AN1057

TAXI™ to Cypress CY7C9689A Hotlink® Transceiver Conversion Series: 1. System Parallel Interface

Associated Project: No
Associated Part Family: CY7C9689A
Software Version: NA
Related Application Notes: None

To get the latest version of this application note, or the associated project file, please visit http://www.cypress.com/go/AN1057.

AN1057 discusses about the system parallel interface built with the CY7C9689A that are directly compatible with the legacy systems made using AMD TAXI™.

Introduction

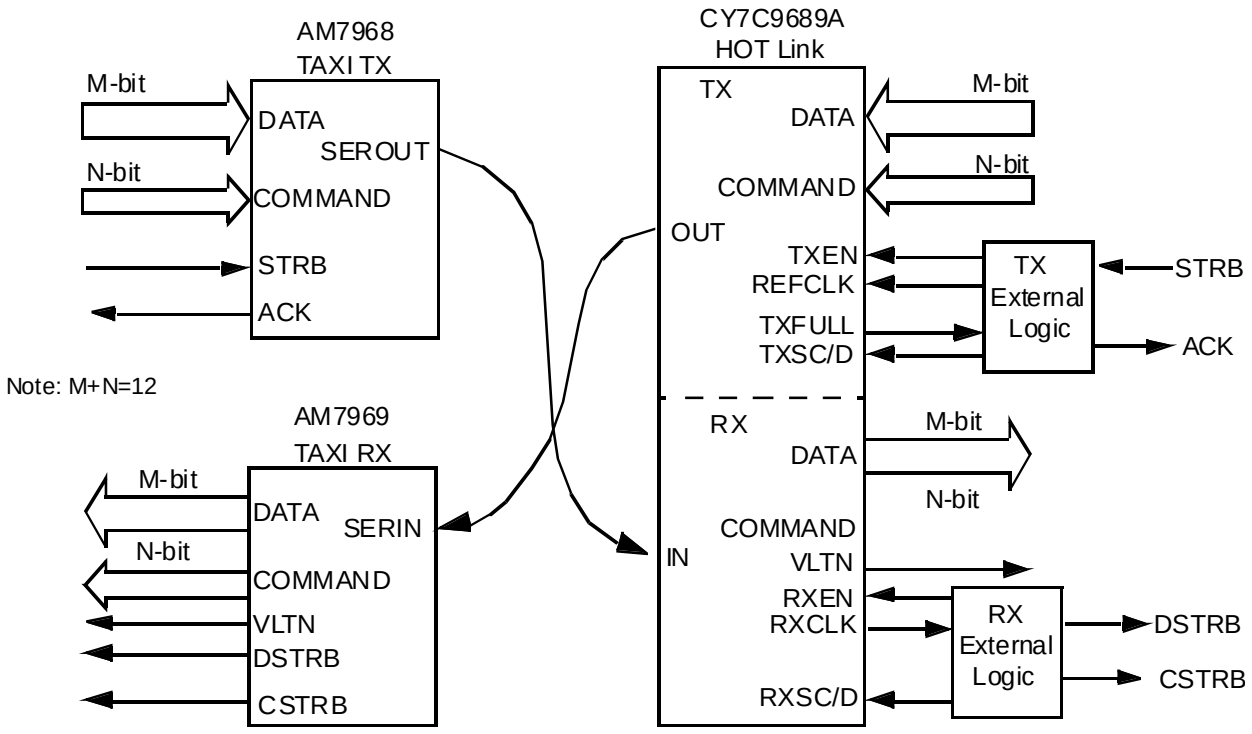
The Cypress CY7C9689A TAXI-compatible HOTLink® Transceiver facilitates point-to-point data communication over high-speed serial links. Systems built with the CY7C9689A are directly compatible with legacy systems made using AMD TAXI chip devices. The CY7C9689A HOTLink Transceiver is functionally equivalent to an AMD AM7968 TAXI transmitter and AM7969 receiver pair, with numerous technology enhancements and extensions.

Time has moved on, and the AMD TAXI chipset has been left behind. The TAXI chipset is a full bipolar design and consumes approximately 3 W of power. The CY7C9689A is designed on 0.35-mm CMOS technology and consumes only 30% of the power of the AMD chipset. The CY7C9689A contains both transmit and receive functions, and integrates separate transmit and receive FIFOs in the same device. The CY7C9689A is available in a 100-pin TQFP package, offering significant board real estate savings. It operates from 50 MBd to 200 MBd, while the TAXI chipset has a maximum serial signaling rate of 175 MBd. In addition, the AMD TAXI chipset has been discontinued, leaving numerous designs and products in jeopardy.

