University of Peradeniya
Department of Computer Engineering

CO326 : Computer Systems Engineering: Industrial Networks

USB Interfaced General Purpose Digital Input Output Bus

Group 04

E/14/158 : Gihan Jayatilaka
E/14/339 : Suren Sritharan
E/14/379 : Harshana Weligampola

Contents

1 Abstract

Interfacing PCs (personal computers) to a wide range of input output devices is a major requirement in developing cyber-physical systems. This project proposes a universal serial bus interfaced digital IO interface to enable PCs to communicate with any arbitrary device. The project designs a protocol for the communication, develops hardware/software for it to be functional and carries out a performance bench-marking.

2 Proposal

2.1 Introduction

The generic personal computers (PC) are not intended for directly interfacing with external voltage signals. Usually this is done through serial, parallel or GPIB interfaces.

2.2 Problem

Modern personal computers are getting rid of all hardware ports except for a few general purpose communication ports. USB ports (but of different versions) seems to be the only consistent port through most systems. There is no way of directly sending voltage signals to the computers anymore. There is a need for a system to interface voltage signals to the computers through USB ports now.

2.3 Proposed Solution

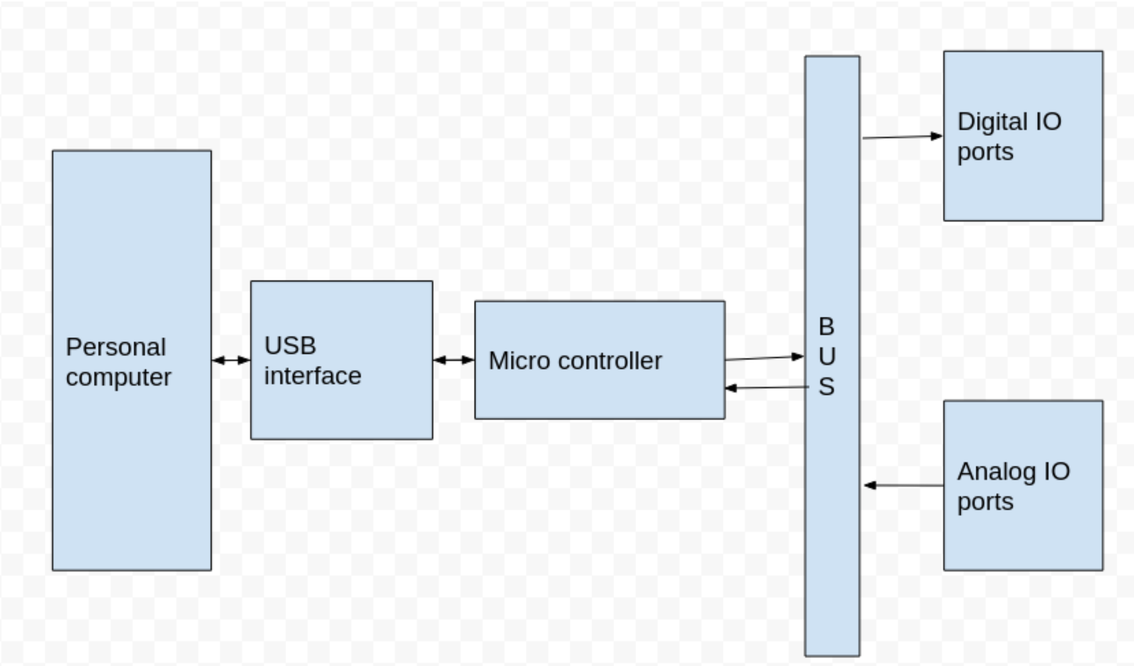

Figure 1: High level diagram of solution

3 Project Work

The details of the project have been divided into 3 main sections.

- Hardware
- PIC software
- PC software

3.1 Hardware

The major component of the hardware interface is the PIC18F4550[1]. The PIC18F4550 is a Programmable IC which can be used to interface a USB with other ports. It supports high speed USB 2.0 interface and has 16 end points. Apart form this other components such as 8 bit registers (74HC245), 8 bit buffers (74HC241), Oscillators, capacitors and resistors were used to create the circuit.

