ = HIGH)[1, 2]

| S.C. Byte Name | S.C. Code Name | | Bits | | Current RD– | | Current RD+ | |
|---|---|---|---|---|---|---|---|---|
| | | | HGF | EDCBA | abcdei | fghj | abcdei | fghj |
| K28.0 | C0.0 | (C00) | 000 | 00000 | 001111 | 0100 | 110000 | 1011 |
| K28.1 | C1.0 | (C01) | 000 | 00001 | 001111 | 1001 | 110000 | 0110 |
| K28.2 | C2.0 | (C02) | 000 | 00010 | 001111 | 0101 | 110000 | 1010 |
| K28.3 | C3.0 | (C03) | 000 | 00011 | 001111 | 0011 | 110000 | 1100 |
| K28.4 | C4.0 | (C04) | 000 | 00100 | 001111 | 0010 | 110000 | 1101 |
| K28.5 | C5.0 | (C05) | 000 | 00101 | 001111 | 1010 | 110000 | 0101 |
| K28.6 | C6.0 | (C06) | 000 | 00110 | 001111 | 0110 | 110000 | 1001 |
| K28.7 | C7.0 | (C07) | 000 | 00111 | 001111 | 1000 | 110000 | 0111 |
| K23.7 | C8.0 | (C08) | 000 | 01000 | 111010 | 1000 | 000101 | 0111 |
| K27.7 | C9.0 | (C09) | 000 | 01001 | 110110 | 1000 | 001001 | 0111 |
| K29.7 | C10.0 | (C0A) | 000 | 01010 | 101110 | 1000 | 010001 | 0111 |
| K30.7 | C11.0 | (C0B) | 000 | 01011 | 011110 | 1000 | 100001 | 0111 |
| Idle | C0.1 | (C20) | 001 | 00000 | –K28.5+,D21.4,D21.5,D21.5,repeat[3] | | | |
| R_RDY | C1.1 | (C21) | 001 | 00001 | –K28.5+,D21.4,D10.2,D10.2,repeat[4] | | | |
| EOFxx | C2.1 | (C22) | 001 | 00010 | –K28.5,Dn.xxx0[5]+K28.5,Dn.xxx1[5] | | | |
| | Follows K28.1 for ESCON Connect–SOF (Rx indication only) | | | | | | | |
| C–SOF | C7.1 | (C27) | 001 | 00111 | 001111 | 1000 | 110000 | 0111 |
| | Follows K28.5 for ESCON Passive–SOF (Rx indication only) | | | | | | | |
| P–SOF | C7.2 | (C47) | 010 | 00111 | 001111 | 1000 | 110000 | 0111 |
| | | | | | Code Rule Violation and SVS Tx Pattern | | | |
| Exception | C0.7 | (CE0) | 111 | 00000 | 100111 | 1000[6] | 011000 | 0111[6] |
| –K28.5 | C1.7 | (CE1) | 111 | 00001 | 001111 | 1010[29] | 001111 | 1010[29] |
| +K28.5 | C2.7 | (CE2) | 111 | 00010 | 110000 | 0101[30] | 110000 | 0101[30] |
| | | | | | Running Disparity Violation Pattern | | | |
| Exception | C4.7 | (CE4) | 111 | 00100 | 110111 | 0101[31] | 001000 | 1010[31] |

**Notes**

1. All codes not shown are reserved.
2. Notation for Special Character Byte Name is consistent with Fibre Channel and ESCON naming conventions. Special Character Code Name is intended to describe binary information present on I/O pins. Common usage for the name can either be in the form used for describing Data patterns (i.e., C0.0 through C31.7), or in hex notation (i.e., Cnn where nn = the specified value between 00 and FF).
3. C0.1 = Transmit Negative K28.5 (–K28.5+) disregarding Current RD when input is held for only one byte time. If held longer, transmitter begins sending the repeating transmit sequence –K28.5+, D21.4, D21.5, D21.5, (repeat all four bytes)... defined in X3.230 as the primitive signal “Idle word.” This Special Character input must be held for four (4) byte times or multiples of four bytes or it will be truncated by the new data. The receiver will never output this Special Character, since K28.5 is decoded as C5.0, C1.7, or C2.7, and the subsequent bytes are decoded as data.
4. C1.1 = Transmit Negative K28.5 (–K28.5+) disregarding Current RD when input is held for only one byte time. If held longer, transmitter begins sending the repeating transmit sequence –K28.5+, D21.4, D10.2, D10.2,(repeat all four bytes)... defined in X3.230 as the primitive signal “Receiver_Ready (R_RDY).” This Special Character input must be held for four (4) byte times or multiples of four bytes or it will be truncated by the new data.
The receiver will never output this Special Character, since K28.5 is decoded as C5.0, C1.7, or C2.7 and the subsequent bytes are decoded as data.
5. C2.1 = Transmit either –K28.5+ or +K28.5– as determined by Current RD and modify the Transmission Character that follows, by setting its least significant bit to 1 or 0. If Current RD at the start of the following character is plus (+) the LSB is set to 0, and if Current RD is minus (–) the LSB becomes 1. This modification allows construction of X3.230 “EOF” frame delimiters wherein the second data byte is determined by the Current RD.
For example, to send “EOFdt” the controller could issue the sequence C2.1–D21.4– D21.4–D21.4, and the HOTLink Transmitter will send either K28.5–D21.4–D21.4–D21.4 or K28.5–D21.5– D21.4–D21.4 based on Current RD. Likewise to send “EOFdti” the controller could issue the sequence C2.1–D10.4–D21.4–D21.4, and the HOTLink Transmitter will send either K28.5–D10.4–D21.4– D21.4 or K28.5–D10.5–D21.4– D21.4 based on Current RD.
The receiver will never output this Special Character, since K28.5 is decoded as C5.0, C1.7, or C2.7, and the subsequent bytes are decoded as data.
6. C0.7 = Transmit a deliberate code rule violation. The code chosen for this function follows the normal Running Disparity rules. Transmission of this Special Character has the same effect as asserting SVS = HIGH.
The receiver will only output this Special Character if the Transmission Character being decoded is not found in the tables.

## Maximum Ratings

Exceeding maximum ratings may impair the useful life of the device. These user guidelines are not tested.

