Application Note

A Simple Console Application for Z8 Encore! XP MCUs

AN034201-1112

Abstract

Console applications are widely used by engineers for ease of project development. For this reason, Zilog has developed a simple console application to allow customers to not only quickly develop their projects, but to provide a simple UI application that employs a common industry command set. Using these commands, developers can enable GPIO as input and output, turn devices on using a relay, or read sensors via the ADC input pin.

The Z8 Encore! XP console application described in this document has been created to occupy a very small footprint (below 3KB) to allow other customer-developed applications to fit into MCU memory. The command set for this application is discussed in the Software Implementation section on page 6.

This console application offers the following features:

- Less than 3KB total footprint
- HyperTerminal connection for commands and responses
- Commonly-used industry command set
- Input and output pin control
- ADC control

Note: The source code file associated with this application note, AN0342-SC01.zip, is available for download from the Zilog website. It has been tested with ZDSII for Z8 Encore! version 5.0.0.

Discussion

A console application is a line command-based program that is used to control the peripherals units of an MCU. These commands are keyed in using a keyboard, and the HyperTerminal emulation program displays the results of these commands and their responses.

The Z8 Encore! XP Console Application is written in the C language and can be downloaded to any of Zilog's Z8 Encore! XP MCUs. This application will accept a line command and return an appropriate response; its command set comprises a simple command and a narrow range of parameters. After the MCU has executed a command, a response message will be returned to the HyperTerminal console, where the response can be seen as a value or a return message.

Commands and Responses

The Z8 Encore! XP Console Application Command Set consists of only eight commands. These commands, listed in Table 1, are each described in this section.

Table 1. Console Application Command Set

SET	ADC
CLR	CBR
GET	RST
CON	?

Note: The Z8 Encore! XP Console Application Command Set is not case-sensitive.

SET

The SET command is used to pull the corresponding [PORT][PIN] to a logic High, which equates to the MCU voltage supply.

Parameters

SET [PORT][PIN] | [PORT] = PA:PH; [PIN] = 0:7.

Syntax

SET<SPACE>[PORT][PIN]<ENTER KEY>

Example

COMMAND	SET PA1
RESPONSE	OK!

CLR

The CLR command is used to pull the corresponding [PORT][PIN] to a logic Low.

Parameters

CLR [PORT][PIN] | [PORT] = PA:PH; [PIN] = 0:7

Syntax

CLR<SPACE>[PORT][PIN] <ENTER KEY>

Example

COMMAND	CLR PA1
RESPONSE	OK!

GET

The GET command is used to get the value of [PORT][PIN].

Parameters

GET [PORT][PIN] | [PORT] = PA:PH; [PIN] = 0:7

Syntax

GET<SPACE>[PORT][PIN] <ENTER KEY>

Example

COMMAND	GET PA1
RESPONSE	1 or 0

CON

The CON command is used to cause a corresponding [PORT][PIN] to be either an input or an output. If [DIRECTION] is equal to 1, then PIN is an input; if [DIRECTION] is equal to 0, then PIN is an output.

Parameters

[PORT], [PIN], [DIRECTION]

Syntax

CON<SPACE>[PORT][PIN][DIRECTION] <ENTER KEY>

Example

COMMAND	CON PA10
RESPONSE	OK!

Note: As the syntax in the above example shows, there is no space between the [PIN] and [DIRECTION] values.

ADC

The ADC command is used to get the ADC value of AN[PARAM].

Parameters

[PARAM] | [PARAM] is 0:7

Syntax

ADC<SPACE>[PARAM] <ENTER KEY>

Example

COMMAND	ADC 0
RESPONSE	ANA0=xxxx

Note: Not all ADC commands are available with each Z8 Encore XP! MCU. See Table 5 on page 15, which lists the Z8 Encore XP! parts pertinent to this application and their available ADC commands.

CBR

The CBR command is used to change the baud rate of the UART peripheral. These baud rates are represented by [PARAM] as indicated in Table 2; the default baud rate is 56kbps.