The circuit designed can be used for 8 bit bidirectional I/O with 2 devices. The design and simulation of the circuit was done using proteus. A figure of the entire circuit is shown in the diagram below.

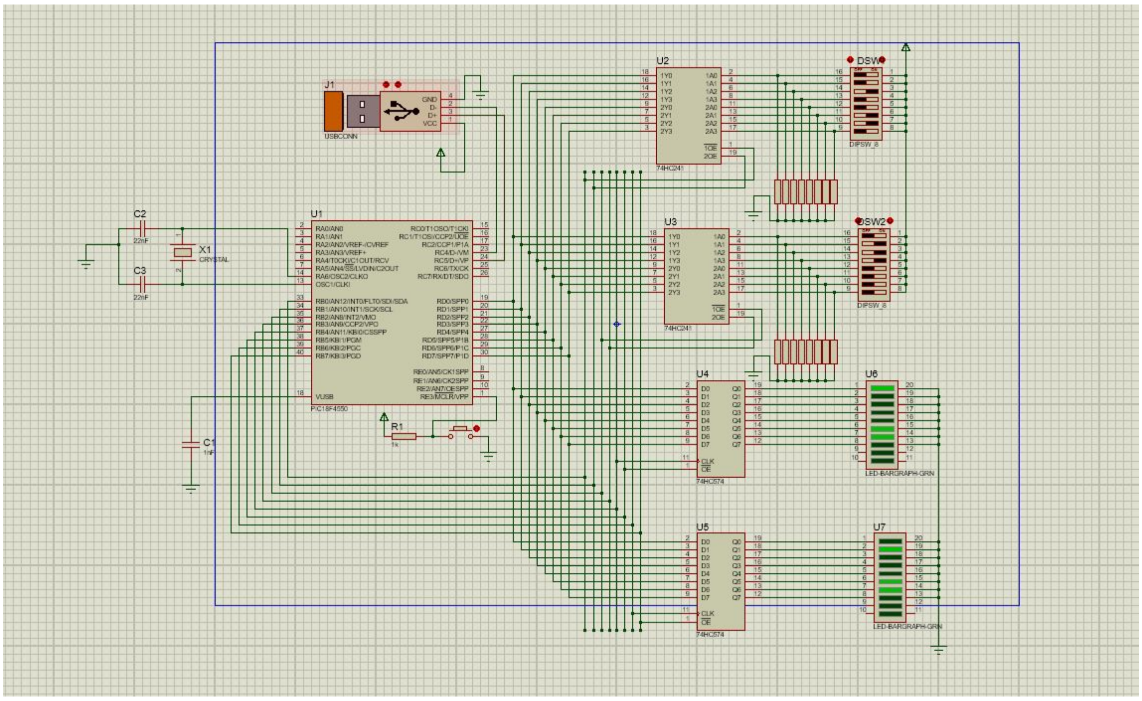

Figure 2: Circuit design

The PIC has 40 pins[1]. However, only a few of those pins were required for our purpose.

4 of the pins are used for power (2 x Vcc, 2 x Vdd). One of these 2 pairs were connected to the power line of the USB connector. The D+ (23) and D- (24) pins are connected to the data transmission lines of the USB connector.

Pins 13 and 14 can be used to connect to an external clock. In the circuit provided, an external clock with 20MHz frequency was connected to these pins as shown in the diagram.

The PIC has 4 set of ports

- RA0- RA6

- RB0- RB7
- RC0- RC7
- RD0- RD7

All 4 ports are 8 bit (or 7 bit) bidirectional IO ports. They can be used for both analog and digital transmission and specific ports support different other protocols (e.g. : SPP, SPI, I2C, etc). However, for our purpose we'll only be using 2 8 bit ports for digital transmission. Note that analog data can also be transmitted using the same ports (with slight modification to the circuit). However, due to the complexity of the new system (introduction of A/D and D/A converters) this wasn't completed.