Storage temperature ..... –65 °C to +150 °C

Ambient temperature with power applied ..... –55 °C to +125 °C

Supply voltage to ground potential ..... –0.5 V to +7.0 V

DC input voltage ..... –0.5 V to +7.0 V

Output current into TTL outputs (LOW) ..... 30 mA

Output current into PECL outputs (HIGH) ..... –50 mA

Static discharge voltage ..... > 4001 V (per MIL-STD-883, Method 3015)

Latch up current ..... > 200 mA

## Operating Range

| Range | Ambient Temperature | $V_{CC}$ |
|---|---|---|
| Commercial | 0 °C to +70 °C | 5 V ± 10% |
| Industrial | –40 °C to +85 °C | 5 V ± 10% |

## CY7B923/CY7B933 Electrical Characteristics

Over the Operating Range[7]

| Parameter | Description | Test Condition | | Min | Max | Unit |
|---|---|---|---|---|---|---|
| **TTL OUTs, CY7B923: RP; CY7B933: $Q_{0-7}$, SC/D, RVS, RDY, CKR, SO** | | | | | | |
| $V_{OHT}$ | Output HIGH voltage | $I_{OH}$ = - 2 mA | | 2.4 | | V |
| $V_{OLT}$ | Output LOW voltage | $I_{OL}$ = 4 mA | | | 0.45 | V |
| $I_{OST}$ | Output short circuit current | $V_{OUT}$ = 0V[8] | | –15 | –90 | mA |
| **TTL INs, CY7B923: $D_{0-7}$, SC/D, SVS, ENA, ENN, CKW, FOTO, BISTEN; CY7B933: RF, REFCLK, BISTEN** | | | | | | |
| $V_{IHT}$ | Input HIGH voltage | | Commercial and industrial | 2.0 | $V_{CC}$ | V |
| | | | Industrial (CKW and FOTO, only) | 2.2 | $V_{CC}$ | V |
| $V_{ILT}$ | Input LOW voltage | | | –0.5 | 0.8 | V |
| $I_{IHT}$ | Input HIGH current | $V_{IN} = V_{CC}$ | | –10 | +10 | µA |
| $I_{ILT}$ | Input LOW current | $V_{IN}$ = 0.0V | | | –500 | µA |
| **Transmitter PECL-Compatible Output Pins: OUTA+, OUTA–, OUTB+, OUTB–, OUTC+, OUTC–** | | | | | | |
| $V_{OHE}$ | Output HIGH voltage ($V_{CC}$ referenced) | Load = 50Ω to $V_{CC}$ – 2V | Commercial | $V_{CC}$ – 1.03 | $V_{CC}$ – 0.83 | V |
| | | | Industrial | $V_{CC}$ – 1.05 | $V_{CC}$ – 0.83 | V |
| $V_{OLE}$ | Output LOW voltage ($V_{CC}$ referenced) | Load = 50Ω to $V_{CC}$ – 2V | Commercial | $V_{CC}$ – 1.86 | $V_{CC}$ – 1.62 | V |
| | | | Industrial | $V_{CC}$ – 1.96 | $V_{CC}$ – 1.62 | V |
| $V_{ODIF}$ | Output differential voltage \|(OUT+) – (OUT–)\| | Load = 50Ω to $V_{CC}$ – 2V | | 0.6 | | V |
| **Receiver PECL-Compatible Input Pins: A/B, SI, INB** | | | | | | |
| $V_{IHE}$ | Input HIGH voltage | | Commercial | $V_{CC}$ – 1.165 | $V_{CC}$ | V |
| | | | Industrial | $V_{CC}$ – 1.14 | $V_{CC}$ | V |
| $V_{ILE}$ | Input LOW voltage | | Commercial | 2.0 | $V_{CC}$ – 1.475 | V |
| | | | Industrial | 2.0 | $V_{CC}$ – 1.50 | V |
| $I_{IHE}$[9] | Input HIGH current | $V_{IN} = V_{IHE}$ Max. | | | +500 | µA |
| $I_{ILE}$[9] | Input LOW current | $V_{IN} = V_{ILE}$ Min. | | +0.5 | | µA |

**Notes**
7. See the last page of this specification for group A subgroup testing information.
8. Tested one output at a time, output shorted for less than one second, less than 10% duty cycle.
9. Applies to A/$\overline{B}$ only.

## CY7B923/CY7B933 Electrical Characteristics

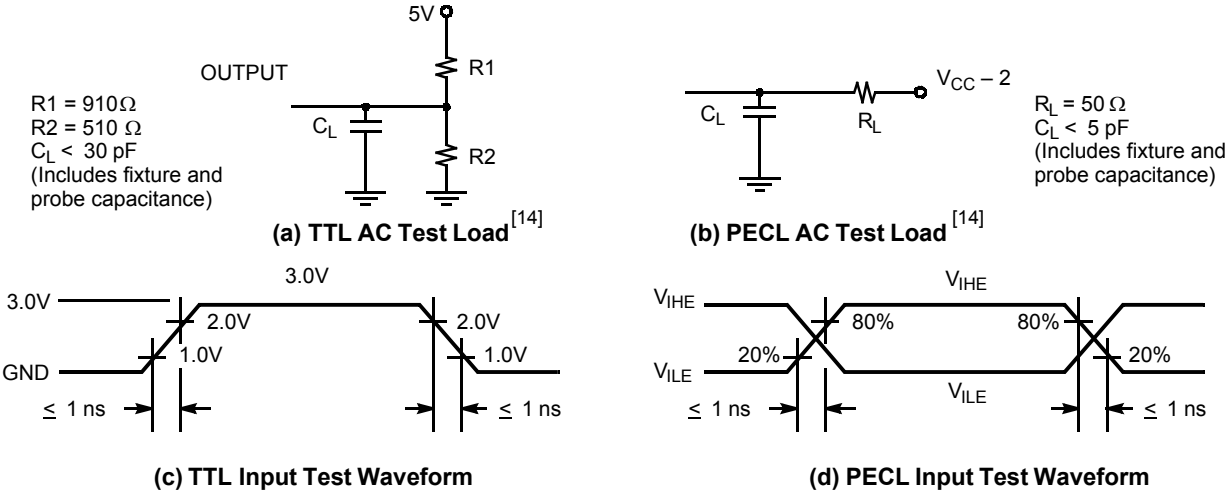
Over the Operating Range[7]

| Differential Line Receiver Input Pins: INA+, INA–, INB+, INB– | | | | | | |
|---|---|---|---|---|---|---|
| $V_{DIFF}$ | Input differential voltage \|(IN+) – (IN–)\| | | | 50 | | mV |
| $V_{IHH}$ | Highest input HIGH voltage | | | | $V_{CC}$ | V |
| $V_{ILL}$ | Lowest input LOW voltage | | | 2.0 | | V |
| $I_{IHH}$ | Input HIGH current | $V_{IN} = V_{IHH}$ Max. | | | 750 | µA |
| $I_{ILL}$[10] | Input LOW current | $V_{IN} = V_{ILL}$ Min. | | –200 | | µA |
| **Miscellaneous** | | | | **Typ** | **Max** | **Unit** |
| $I_{CCT}$[11] | Transmitter power supply current | Freq. = Max. | Commercial | 65 | 85 | mA |
| | | | Industrial | 75 | 95 | mA |
| $I_{CCR}$[12] | Receiver power supply current | Freq. = Max. | Commercial | 120 | 155 | mA |
| | | | Industrial | 135 | 160 | mA |

