PARALLAX INC
599 Menlo Drive, Suite 100
Rocklin, California 95765, USA
Office: (916) 624-8333
Fax: (916) 624-8003

General: info@parallaxinc.com
Technical: javelintech@parallaxinc.com
Web Site: www.javelinstamp.com
Other: www.parallaxinc.com

Javelin Application Note

Using the Philips PCF8574 Remote I/O Expander

Introduction

The Philips PCF8574 provides an easy way to add eight bits of I/O expansion to the Javelin. Using the I^2C bus and protocol, the Javelin needs just two pins to communicate with the PCF8574. These pins can be shared with other I^2C devices, including up to seven additional PCF8574 nodes for a total I/O expansion of 64 quasi-bidirectional I/O bits on one bus.

This application note demonstrates the PCF8574 with mixed I/O bits on the same device.

Library Classes Used

- I2C.java
- PCF8574.java

Background

The PCF8574 provides general-purpose I/O expansion for microcontrollers. The quasi-bidirectional nature of the PCF8574 pins means there is no control byte for data direction; one simply reads from or writes to the device.

This can be a bit tricky when mixed I/O pins are used on the same PCF8574. In this case, pins that are designated as inputs should be masked (with a "1") during write operations to the device. Since inputs and outputs of the PCF8574 are designed to be active-low, writing a "1" bit to an input pin will prevent a false input on the next read cycle. The PCF8574 class handles this detail by maintaining a pin directions value. Using this value (passed to the constructor), input pins on the PCF8574 are protected.

Program Explanation

This program starts by creating a shared I^2C bus on pins 0 (SDA) and 1 (SCL). The I^2C bus object will be passed to the PCF8574 object. Using this strategy, several I^2C objects can share the same software bus; just as the physical devices share the physical bus.

Next, the PCF8574 object is created. The parameters passed to the constructor are the I^2C bus object, the device address (0 – 7) and the I/O pins definition. For the I/O pins definition, a "1" bit signifies an input, a "0" bit signifies an output.

Finally, a string buffer is created for our display messages, as well as an integer to hold the switch inputs returned from the PCF8574.

The program code starts by clearing the message window, then checking for the presence of the defined PCF8574 device. If it is not present, an error message will be displayed. Assuming the PCF8574 is connected properly, the program will display a four-bit counter on the its LEDs and read the and display the four switch input bits.

As stated earlier, the inputs and outputs of the PCF8574 are active low. For this reason, values written to and read from the device are inverted with the bit-wise inversion operator (~). Since the inputs appear on the upper four bits of the PCF8574, the value read back is shifted right by four bits and then masked to make sure that the bits four through seven of the *switches* value are cleared.

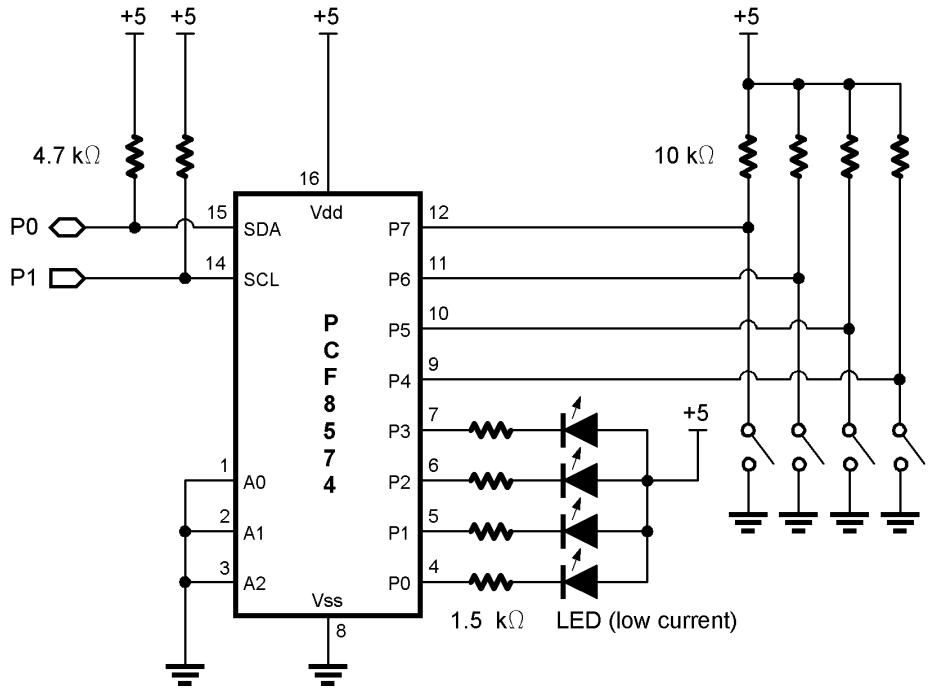
Data for display is constructed in a string buffer to conserve memory (since string objects cannot be recovered). The inputs are actually scanned five times between each counter write cycle. This makes the inputs very responsive while providing a delay between *counter* write cycles. To display the count at a specific interval, a timer object could be created to control the LED display.

