### Zilog Application Note Using the IrDA Port of the Z8 Encore! XP® MCU

AN018402-1207

### Abstract

This Application Note describes the communication between two Zilog's Z8 Encore! XP<sup>®</sup> MCUs via their IrDA ports. This IrDA application uses two Z8F6423 MCU development boards, each equipped with Zilog's Slim Series SIR Transceiver. Each of the Z8F6423 development board is mounted onto one of the Zilog's Modular Development System General-Purpose (MDS-GP) boards. For demonstration purposes, temperature readings are transmitted via the IrDA ports and displayed in a HyperTerminal window.

**Note:** The source code file associated with this Application Note, AN0184– SC01.zip is available for download on <u>www.zilog.com</u>.

Table 1 lists the part numbers associated with the Zilog<sup>®</sup> products used to demonstrate the IrDA application.

#### Table 1. Demonstration Products and Part Numbers

Quantity	Product	Part Number
1	Slim Series SIR Transceiver	ZHX1810
2	Z8F642 Devel- opment Kit	Z8F64200100KIT
2	MDS-GP board	ZGENPRP0100MDS

### Z8 Encore! XP<sup>®</sup> Flash Microcontrollers

Zilog's Z8 Encore! XP<sup>®</sup> products are based on the new eZ8 CPU and introduce Flash memory to Zilog's extensive line of 8-bit microcontrollers. Flash memory in-circuit programming capability allows for faster development time and program changes in the field. The high-performance register-to-register based architecture of the eZ8 core maintains backward compatibility with Zilog's popular Z8<sup>®</sup> MCU.

Z8 Encore! XP MCUs combine a 20 MHz core with Flash memory, linear-register SRAM, and an extensive array of on-chip peripherals. These peripherals make the Z8 Encore! XP MCU suitable for a variety of applications including motor control, security systems, home appliances, personal electronic devices, and sensors.

### **Discussion**

The Z8 Encore! XP MCU is equipped with a number of built-in peripherals, such as SPI, I<sup>2</sup>C, and UART, that are used in conjunction with software to communicate with personal computers and other MCUs/ICs.

An important built-in peripheral of the Z8 Encore! XP MCU is the IrDA Encoder/Decoder (endec). The Z8 Encore! XP MCU can communicate with other IrDA devices through the UART0 console.

### **Theory of Operation**

A number of parameters, such as signal format, voltage level, and type of trans-mission, are significant when applied to the transmission and reception of data through the IrDA port.

The IrDA transmit signal uses the return to zero inverted (RZI) signal format, which inverts a portion of the signal and returns the remainder of the signal to zero. Each signal contains leading, data, and trailing edges. If the data to be transmitted is a logic 0, then the leading and trailing edges are at logic 0 (Return to Zero), and the data is at logic 1 (inverted). If the data to be transmitted is a logic 1, then both the leading and trailing edges, as well as the data, are at logic 0.

The IrDA receive signal uses the return to one (RTO) signal format, which returns the signal to High when the data is zero. When the data is High, the IrDA Receive signal is High throughout its period.

The two formats of the SIR signal are also known as asynchronous formats, because the frame contains start and stop bits, and only one byte of data is transmitted or received in each frame.

For details on UART and infrared data communication, refer to Z8 Encore! XP<sup>®</sup> 64K Series Flash Microcontrollers Product Specification (PS0199), available for download at <u>www.zilog.com</u>.

### **Description of Components**

The application described in this document uses IrDA transceivers for wireless communication between different Z8F6423 MCU development boards. The ZHX1810 transceiver on the Z8F6423 MCU Development Board, which is, in turn, mounted on the MDS-GP Board (see Figure 1 on page 4), are used to demonstrate the usage of the IrDA port. Additionally, an I<sup>2</sup>C-compliant temperature sensor featured on the MDS-GP Board, the MAX6625 IC, is also used to demonstrate infrared communication.