## Capacitance[13]

| Parameter | Description | Test Conditions | Max | Unit |
|---|---|---|---|---|
| $C_{IN}$ | Input capacitance | $T_A = 25\,°C$, $f_0 = 1$ MHz, $V_{CC} = 5.0V$ | 10 | pF |

**Figure 9. AC Test Loads and Waveforms**




(a) TTL AC Test Load[14]

(b) PECL AC Test Load[14]

(c) TTL Input Test Waveform

(d) PECL Input Test Waveform

**Notes**

10. Input currents are always positive at all voltages above $V_{CC}/2$.
11. Maximum $I_{CCT}$ is measured with $V_{CC}$ = Max., one PECL output pair loaded with 50 ohms to $V_{CC}$ – 2.0V, and other PECL outputs tied to $V_{CC}$. Typical $I_{CCT}$ is measured with $V_{CC}$ = 5.0V, $T_A$ = 25°C, one output pair loaded with 50 ohms to $V_{CC}$ – 2.0V, others tied to $V_{CC}$, $\overline{BISTEN}$ = LOW. $I_{CCT}$ includes current into $V_{CCQ}$ (pin 9 and pin 22) only. Current into $V_{CCN}$ is determined by PECL load currents, typically 30 mA with 50 ohms to $V_{CC}$ – 2.0V. Each additional enabled PECL pair adds 5 mA to $I_{CCT}$ and an additional load current to $V_{CCN}$ as described. When calculating the contribution of PECL load currents to chip power dissipation, the output load current should be multiplied by 1V instead of $V_{CC}$.
12. Maximum $I_{CCR}$ is measured with $V_{CC}$ = Max., RF = LOW, and outputs unloaded. Typical $I_{CCR}$ is measured with $V_{CC}$ = 5.0V, $T_A$ = 25°C, RF = LOW, $\overline{BISTEN}$ = LOW, and outputs unloaded. $I_{CCR}$ includes current into $V_{CCQ}$ (pins 21 and 24). Current into $V_{CCN}$ (pin 9) is determined by the total TTL output buffer quiescent current plus the sum of all the load currents for each output pin. The total buffer quiescent current is 10mA max., and max. TTL load current for each output pin can be calculated as follows:

$$\frac{I_{I\,CCN}}{TTLPin} = \left[\frac{0.95 + (V_{CCN} - 5) * 0.3}{R_L} + C_L * \left[\frac{V_{CCN}}{2} + 1.5\right] * F_{pin}\right] * 1.1$$

Where $R_L$ = equivalent load resistance, $C_L$ = capacitive load, and $F_{pin}$ = frequency in MHz of data on pin. A derating factor of 1.1 has been included to account for worst process corner and temperature condition.

13. Tested initially and after any design or process changes that may affect these parameters, but not 100% tested.
14. Cypress uses constant current (ATE) load configurations and forcing functions. This figure is for reference only.

## Transmitter Switching Characteristics

Over the Operating Range[7]

| Parameter | Description | 7B923-155 | | 7B923 | | 7B923-400 | | Unit |
|---|---|---|---|---|---|---|---|---|
| | | Min | Max | Min | Max | Min | Max | |
| $t_{CKW}$ | Write clock cycle | 62.5 | 66.7 | 30.3 | 62.5 | 25 | 62.5 | ns |
| $t_B$ | Bit time[15] | 6.25 | 6.67 | 3.03 | 6.25 | 2.5 | 6.25 | ns |
| $t_{CPWH}$ | CKW pulse width HIGH | 6.5 | – | 6.5 | – | 6.5 | – | ns |
| $t_{CPWL}$ | CKW pulse width LOW | 6.5 | – | 6.5 | – | 6.5 | – | ns |
| $t_{SD}$ | Data setup time[16] | 5 | – | 5 | – | 5 | – | ns |
| $t_{HD}$ | Data hold time[16] | 0 | – | 0 | – | 0 | – | ns |
| $t_{SENP}$ | Enable setup time (to insure correct $\overline{RP}$)[17] | $6t_B + 8$ | – | $6t_B + 8$ | – | $6t_B + 8$ | – | ns |
| $t_{HENP}$ | Enable hold time (to insure correct $\overline{RP}$)[17] | 0 | – | 0 | – | 0 | – | ns |
| $t_{PDR}$ | Read pulse rise alignment[18] | –4 | 2 | –4 | 2 | –4 | 2 | ns |
| $t_{PPWH}$ | Read pulse HIGH[18] | $4t_B–3$ | – | $4t_B–3$ | – | $4t_B–3$ | – | ns |
| $t_{PDF}$ | Read pulse fall alignment[18] | $6t_B–3$ | | $6t_B–3$ | – | $6t_B–3$ | – | ns |
| $t_{RISE}$ | PECL output rise time 20 to 0% (PECL test load)[13] | – | 1.2 | – | 1.2 | – | 1.2 | ns |
| $t_{FALL}$ | PECL output fall time 80 to 20% (PECL test load)[13] | – | 1.2 | – | 1.2 | – | 1.2 | ns |
| $t_{DJ}$ | Deterministic jitter (peak-peak)[13, 19] | – | 35 | – | 35 | – | 35 | ps |
| $t_{RJ}$ | Random jitter (peak-peak)[13, 20] | – | 175 | – | 175 | – | 175 | ps |
| $t_{RJ}$ | Random jitter ($\sigma$)[13, 20] | – | 20 | – | 20 | – | 20 | ps |

**Notes**

15. Transmitter $t_B$ is calculated as $t_{CKW}/10$. The byte rate is one tenth of the bit rate.
16. Data includes $D_{0–7}$, SC/$\overline{D}$, SVS, $\overline{ENA}$, $\overline{ENN}$, and $\overline{BISTEN}$. $t_{SD}$ and $t_{HD}$ minimum timing assures correct data load on rising edge of CKW, but not $\overline{RP}$ function or timing.
17. $t_{SENP}$ and $t_{HENP}$ timing insures correct $\overline{RP}$ function and correct data load on the rising edge of CKW.
18. Loading on $\overline{RP}$ is the standard TTL test load shown in part (a) of AC Test Loads and Waveforms except $C_L$ = 15 pF.
19. While sending continuous K28.5s, $\overline{RP}$ unloaded, outputs loaded to 50Ω to $V_{CC}$–2.0V, over the operating range.
20. While sending continuous K28.7s, after 100,000 samples measured at the cross point of differential outputs, time referenced to CKW input, over the operating range.