Fortunately, the CY7C9689A is a simple replacement for the TAXI chipset that requires little change in surrounding system logic. The CY7C9689A allows the continued use and manufacturing of these legacy systems with minimal impact to the equipment, and enables new designs with up-to-date technology. By replacing the TAXIchip devices with the CY7C9689A, it is possible to maintain compatibility with those systems deployed in the field, while increasing integration, lowering power requirements, and improving reliability.

Although the CY7C9689A is not pin-for-pin compatible with the TAXI device, the CY7C9689A parallel interface and serial interface can be mapped onto the TAXI functional pins with minimal external logic. Since the CY7C9689A was designed with modern systems in mind, it has many enhanced functions that the TAXI lacks. The objective of this document is to show the basic configuration of the CY7C9689A for replacing the TAXI chipset in the most general applications.

Figure 1. TAXI vs. CY7C9689A System Parallel Interface



Overview

The TAXI chipset, when operating in 8- or 10-bit mode, uses 4B/5B or 5B/6B encoding and decoding respectively. The encoded data is further enhanced for serial transmission by NRZI coding. The CY7C9689A Transceiver uses the same 4B/5B, 5B/6B, and NRZI encoding and decoding. This allows direct communication between the CY7C9689A and TAXI devices at the serial interface. Figure 1 shows how a CY7C9689A transceiver would be connected in a serial link with the TAXI chipset.

While both the serial and parallel interfaces of the CY7C9689A are compatible with the corresponding interface of the AMD TAXI devices, the devices are not identical. This application note discusses the replacement of TAXI chipset with the CY7C9689A in existing system interfaces.

Converting TAXI to CY7C9689A Parallel Interface

The parallel interface of the TAXI transmitter and receiver is based on a strobe/acknowledge interface that is seldom used in present-day designs. The transmit and receive parallel-data interface on the CY7C9689A makes use of a much more predictable and easier to control clocked (synchronous) interface. Although this clocked interface is different from that of the original TAXI, it can generally be interfaced with existing system logic with minimal changes.

Transmitter System Interface Bus Mapping

The AM7968 TAXI transmitter parallel interface consists of an N-bit data bus, an M-bit command bus (N = 8, 9, or 10, where N+M=12), a write strobe, and an acknowledge signal. For example, when configured for 8-bit data, the TAXI data bus is eight bits wide and the command bus is four bits wide.

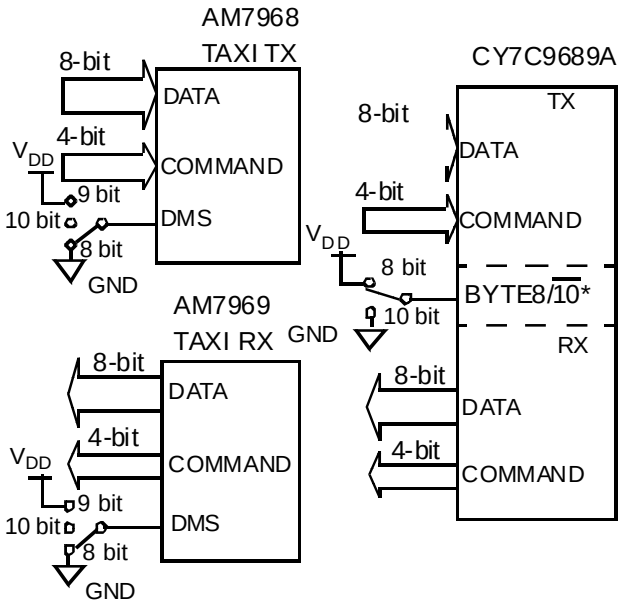
The CY7C9689A data and command buses are organized in exactly the same fashion. Unlike the TAXI devices (which support 8, 9, or 10-bit data paths), the CY7C9689A only supports 8- and 10-bit modes. Within the CY7C9689A, the data path bus-width is controlled by the BYTE8/$\overline{10}$ signal. When configured to transport 8-bit data (BYTE8/$\overline{10}$ = HIGH), the CY7C9689A data bus is eight bits wide and its command bus is four bits wide. When configured to transport 10-bit data (BYTE8/$\overline{10}$ = LOW), the CY7C9689A data bus is ten bits wide and its command bus is two bits wide. With the TAXI transmitter and receiver, the data bus width is configured using the DMS signal. Figure 2 shows the setting of the DMS and BYTE8/$\overline{10}$ signals to configure both the TAXI and CY7C9689A for 8-bit data and 4-bit command bus-widths.