## Developing the IrDA Application with the Z8 Encore! XP<sup>®</sup> MCU

The communication protocols defined for infrared communication consist of three layers:

- 1. Infrared Physical Layer (IrPHY)
- 2. Infrared Link Access Protocol (IrLAP)
- 3. Infrared Link Management Protocol (IrLMP)

These layers form the core of the IrDA architecture and are necessary for a device to be IrDA-compliant. For more information on the implementation of the IrDA stack layers, refer to *IrDA Framer Implementation on the Z8 Encore! XP*<sup>®</sup> *MCU Application Note* (AN0152) available for download at www.zilog.com.

The UART and the IrDA endec of the Z8 Encore! XP MCU must be initialized to communicate using the IrDA port. Table 2 displays the control register used for initializing the IrDA endec. Setting bit 0 (IREN) of this control register enables the endec of the Z8 Encore! XP MCU.

Bits	7	6	5	4	3	2	1	0
Field	MPMD[1]	MPEN	MPMD[0]	MPBIT	DEPOL	BRGCTL	RDAIRQ	IREN
Reset	0	0	0	0	0	0	0	0
R/W	R/W							
Address	F43h and F4Bh							

>

#### Table 2. UxCTL1 Control Register

**Note:** Set bit 0 before initializing the Alternate Function Subregister of the GPIO port to prevent spurious signals during IrDA transmission.

Table 3 displays the data transfer format used for IrDA communication. The IrDA application uses a customized communication protocol for simplicity and ease of understanding.

Table 3. Customized Protocol for IrDA

BOF	Addr1	Addr2	Command	Data	EOF	Checksum

BOF, Addr1, Addr2, Command, EOF, and Checksum are all one byte in length. The size of data varies from one byte to 50 bytes. Transmission is performed by sending an entire frame, beginning with a start byte (BOF = 0.2h in this application), followed by the address of the target Z8 Encore! XP<sup>®</sup> MCU Development Board (Addr1 = 00h in this application), the address of the transmitting Z8 Encore! XP MCU Development Board (Addr2 = 00h in this application), the command (04h for display and 08h for reading temperature), the data, the end of frame (EOF = 15h in this application), and the checksum for the entire frame. The address fields determine which of the Z8 Encore! XP MCU development boards must receive the data. The address fields are followed by either of the two commands described below.

**Display Command**—Displays data in the Hyper-Terminal application.

**Read Temperature Command**—Reads temperature from the temperature sensor on the MDS-GP Board. The data being transmitted is displayed as a string in the HyperTerminal application. If the command issued is a **Read Temperature** command, then the data is omitted. An EOF signal indicates the end of transmission.

#### **Hardware Architecture**

Figure 1 on page 4 displays a basic block diagram for IR communication. The IrDA application uses two PCs in conjunction with the HyperTerminal application to display commands and string messages. The two Z8F642x development boards mounted on the MDS-GP boards communicate with each other through the IrDA transceivers connected to the UART0 devices on each board. A serial RS-232 cable connects the PCs to the respective MDS-GP boards. For more details on mounting the MDS-GP Board (ZGENPRP0100MDS) to the Z8F642x Development Board, see the document referenced in References on page 12.

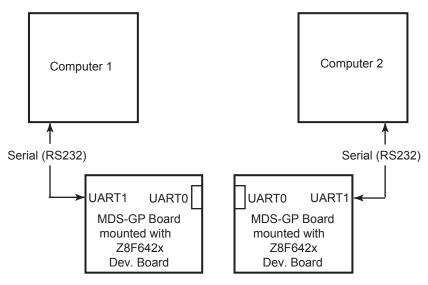


Figure 1. Block Diagram for Communication through IR

#### **Software Implementation**

The software implemented for this IrDA application aims at communication between the two IrDA devices that are each featured on the Z8F642x Development Boards.

The following two processes are used as examples to demonstrate communication using the IrDA port:

- Transmission of a line of text input from the HyperTerminal application associated with one Z8F642x development board to another Z8F642x development board.
- Request for specific data to be transmitted between two Z8F642x development boards. In the IrDA application, the temperature reading from the on-board I<sup>2</sup>C temperature sensor of the MDS-GP Board is also the data that is transmitted from one board to the other.