## Receiver Switching Characteristics

Over the Operating Range [7]

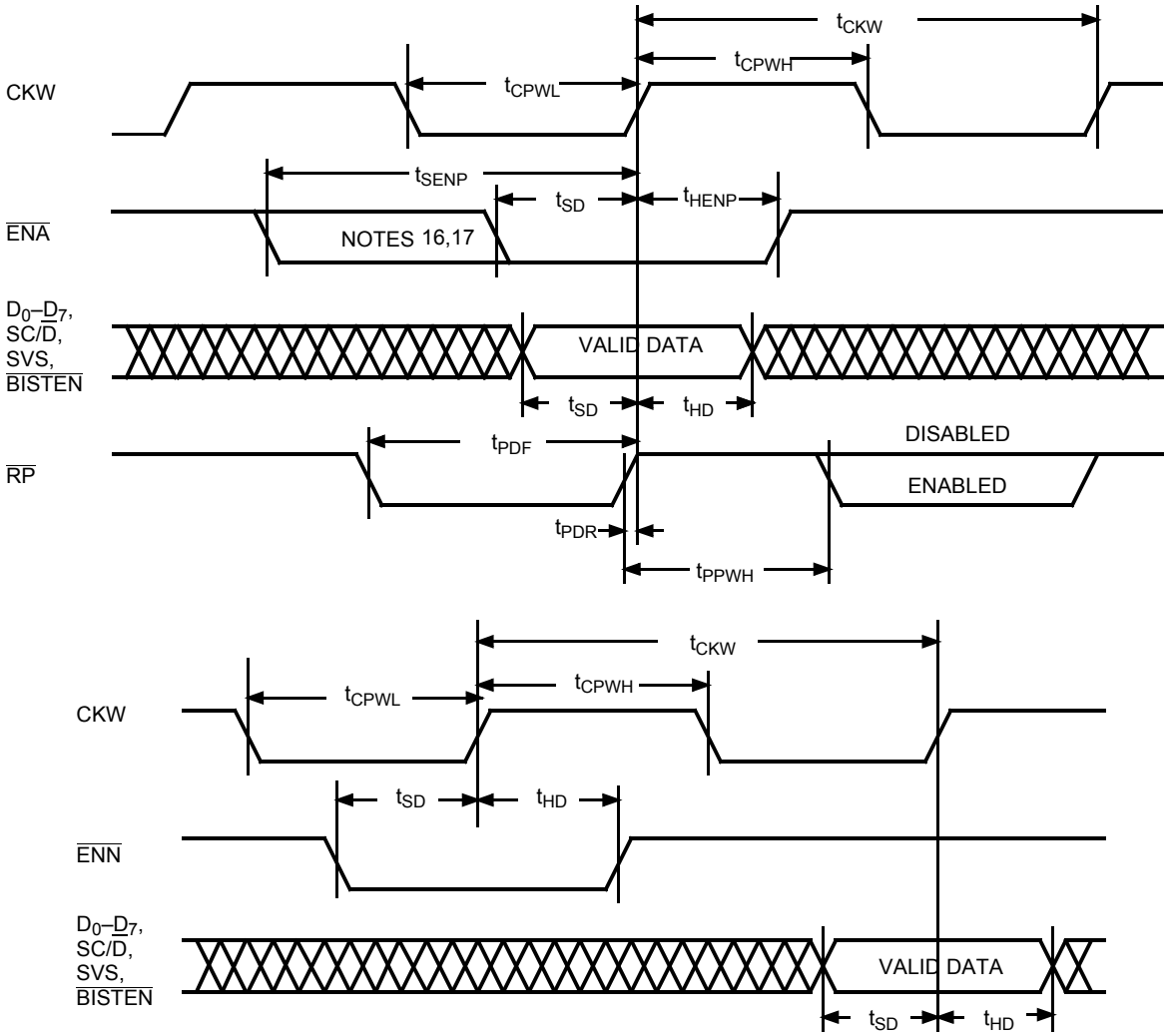
| Parameter | Description | 7B933-155 | | 7B933 | | 7B933-400 | | Unit |
|---|---|---|---|---|---|---|---|---|
| | | Min | Max | Min | Max. | Min | Max | |
| $t_{CKR}$ | Read clock period (no serial data input), REFCLK as reference[21] | –1 | +1 | –1 | +1 | –1 | +1 | % |
| $t_B$ | Bit time[22] | 6.25 | 6.67 | 3.03 | 6.25 | 2.5 | 6.25 | ns |
| $t_{CPRH}$ | Read clock pulse HIGH | $5t_B$–3 | – | $5t_B$–3 | – | $5t_B$–3 | – | ns |
| $t_{CPRL}$ | Read clock pulse LOW | $5t_B$–3 | – | $5t_B$–3 | – | $5t_B$–3 | – | ns |
| $t_{RH}$ | $\overline{RDY}$ hold time | $t_B$–2.5 | – | $t_B$–2.5 | – | $t_B$–2.5 | – | ns |
| $t_{PRF}$ | $\overline{RDY}$ pulse fall to CKR rise | $5t_B$–3 | – | $5t_B$–3 | – | $5t_B$–3 | – | ns |
| $t_{PRH}$ | $\overline{RDY}$ pulse width HIGH | $4t_B$–3 | – | $4t_B$–3 | – | $4t_B$–3 | – | ns |
| $t_A$ | Data access time[23, 24] | $2t_B$–2 | $2t_B$+4 | $2t_B$–2 | $2t_B$+4 | $2t_B$–2 | $2t_B$+4 | ns |
| $t_{ROH}$ | Data hold time[23, 24] | $t_B$–2.5 | – | $t_B$–2.5 | – | $t_B$–2.5 | | ns |
| $t_H$ | Data hold time from CKR rise [23, 24] | $2t_B$–3 | – | $2t_B$–3 | – | $2t_B$–3 | | ns |
| $t_{CKX}$ | REFCLK clock period referenced to CKW of transmitter[25] | –0.1 | +0.1 | –0.1 | +0.1 | –0.1 | +0.1 | % |
| $t_{CPXH}$ | REFCLK clock pulse HIGH | 6.5 | – | 6.5 | – | 6.5 | – | ns |
| $t_{CPXL}$ | REFCLK clock pulse LOW | 6.5 | – | 6.5 | – | 6.5 | – | ns |
| $t_{DS}$ | Propagation delay SI to SO (note PECL and TTL thresholds)[26] | – | 20 | – | 20 | – | 20 | ns |
| $t_{SA}$ | Static alignment[13, 27] | – | 100 | – | 100 | – | 100 | ps |
| $t_{EFW}$ | Error-free window[13, 28] | $0.9t_B$ | – | $0.9t_B$ | – | $0.9t_B$ | – | – |