Transmitter Interface Control Signaling

TAXI Transmitter STRB and ACK

Data is latched into the TAXI transmitter using a strobe pulse called STRB. The STRB input on the TAXI transmitter can often be treated as an asynchronous signal with respect to the reference clock input and the CLK output. The TAXI transmitter latches information present on the data and command buses on the rising edge of the STRB input. When this captured information is transferred out of the TAXI-transmitter input register, it provides an acknowledge pulse (ACK) back to the system to confirm reception of the data or command information, and to indicate that the input register is ready to accept new data/command information. This asynchronous STRB/ACK interface was appropriate at the time when the TAXI chipset was introduced into the market.

Figure 2. 8-bit Data Bus Configuration



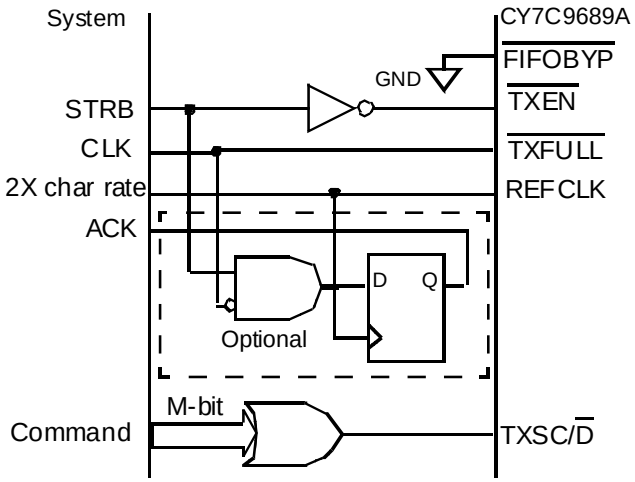
Although this strobed interface is (by definition) asynchronous, most applications implement STRB synchronous to the character-clock output (CLK) of the TAXI transmitter. This is necessary (when the TAXI transmitter is loaded at or near the character-clock rate) to avoid the “Strobe Stayout Area” or “t_6 window” located around the falling edge of CLK. To ensure that STRB assertion remains outside of the stayout area, most systems generate STRB synchronous to CLK.

Converting STRB to $\overline{\text{TXEN}}$

The parallel interface of the CY7C9689A no longer makes use of a strobe to capture data. It follows the more modern construct of a synchronous interface. However, using a small amount of external logic, it is possible to emulate the strobe-based interface of a TAXI transmitter on the CY7C9689A. This external transmit logic is shown in Figure 3, and requires that the CY7C9689A be configured for use with a 2x reference clock (REFCLK) input.

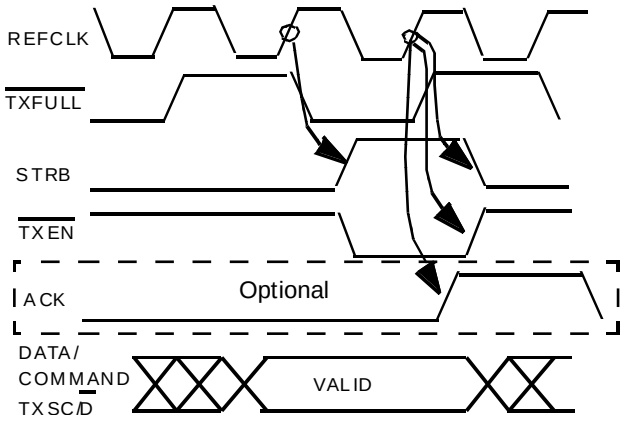
When the FIFOs in the CY7C9689A are bypassed ($\overline{\text{FIFOBYP}}$ = LOW) and the REFCLK input is configured to operate at twice the character rate, the $\overline{\text{TXFULL}}$ flag becomes a character-rate clock output. $\overline{\text{TXFULL}}$ also indicates which REFCLK cycle the transmitter is ready to accept data. The transmitter captures data on each rising edge of REFCLK when both $\overline{\text{TXFULL}}$ and $\overline{\text{TXEN}}$ are LOW (see Figure 4). When the system STRB output is synchronous to the system CLK (i.e., $\overline{\text{TXFULL}}$), then the system STRB signal can be connected to the $\overline{\text{TXEN}}$ input of the CY7C9689A through an inverter.