The two ports used in our circuit are:

- RD0 - RD7 : 8 bit Data port
- RB0 - RB7 : 8 bit Control port

The D port is used to transmit a byte of data. The data can either be written or read from these 8 data lines. The operation (read / write) depends on the port configuration which is set by the protocol (defined below)

The B port is used to select the device to which the data must be sent to / the device from which the data must be read. The control lines are used to control the enable pins of the registers and buffers. The values of these pins are set based on the protocol defined below.

3.2 PIC Software

3.2.1 Protocol

The device is used to communicate between the PC and any unknown device. There are two 8-bit input ports and two 8-bit output ports. This device can read any data on the input port or send any data to the output port. In order to achieve this functionality, a specific protocol is defined for the PC and PIC-device communication.

There are 3 types of data transmitted from PC to the device.

- Data byte
- Address byte
- Control byte

The Control bytes are identified from the first 4 bits of the bytes. If the first 4 bytes are '0101XXXX', then it is categorized as a control byte.
There are 2 main types of control bytes (Read/Write). This read/write type of the control byte is encoded in the most significant bit.

- 1 - Write
- 0 - Read

If it is a read control byte, then the device waits till the PC sends the address of the read port. In the current architecture, the addresses for the input ports are defined as '0' and '1'. When a correct address is received the device sends a data byte of the corresponding port to the PC.

If it is a write control byte, then the device waits till the PC sends the address of the write port. In the current architecture, the address for the output ports are defined as '2' and '3'. After receiving a correct address, the device waits till the data byte from the PC. After that, the device writes the data to the corresponding output port.

In summary, there are 3 main steps to read/write byte through the device.

1. Send control byte
2. Send address
3. Read/Write data

Control Byte

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
CTRL3	CTRL2	CTRL1	CTRL0	-	-	-	R/W

bit0	0 - Read
	1 - Write
bit1,bit2,bi3	Undefined
bit4,bit5,bit6,bit7	0101 - Control byte

3.2.2 PIC Software

There are four types of packets in the protocol. The packets can be 8 bits or 16 bits long. (16 bit is used by utilizing two 8 bit ports) PIC software sets relevant bits and sends clock signals to the registers in input/output ports when correct sequence of data is received to the device as mentioned in the PIC protocol section. Also, the software handles the USB communication between the PC and the device.

3.3 PC software

PC side sofware was written in JAVA. The whole system was written on top of the USB library jSerialComm [2].

3.3.1 Library

The library is written in JAVA to enable the communication between the PC and the devices connected to the bus.
There are two device types in the library as,

- Input device
- Output device

Both these device types have

- 8 bit data
- 4 bit address
- A device ID

The only functionality of the input devices is

- **read()**
  This function call returns a byte corresponding to the digital read from the particular input device.

The output devices have functionalities

- **read()**
  This function call returns a byte corresponding to the digital signal already written to the output device.
- **write()**
  This function call writes a byte of digital data to the output device.

3.3.2 GUI

A GUI was implemented for easy usage of the system. The GUI has 2 functionalities

Configure the devices connected to the bus

Read from or write to the devices connected to the bus

The GUI starts from a wizard. The first window asks for the number of devices connected to the bus.

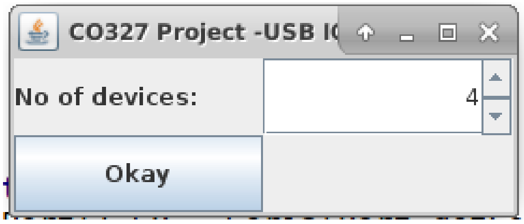

Figure 3: Number of devices

Then the user have to input the information about the devices connected to the bus – the device name and whether it is an input device or an output device.

