EECS 373 Projects

Winter 2004

Introduction

EECS 373 projects are intended to tie together all of the concepts you have learned in the labs, and to provide you with an introductory experience in designing a complete microprocessor-based system from "scratch". You will have to:

1. Choose a project.
2. Write a general specification.
3. Submit a proposal.
4. Implement the project.
5. Demonstrate the project to the lab staff.
6. Document your final design.

Schedule

Proposal Due Date: Due Tuesday in Lecture, March 23, 2004.

Project Advising: Project advising is available Monday, March 15 from 2 to 5, Wednesday, March 17 from 10:00am to 12:00 and Friday, March 19 from 1 - 4 in Matt Smiths office 3217 eecs.

Final Proposal Review: Wednesday and Thursday March 24 and 25. See signup sheet on lab door for times.

Project Lab Time: Monday, March 22 - Demonstration time.

Project Demonstrations and Due Date: Projects are due the last day of class April 21, 2004. A demonstration sheet will be posted for time slots. You may demonstrate earlier by request.

Late Penalty: Projects may be submitted late up to the scheduled exam time April 29, 2004 at a fixed 10% reduction. For example, if your project score is 80% you will receive 70% for your final project score.

Grading

50% – Successful completion. Does your project achieve what you set out to do, as described in your specification? Does your project represent at least three of the lab topic categories listed bellow.

40% – Difficulty components. Each adequate difficulty component earns 10%. Difficulty components are generally defined as aspects of your project that extend beyond applications you have already encountered in the lab. For groups larger then two, difficulty components are proportionally decreased in value. See under Project Requirements for more detail.

10% – Final documentation.

Note: A project that is successful completed with adequate documentation and two difficulty components will earn 80% or roughly a low B.

Choosing a Project

Requirements

Your project should apply the concepts and skills you have learned over the term to a problem or application different from what you have done so far in the lab. Specifically, your project must demonstrate your ability in at least three of the following categories:

1. Bus interfacing
2. Serial communications
3. Interrupts
4. Built-in MPC823 devices
5. Analog-to-digital and/or digital-to-analog conversion

To earn difficulty components, your project must demonstrate your ability to apply your knowledge in these categories to an application different from the labs. Note that while complex software-only components can make your overall project more impressive, they do not directly fulfill any of these five categories.

One example of a very basic 373 project is a digital alarm clock. Many typical alarm clock functions such as keeping and displaying the current time, setting the time, setting the alarm, snooze (alarm delay), and alarm activation can be realized with the EECS 373 I/O board and MPC823 devices you have already seen. This project would demonstrate your competence in bus interfacing (for the switch inputs and LED display), built-in MPC823 devices (the RTC), and interrupts (using the RTC interrupts to keep time). However, in its most basic form this project does not extend any of these areas beyond what you have previously developed in the labs. To meet this requirement, you could consider adding an alternate display device such as a multi-digit display allowing a full readout of hours, minutes and seconds. Alternatively, a low-cost condenser microphone and simple op-amp circuit can be used to add a ‘clap’-activated snooze feature, or combined with a DAC and a simple audio amplifier to play back a digitized audio message when the alarm goes off. More significant extensions beyond what you have already done in the lab will lead to a higher potential grade on the project, assuming you are able to complete your project successfully. For more ideas, see the project examples listed below.

Resources

While it is tempting to dream of ambitious projects, the scope of your project will be necessarily limited by the following resources:

1. **Time.** Your time is probably the most restricted resource. The scheduled project period is approximately 3 weeks. You can get a head start by researching and planning early.
2. **Group Members.** Group sizes can range from 2 to 4 people. Most project groups are either the same as a lab group or a combination of two existing lab groups. You may split a lab group to form separate project groups only if it is mutually acceptable to all current lab group members. Project group members do not need to be registered for the same lab section. The difficulty of your project needs to reflect the size of your group. For groups larger then two, difficulty components are decreased in value.
3. **Hardware**

   3.1 **MPC823FADS Board Set**

   The MPC823FADS boards have a large number of built-in peripherals, including Ethernet and USB ports and a video controller. Some of these devices have been made to work by previous project groups; if available, you may build on these prior projects. For some interfaces (particularly USB), it is straightforward to get the physical interface hardware to work, but

extremely complex to develop the software protocol. Because the focus of this course (and the project) is on hardware/software interfacing, projects consisting entirely of complex software development are discouraged.

3.2 **EECS 373 I/O Board**

In addition to the switches, LED displays, and A-to-D converter you have used in the labs, there is also a D-to-A converter on the I/O boards. See the course web page for data sheets on these devices. The D-to-A circuit on the original board design is broken, but a few of the boards have been fixed (look for soldered-on wires in the D-to-A section).

3.3 **Test Points Connector**

The 373 I/O test points can be used as I/O signals. Up to 15 signals are available for general-purpose use. An additional 17 signals are available at the test-point connector but are shared with on-board devices such as the LED displays. (See the lab board schematics on the web page.) These additional signals are available for I/O if you are not using the corresponding on-chip device.