**Notes**

21. The period of $t_{CKR}$ will match the period of the transmitter CKW when the receiver is receiving serial data. When data is interrupted, CKR may drift to one of the range limits above.
22. Receiver $t_B$ is calculated as $t_{CKR}$/10 if no data is being received, or $t_{CKW}$/10 if data is being received. See note.
23. Data includes $Q_{0-7}$, SC/$\overline{D}$, and RVS.
24. $t_A$, $t_{ROH}$, and $t_H$ specifications are only valid if all outputs (CKR, $\overline{RDY}$, $Q_{0-7}$, SC/$\overline{D}$, and RVS) are loaded with similar DC and AC loads.
25. REFCLK has no phase or frequency relationship with CKR and only acts as a centering reference to reduce clock synchronization time. REFCLK must be within 0.1% of the transmitter CKW frequency, necessitating a ±500-PPM crystal.

## Switching Waveforms

**Figure 10. Switching Waveforms for the CY7B923 HOTLink Transmitter**




**Notes**

26. The PECL switching threshold is the midpoint between the PECL– $V_{OH}$, and $V_{OL}$ specification (approximately $V_{CC}$ – 1.35V). The TTL switching threshold is 1.5V.
27. Static alignment is a measure of the alignment of the Receiver sampling point to the center of a bit. Static alignment is measured by sliding one bit edge in 3,000 nominal transitions until a byte error occurs.
28. Error Free Window is a measure of the time window between bit centers where a transition may occur without causing a bit sampling error. EFW is measured over the operating range, input jitter < 50% Dj.

## Switching Waveforms (continued)

**Figure 11. Switching W0aveforms for the CY7B933 HOTLink Receiver**

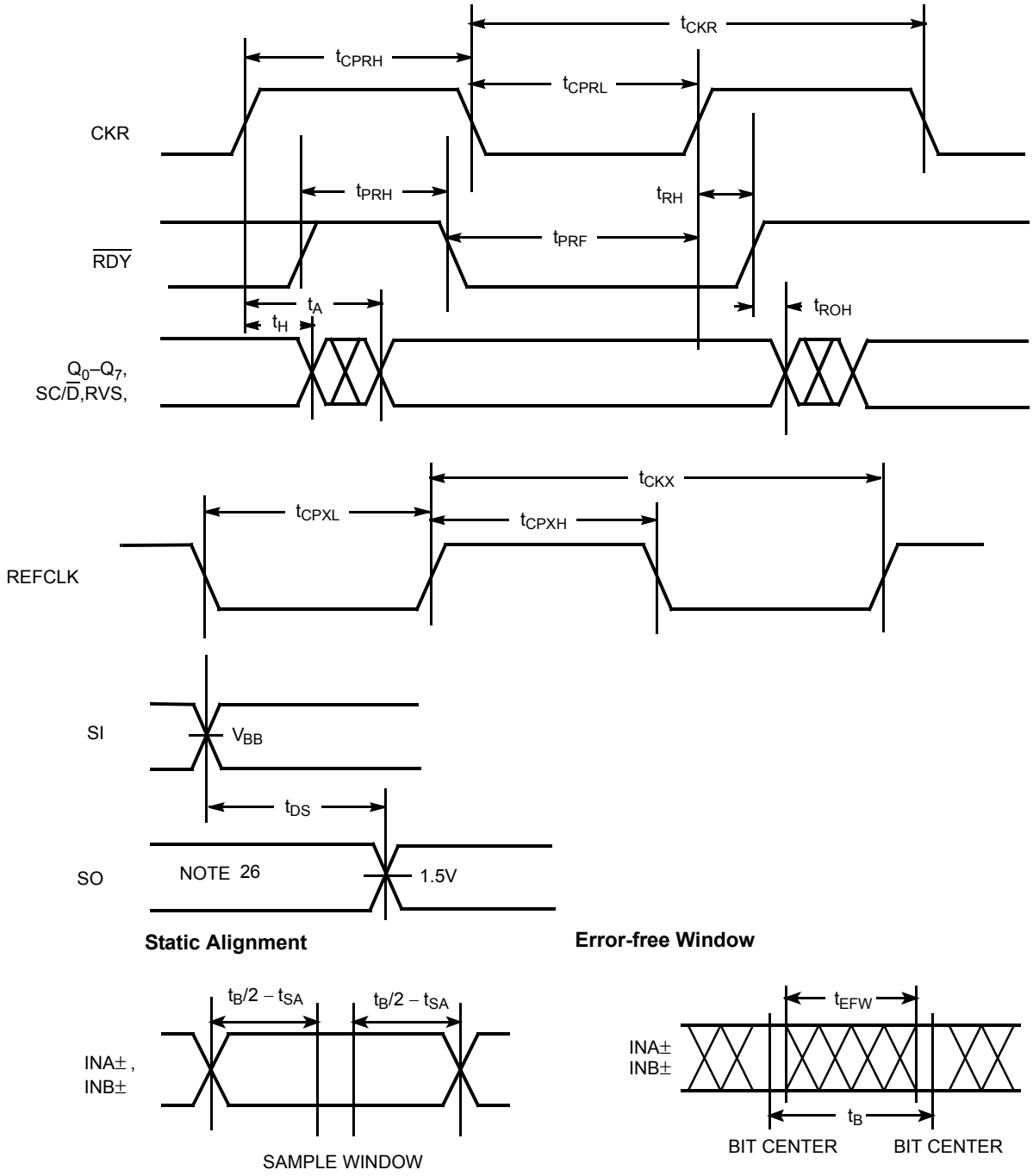
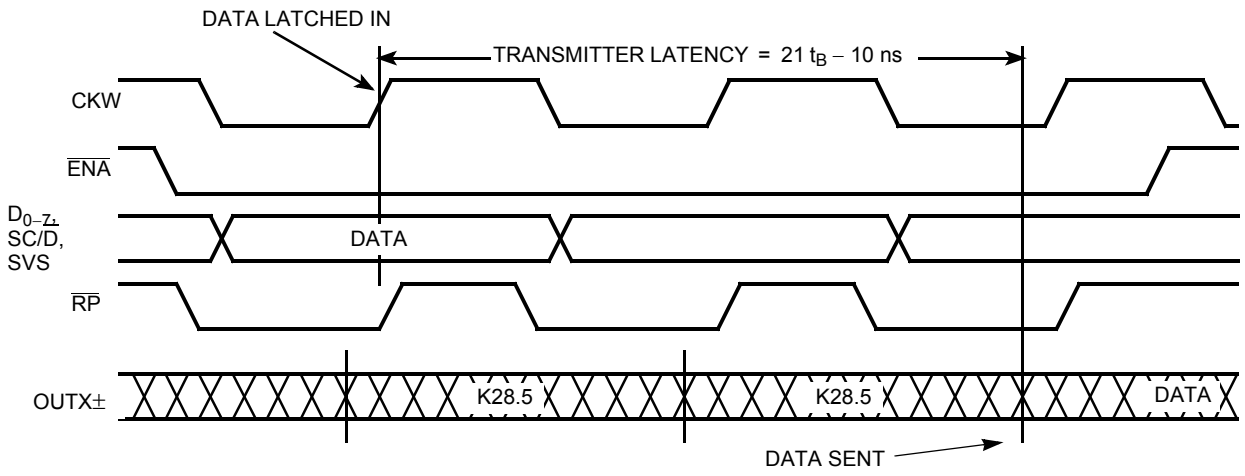


