Multiprocessing Computer System Design
A course with Hardware - Software System Design Approaches

Omer Farook, Chandra R. Sekhar, and Essaid Bouktache
Purdue University Calumet
ofarook@purdue.edu, bouktae@calumet.purdue.edu, crsekhar@calumet.purdue.edu

Abstract

The paper expounds the current status of the subject content area for the course "Multiprocessing Computer System Design". This is a 400 level course that has climbed through the evolutionary ladder of myriad of Technological changes and has kept its currency. The course integrates many of the earlier subject content areas and provides the students the ability to customize a viable system design utilizing the power of multiple processors. The paper examines the subject area in its current status and provides an overview of hardware interfaces. Software design tools and platforms utilized.

The paper discusses in detail a list of laboratory exercises which glue the hardware components with the software in achieving the desired system's functionality. The course culminates toward the end in a Final Class Project. The paper also discusses the pedagogical underpinnings of the course that are as important as the content of the course. The course utilizes the pedagogy of collaborative learning and student empowerment, so that the students take charge of their own learning process.

In its current form the course is PC centric and utilizes the power of native PC processor, but the system design incorporates additional Microcontrollers for dedicated functionality. Thus the system design methodology is multiprocessing in approach. Through this course the Students in Electrical and Computer Engineering Technology program develop the design template that they utilize further in completing a two course sequence of Senior Design and become industry ready.

I. Introduction

The need to go beyond the basics of providing an introductory course in the microprocessor or microcontroller in Engineering and Engineering Technology type curriculums has long been over due. The subject matter covered in System Design has matured to the extent that it has been the subject of curriculum content in the form of two or more courses in most of the
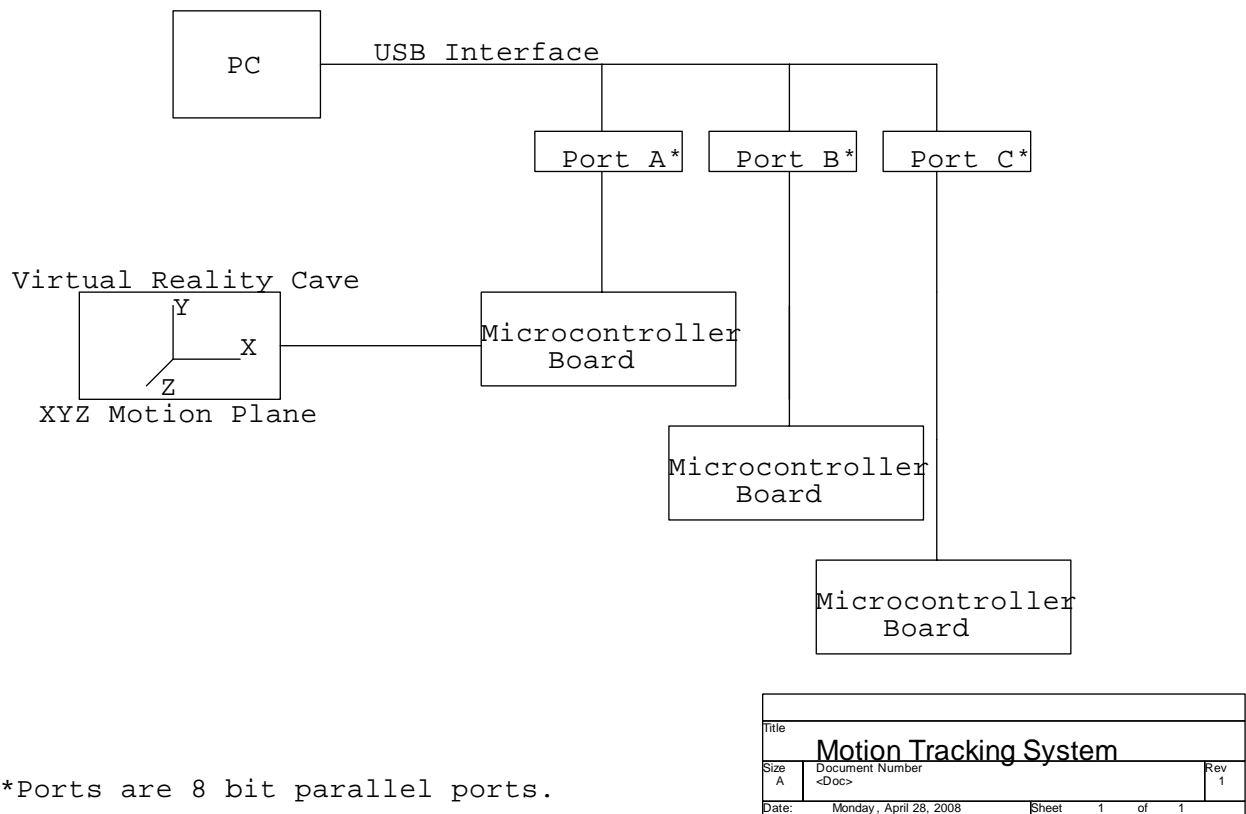
universities[1]. The subject course which is the subject of this paper is a 400 level course in the Electrical and Computer Engineering Technology Department at Purdue University Calumet. This is preceded by two courses: 1) a C or C^{++}, programming course, that covers the C or C^{++} language constructs with emphases on bit manipulation, 2) an introductory microcontroller based Embedded System design course, that covers the architecture of the microcontroller along with its software design implemented with C language. The Embedded System Design course is based on Microchip's 16F88 microcontroller[2].