Table 2. CBR Command: Baud Rate Parameter Values

Parameter Value	Baud Rate (bps)
0	9600
1	19200
2	38400
3	57600
4	115200

Parameter

[PARAM]

Syntax

CBR<SPACE> [PARAM] <ENTER KEY>

Example

COMMAND	CBR 0
RESPONSE	OK!

RST

Issuing the RST command executes a system reset to the MCU.

Parameters

None.

Syntax

RST<ENTER KEY>

Example

COMMAND	RST
RESPONSE	OK!

?

The ? command is used to display a list of commands.

Parameters

None.

Syntax

? <ENTER KEY>

Example

COMMAND	**?**	**Description**
RESPONSE	SET	Set port pin
	CLR	Clear port pin
	GET	Get port pin status
	CON	Configure port pin
	ADC	Get ADC value
	CBR	Change baud rate
	RST	Reset
	?	Show list of commands

Note: If a command not listed in Table 1 on page 2 is entered, or if a command is entered incorrectly, the application will return `Error!`.

Hardware Implementation

This console application was implemented using the Z8F6423, Z8F1680, Z8F0822, and Z8F082A development boards, with serial connections as shown in Figure 1. A serial cable is used to connect these development boards to a PC using HyperTerminal.

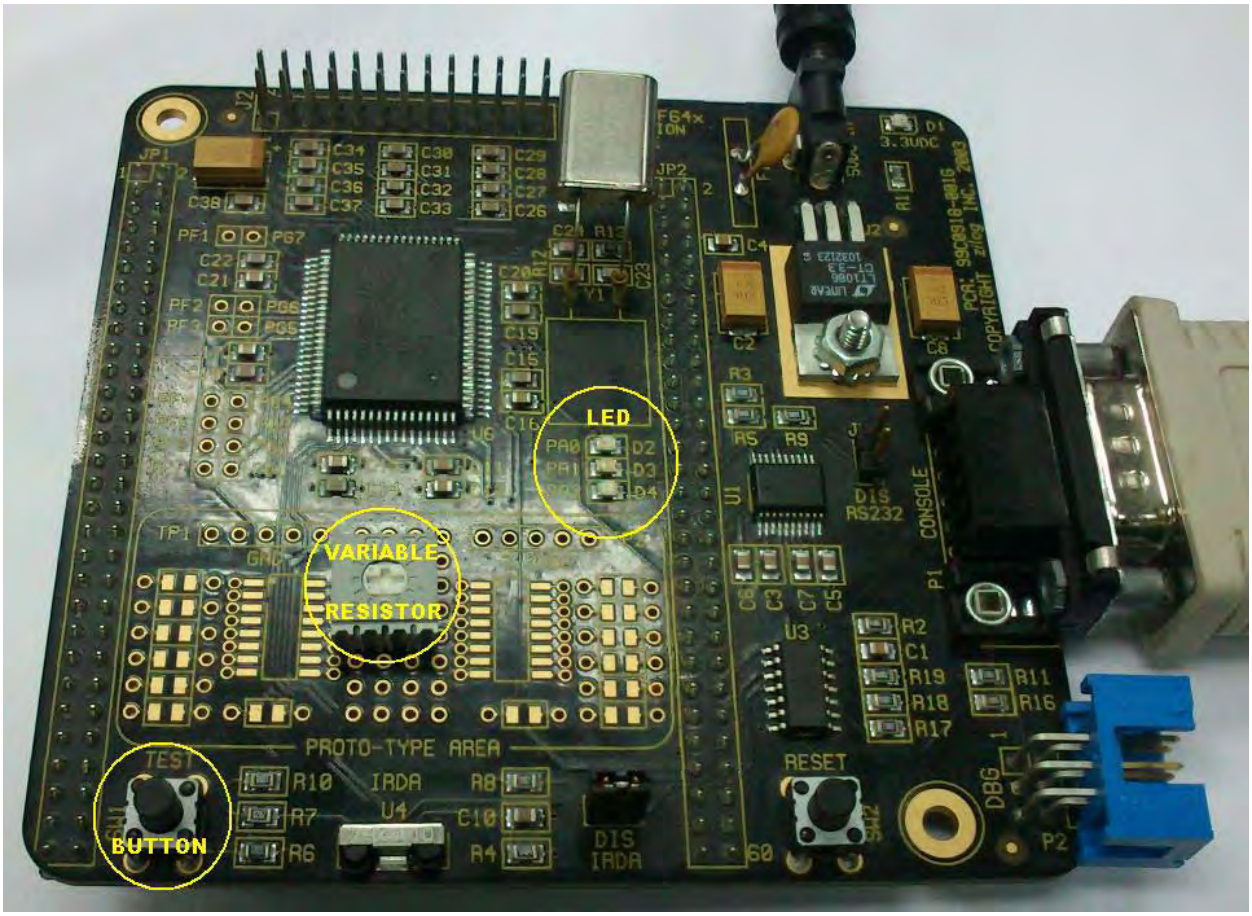

Figure 1. A Serially-Connected Z8F64xx Development Board

Software Implementation

The source code was written in C and the routines were modularized for users to copy and use in their their application requirements.

At code initialization, only UART0 is configured, with a baud rate of 56kbps. The HyperTerminal console will display the message, `Zilog Console Application v1.0`, as shown in Figure 2.