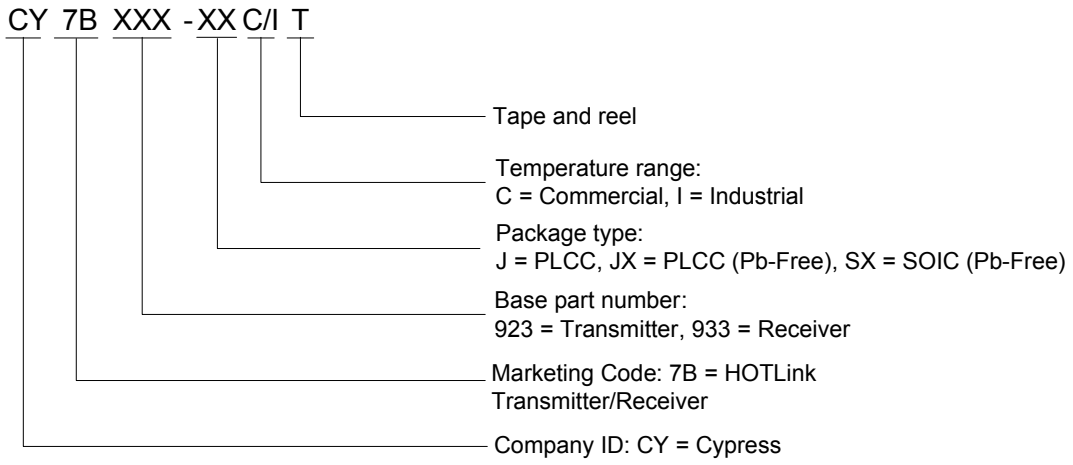
**Figure 12. CY7B923 Transmitter Data Pipeline**



## Ordering Information

| Speed | Ordering Code | Package Name | Package Type | Operating Range |
|---|---|---|---|---|
| Standard | CY7B923-JXC | J64 | 28-pin PLCC (Pb-free) | Commercial |
| | CY7B923-JXCT | J64 | 28-pin PLCC (Pb-free) | |
| | CY7B923-JXI | J64 | 28-pin PLCC (Pb-free) | Industrial |
| | CY7B923-JXIT | J64 | 28-pin PLCC (Pb-free) | |
| | CY7B923-SXC | S21 | 28-pin SOIC (Pb-free) | Commercial |
| | CY7B923-SXCT | S21 | 28-pin SOIC (Pb-free) | |
| 400 | CY7B923-400JXC | J64 | 28-pin PLCC (Pb-free) | Commercial |
| | CY7B923-400JXCT | J64 | 28-pin PLCC (Pb-free) | |
| Standard | CY7B933-JXC | J64 | 28-pin PLCC (Pb-free) | Commercial |
| | CY7B933-JXCT | J64 | 28-pin PLCC (Pb-free) | |
| | CY7B933-JXI | J64 | 28-pin PLCC (Pb-free) | Industrial |
| | CY7B933-JXIT | J64 | 28-pin PLCC (Pb-free) | |
| | CY7B933-SXC | S21 | 28-pin SOIC (Pb-free) | Commercial |
| | CY7B933-SXCT | S21 | 28-pin SOIC (Pb-free) | |
| 400 | CY7B933-400JXC | J64 | 28-pin PLCC (Pb-free) | Commercial |
| | CY7B933-400JXCT | J64 | 28-pin PLCC (Pb-free) | |