II. Definition of Multiprocessing Computer System Design

The subject of this paper is defined as the utilization of multiple processors or microcontrollers which provides or defines the system's hardware functionality. The system's software functionality is made up of a host of discrete software pieces that are implemented or targeted on host of microcontrollers or microprocessors. In the case of a distributed systems[3] the subject hardware and software is implemented at distributed geographical locale. Thus **Multiprocessing** is the use of two or more central processing units (CPUs) within a single computer system and *it* further refers to the execution of multiple concurrent software processes in a system as opposed to a single process at any one instant of time.

III. A Generic Multiprocessing Computer System

In Figure 1., a generic **Multiprocessing Computer System** is represented. The system is a skeletal representation of hypothetical Motion Tracking System. For the purposes of subject matter development the representation is kept bare minimum.

Figure 1. A Generic Multiprocessing Computer System

The subject hypothetical system is PC centric. PC with its rich functional resources could implement most if not all the functionality of the system. The Motion Tracking system will monitor say XYZ motion of a point (or multiple points) of a subject in Virtual Reality Cave. Hardware implementation of such a system is achieved in Laboratory by 1) PC with, 2) USB to 3 Parallel ports interface, 3) Microcontroller board connected to 4) gyroscopes, accelerometers and magnetometers etc.

The software design of this hypothetical system will consist of 1) an application program that is running on PC and 2) an embedded software program targeted to the specific microcontroller that is utilized. Depending upon the software designed and functional mapping the system could have the functionality of monitoring, recording, displaying and controlling the desired system's parameters.

Through this simple hypothetical system we could demonstrate that the level of students' learning has been elevated to appreciate the concepts and aspects of modularized system design. Students understand, appreciate and put into practice the software design concepts and aspects of structured programming and portability and reusability of code. Students not only are cognitive of portability of system's functions between hardware and software but as well implement it in their own system design.

IV. PC Centric System

PC is chosen as the center piece of the system as it provides a host of desirable features and functions at a cost - benefit factor which cannot be ignored. It provides standardized interface in terms of display devices, input devices, storage devices and connectivity. The most important feature of all is the PC's operating system. PC's operating systems lend a wealth of functionality at a very low cost (Windows) or at no cost (Linux).

V. PC Software Design Considerations

Operating System:

The very first consideration is with the regard to choice of operating system. The choice probably is to be made between Windows and Linux. A comparative advantage of one operating system over the other deserves a separate discussion and is beyond the scope of this paper. Windows has two main lines. The older flavors are referred to as "Win9x" and consist of Windows 95, 98, 98SE and Me. The newer flavors are referred to as "NT class" and consist of Windows NT3, NT4, 2000, XP and Vista. The flavors of Linux are referred to as distributions. All the Linux distributions released around the same time frame will use the same kernel (the guts of the Operating System)[4]. Due to University Lab support policies presently we use Windows (this is in no way an endorsement of windows).