As you know, the test points are configured by default as outputs. They can be used to control binary devices such as relays, steering controls, etc. For example, a single bit can be used to turn a relay on and off with the help of a transistor driver. Any test point can alternatively be configured as an input by changing the output buffer and OPAD pair in the Xilinx macro to an input buffer and IPAD pair. As inputs, they can be used to monitor binary sensors such as light sensors, magnetic switches, etc.

3.4 **Additional FPGA**

Some boards are equipped with an additional FPGA which interfaces to the MPC823 bus, the primary FPGA, and an additional 40-pin connector (similar to the test points). This FPGA is useful if your project requires a larger FPGA-based logic module or additional I/O connections.

3.5 **External Logic and Analog Components**

It may be necessary to use logic devices such as gates, multiplexers, registers etc. outside of the FPGA. Many of these components are available in the lab and are electrically compatible with the FPGA. Simple analog components such as transistors, capacitors, resistors, and op-amps (like those used in EECS 215) are available for signal gain, filtering or buffering.

3.6 **I/O Devices**

- **Transducers**: Speakers, microphones, light sensors, acoustic distance sensors, relays, stepper motors, thermistors, strain gauges, pressure transducers, etc.
- **Displays**: Seven-segment or alphanumeric LED displays, LCD displays
- **Others**: There are many kinds of sensors, transducers, etc. that are available. Discuss possibilities with Matt during the proposal phase or search the web.

4. **Software Tools**

4.1 MetaWare Assembler and SingleStep Debug environment.

4.2 MetaWare C Compiler

The MetaWare C Compiler is available to develop more complex PowerPC software.

4.3 Other Compilers

Other C, Java, etc. compilers available on CAEN can be used to develop applications that run under Windows on the PC. The MPC is then used as an I/O interface to a controller such as a game controller. Controller status is sent to the Windows application via the serial port.

4.4 Xilinx FPGA Synthesis

Schematic entry, ABEL or Verilog synthesis.

General Specification

The general specification is a high-level but complete description of your project. It should be one to two pages long, and include the following three sections. Under each section, we have provided an example based on the simple alarm clock project described above.

1. **Identify and describe the functions of your project**.

   Example:

   Alarm Clock Functions

   Set Time

   Set Alarm

   Ten-Minute Snooze

   Alarm Activated by Turning on Buzzer

2. **Specify I/O**

   Example:

   DIP switches: Used to enter time.

   Pushbutton 1: Used to load time.

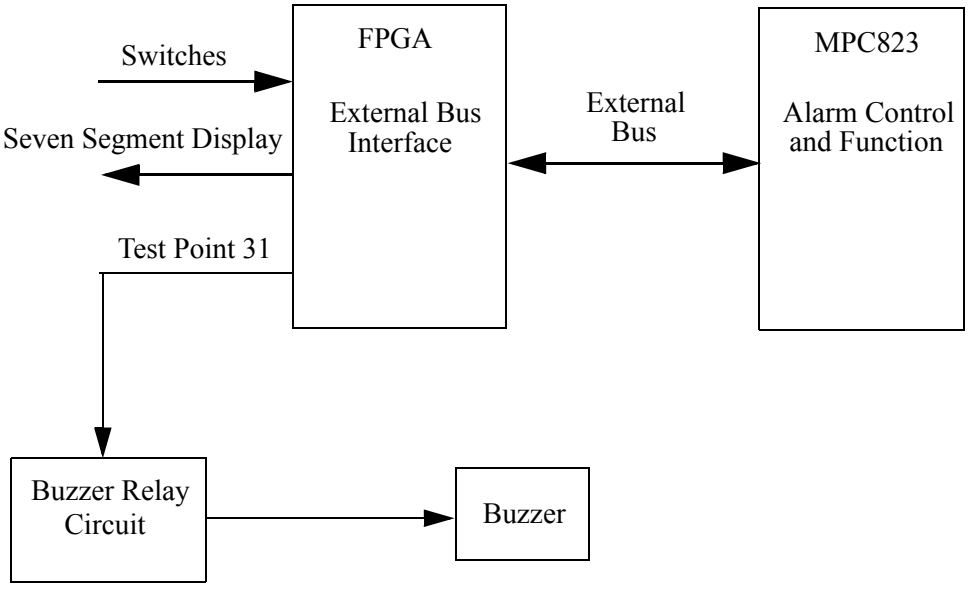
   Pushbutton 2: Used to set snooze.

   Toggle switches: Used to set modes.

   Testpoint 31: Used to switch on buzzer relay.

3. **Component or Functional Block Diagram**

   Example:



Proposal

1. **Content**

   The proposal is a snapshot of your general specification at the time of the proposal due date. It should contain the three parts of the general specification listed above. The proposal gives us a concrete, complete plan to evaluate for adequacy and feasibility.

2. **Pre-Proposal Advising**

   You are strongly urged to consult with Matt Smith and/or other course staff on your project ideas before submitting your proposal. The closer your proposal is to an adequate and feasible project, the more quickly you can get started on implementing it. Set up an appointment with Matt via email or talk to him during his lab sections or open lab.

