EE459L Project

Spring 2008

Alarm Clock

Project Overview

The project this semester is a prototype for a digital clock with a seven-day alarm function that allows the user to set different alarm times for each day of the week. This would allow a student to set different times to be awakened according to their class schedule. Clocks like this are commercially available from several sources so it's not a new idea.

Other features of the clock are the ability to set alerts sounds to go off at various times to remind the student that it is time to go to class, etc. These would be different than the alarms in that they would just go off once and not continue. That way they could go off if the student isn't around and not keep going on and annoy the neighbors or roommates.

Since setting all these alarm and alert times from controls on the clock could be a chore, the clock should have a way of downloading the clock setting from some other device (computer, PDA, etc.) over a wired or wireless link.

Design Requirements

The requirements stated below for the basic project can be implemented in a variety of ways. It is up to the design team to decide how to build their project. In all projects involving microcontrollers there are ways to build the project that require more hardware and less software, and vice-versa. The design team must decide whether to implement the various parts of the design in hardware or software. Design trade-offs in areas such as reliability, manufacturability, ease of use, cost, etc., should also be analyzed to determine which is the best one to use. In addition to the baseline requirements, students are expected to think of other features that would add value to the product and if possible add some of these features to the project. Others features can be described in the final report as features that should be considered for inclusion if and when the device goes into production. Possible design options can be found by studying the features that other manufacturers have included in similar devices.

This alarm clock is intended for use by students or other people who may have differing times they wish to be awakened each day of the week. The user must have the ability to program a different alarm time (or no alarm) for each day of the week. As with any clock, the user must also be able to set the current time of the day. Since this clock has different alarms settings for each day, they also need to be able to set the current day. When an alarm time is reached, the clock should produce some sort of audible sound. The alarm continues to sound until silenced by the user.

The clock should have the ability to generate sounds at various times to alert the user to upcoming events. The alert times are separate from the alarm times and the clock should be able to store up to eight alert times for the seven-day period. When an alert time is reached, the clock makes some sort of audible sound and then stops making the sound without requiring the user to silence it.

Since this clock is intended for use by students who might have it in their apartment or dorm room near their computer, it should have some sort of interface (Bluetooth, RS-232, USB, etc.) that allows the user to download the alarm and alert times from their computer. This would make it possible for the user to enter the alarm and alert times on the computer and then have the settings sent to the clock. Presumably this would be easier than setting the times using the buttons on the clock. It is not necessary to develop a fancy program that the customer would use to enter the times. Any quick-and-dirty program that shows that the times can be downloaded is acceptable.

The project consists of several components that are all controlled by a microcontroller.

Inputs

The clock needs inputs of some sort to allow the user to program it for alarm and alert times, and to set the current time and day into the clock. The design team should select the input controls based on the type and amount of information that needs to be stored. Some options are:

- Buttons
- Dials
- Keypad
- Remote control device such as used with a TV

The design teams should examine the relative merits of the different types of input devices to determine which is the best one to use. One issue that needs be addressed is how the input devices will interface to the microcontroller. Some types of input require more I/O lines to the micro than others and the micro only has a limited number of I/O lines.

Outputs

The output display for the clock should be some sort of numerical display that can be seen in the dark. It should not be so bright as to light up the room at night but the numbers should be readable. The digits on the display should be large enough that they can be read from several feet away. The design team must decide what other information needs to be displayed (AM/PM, alarm on/off, etc.) The output also needs to provide the user with any pertinent information when the clock is being programmed for alarm and alert times. As with the inputs, the number of available microcontroller I/O lines will have affect the choice of output display. It is permissible to use a multi-digit display module instead of multiple individual digits if one can be found that is suitable.

The audio output for the clock can be a simple tone of some sort. A longer tone can be used for the alarm and shorter ones for the alerts. The design team must decide whether to use a commercially available module for producing the tones or whether to do it in hardware and software that they construct.

Memory

The clock should have the ability to retain the alarm and alert times if power is lost. The user should not have to re-enter all their alarm and alert times if they have to unplug the clock for some reason. The settings can be stored in some sort of non-volatile memory such as an EEPROM.