Structured vs. Object oriented Programming (OOP):

The subject course utilizes strict use of Structured C^{++} programming. Pointers are used for all inter-functional communication. OPP methodology is not used for this course.

Graphical User Interface (GUI) vs. Console Programming:

Both Linux and Windows provide a GUI and a command line interface. This further provides another choice between the software designs of one genus over the other. In the course under discourse for this paper the program design is performed with Console type programs. Even though the GUI programs are desirable, but the learning curve for GUI program is deep and could not be accommodated with the time frame for this course. However in the Electrical and Computer Engineering Technology curriculum there is another 400 level course that exclusively deals with GUI and Object Oriented Programming (OOP) methodology.

VI. System's other Processors

For the purpose of this course we utilize mostly Microchip 16F88 microcontroller. Any other type of microcontrollers (Atmel, Infineon, Freescale, STMicroelectronics, Texas Instruments, Analog Devices, etc.) could as well be used. In this course we use Microchip for its market share in the embedded applications and due to the fact that students learn about the Microchip PIC microcontroller in the course that is a prerequisite to this course. .

Microchip PIC Software Development Tools:

The course utilized C language for all the software development of 16F88 microcontroller. The compiler for the course that we utilized is Custom Computer Service's CCS v-4 C compiler. This could be invoked from the MPLAB, the Integrated Development Environment (IDE) that is freely available from Microchip Technologies. The students could do the software development in MPLAB IDE and also could perform the simulation of the software as well. The other commercially available C Compilers are: 1) HI-Tech PICC v.9.50. 2) IAR Embedded Workbench v.2.21. 3) Forest Electronics C Compiler v.14. 4) B Knudsen CC5X and CC8E C Compiler and 5) Source boost C Compiler.

Microchip PIC Hardware Design and Development Tools:

The authors have designed an in house hardware development platform, whose schematic is provided in Figure 2. The development board provides headers for accessing all the ports along with VDD and VSS. It has also a Serial port interface to access the PC and communicate via Rs232 port for down loading the program. By adding another serial port interface (not in the schematic), the development board could utilize the monitor display of PC as its extension. It has on board opto-isolators to protect the ports inputs and outputs. There are 2 seven segment displays, eight dip switches and eight toggle switches and eight LEDS for immediate interfacing.
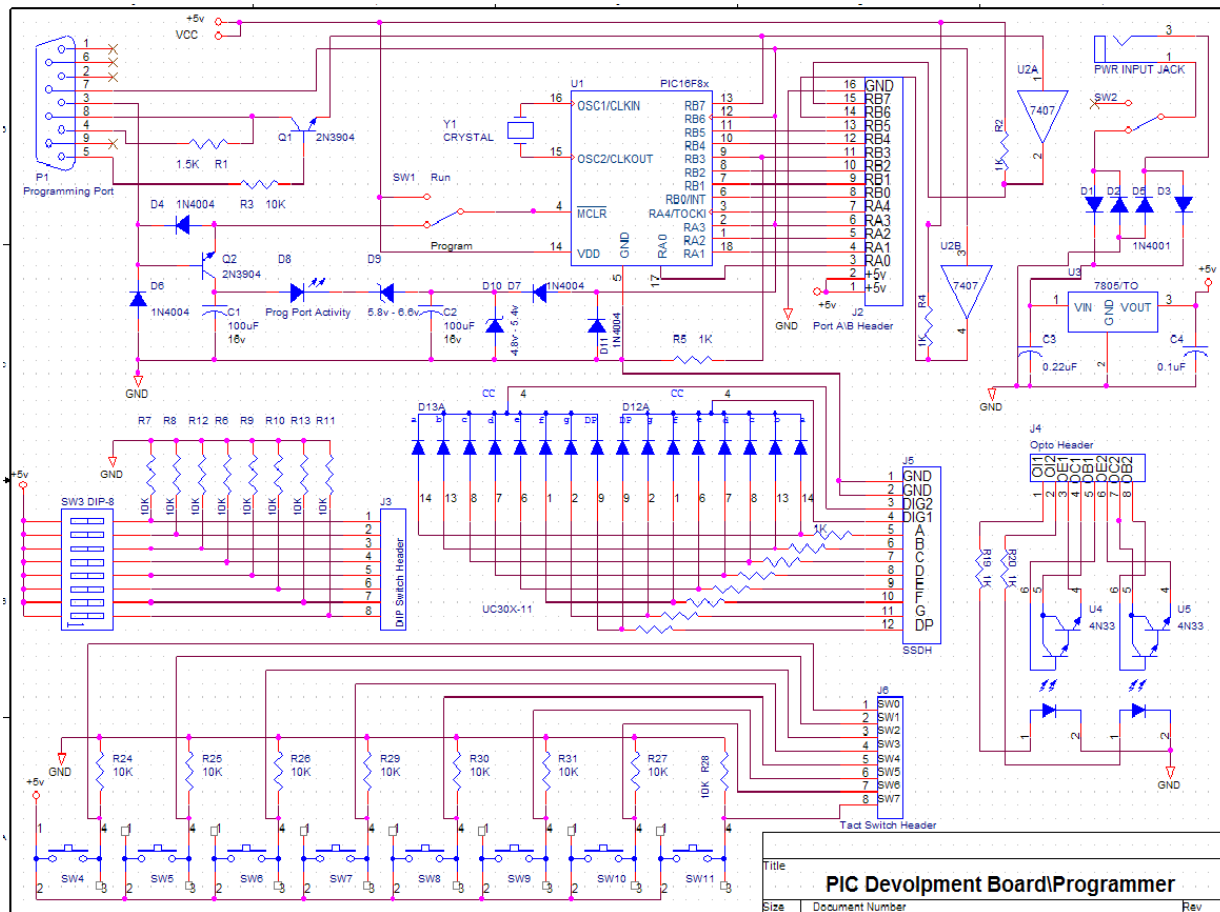
Figure 2. PIC 16F88 Design and Development Platform

