This design example shows the capability of Altera® MAX® II, MAX V and MAX 10 to provide general purpose I/O (GPIO) pin expansion via an industry standard I²C bus.

To reduce package size and pin count, the number of general purpose I/Os are limited in many microprocessor-based systems. However, if the system has an I²C interface, this design example shows how to add additional GPIO pins via the I²C bus.

The supported Altera devices are an excellent choice to implement industry standard interfaces, such as the I²C. Their low power, easy power-on feature, and internal oscillator make them ideal programmable logic devices to implement applications such as I²C interfaces to provide GPIO pin expansion.

Related Information
• Design Example for MAX II
  Provides the MAX II design files for this application note (AN 494).
• Design Example for MAX 10
  Provides the MAX 10 design file for this application note (AN 494).
• Power Management in Portable Systems Using MAX II CPLDs
• MAX II CPLD Design Guidelines

GPIO Pin Expansion and I²C

In some cases, it may be required to have access to the GPIO pins from a relatively long PCB trace path within the system (such as in the two different parts of a clamshell cell phone). Because the I²C interface is a two-wire system, the design provides multiple input and output pins at the remote end with just a common two-wire trace. This provides increased design flexibility and also adds to the physical compactness of the entire system. It also enables smaller packaging and a reduced pin count. Devices such as fan controllers, LED status displays, and status indicators can be easily connected and controlled via the general purpose output pins. Similarly, devices such as reset pins and push button switches can be easily coupled to the general purpose inputs provided on the device to serve various applications.
Figure 1: GPIO Pin Expansion via an I^2C Bus

I^2C Interface for GPIO Pin Expansion

The supported Altera device acts as a slave on the I^2C bus and has two pins on its I^2C interface: the I^2C clock SCL and the I^2C data line SDA. The host system, which acts as an I^2C master, communicates with the device (which acts as an I^2C slave). The device presents eight general purpose input ports and eight general purpose output ports to the host. Data, which is transmitted serially over the I^2C bus, is received in parallel at the GPIO pins. This way, all eight general purpose I/Os can be read or written at the same time.

I^2C Interface

For the I^2C interface, the device (I^2C slave) has a built-in 7-bit address and follows the general I^2C protocol. The start signal is sent by the master, followed by the 7-bit address and an R/W bit. When the address broadcast on the I^2C bus matches a slave device’s address, an ACK (acknowledge) signal is sent by the device followed by data according to the read or write signal sent by the master. This is then followed by another ACK signal. The exchange of data continues in this manner until the Stop (P) signal is sent by the master.

Table 1: I^2C Interface Pin Description

<table>
<thead>
<tr>
<th>Signal</th>
<th>Purpose</th>
<th>Direction</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCL</td>
<td>I^2C Clock</td>
<td>Output</td>
</tr>
<tr>
<td>SDA</td>
<td>I^2C Serial Data</td>
<td>Bidirectional</td>
</tr>
</tbody>
</table>
Figure 2: I2C Signal Format

<table>
<thead>
<tr>
<th>S</th>
<th>ADDRESS</th>
<th>R/W</th>
<th>ACK</th>
<th>DATA</th>
<th>ACK</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>Master Write</td>
<td>Slave Write</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

S = Start (SCLK high, SDA high to low)
R/W = Read/Write (1 for Read, 0 for Write)
ACK = Acknowledgement (SDA held low by receiver)
P = Stop (SCLK high, SDA low to high)
Default Slave Address = 0000000 (00h)

GPIO Interface

Whenever the master issues a write condition (R/W=0), the data received on the I2C bus is used to update the general purpose output pins until a stop or a repeat start condition is encountered. Similarly, when the I2C master issues a read condition (R/W = 1), the values at the general purpose input pins are sampled at the ACK bit and transmitted serially over the I2C bus. This process continues until the master issues a stop or repeat start.

Table 2: GPIO Pin Description

<table>
<thead>
<tr>
<th>Signal</th>
<th>Purpose</th>
<th>Direction</th>
</tr>
</thead>
<tbody>
<tr>
<td>8-bit Input</td>
<td>General</td>
<td>In</td>
</tr>
<tr>
<td>8-bit Output</td>
<td>General</td>
<td>Out</td>
</tr>
</tbody>
</table>

GPIO Pin Expansion Using I2C Bus Interface Using MAX II Devices

The detailed description of the implementation is based on the MAX II devices. This application can also be implemented in MAX V and MAX 10 devices.
Implementation involves using this design example source code and allocating \(^2\)C bus lines and GPIO pin expansion inputs and outputs to MAX II GPIOs. GPIO pin expansion is demonstrated on the MDN-B2 demo board with the help of an \(^2\)C simulator that is created using a PC parallel port and interfacing hardware to create an \(^2\)C compliant two-wire bus.