Figure 3. TX External Logic



With the circuit in Figure 3, the STRB signal can pulse once per character as it does in a normal TAXI system. It may also remain HIGH when characters to be sent are present on the command/data bus. To prevent duplicate or invalid characters from being sent, STRB must be deasserted after the last valid character is captured. If STRB stays asserted after acknowledgement, anything on the data/command bus will be latched on the subsequent REFCLK cycle when $\overline{\text{TXFULL}}$ is LOW. This information must also remain valid until the hold time (from the rising edge of REFCLK) is satisfied.

Figure 4. TX External Logic Timing



When $\overline{\text{TXEN}}$ is asserted (and $\overline{\text{TXFULL}}$ is LOW), latching of the information on the data/command bus is guaranteed at the correct REFCLK cycle. Therefore, the ACK handshake signal is optional and can be omitted (if not required by the system).

Converting STRB to $\overline{\text{TXEN}}$ (synchronous to CLK/$\overline{\text{TXFULL}}$)

With a slight change to the circuit in Figure 3, it is possible to alter the method of enabling write operations to the CY7C9689A transmit interface (see Figure 5).

Here the CY7C9689A does not directly use the STRB signal. Instead, it captures the information on the data/command bus on the rising edge of REFCLK when both $\overline{\text{TXFULL}}$ and $\overline{\text{TXEN}}$ are LOW. This allows STRB and $\overline{\text{TXEN}}$ to be generated relative to the rising edge of $\overline{\text{TXFULL}}$ /CLK. This eliminates the need for designing external logic at a 2X character rate. (REFCLK must still operate at twice the character rate.)

Figure 5. Alternative TX External Logic Timing

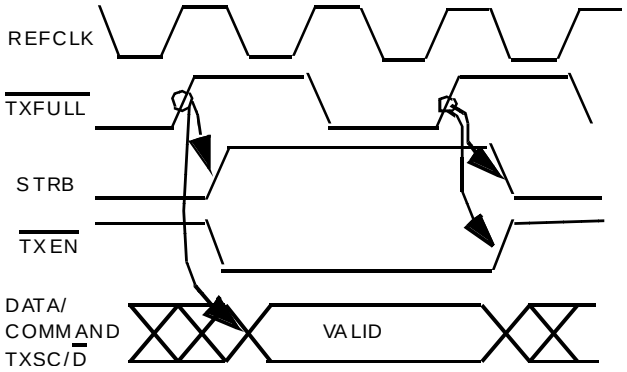
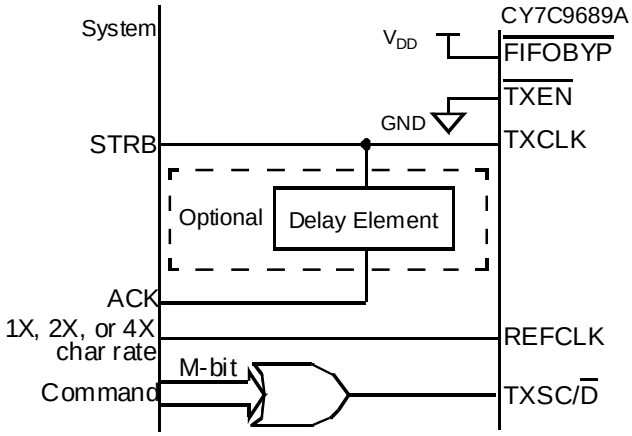


Figure 6. Asynchronous TX External Logic



When STRB/$\overline{\text{TXEN}}$ is generated relative to $\overline{\text{TXFULL}}$, latching of the information on the data/command bus is guaranteed to occur within the correct REFCLK cycle. Again, the ACK handshake signal can be omitted.