This section discusses the main routine and the interrupt service routines (ISRs) implemented for the UART and the IrDA peripherals. In addition to initializing the UARTs for transmission and reception of data, the main routine also initializes the I<sup>2</sup>C peripheral, sets up the IrDA port for the transmit and receive interrupts, and displays a menu in the Hyper-Terminal application.

Depending on the command received from the HyperTerminal application, the transmission or reception of data occurs. For details on the software implementation, see Appendix B—Flowcharts on page 14. Figure 2 through Figure 5 on page 11 display the screen shots from the HyperTerminal application.

In this IrDA application, an ISR that allows a Z8 Encore! XP<sup>®</sup> MCU development board to receive commands and addresses from a transmitting Z8 Encore! XP MCU development board (in concert with the HyperTerminal application), is implemented for both UART0 and UART1. This ISR receives the command entered through the HyperTerminal application and transmits the acquired data through an infrared signal. Another ISR implemented for the IrDA Receiver sets the process data Flag, provided that the data is equal to EOF. If the data is not equal to EOF and the data counter is not equal to the terminal count, then the ISR increments the data counter.

The process data routine parses the data received through IR transmission into an address, a command, and data. When a **Display** command is issued, the Display Flag is set to display the data string in the HyperTerminal application. When the **Read Temperature** command is issued, the process data routine reads the temperature from the temperature sensor and returns this value to the requestor board. For flow charts representing the software implementation of this infrared data communication, see Appendix B—Flowcharts on page 14.

Three APIs have been developed for this demonstration so that the  $I^2C$  protocol can communicate with the temperature sensor on the MDS-GP Board. For more details about the temperature sensor, see the documents referenced in References on page 12.

The three APIs used in this demonstration are:

- 1. Send\_Data\_String
- 2. Read\_I2C\_Data
- 3. I2C

Each of these APIs is described in Demonstration APIs on page 6.

#### **Demonstration APIs**

#### void Send\_Data\_String(char Command\_Type,int Data\_Type)

#### Description

Function that sends the command type and the data string to be displayed.

#### Argument (s)

Command_Type	Distinguishes between Display and Read commands.
Data_Type	Determines whether data is a temperature reading or a string.

#### Return Value (s)

void

#### void Read\_I2C\_Data (void)

#### Description

Reads the I<sup>2</sup>C data.

#### Argument (s)

void

#### Return Value (s)

void

#### I<sup>2</sup>C API

#### Description

Initializes the I<sup>2</sup>C Protocol and reads the data from the MAX6625 temperature sensor on the MDS-GP Board.

### Testing

This section lists the basic setup, equipment used, and the procedure followed to test the IrDA application.

#### Setup

Figure 1 on page 4 displays the basic setup used for testing the IrDA application. The IrDA application uses two PCs in conjunction with the HyperTerminal application to display commands and string messages. The two Z8F642x development boards, mounted on the respective MDS-GP boards, communicate with each other through IrDA transceivers connected to the UART0 devices on each board. A serial RS-232 cable connects the PCs to their respective MDS-GP boards.

#### **Equipment Used**

The following equipments are used to test the IrDA application:

- Two Z8F6423 development boards, each mounted on one of two Modular Development System General Purpose boards
- PCs equipped with a HyperTerminal application configured to the following settings:
  - 9600 bps baud rate
  - 8 data bits
  - No parity
  - One stop bit
  - No flow control
- Two RS-232 cables
- Two 10-pin-to-DB9 serial cables

#### **Jumper Settings**

The jumper settings are as follows:

- Insert a jumper at J2 (RS 232-2) on the MDS-GP Board
- Insert a jumper at J2 (IrDA) on the MDS-GP Board

• Insert a jumper at JP3 (Disable IrDA) on the Z8F642x Development Board

#### Procedure

Follow the steps below to test the IrDA application:

- 1. Download the AN0184-SC01.zip file from www.zilog.com. Extract its contents to a folder on your PC.
- 2. Launch ZDSII for Z8 Encore!.
- 3. Connect the MDS-GP Board to a power source. Ensure that the necessary jumpers are placed appropriately.
- 4. Connect the hardware connections as displayed in Figure 1.
- 5. Download the source code to both Z8F642x development boards using ZDSII.
- 6. Launch the HyperTerminal application and configure it according to the specifications provided in the Equipment Used.
- 7. Reset both Z8F642x development boards. Resetting the boards clears the display in the HyperTerminal application, and displays the HyperTerminal prompt.

#### Results

For the requestor board, the HyperTerminal application initially displays the menu. The menu options are: **Transmit Data to be Displayed [D]** and **Read Temperature of Other Boards [R]**. The results obtained for each of these options are described below.

**Read Temperature Command** —If the command entered is **R**, then a prompt for the destination address is displayed in the HyperTerminal window representing the requestor board. As a result, the HyperTerminal window representing the target board now displays the menu (menu options are same as that for the requestor board). After receiving a **Read Temperature** request, the target board prints that it is transmitting the read temperature; after transmitting the data, it displays the main menu a second time. The requestor board then receives the tempera-



ture reading from the target board and prints the temperature reading in its HyperTerminal window. board, respectively, for the **Read Temperature** command.

Figure 2 and Figure 3 display screen shots of the HyperTerminal windows of the requestor and target

test - Hyper		-02
Edit Hew	Cal Transfer Help	
6	5 <u>08</u>	
[D]	Transmit Data to be DISPLAYED.	
[R]	Read TEMPERATURE of Other Boards	
IrDA>r		
	Enter Destination Address: 0 Read Temperature	
[D]	Transmit Data to be DISPLAYED.	
[R]	Read TEMPERATURE of Other Boards	
IrDA>		
	Temperature Reading is: 021.1 degree Celcius from ID [0]	
[D]	Transmit Data to be DISPLAYED.	
[R]	Read TEMPERATURE of Other Boards	
IrDA>		

Figure 2. Requestor Board HyperTerminal for the Read Temperature Command



ending Temperature Reading	
ransmit Data to be DISPLAYED.	
ead TEMPERATURE of Other Boards	
	Transmit Data to be DISPLAYED. ead TEMPERATURE of Other Boards ending Temperature Reading ent Temperature Reading. ransmit Data to be DISPLAYED. ead TEMPERATURE of Other Boards

Figure 3. Target Board HyperTerminal for the Read Temperature Command

**Display Command**—If the command entered in HyperTerminal is **D**, then the requestor board prompts the user to enter the text to be transmitted and the destination address; then transmits the data. The HyperTerminal window for the target board now displays the menu (menu options are same as that for the requestor board). After receiving data, the target board processes it and prints the received text data in its HyperTerminal window.

Figure 4 and Figure 5 display screen shots of the HyperTerminal windows of the requestor and target board, respectively, for the **Display** command.



st - Hyperi Edit Mew	ferminal Cat Transfer Help	-
303	<u>9 00 2</u>	
[D]	Transmit Data to be DISPLAYED.	
[R]	Read TEMPERATURE of Other Boards	
[rDA>d		
	Type Text to Display	
	Press Enter to Transmit	
lext>₩	elcome to ZiLOG	
	Enter Destination Address: 0 Data Display Transmission Success!	
[D]	Transmit Data to be DISPLAYED.	
[R]	Read TEMPERATURE of Other Boards	
IrDA>		

Figure 4. Requestor Board HyperTerminal for the Display Command



fle Edit View Gall Transfer Help	-0-2
	_
IP       PIB       PIB       PIB         ID1       Iransmit Data to be DISPLAYED.         IR1       Read TEMPERATURE of Other Boards         IrDA>       Data Received >>Welcome to ZiLOG         from ID I01       Iransmit Data to be DISPLAYED.         IR1       Read TEMPERATURE of Other Boards         ID1       Iransmit Data to be DISPLAYED.         IR1       Read TEMPERATURE of Other Boards         IrDA>	
Connected 0:10:53 Auto detect \$7600 B-N-1 \$CTIOL GAPS NUM Capture Protection	

Figure 5. Target Board HyperTerminal for the Display Command

#### Conclusion

The program developed for IrDA data communication was implemented success-fully using two Z8F642x development boards, each mounted on an MDS-GP board. The data displayed in the HyperTerminal application and the temperature readings obtained were as expected.