3. **Due Date**

   As stated above.

4. **Proposal Review**

   Your proposal will be reviewed for feasibility and adequacy. Matt and/or Prof. Brehob will review the projects with each group during the week the proposals are due. A sign-up sheet will be placed in the lab for 15-minute appointments.

Implementation

1. Identify tasks within the project

   For example, consider the alarm clock project:

   Tasks

   Hardware

   1) Develop external buzzer circuit

   2) Develop switch FPGA hardware switch interface

   3) Develop display FPGA hardware interface

   Software

   1) Write interrupt handler

   2) Write I/O interfaces for switches, displays, etc.

   3) Write set time routine

   4) etc.

2. **Distribute tasks to group members based on preference and ability.**

3. **Integrate tasks and test system.**

Demonstration and Documentation

You will demonstrate your project to the lab staff with all group members present. All group members should be prepared to answer questions on the project. Project demonstrations are scheduled as stated above. A sign-up sheet for 15-minute demonstration blocks will be placed in the lab.

Your final project documentation should provide your final specification and a detailed discussion of any ways in which your implementation failed to meet that specification. Also include a brief description of your implementation and a discussion of the challenges you encountered. We may also require electronic submission of your Xilinx schematics and code (TBD).

Examples

The following examples are intended to illustrate typical 373 projects. Keep in mind that you are not limited to these type of projects. Please discuss possibilities with Matt Smith or other course staff.

Feed-Back Control Examples

Constant Motor Speed Control

A small DC fan motor's speed can be modulated with a little hardware and one of the testpoints acting as an output. A small light sensor can be attached to the motor to drive an interrupt that can be timed to measure the RPM of the motor. This combination will provide a basic feedback loop to control the motor speed. A simple algorithm can be implemented that will adjust the motors speed to small load variations.This project demonstrates:

1. Bus Interfacing

   DC modulation of motor with single bit output. Motor rotational rate displayed on 7 segment displays.

2. Serial Communications

   Motor rotation rate displayed on HyperTerminal. Motor target speed set through HyperTerminal interface.

3. Interrupts

   Light sensor output drives external interrupt.

4. MPC Devices

   Interval between light sensor interrupts timed to measure rotational rate.

5. A to D and D to A Conversion

   Not used.

Light Tracker

A light tracker can be made by fixing a panel on the shaft of servo motor with light sensors mounted on opposing ends of the panel. A light source like a flashlight can be tracked by measuring the light difference between the two sensors. The motor can rotate the panel until the light level difference between the two sensors is minimum.

Thermostat

A thermostat can be realized using a simple temperature sensor and fan. A heat source such as a load resistor can be temperature controlled by modulating a fan or regulating the current flow in the resistor. Temperature set points can be entered via HyperTerminal or the I/O board. Temperature can be displayed with HyperTerminal or the seven-segment LEDs.

Game Examples

Pong Game

Pong can be implemented using HyperTerminal as a display and an extended ASCII character control set. The position controls can be potentiometers that are read via the A/D converter channels. Sound can also be added via one of the testpoint outputs by modulating an amplified speaker.

1. Bus Interfacing

   A to D interfacing. Sound control bit. Score or game mode displayed on 7 segment display.

2. Serial Communications

   HyperTerminal connection used for game display.

3. Interrupts

   Game display driven by periodic interrupts.

4. MPC Devices

   CPM timers used to time frame display, trajectory time, sample A to D converter.

5. A to D and D to A Conversion

   Used to interrupt potentiometer position to control paddle position.

Other Game Projects

Enhanced game graphics can be provided by using Visual C or Java graphics libraries running under Windows on the PC. Most of these compilers have serial drivers that can be used to interface to the MPC823 serial port. The MPC823 can be used as a game controller to emulate a simple game controller such as some of the earlier Nintendo controllers.

Measurement Examples

Weight Scale

A weight scale can be realized with a weight transducer. Such transducers change resistance as a function of force. The resistance can be measured indirectly by measuring the voltage across it using the A/D converter. The following categories can be demonstrated:

1. Bus Interfacing

   Interface logic for A to D, LED display.

2. Serial Communications

   Use HyperTerminal for Display.

3. Interrupts

   Used to drive periodic measurement with A to D.

4. MPC Devices

   CPM timers to control measurement sample time.

5. A to D and D to A Conversion

   Used to measure weight as a change of resistance.

Temperature Measurement

Temperature sensors can be measured indirectly as a change in voltage.

Distance Measurement

Ultrasonic ranging sensors can be used to measure distance. The sensor produces a periodic pulse with a width that is proportional to distance. The period of the pulse is measured with timers to determine distance. Range is approximately 10 feet. Resolution is a few inches.

Speed Measurement

Microwave transducers can be used to measure low speeds. The sensor produces an output frequency that is proportional to speed. The frequency can be measured by measuring its period with a timer. This is how a police speed radar works.