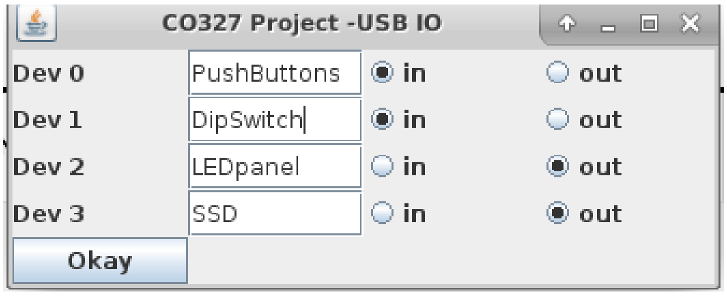

Figure 4: Device information prompt

Once the device information is inserted, the sofwtare reaches the control panel.

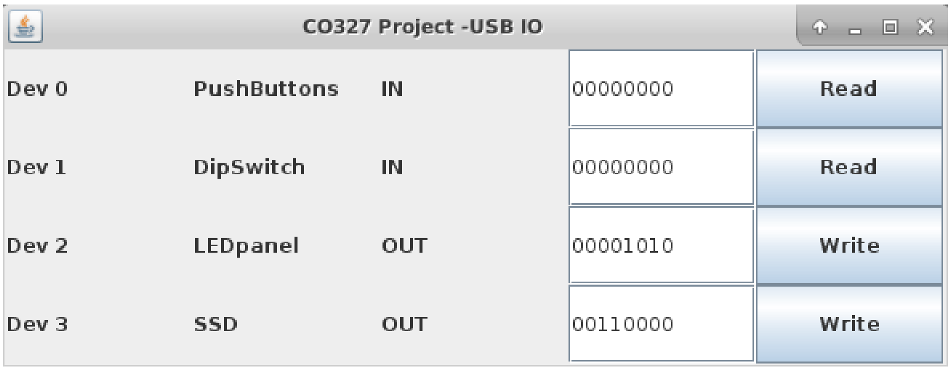

Figure 5: Control panel

The control panel is used to write data to the devices or read data from devices. The given screen capture is for a bus with two input devices and two output devices connected.

3.4 PCB Design

A PCB was designed for the project using Fritzing software[3].

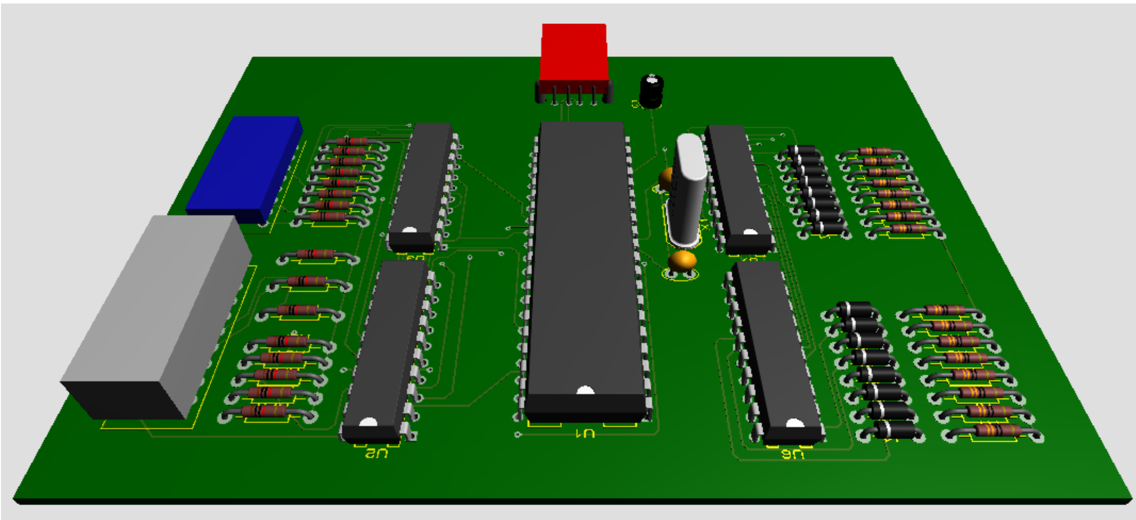

Figure 6: PCB Design, 3d View

The design (top view, bottom view, design files, gerber files for fabrication) is attached in the appendix.