Details about setting up an \(^2\)C environment is described in the Dallas Semiconductor’s Maxim application note AN3230. This utility program uses the parallel port and its interfacing hardware to interact with the MAX II device and provides the SDA and SCL connections, as required on an \(^2\)C two-wire system. When implemented, this design allows inputs from the MDN-B2 demo board (set via DIP switches) to reach the \(^2\)C master. Similarly, data sent by the \(^2\)C master is available on the GPIO output ports (connected to LEDs on the demo board) of the MAX II device. The \(^2\)C master in this demonstration is the user interface on the PC running the parallel port \(^2\)C software.

The following details the implementation of this design example on the MDN-B2 demo board.

### Table 3: EPM240G Pin Assignments

<table>
<thead>
<tr>
<th>Signal</th>
<th>Pin</th>
<th>Signal</th>
<th>Pin</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCLK</td>
<td>pin 39</td>
<td>SDA</td>
<td>pin 40</td>
</tr>
<tr>
<td>GPIO_output[0]</td>
<td>pin 69</td>
<td>GPIO_output[1]</td>
<td>pin 70</td>
</tr>
<tr>
<td>GPIO_input[0]</td>
<td>pin 55</td>
<td>GPIO_input[1]</td>
<td>pin 56</td>
</tr>
</tbody>
</table>
Note: Assign unused pins **As input-tristated** in the Quartus® II software. You must also enable the **Auto Open-Drain** setting on the SCLK and SDA pins. To do this, on the Assignments menu, click **Settings** and then select **Analysis and Synthesis Settings** to enable the **Auto Open-Drain** setting. These settings are followed by a compilation cycle.

Related Information
Dallas Semiconductor’s Maxim application note AN3230
Provides details about setting up an I2C environment

Design Demonstration on the MDN-B2 Demo Board

To demonstrate this design on the MDN-B2 demo board, perform the following steps:

1. Turn on the power to the demo board using the slide switch SW1.
2. Download the design to the device through the JTAG header JP5 on the demo board and a conventional programming cable (ByteBlaster™ II or USB-Blaster™).
3. Keep SW4 on the demo board pressed before and during the start of the programming process. After programming, turn off the power and remove the JTAG connector.
4. To set up a parallel port driven I2C environment on your PC, perform the following:
   a. Download a software utility, such as the Maxim parallel port utility, to communicate with the slave in the I2C defined protocol. Install the parallel port software. (The ParDS2W.exe program downloaded from parallel port software from Direct-IO is used in this example.)
   b. You must install a parallel port driver to enable access to the parallel port in Windows XP or Windows 2000 for this parallel port utility.
   c. After installation, you must configure the Direct-IO program. Open the Windows control panel and click the Direct IO icon. Enter the **Begin** and **End** addresses of your parallel port (normally, this is 378 through 37F; however, confirm your PC’s parallel port address by looking at the settings in Control Panel/System/Hardware/Device Manager/Ports/ECP Printer port (LPT)/Resources)
   d. Configure the parallel port to ECP by changing the BIOS settings when you start up your PC.
   e. Next, select the **Security** tab of the Direct IO control panel and browse to the directory path of the ParDS2W.exe program. Click **Open** and then click **Add** to add the program. The path of this utility is shown in the **Allowed Processes** field. Click OK.
   f. Attach the parallel port I2C dongle that is supplied along with the MDN-B2 demo board. Use an extension chord, if necessary, to extend the parallel port connection closer to your demo board.
   g. Attach the 4-pin socket on the pig tail of the I2C parallel port dongle to the I2C header (JP3) of the demo board so that the red mark on the socket meets pin 1 on the JP3 header.
   h. Open the ParDS2W program, select the appropriate parallel port address of your PC (as seen when configuring Direct IO) and set the **2-Wire Device Address** to 00h.
   i. Finally, you can test the I2C setup on the **Test Circuit** tab to see if you have a Test PASS message in the Status window. If you do, the I2C environment is set.
5. With the parallel port utility you can now perform write and read operations in I2C using the 2-Wire Functions.
6. To perform a write I2C operation, click **Start** and then click **Write Byte**. Enter a hex byte in the field adjacent to Write Data and click **Write Data**. Observe the corresponding value on the eight red LEDs. Click **Stop** after each write operation.
7. Similarly, a read operation is performed by clicking **Start** and then **Read Byte**. The Read window displays the settings on the SW5 dip switch on the demo board. Click **Stop** after each read operation.
## Acknowledgments

Design example adapted for Altera MAX 10 FPGAs by:
Orchid Technologies Engineering and Consulting, Inc.
Maynard, Massachusetts 01754
TEL: 978-461-2000
WEB: [www.orchid-tech.com](http://www.orchid-tech.com)
EMAIL: [info@orchid-tech.com](mailto:info@orchid-tech.com)

## Document Revision History

<table>
<thead>
<tr>
<th>Date</th>
<th>Version</th>
<th>Changes</th>
</tr>
</thead>
</table>
| September 2014 | 2014.09.22 | • Added MAX V and MAX 10 devices.  
|                |         | • Updated template.  
|                |         | • Restructured document.                        |
| December 2007  | 1.0     | Initial release.                                 |