Code Listing

```
// PCF8574 demonstration program
// -- by Jon Williams
// -- Applications Engineer, Parallax
// -- jwilliams@parallaxinc.com
// -- Updated: 20 July 2002

/*  PCF8574 connections (device Addr = 0):
 *
 *  Pin  1: A0 --> Vss
 *  Pin  2: A1 --> Vss
 *  Pin  3: A2 --> Vss
 *  Pin  4: P0 --> 1.5K --> LED [cathode] --> Vdd
 *  Pin  5: P1 --> 1.5K --> LED [cathode] --> Vdd
 *  Pin  6: P2 --> 1.5K --> LED [cathode] --> Vdd
 *  Pin  7: P3 --> 1.5K --> LED [cathode] --> Vdd
 *  Pin  8: Vss
 *  Pin  9: P4 --> 10K --> Vdd; N.O. switch --> Vss
 *  Pin 10: P5 --> 10K --> Vdd; N.O. switch --> Vss
 *  Pin 11: P6 --> 10K --> Vdd; N.O. switch --> Vss
 *  Pin 12: P7 --> 10K --> Vdd; N.O. switch --> Vss
 *  Pin 13: Int\  (not used)
 *  Pin 14: SCL --> I2C bus (pulled up to Vdd through 4.7K)
 *  Pin 15: SDA --> I2C bus (pulled up to Vdd through 4.7K)
 *  Pin 16: Vdd
 */

import stamp.core.*;
import stamp.peripheral.I2C;
import stamp.peripheral.io.PCF8574;

public class demoPCF8574 {

  final static char CLR_SCR = '\u0010';
  final static char HOME    = 0x01;
```

```
  public static void main() {

    // create bus for I2C devices [SDA = pin0, SCL = pin1]
    I2C i2cBus = new I2C(CPU.pin0, CPU.pin1);
    // create PCF8574 object with mixed I/O
    // -- device address is 0
    // -- bits 0-3 are outputs; 4-7 are inputs
    PCF8574 ioPort0 = new PCF8574(i2cBus, 0x00, 0xF0);
    // create buffer for screen messages
    StringBuffer msg = new StringBuffer();
    // storage for switch inputs
    int switches;

    // start of code
    System.out.print(CLR_SCR);
    if (ioPort0.isPresent()) {
      while(true) {
        // create binary counter for LEDs
        for(int counter = 0; counter <= 15; counter++) {
          // write [inverted for active-low] counter bits to PCF8574
          ioPort0.write(~counter);
          // scan inputs between LED write cycles
          for (int scan = 0; scan < 5; scan++) {
            // get switch inputs
            // -- inputs are inverted for active low
            // -- shifted to align with bit 0 of variable
            switches = ~ioPort0.read() >> 4 & 0x0F;
            // update message screen
            msg.clear();
            msg.append(HOME);
            msg.append("LEDs = ");
            msg.append(counter);
            msg.append("  \n\r");
            msg.append("Switches = ");
            msg.append(switches);
            msg.append("  ");
            // show binary version of switches
            for (int swBit = 0x08; swBit > 0x00; swBit >>= 1) {
              if ((switches & swBit) == 0)
                msg.append("0");
              else
                msg.append("1");
            }
            msg.append(" ");
            System.out.print(msg.toString());
          }
        }
      }
    }
    else {
      System.out.print("Error: PCF8574 not found");
    }
  }
}
```

Schematic



Sources

- Parallax
  http://www.parallaxinc.com/html_files/component_shop/plus_pack.asp

Reference Documents and Information

- PCF8574_2.PDF
  http://www.semiconductors.philips.com/pip/pcf8574p

Author Information

- Jon Williams
  Applications Engineer, Parallax
  jwilliams@parallaxinc.com

- Published: July 20, 2002