### Summary

This Application Note discusses the usage of the IrDA endec featured in Zilog's Z8 Encore! XP<sup>®</sup> Microcontrollers. Enabling and configuring the Z8

Encore! XP MCU's corresponding control registers, combined with its ease of programming, allows you to utilize the IrDA port of the Z8 Encore! XP MCU. For details on the implementation of the IrDA stack layers, refer to *IrDA Framer Implementation on the Z8 Encore! XP*<sup>®</sup> *MCU Application Note (AN0152)*, available for download at www.zilog.com.

The usage of the I<sup>2</sup>C protocol for reading temperature on the MDS-GP Board is also briefly discussed. With the IrDA function, monitoring or reading temperatures from other boards can be performed over a wireless medium.

### References

The documents associated with Z8 Encore! XP<sup>®</sup> family of products available on <u>www.zilog.com</u> are provided below:

- eZ8 CPU Core User Manual (UM0128)
- Z8 Encore! XP<sup>®</sup> 64K Series Flash Microcontrollers Product Specification (PS0199)
- Z8 Encore! XP<sup>®</sup> 64K Series Flash Microcontrollers Development Kit User Manual (UM0151)
- IrDA Protocols, <u>http://www.irda.org</u>
- IrDA Framer Implementation on the Z8 Encore! XP<sup>®</sup> MCU Application Note (AN0152)
- MDS General Purpose Board Product Brief (PB0146)
- MDS General Purpose Board Quick Start Guide (QS0045)
- Modular Development System General Purpose Board User Manual (UM0169)
- Zilog Developer Studio II–Z8 Encore! User Manual (UM0130)
- Data sheet for MAX6625 and MAX6626, http://www.maxim-ic.com

# <mark>z</mark>ilog<sup>a</sup>

### Appendix A—Glossary

Table 4 lists the abbreviations and definitions used inthis Application Note.

#### Table 4. Abbreviation and Definition

Term/Abbreviation	Definition
BOF	Beginning of Frame
EOF	End of Frame
IrDA	Infrared Data Association
RTO	Return to One Signal Format
RZI	Return to Zero Inverted Signal Format
SIR	Serial Infrared
UART	Universal Asynchronous Receiver/Transmitter

### Appendix B—Flowcharts

This Appendix displays the flowcharts for the IrDA application. Figure 6 displays the initialization of UART0 for IrDA communication, UART1 for communication via HyperTerminal, and I<sup>2</sup>C APIs for reading temperature.

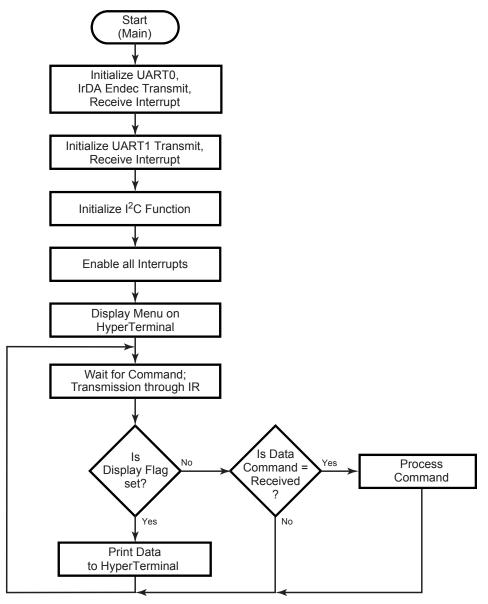


Figure 6. Main Routine

Figure 7 displays the UART1 interrupt service routine.