### Ordering Code Definitions

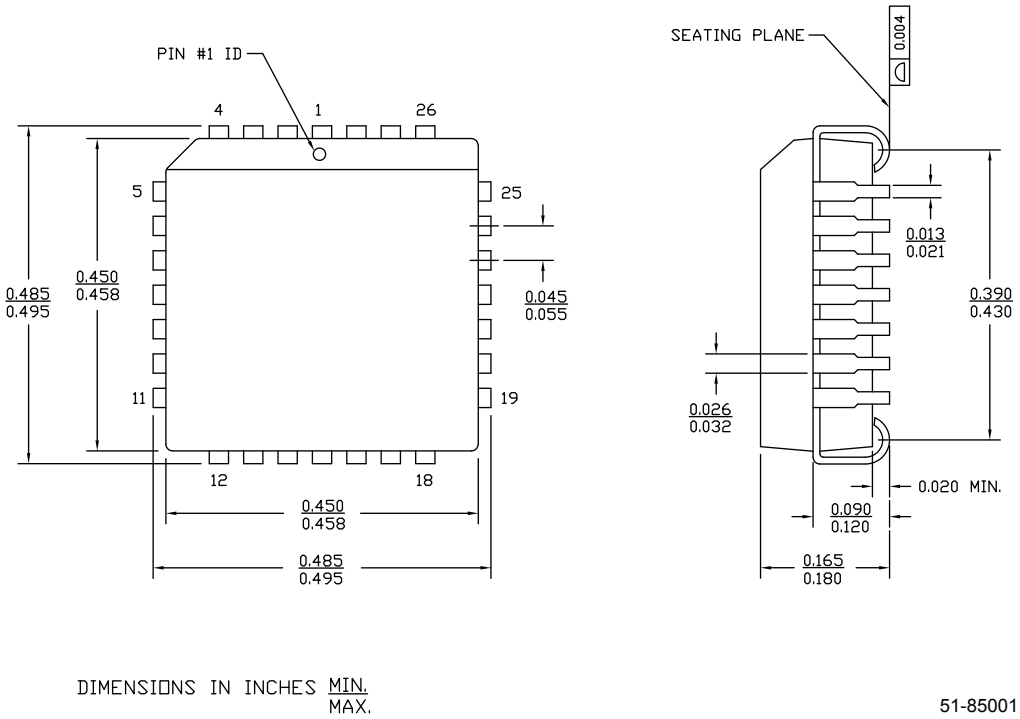



**Notes**

29. C1.7 = Transmit Negative K28.5 (–K28.5+) disregarding Current RD.
The receiver will only output this Special Character if K28.5 is received with the wrong running disparity. The receiver will output C1.7 if –K28.5 is received with RD+, otherwise K28.5 is decoded as C5.0 or C2.7.
30. C2.7 = Transmit Positive K28.5 (+K28.5–) disregarding Current RD.
The receiver will only output this Special Character if K28.5 is received with the wrong running disparity. The receiver will output C2.7 if +K28.5 is received with RD–, otherwise K28.5 is decoded as C5.0 or C1.7.
31. C4.7 = Transmit a deliberate code rule violation to indicate a Running Disparity violation.
The receiver will only output this Special Character if the Transmission Character being decoded is found in the tables, but Running Disparity does not match. This might indicate that an error occurred in a prior byte.

## Package Diagrams

**Figure 13. 28-Pin Plastic Leaded Chip Carrier J64**

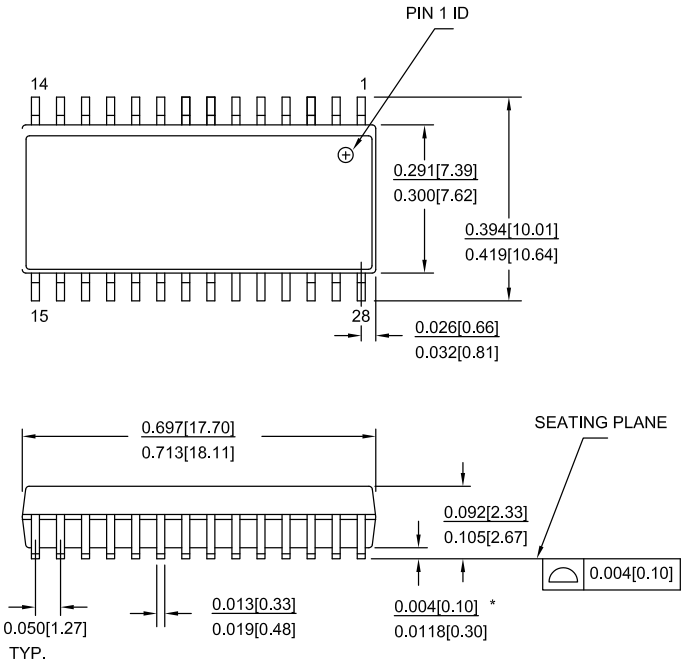


## Package Diagrams (continued)

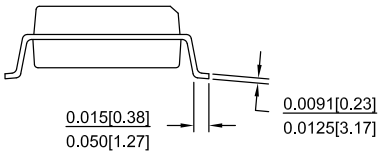
**Figure 14. 28-Pin (300-Mil) Molded SOIC S21**

NOTE :

1. JEDEC STD REF MO–119
2. BODY LENGTH DIMENSION DOES NOT INCLUDE MOLD PROTRUSION/END FLASH,BUT DOES INCLUDE MOLD MISMATCH AND ARE MEASURED AT THE MOLD PARTING LINE. MOLD PROTRUSION/END FLASH SHALL NOT EXCEED 0.010 in (0.254 mm) PER SIDE
3. DIMENSIONS IN INCHES MIN. / MAX.




| PART # | |
|---|---|
| S28.3 | STANDARD PKG. |
| SZ28.3 | LEAD FREE PKG. |
| SX28.3 | LEAD FREE PKG. |




51-85026 *H

## Acronyms

**Table 5. Acronyms Used in this Document**

| Acronym | Description |
|---|---|
| AC | alternating current |
| BIST | built-in self-test |
| CDR | clock/data recovery |
| CML | current mode logic |
| DC | direct current |
| DVB | digital video broadcasting |
| ECL | emitter coupled logic |
| I/O | input/output |
| JTAG | joint test action group |
| LFI | link fault indicator |
| LFSR | linear feedback shift register |
| LPEN | local loopback input |
| PECL | positive-ECL |
| PLL | phase-locked loop |
| TTL | transistor-transistor logic |
| VCO | voltage controlled oscillator |

## Document Conventions

### Units of Measure

**Table 1-1:**

| Symbol | Unit of Measure |
|---|---|
| °C | degrees Celsius |
| MHz | megahertz |
| µA | microampere |
| µs | microsecond |
| Mbps | megabits per second |
| mA | milliampere |
| mm | millimeter |
| ms | millisecond |
| mV | millivolt |
| nA | nano ampere |
| ns | nanosecond |
| nV | nano volt |
| Ω | ohm |
| pF | picofarad |
| pp | peak-to-peak |
| ps | picosecond |
| sps | samples per second |
| V | volt |