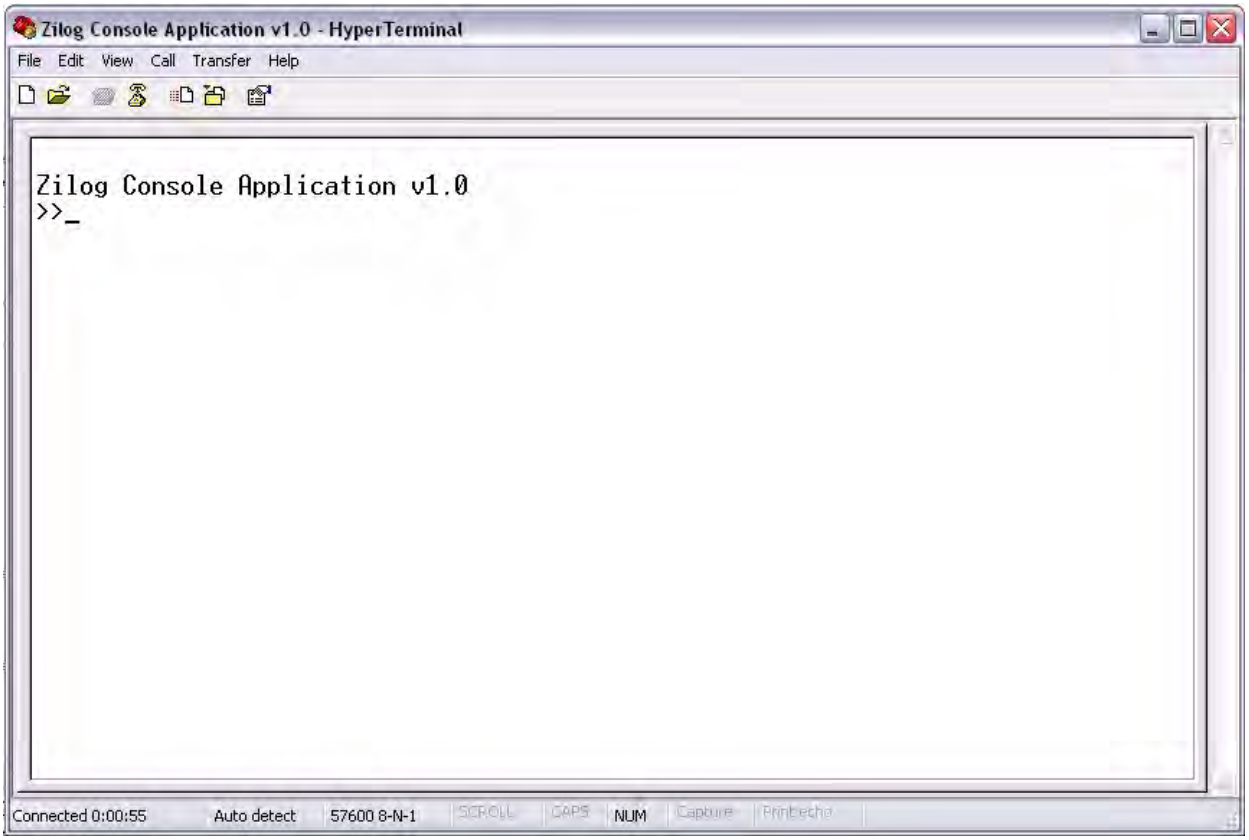

Figure 2. HyperTerminal Window Displaying Zilog Console Application v1.0

Users can enter a command and press the Enter key for command execution. The console application accepts the command while waiting for the command terminator (`0x0A`) to be received; this command terminator indicates the end of a command message. When this `0x0A` value has been received, the console application will process the command message, which is divided into three parts: the Command, the Separator (`0x20`) and the Parameter. After processing the command, the console application will return a message indicating either values or a status.

Table 3 lists the functions and macros of the console application.

Table 3. Console Functions and Macros

Function	Description
void get_input(void)	Accepts user input.
void reset_board(void)	Resets the board.
void read_adc(unsigned char pin_number)	Reads the ADC value of a give pin_number.

Table 3. Console Functions and Macros (Continued)

Function	Description
void get_input(void)	Accepts user input.
void init_uart0(unsigned char baud)	Changes the baud rate of the UART0 to the entered parameter, as follows: **Baud Parameter Index** 0: 9600 1: 19200 2: 38400 3: 57600 4: 115200
#define help()	Displays a list of commands.
*void gpio_bit(unsigned char casenum,unsigned char *c)*	Performs the specific process indicated by case number (0) CON, (1) CLR, (2) SET and (3) GET).

Equipment Used

The tools used to create and test this application are:

- ZDSII – Z8 Encore! v5.0.0
- Z8 Encore! XP MCUs:
  - Z8F6423
  - Z8F1680
  - Z8F0822
  - Z8F082A
- 5V DC power supply
- Serial cable

Testing/Demonstrating the Application

This section discusses a methodology for demonstrating this application and testing its software.

Hardware Setup

Figure 1 on page 6 highlights the test elements on the Z8F64xx Development Board. The LEDs illuminate when the I/O pin are set to High or Low, and the BUTTON switch is used to indicate a GPIO High or Low condition. The VARIABLE RESISTOR is used to vary the ADC input voltages.

Software Configuration

After the hardware is correctly configured, observe the following procedure to correctly compile, load and run the software.

1. Launch HyperTerminal by clicking the Windows **Start** button at the lower left of your desktop and entering **Run** to open the Command Prompt. Enter `hypertrm.exe` into the Command Prompt and press the Enter key on your keyboard. The Connection Description dialog box will appear.
2. Enter any name you prefer into the **Name:** field and click **OK.** The COM1 Properties dialog box will appear. In this dialog, ensure that HyperTerminal is configured to match the values shown in Figure 3, then click **OK**.