The Embedded System Hardware – Software Development Platform:

The MPLAB IDE v7.61 by Microchip is the core development platform for the software. MPLAB is a freely down loadable from Microchip's website. The MPLAB IDE provides an integrated development platform in which we can do software development, which consist of an editor with all its functionality. The C compiler (in our case CCS v-4 C compiler) is invoked from within the MPLAB. After the compilation the MPLAB also provides a simulation mode that allows the simulation and testability of the code that allow us to monitor data, variables and all the special Purpose registers of the subject microcontroller. After satisfying with the simulation the next step is to down load the source code file which is the Standard Intel Hex Format. The PIC 16F88 Design and Development Platform (Figure

2) along with the respective interface circuitry which forms our respective Embedded System is connected to PC. The designed software file in the form of Standard Intel Hex Format is then down loaded using the WinPic, open source programmer which is again free for download. Thus the whole development platform described here with the exception of the compiler is open architecture and utilizes open or free software.

VII. Inter-Processor and Processor-PC Communication

There are a number of Microcontrollers that are USB 2.0 compliant and they could serve as a bridge between Processors and PC for USB communication. Additionally there are a number of commercially available interfaces that provides from PC via USB a number of Parallel I/O Ports which could be interfaced with the Microcontroller Parallel I/O Ports (one such device is: Measurement Computing's USB-based, 24-bit digital I/O module, USB1024-LS[5]. Yet another method is to establish RS-32 Serial communication established between PC and UART of the microcontroller. 16F88 also provide I²C (Inter-Integrated Circuit) bus for a multi-master serial computer type applications bus.