## Document History Page

| Document Title: CY7B923/CY7B933, HOTLink® Transmitter/Receiver<br>Document Number: 38-02017 | | | | |
|---|---|---|---|---|
| **Revision** | **ECN** | **Orig. of Change** | **Submission Date** | **Description of Change** |
| ** | 105855 | SZV | 03/28/01 | Changed from Spec number: 38-00189 to 38-02017 |
| *A | 112164 | REV | 03/25/02 | Changed OUTA± pin description to improve consistency with diagram.<br>Changed INA± pin description to include what to do with unused pairs of inputs.<br>Changed Equation in note 6–old one made no sense. |
| *B | 114562 | BSS | 03/27/02 | Changed Hotlink™ Transmitter/Receiver to Hotlink® Transmitter/Receiver. |
| *C | 125525 | OOR | 04/01/03 | Removed all references to Military parts (Obsolete): CY7B923-LMB, CY7B933-LMB |
| *D | 132104 | KKV | 12/22/03 | Minor change: reset Valid Data Characters (SC/$\overline{D}$ = LOW) table format to single-column pages |
| *E | 393422 | PCX | See ECN | Added Pb-Free Logo<br>Added Pb-Free parts to Ordering Information:<br>CY7B923-400JXC, CY7B923-JXC, CY7B923-JXI, CY7B923-SXC, CY7B933-400JXC, CY7B933-JXC, CY7B933-JXI, CY7B933-SXC, CY7B933-SXI |
| *F | 2896112 | CGX | 03/19/10 | Removed obsolete parts in ordering information table<br>Updated package diagrams |
| *G | 3028517 | FRE | 09/13/2010 | Removed references to obsolete low-speed and military parts.<br>Added ordering code definition.<br>Updated 28-pin PLCC package diagram.<br>Updated to new template. |
| *H | 3059305 | SAAC | 10/14/2010 | Reviewed Content - No change |
| *I | 3400761 | SAAC | 10/10/2011 | Removed reference to LCC from Features section on page 1.<br>Removed following obsolete parts:<br>CY7B923-JC<br>CY7B933-JC<br>Updated package diagrams. |
| *J | 4571788 | YLIU | 11/17/2014 | Updated Package Diagrams:<br>spec 51-85026 – Changed revision from *F to *H.<br><br>Updated to new template.<br><br>Completing Sunset Review. |
| *K | 5959791 | YLIU | 11/07/2017 | Updated package diagram (51-85001) |

## Sales, Solutions, and Legal Information

### Worldwide Sales and Design Support

Cypress maintains a worldwide network of offices, solution centers, manufacturer's representatives, and distributors. To find the office closest to you, visit us at Cypress Locations.

### Products

| | |
|---|---|
| ARM® Cortex® Microcontrollers | cypress.com/arm |
| Automotive | cypress.com/automotive |
| Clocks & Buffers | cypress.com/clocks |
| Interface | cypress.com/interface |
| Internet of Things | cypress.com/iot |
| Memory | cypress.com/memory |
| Microcontrollers | cypress.com/mcu |
| PSoC | cypress.com/psoc |
| Power Management ICs | cypress.com/pmic |
| Touch Sensing | cypress.com/touch |
| USB Controllers | cypress.com/usb |
| Wireless Connectivity | cypress.com/wireless |

### PSoC® Solutions

PSoC 1 | PSoC 3 | PSoC 4 | PSoC 5LP

### Cypress Developer Community

Forums | WICED IOT Forums | Projects | Video | Blogs | Training | Components

### Technical Support

cypress.com/support

© Cypress Semiconductor Corporation, 2001-2017. This document is the property of Cypress Semiconductor Corporation and its subsidiaries, including Spansion LLC ("Cypress"). This document, including any software or firmware included or referenced in this document ("Software"), is owned by Cypress under the intellectual property laws and treaties of the United States and other countries worldwide. Cypress reserves all rights under such laws and treaties and does not, except as specifically stated in this paragraph, grant any license under its patents, copyrights, trademarks, or other intellectual property rights. If the Software is not accompanied by a license agreement and you do not otherwise have a written agreement with Cypress governing the use of the Software, then Cypress hereby grants you a personal, non-exclusive, nontransferable license (without the right to sublicense) (1) under its copyright rights in the Software (a) for Software provided in source code form, to modify and reproduce the Software solely for use with Cypress hardware products, only internally within your organization, and (b) to distribute the Software in binary code form externally to end users (either directly or indirectly through resellers and distributors), solely for use on Cypress hardware product units, and (2) under those claims of Cypress's patents that are infringed by the Software (as provided by Cypress, unmodified) to make, use, distribute, and import the Software solely for use with Cypress hardware products. Any other use, reproduction, modification, translation, or compilation of the Software is prohibited.

TO THE EXTENT PERMITTED BY APPLICABLE LAW, CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS DOCUMENT OR ANY SOFTWARE OR ACCOMPANYING HARDWARE, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. To the extent permitted by applicable law, Cypress reserves the right to make changes to this document without further notice. Cypress does not assume any liability arising out of the application or use of any product or circuit described in this document. Any information provided in this document, including any sample design information or programming code, is provided only for reference purposes. It is the responsibility of the user of this document to properly design, program, and test the functionality and safety of any application made of this information and any resulting product. Cypress products are not designed, intended, or authorized for use as critical components in systems designed or intended for the operation of weapons, weapons systems, nuclear installations, life-support devices or systems, other medical devices or systems (including resuscitation equipment and surgical implants), pollution control or hazardous substances management, or other uses where the failure of the device or system could cause personal injury, death, or property damage ("Unintended Uses"). A critical component is any component of a device or system whose failure to perform can be reasonably expected to cause the failure of the device or system, or to affect its safety or effectiveness. Cypress is not liable, in whole or in part, and you shall and hereby do release Cypress from any claim, damage, or other liability arising from or related to all Unintended Uses of Cypress products. You shall indemnify and hold Cypress harmless from and against all claims, costs, damages, and other liabilities, including claims for personal injury or death, arising from or related to any Unintended Uses of Cypress products.

Cypress, the Cypress logo, Spansion, the Spansion logo, and combinations thereof, WICED, PSoC, CapSense, EZ-USB, F-RAM, and Traveo are trademarks or registered trademarks of Cypress in the United States and other countries. For a more complete list of Cypress trademarks, visit cypress.com. Other names and brands may be claimed as property of their respective owners.

## Looking for pricing, stock, or lifecycle information?

Click below to explore more details on WIN SOURCE:

- View CY7B933-JXIT on WIN SOURCE
- Cypress Semiconductor Corp Information

## Optimize Your Supply Chain with WIN SOURCE Solutions

- Global Sourcing Solution
- Obsolete Management
- Cost Control Management
- Shortage Management
- Alternative Solution
- Excess Inventory Management