Timing

Most commercial alarm clocks keep track of time by counting the cycles of the 120 Volt, 60Hz AC power line. In the interest of safety, we do not want to have any 120 Volt power connected to our project boards. Projects teams should use some other method to keep track of time:

- use the microcontroller's internal timers to generate interrupts
- an oscillator (e.g. NE555 timer) on the project board
- an external low voltage AC power supply (wall-wart transformer)
- dividing a higher frequency clock down to a suitable frequency

For the purposes of this prototype, accuracy is not very important so any method that makes the clock run with less than about a 5% error is acceptable.

When testing the clock it would be very useful to have the option to make it run much faster than normal. For example, the design might include a switch that makes the clock advance one minute for every second of actual time, or one day for every minute, etc. This would make it easier to check whether the alarms and alerts are working.

Optional Features

The design team is free to add additional features to the alarm clock as they see fit. The additional features should add value to the project and hopefully will result in more people choosing to buy it. However the team must remember that each added feature will add to the product's cost so adding lots of interesting features may force the price above what many consumers would be willing to pay. Teams must remember that they are responsible for coming up with a design that they can build during the time scheduled. Teams must be careful to avoid "mission creep" where more and more features are added to the design until it becomes too bloated to be constructed. Do not allow optional features to get in the way of designing and constructing the baseline project.

Some optional feature that might be considered:

- Battery backup in case there is a power outage. When running off the battery it should keep time and still sound the alarm at the specified time. Since the display may consume a significant amount of power it is permissible to shut off the display while running on batteries in order to extend the time the clock can operate.
- Downloadable alarm and alert tones. Rather than have the clock sound a simple tone for the alarm and alerts, the user has the ability to download ringtones or other audio data to the clock from a computer.

Milestones

Each team is required to complete certain parts of the project and demonstrate that they have them working by specified dates. The purpose of this is to make sure no team gets too far along in the semester without having some very fundamental parts of the project working.

1. Power circuit. Install the connector for power and ground and confirm that you can hook your board to the lab power supply and apply power to it without shorting out the power source.
2. Timing circuit: Show that the logic board master clock is functioning and that any lower frequency clocks are being properly generated from the master clock. This can be shown using one of the oscilloscopes in the lab.
3. Microcontroller: Install the microcontroller and demonstrate that it is functioning and that you can program it. Write a small program that does something to confirm that the microcontroller is working. It is very important to get the micro working early in the project so it can then be used to test other parts of the project.
4. Display: Show that you can output something on your choice of display. Your display can be a very useful part of your debugging since once it is working you can use it to output information about what is happening with the rest of the system.
5. Inputs: Demonstrate that your input controls can be read by the microcontroller. For example, if you are using pushbuttons, show that each time the button is pressed something changes on the display
6. Clock: Show that your microcontroller produce a simple clock on the output. This demonstrates that it can determine when a second or minute has elapsed and it is time to update the clock display.

Specifications

The following are some specifications as to how the project is to be constructed.

1. The use of a commercial clock module in your design must be approved by the instructor. The goal of the class is to design and build a clock, not to buy a module that does 99% of the task and just hook up power to it.
2. Most of the project circuitry must be built on a single prototyping board. These are provided in the class and are approximately 6 by 9 inches. If teams choose to build their project on some other board they must first get permission from the instructor.
3. All signal and power connections to the logic board must be made using connectors and wiring that make for a reliable connection. For example, using clip leads to attach to a pin of an IC socket should not be used for anything more than the very first test of the board. A person trying to use your board should be able to see immediately where the power and signals are connected. They shouldn't have to be told to "use a jumper to connect the +5 volts to this pin over here."
4. The design should be based on a Freescale MC68HC908JL16 microcontroller. Projects may use multiple microcontrollers if needed. The use of a different microcontroller must be approved by the instructor.
5. The microcontroller should be installed in a zero-insertion force (ZIF) socket in order to make it easy to remove it for reprogramming. When planning where to mount the various IC sockets on the board, make sure to leave extra room for the ZIF socket since it is longer and wider than a normal socket.