VIII. A Typical Lab Example of Multiprocessing System

System's Description:

Design a system that consists of a PC and Development board with 16F88. The interface between PC and 16F88 parallel ports is achieved via PC's USB port with Measurement Computing's USB 1024-LS. The process assigned to 16F88 is to count from 0 to 255. The counting increments every second and the binary value is outputted on Port B of 16F88. The PC will capture a value when prompted by the user. It will then output that value not only to the monitor but also will record it to a data file. The system will be display the value along with a time stamp of the occurrence

Hardware Layout of the System:

The system consists of, 1) A PC, 2) 16F88 System Board and 3) Measurement Computing USB interface USB 1024-LS. The skeletal layout of the system is depicted in Figure 3.
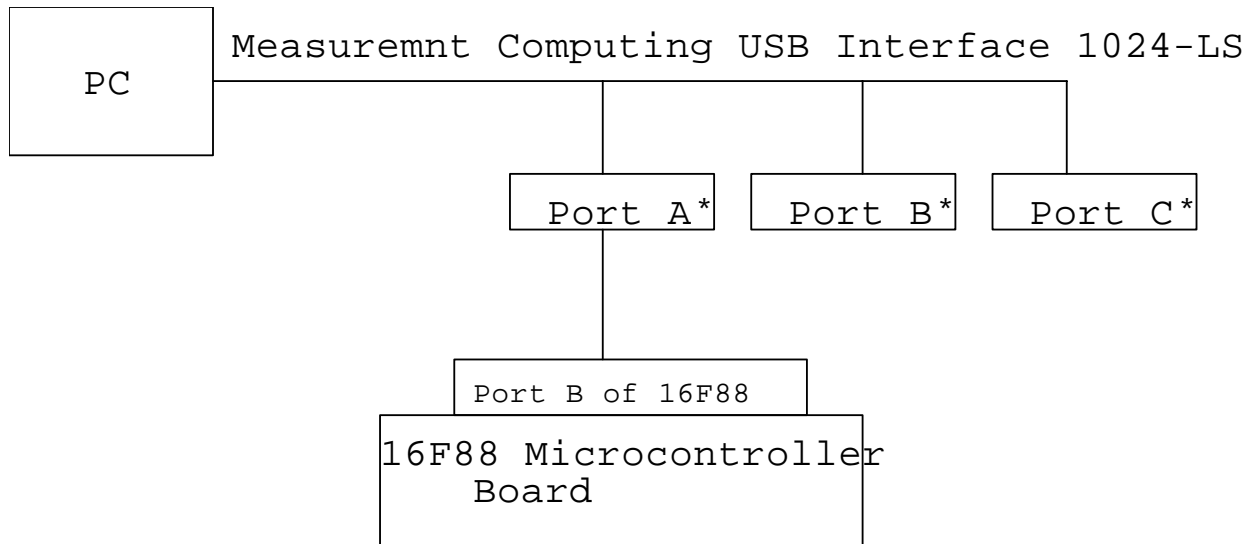
```
+-----------+   Measuremnt Computing USB Interface 1024-LS
|           |_____
|    PC     |                    |           |             |
|           |                    |           |             |
+-----------+              +-----------+ +----------+ +----------+
                           | Port A*   | | Port B*  | | Port C*  |
                           +-----------+ +----------+ +----------+
                                 |
                                 |
                           +---------------------+
                           | Port B of 16F88     |
                           +---------------------+
                           | 16F88 Microcontroller|
                           |       Board          |
                           |                      |
                           +----------------------+
```

Figure 3. A Continuous Counting System

IX. Software Design for the 16F88:

#include<16f88.h>
#use delay (clock=8000000)
#fuses NOLVP, NOWDT, INTRC, PUT

void main(void)
{
setup_oscillator(OSC_8MHZ);
setup_adc_ports(NO_ANALOGS);
set_tris_a(0x01);
set_tris_b(0x00);

while (1)
{
        int number;
        number++;
        output_b(number);
        delay_ms(1000);

```
        if(number==255)
        {
        number=0;
        }
}
} }
```