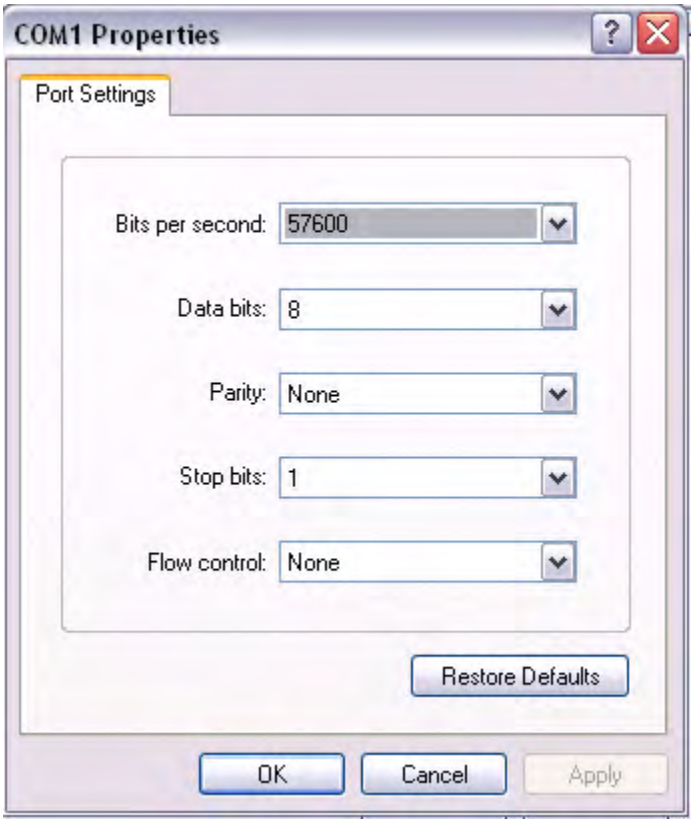

Figure 3. HyperTerminal Settings

3. Download ZDSII – Z8 Encore! v5.0.0 and follow the prompts as they guide you through the ZDSII installation.
1. Download the AN0342-SC01.zip file from the Zilog website and unzip it to your PC's hard drive.
2. Launch the ZDSII IDE.

3. From the **File** menu in ZDSII, select **Open Project** to display the Open dialog box. In this dialog, browse to and open the AN0342-SC01 folder, find the `AN0342.zdsproj` file, and click **Open** to open the project.
4. From the **Project** menu, select **Settings...** to open the Project Settings dialog. In the left panel of this dialog, click **Code Generation**. In this Code Generation panel, ensure that the checkbox labeled **Limit Optimizations for Easier Debugging** is deselected and that **Memory Model:** is set to **Large**.
5. In the left panel, click Debugger. In the Target pane, select the `AN0342_Z8F6423` project by clicking its checkbox.
6. In this same Debugger window, click the **Setup** button, which is just below the Target pane. A small dialog box labeled **Configure Target -- AN0342_Z8F6423** will appear. In this dialog, ensure that the Clock Frequency is set to the **20** (MHz) value. Additionally, ensure that the crystal oscillator on your development board is a 20MHz crystal. Click **OK**.
7. A dialog box will appear, displaying the following message:

   ```
   The project settings have changed since the last build. Would
   you like to rebuild the affected files?
   ```

   Click the **Yes** button to rebuild the project.
8. In the ZDSII main window, select **Go** from the **Debug** menu to step through the debug process. Alternatively, click the Go icon ( ) in the ZDSII toolbar, or press F5 on your keyboard.

Note: If ZDSII should indicate that No USB Smart Cable has been detected, ensure that the USB Smart Cable is properly connected to the PC.