Interfacing Asynchronous STRB and ACK to CY7C9689A

In spite of the TAXI transmitter's "Strobe Stayout Area" requirement, some legacy systems with low data-rate requirements still use asynchronous STRB/ACK handshaking.

To interface the CY7C9689A to these asynchronous system interfaces, the internal FIFO must be enabled ($\overline{TXEN}$ = HIGH). Figure 6 shows the circuit connections between the asynchronous system interface and the CY7C9689A. The delay element used for ACK generation must be such that the FIFO cannot be loaded faster than the characters are read from the FIFO and transmitted. This allows the system to be built without monitoring the $\overline{TXFULL}$ FIFO status flag. If the generation of STRB pulses is self-limiting (due to the speed at which characters are made available by the system), the generation of ACK may be as simple as a direct connection between STRB and ACK.

The operation of this asynchronous interface is similar to the TAXI transmitter input latch. The system STRB is connected directly to the TX FIFO write clock (TXCLK). This latches data or commands into the FIFO input register at the rising edge of the TXCLK. ACK is generated directly by STRB (or a time delayed form of STRB) as shown in Figure 7. Upon the falling edge of STRB/TXCLK, the data in the input register is transferred into the FIFO. Upon the next rising edge of STRB/TXCLK (which writes the next data/command character into the input register), the internal FIFO pointers are updated, and the internal transmit state machine reads this character from the TX FIFO for serial transmission.

Because the TX FIFO requires a following rising edge on TXCLK to update the internal FIFO pointers, it may be necessary to write a dummy SYNC character to the command bus after the last data/command character to allow that character to be transmitted. This extra write operation ensures that the last data/command character of a burst or packet is transmitted before the first character of the next burst.

Data and Command Transmission

The TAXI transmitter switches between Data and Command transmission by examining the Command bus inputs. An all zero pattern on the Command bus (when a character is captured) causes Data to be sent. Any non-zero pattern on the Command bus causes the associated command character to be sent.

The system logic for a TAXI transmit interface normally contains an internal signal which determines when the Command bus needs to be “zeroed” for transmission of data. In the CY7C9689A, the TXSC/$\overline{D}$ input serves this same purpose. It determines whether a Data or Command character is captured on the current write cycle. When TXSC/$\overline{D}$ is sampled LOW, a Data character is captured. If TXSC/$\overline{D}$ is sampled HIGH, a Command character is captured. If a signal is not available to drive TXSC/$\overline{D}$, a simple OR gate can be used to combine all the Command signals to drive the TXSC/$\overline{D}$ input (as shown in Figure 3).

Receiver System Interface Bus Mapping

Just like the Transmit Data and Command Bus, the Receive Data and Command Bus mapping of the CY7C9689A is exactly the same as the TAXI in 8- and 10-bit modes. The CY7C9689A also has a Violation (VLTN) output to indicate when characters are received that do not match any valid data or command character.

Figure 7. Asynchronous TX External Logic Timing

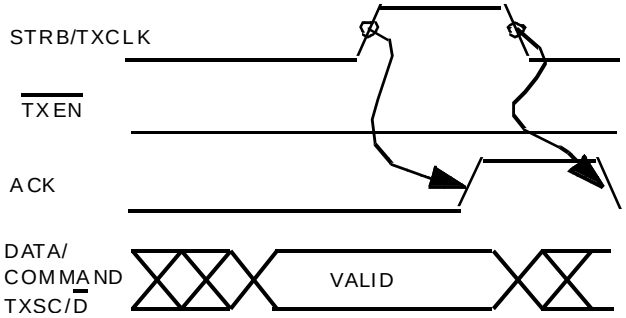
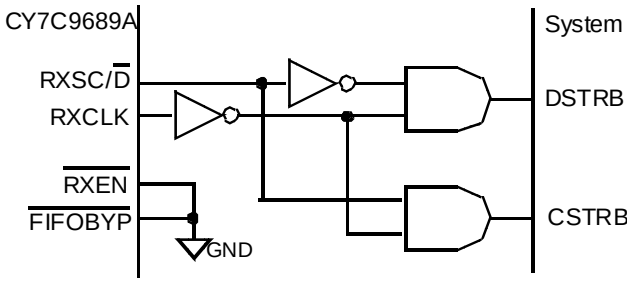


Figure 8. RXSC/$\overline{D}$ to DSTRB and CSTRB Translation



Receiver System Interface Control Signaling

TAXI receiver DSTRB and CSTRB

Upon reception of valid serial data, the TAXI receiver decodes the incoming data and presents the decoded Data or Command character on the Data/Command bus. It then notifies the system that a Data or Command character is available on the bus by asserting the data strobe (DSTRB) or command strobe (CSTRB) respectively.