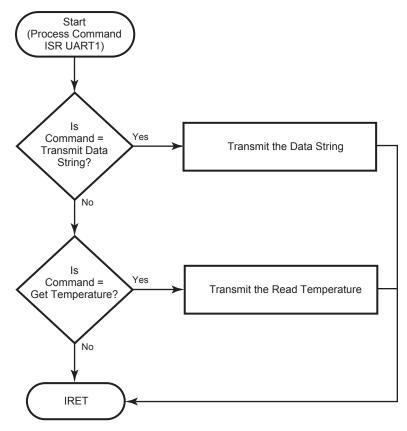


Figure 7. UART1 Interrupt Service Routine

Figure 8 displays the interrupt service routine for the IrDA receiver.

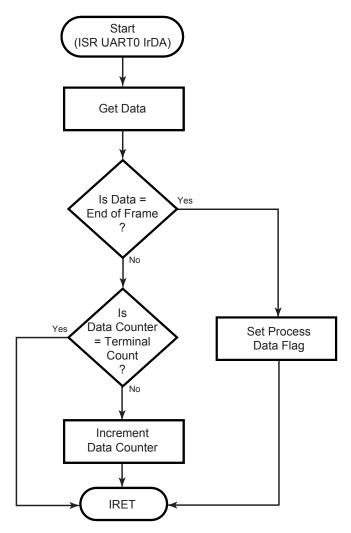


Figure 8. IrDA Receiver Interrupt Service Routine

Figure 9 displays the process data routine in which the data received through IR transmission is parsed.

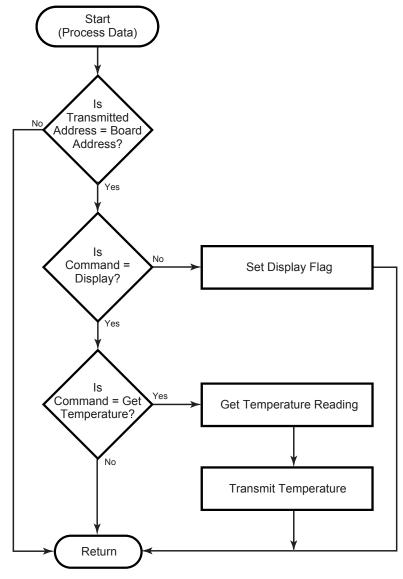
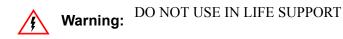


Figure 9. Process Data Routine



#### LIFE SUPPORT POLICY

ZILOG'S PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS PRIOR WRITTEN APPROVAL OF THE PRESIDENT AND GENERAL COUNSEL OF ZILOG CORPORATION.

#### As used herein

Life support devices or systems are devices which (a) are intended for surgical implant into the body, or (b) support or sustain life and whose failure to perform when properly used in accordance with instructions for use provided in the labeling can be reasonably expected to result in a significant injury to the user. A critical component is any component in a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system or to affect its safety or effectiveness.

#### **Document Disclaimer**

©2007 by Zilog, Inc. All rights reserved. Information in this publication concerning the devices, applications, or technology described is intended to suggest possible uses and may be superseded. ZILOG, INC. DOES NOT ASSUME LIABILITY FOR OR PROVIDE A REPRESENTATION OF ACCURACY OF THE INFORMATION, DEVICES, OR TECHNOLOGY DESCRIBED IN THIS DOCUMENT. ZILOG ALSO DOES NOT ASSUME LIABILITY FOR INTELLECTUAL PROPERTY INFRINGEMENT RELATED IN ANY MANNER TO USE OF INFORMATION, DEVICES, OR TECHNOLOGY DESCRIBED HEREIN OR OTHERWISE. The information contained within this document has been verified according to the general principles of electrical and mechanical engineering.

Z8, Z8 Encore!, and Z8 Encore! XP are registered trademarks of Zilog, Inc. All other product or service names are the property of their respective owners.