9. In a moment, the HyperTerminal window will appear, as seen in Figure 2 on page 7.

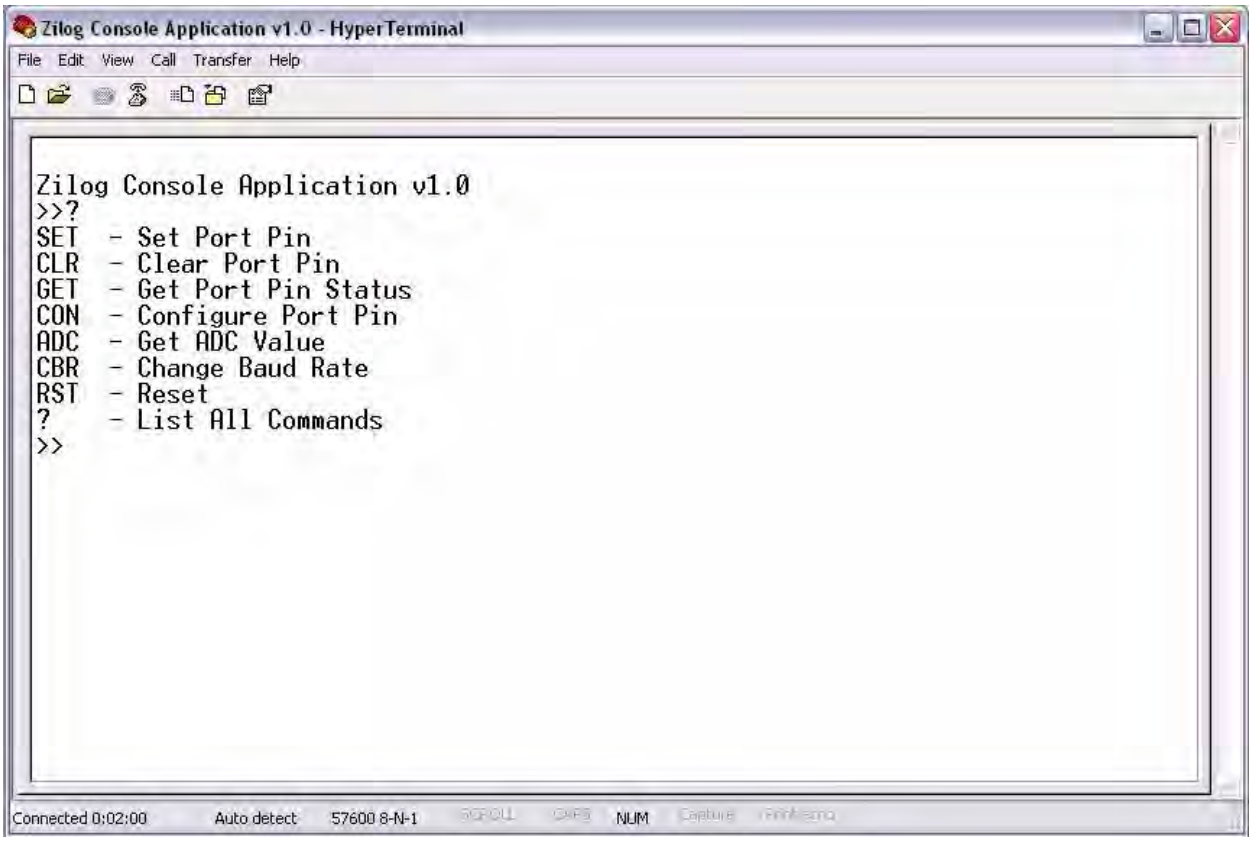

Figure 4. HyperTerminal Displaying A List Of Commands

10. Enter any command in its proper format, as discussed in the Software Implementation section on page 6.

Results

Using the SET and CLR commands in HyperTerminal, the Z8 Encore! XP MCU's GPIOs were configured as outputs that illuminated the LED on the Z8F6423 Development Board. Using the GET and CON commands, these GPIOs were configured as inputs to get the status of a switch attached to the input pin (whether logic High or Low). Using the ADC command, the software read the ADC value to get the voltage level of the potentiometer in the analog input pin. With the CBR command, the baud rate was successfully changed to a preferred rate.

Summary

This document discusses a simple console application for Z8 Encore! XP microcontrollers. The application can turn a specific I/O pin as input or output, make it Low or High, and read I/O pin status if Low or High. Additionally, the application can read the voltage level of the ADC I/O pin.

References

The following documents are associated to Zilog's Z8 Encore! XP series of MCUs and are available free for download from the Zilog website.

- Z8 Encore! XP F64xx Series Product Specification (PS0199)
- Z8 Encore! XP F1680 Series Product Specification (PS0250)
- Z8 Encore! XP F0823 Series Product Specification (PS0243)
- Z8 Encore! XP F082A Series Product Specification (PS0228)

Appendix A. Flowcharts

Figure 5 shows the flow of the main application.