CY7C9689A Synchronous Parallel Interface Generation of DSTRB and CSTRB

When configured as a synchronous interface, the internal receive FIFO of the CY7C9689A is bypassed ($\overline{\text{FIFOBYP}}$ = LOW), and RXCLK becomes the recovered clock output. The data and command buses on the CY7C9689A receive interface are synchronized with the RXCLK output. Unlike REFCLK (which may be configured for 1X or 2X character-rate operation), RXCLK switches only at the recovered character-rate frequency.

With the receive FIFO bypassed, the Data/Command bus (with VLTN) is updated at every RXCLK cycle. $\overline{\text{RXEN}}$ must be asserted LOW to enable the Data/Command Bus. The RXSC/$\overline{\text{D}}$ output signal indicates whether the information on the Data/Command bus is a data character or a command character.

As the transmit parallel interface, it is easy to translate this receive synchronous interface to a strobe based interface. The circuitry shown in Figure 8 translates RXCLK and RXSC/$\overline{\text{D}}$ into DSTRB and CSTRB.

CY7C9689A Asynchronous Parallel Interface Generation of DSTRB and CSTRB

When the internal FIFOs are enabled to allow asynchronous operation ($\overline{\text{FIFOBYP}}$ = HIGH), RXCLK becomes an input. A free running character-rate clock must be used to drive the RXCLK input. This clock may be completely asynchronous to the internal recovered clock, and is usually operated faster than the received character-rate to prevent the receive FIFO from overflowing (with the resulting loss of data). When the FIFO is empty, the RXEMPTY flag is asserted to indicate that the information on the data/command bus is not valid.

Figure 9 shows an implementation of DSTRB and CSTRB with the CY7C9689A internal FIFOs enabled. This circuit assumes the free-running character-rate clock has 40/60 (or better) duty cycle, to ensure that the received data and flag are stable before the falling edge of the clock.

CY7C9689A Static Control Configuration

To emulate the TAXI chipset, the CY7C9689A can be configured to its most basic functional form. The configurations of CY7C9689A static inputs for TAXI parallel interface emulation are summarized in Table 1.

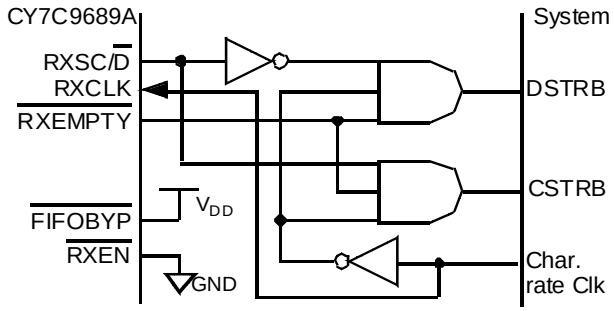
Table 1. CY7C9689A Static Input Configurations for TAXI Asynchronous Parallel Interface Emulation

Static Signals	50–100 MBd	100–200 MBd
$\overline{\text{FIFOBYP}}$	High	High
$\overline{\text{ENCBYP}}$	High	High
EXTFIFO	Low	Low
SPDSEL	Low	High
RANGESEL	Low	High
RXMODE[1]	Low	Low
RXMODE[0]	Low	Low
$\overline{\text{TXBISTEN}}$	High	High
$\overline{\text{RXBISTEN}}$	High	High
RFEN	High	High
$\overline{\text{CS}}$	Low	Low

$\overline{\text{FIFOBYP}}$

The TAXI transmitter does not have integrated FIFOs (beyond the two-character FIFO used for synchronizing the internal state machine with the external asynchronous strobe-base interface).

Figure 9. RXSC/$\overline{\text{D}}$ to DSTRB and CSTRB Translation



For synchronous parallel interface implementations, the CY7C9689A internal 256-character FIFOs must be bypassed by asserting $\overline{\text{FIFOBYP}}$ LOW. For asynchronous parallel interface implementation, the internal FIFOs must be enabled by asserting $\overline{\text{FIFOBYP}}$ HIGH.