X. Software Design for the PC:

```cpp
#include<iostream.h>
#include<fstream.h>
#include<windows.h>
#include<stdio.h>
#include<conio.h>
#include<time.h>
#include"C:\Program Files\Measurement Computing\DAQ\C\cbw.h"

using namespace std;

unsigned short DataValue;

int t2=0, capture=0;

void output(void);

ofstream a_file("c:\\capture.dat");

void main(void)
{
int BoardNum=0,
PortNum=0,
Direction=1;

PortNum=FIRSTPORTA;
```

```cpp
Direction=DIGITALIN;
cbDConfigPort (BoardNum, PortNum, Direction);

cout<<"Welcome to the 16F88 Time Continium Grab Master!"<<endl;

while(1)
{

cout<<"Enter a 1 to capture a value: ";
cin>>capture;

if(capture == 1)
    {
    cbDIn(BoardNum, PortNum, &DataValue);
    output();
    }

}
}

void output(void)
{
  t2++;
  time_t rawtime;
  struct tm *timeinfo;
  time(&rawtime);
  timeinfo = localtime(&rawtime);

  cout<<"Captured Number "<<t2<<": "<<DataValue<<endl;
  a_file<<"Captured Number "<<t2<<": "<<DataValue<<" "<<"Time Captured: ";
  a_file<<asctime (timeinfo)<<endl;
  capture=0;
}
```

XI. Pedagogy of the Course

The pedagogy of the course is based on Outcome Based Education[6], and utilizes the interactive model of learning. All the students maintain an online portfolio of their work. The system designed in the laboratory to perform a specific task is the core measurement as the learning outcome of the course. The laboratory performance of the course is performed in teams of three students. This mode provides a platform for horizontal learning through active and engaged discourse and discussion. Students are empowered to charter their learning and feed their curiosity. The course culminates in a Final Project which is assessed based upon its comprehensiveness and originality. Students are required to master the soft skills of comprehensive report writing on a weekly basis and of Technical Project Report writing and project oral presentation based upon the Team's Final Project. These classroom practices and laboratory environment provides a challenging and invigorating environment that prepares them for a lifelong learning process and career path.

XII. Conclusion

The paper has provided to the reader the philosophical framework and turnkey path way for the subject of Multiprocessing System Design. The subject matter is to be pursued in parallel at dual tiers of hardware and software system design details. The subject matter Multiprocessing System Design provides to the student in class room many of the same realistic challenges faced by a System Design practitioner. The authors sincerely hope that many academicians by offering such a course in their respective curriculums will provide to students a frame work to integrate hardware and software and learn the challenges faced when multiple processors are to be integrated into a whole system.

Bibliography

[1]   Omer Farook, Chandra R. Sekhar, Jai P. Agrawal, Essaid Bouktache, Ashfaq
           Ahmed, and Mohamed Zainulabedin, "Electrical And Computer Engineering
           Technology Curriculum, From The System Design's Perspective",
           *Proceedings of the 2004 American Society for Engineering Education Annual
           Conference & Exposition,* June 20-23, 2004 Salt Lake City Utah.

[2]   Microchip Technology Inc.
           2355 West Chandler Blvd.
           Chandler, Arizona, USA 85224-6199
           (480) 792-7200

[3]   "Distributed Systems and Algorithmic Approach" by Sukumar Ghosh.
           Chapman and Hall ISBN- 1-58488-464-5.

[4]   http://en.wikipedia.org/wiki/Comparison_of_Windows_and_Linux

[5]   http://www.measurementcomputing.com

[6]   Omer Farook, Jai P. Agrawal, Chandra R. Sekhar, Essaid Bouktache, Ashfaq
           Ahmed and Mohammad Zahraee "Outcome Based Education And
           Assessment", Proceedings of the 2006 American Society for Engineering
           Education Annual Conference & Exposition June 20 -23, 2006. Chicago, IL.

Biography

CHANDRA R. SEKHAR

Professor Chandra R. Sekhar is a member of the faculty of the Electrical and Computer
Engineering Technology at Purdue University Calumet. Professor Sekhar's primary teaching
and research focus is in the areas of Biomedical and Process Control Instrumentation and
Clinical Engineering.

OMER FAROOK

Professor Omer Farook is a member of the faculty of the Electrical and Computer Engineering Technology Department at Purdue University Calumet. Professor Farook's current interests are in the areas of Embedded System Design, Hardware – Software Interfacing, Digital Communication, Networking, Image processing and Biometrics.

ESSAID BOUKTACHE

Professor Essaid Bouktache is the Department Head of the Engineering Technology Department at Purdue University Calumet. His research and teaching interests include Digital Signal Processing, Computer Networks, and Digital Communications. Professor Bouktache has been with Purdue since 1992 and is a member of IEEE and ASEE.