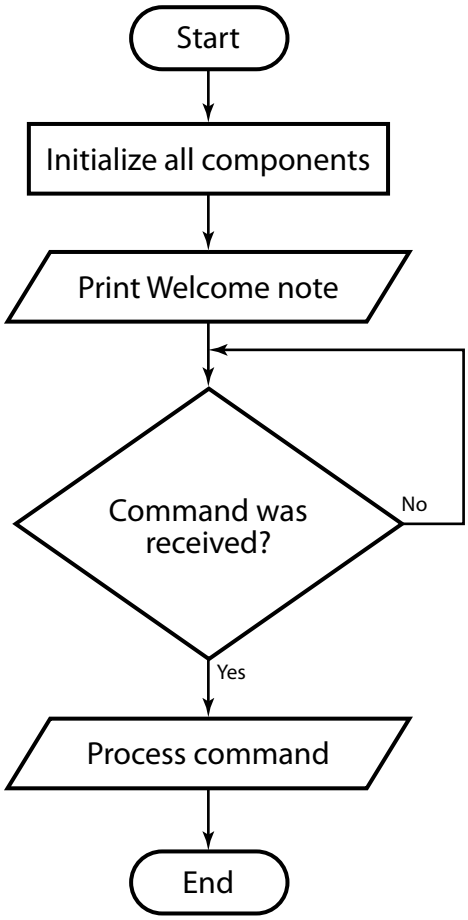

Figure 5. The Main Software Flow

Appendix B. Commands

The syntax of each of the Z8 Encore! XP Console Application Command Set's eight commands, and their responses in HyperTerminal, are listed in Table 4.

Table 4. Commands and Responses

Command	Response	Description
SET<SPACE>[PORT][PIN]<ENTER KEY>	OK!	Set specific port pin.
CLR<SPACE>[PORT][PIN][DIRECTION]<ENTER KEY>	OK!	Clear specific port pin.
CON<SPACE>[PORT][PIN]<ENTER KEY>	OK!	Configure specific port pin as input or output.
GET<SPACE>[PORT][PIN]<ENTER KEY>	1 or 0	Get the status of a specific port pin.
ADC<SPACE>[PARAM]<ENTER KEY>	AN[PARAM]=xxx	Read the voltage level at a specific analog pin.
CBR<SPACE>[PARAM]<ENTER KEY>	OK!	Change the UART0 baud rate.
RST<ENTER KEY>	OK!	Reset the MCU.
?<ENTER KEY>	SET Set port pin CLR Clear port pin GET Get port pin status CON Configure port pin ADC Get ADC value CBR Change baud rate RST Reset ? Show a list of commands	List all commands.

Appendix C. ADC Commands

Table 5 shows the analog inputs for each of the Z8 Encore! XP MCUs, indicated by part number.

Table 5. ADC Commands

	Z8 Encore! XP MCU Part Number			
Analog Input	**80-Pin Z8F642x**	**28-Pin Z8F1680**	**28-Pin Z8F082x**	**28-Pin Z8F082A**
ADC0	√	√	√	√
ADC1	√	√	√	√
ADC2	√	√	√	√
ADC3	√	√	√	√
ADC4	√			
ADC5	√			
ADC6	√			
ADC7	√			

Customer Support

To share comments, get your technical questions answered, or report issues you may be experiencing with our products, please visit Zilog's Technical Support page at http://support.zilog.com.

To learn more about this product, find additional documentation, or to discover other facets about Zilog product offerings, please visit the Zilog Knowledge Base at http://zilog.com/kb or consider participating in the Zilog Forum at http://zilog.com/forum.

This publication is subject to replacement by a later edition. To determine whether a later edition exists, please visit the Zilog website at http://www.zilog.com.

Warning: DO NOT USE THIS PRODUCT IN LIFE SUPPORT SYSTEMS.

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

 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.