ENCBYP

The 4B/5B or 5B/6B encoder and decoder of the CY7C9689A are enabled by asserting ENCBYP HIGH. When BYTE8/10 is asserted HIGH (8-bit mode), the dual 4B/5B encoders are enabled. When BYTE8/10 is asserted LOW (10-bit mode), the dual 5B/6B encoders are enabled.

EXTFIFO

The external FIFO enable input (EXTFIFO) changes the parallel interface timing of the CY7C9689A for direct connect to an external synchronous FIFO like the Cypress CY7C42XX 18-bit FIFO. Most existing TAXI systems interface to system logic devices (e.g., an ASIC, FPGA, or CPLD) and do not directly interface to FIFOs. For the implementations discussed in this document, we assume that the CY7C9689A does not connect to an external synchronous FIFO, therefore EXTFIFO must be LOW.

SPDSEL and RANGESEL

For emulating the TAXI parallel interface using the external logic discussed in this application note, it is recommended that REFCLK operate at twice the character rate. For serial links operating between 50–100 MBd, RANGESEL and SPDSEL must be LOW. The REFCLK frequency should range from 10 MHz to 20 MHz in 8-bit mode, and from 8.33 MHz to 16.67 MHz in 10-bit mode.

For 100–200 MBd operation, RANGESEL and SPDSEL must be HIGH. The REFCLK frequency should range from 20 MHz to 40 MHz in 8-bit mode, and 16.67 MHz to 33.33 MHz in 10-bit mode.

RXMODE[1:0]

The CY7C9689A has built-in SYNC-character filtering for received data. Filtering modes are controlled by the RXMODE[1:0] inputs. These character filters can be used to remove SYNC characters in a controlled fashion. Since the TAXI receiver does not have a character filter, and all received SYNC characters are output with a CSTRB pulse, both RXMODE inputs should be configured LOW to emulate the normal operation of the TAXI receiver. This causes all received characters and commands (including all SYNC characters) to be written to the output register for synchronous operation or into the receive FIFO for asynchronous operation.

TXBISTEN and RXBISTEN

The CY7C9689A has Built-In-Self-Test (BIST) functions on both the transmitter and receiver. The BIST state machine in the transmitter provides a predictable but pseudo-random sequence that can be matched to an identical state machine in the receiver. This is a powerful feature for serial link testing and debugging. For normal link operation, BIST must be turned off on both the transmitter and receiver by deasserting TXBISTEN and RXBISTEN to HIGH. Because this BIST capability is not present in the TAXI devices, it can only be used for link validation with other CY7C9689A devices.

RFEN

Reframe enable (RFEN) input controls the internal character framer of the CY7C9689A. This input must be HIGH for the framer to adjust the internal character boundaries upon reception of SYNC characters. To emulate the TAXI receiver, the RFEN input must be HIGH.

CE

The Chip Enable (CE) input must be LOW to enable the transmit and receive parallel interfaces. This enables the FIFO status indicator/flags at all times. Without CE asserted, the TXEN and RXEN interface enables are not recognized.

Summary

This concludes how to convert TAXI to CY7C9689A parallel interface with Transmitter/Receiver Bus Mapping along with static input configuration.

Document History

Document Title: AN1057 - TAXI™ to Cypress CY7C9689A HOTLink® Transceiver Conversion Series: 1. System Parallel Interface

Document Number: 001-34342

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	1505083	YIS	04/09/2000	New application note.
*A	3059376	SAAC	10/14/2010	No technical updates.
*B	3416014	SAAC	10/20/2011	Added Abstract. Changed part number from CY7C9689 to CY7C9689A throughout the document. Updated CY7C9689A Static Control Configuration: Updated $\overline{\text{TXBISTEN}}$ and $\overline{\text{RXBISTEN}}$: Removed the line "For more information, refer the Application Note "HOTLink Built-In-Self-Test". Added Summary. Updated to new template.
*C	4572887	YLIU	11/18/2014	Updated to new template. Completing Sunset Review.
*D	5809717	AESATMP9	07/11/2017	Updated logo and copyright.

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 | PSoC 6

Cypress Developer Community

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

Technical Support

cypress.com/go/support

All other trademarks or registered trademarks referenced herein are the property of their respective owners.

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709

©Cypress Semiconductor Corporation, 2000-2017. 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.