3.5 Work distribution

Task	Gihan	Harshana	Suren
Hardware design	✓	✓	
Hardware simulation			✓
Hardware implementation			✓
Hardware testing			✓
PIC protocol design	✓	✓	
PIC software design		✓	
PIC programming			✓
PIC testing			✓
PC software design	✓		
PC software implementation	✓		
PC-PIC testing and performance analysis			
Documentation	✓	✓	✓

Table 1: Contribution

4 Results and discussion

4.1 Performance analysis

The performance analysis was done by a script that continuously send and receive bits.

Test setup

- 2 input width bytes (registers)
- 2 output width bytes (registers)
- Program (See appendix)

The following results were achieved.

Experiment	Bytes written / sec	Bytes read / sec	Bytes transferred / sec
Read only	122	0	122
Write only	0	130	130
Read and write	51	47	98

Table 2: Caption

4.2 Achieved objectives

- Completed the hardware prototype
- Completed the PIC software
- Completed the PC software
- Completed the benchmarking
- Completed the documentation
- Completed the PCB design

4.3 Unachieved objectives

- Fabricating the PCB

4.4 Advantages and disadvantages of the approach

4.5 Advantages

- Works on almost every computer with a USB port.
- Works on any operating system (since the software runs on java virtual machine)
- Low cost (PIC micro-controllers, registers and other hardware are relatively cheaper)
- Intuitive GUI.

4.6 Disadvantages

- Slow data transfer (since the system depends on the high level java based APIs for serial communication with the PIC). To solve this a low level solution (Device Driver) would be required.
- The PCB design is complicated due to the presence of a bus.

5 Future work

The interface is based on a user defined protocol.The protocol is a simple protocol with send and receive commands. Even though the provided solution performs well under controlled condition, under a real life scenario with high data rates, the performance (speed) of the interface may be insufficient. Although the

6 Photos of group members


E/14/158


E/14/339


E/14/379

7 Appendix

7.1 Code

Github project :	https://github.com/gihanchanaka/CO326-Computer-Interfacing/tree/master/Project-USB/
PC GUI:	https://github.com/gihanchanaka/CO326-Computer-Interfacing/tree/master/Project-USB/USBIO-GUI_Maven/src/main/java
PC library:	https://github.com/gihanchanaka/CO326-Computer-Interfacing/blob/master/Project-USB/USBIO-GUI_Maven/src/main/java/Device.java
PIC software:	https://github.com/harshana95/CO326-Computer-Interfacing/blob/master/Project-USB/USB-PIC-IO/algo/USB-PIC-IO-Algo.c

7.2 Software

Windows XP image for simulation with all drivers	https://goo.gl/1PbKui
Flowcode[4] for PIC hex file compilation	https://goo.gl/p1sWYu

7.3 Other material

PCB Design	https://github.com/gihanchanaka/CO326-Computer-Interfacing/tree/master/Project-USB/PCB
PCB Design (gerber files)	https://github.com/gihanchanaka/CO326-Computer-Interfacing/tree/master/Project-USB/PCB/gerberFiles

References

[1] *PIC 18F4550 Datasheet*. `http://www.microchip.com`. Accessed: 2015-03-12.

[2] *jSerialComm - Platform-independent serial port access for Java*. `http://fazecast.github.io/jSerialComm/`. Accessed: 2019-02-24.

[3] André Knörig and Brendan Howell. "Advanced prototyping with fritzing". In: Jan. 2010, pp. 341–344. DOI: `10.1145/1709886.1709970`.

[4] Dogan Ibrahim. "Using flowcode in graphical embedded system programming". In: *ELECTRONICS WORLD* 119 (Feb. 2013), pp. 12–15.