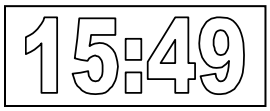
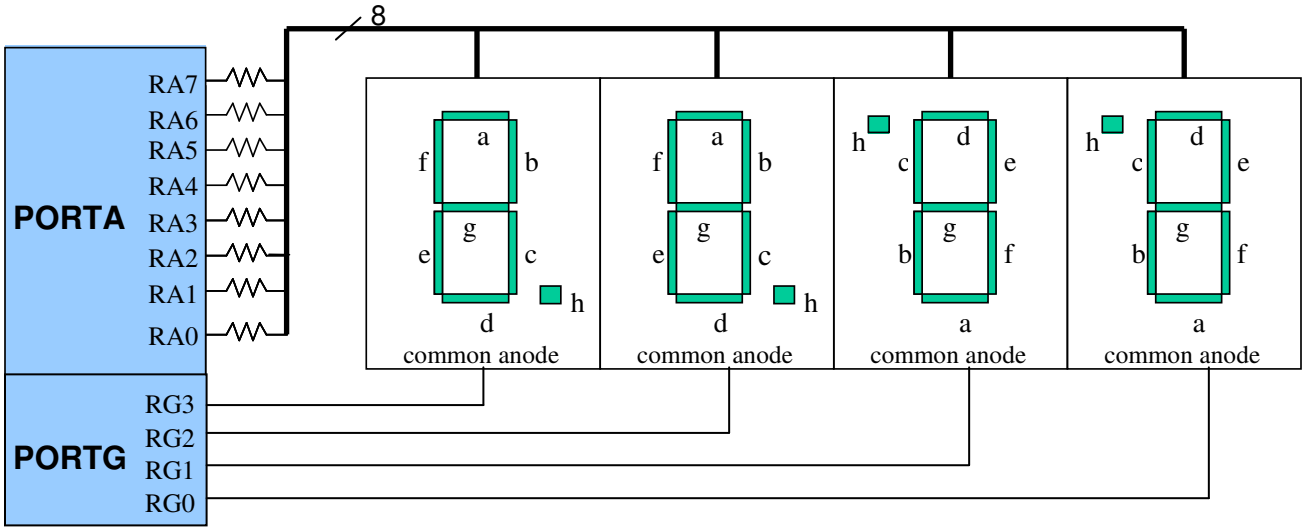
Laboratory #5
Kitchen Timer

Objective: To make use of the PIC32 parallel ports and timer module(s) to implement a kitchen timer with two operating modes (SET & RUN modes) using the keypad and the four digit 7-segment displays for user I/O functions.



Task:

The I/O expansion board attached to the Explorer 16 Development Board has a 16-key (4x4) keypad module and a four-digit 7-segment display module on it. In the previous lab assignment you have implemented a device driver for the keypad module, which you will be using (with small modification as described below) for the kitchen timer application. Refer to the lecture and Lab#4 documents for information about the hardware configuration for the keypad. The configuration for the four digit 7-segment display units is provided below:



Since data inputs for the four 7-segment displays are all connected to a common PORTA (RA7:0), a time de-multiplexing scheme needs to be employed using the display enable signals to select one display unit at a time for outputting data. The common-anode terminals of each of the four 7-segment display units (in the order most to least significant positions) are controlled by the lower nibble of PORTG (i.e. RG3, RG2, RG1, and RG0). Therefore, to display a four digit number (such as 15:49), you need to drive the corresponding display units in a sequential manner enabling each display and writing to it for a short period of time (about 1 millisecond). This process needs to be repeated periodically to generate a steady display.

The kitchen timer will have two modes of operation: the SET mode and the RUN mode.

When your program initially starts it always starts in the SET mode. While in the SET mode, your program waits for the user to enter time by using the keypad input. Each key-press on the keypad will "shift-in" a new digit on the right hand side of the four digit 7-segment display in a manner similar to a "queue" data structure. The old digits on the display give way by shifting to the left as new digits are entered from the keypad.

Once you set up the time for your kitchen timer using the inputs from the keypad you can then press the "#" key to enter the RUN mode. During the RUN mode of operation your program counts down by one second per second until the count reaches zero. The user should be able to switch back and forth between the SET mode and the RUN mode at any time, by pressing the same "#" key for mode control.

User attempts to enter inputs that are not from the decimal digits ('0' to '9') or the '#' character could be quietly ignored by your application.

As part of satisfying the functional requirements of this project you are also required to keep the following points into account:

1) You need to revise your keypad device driver (the *keyin()* function described in Lab#4) so that any software-based time delay operations are replaced by timer-based delay function (using either core timer or standard timer function) as necessary for generating more precise time delays.

2) The time display on the four 7-segment LED display units should always appear steady no matter what mode of operation we are in. For this to work correctly you need to keep the display refreshed at periodic intervals of time. This naturally calls for the use of a periodic timer interrupt. Therefore, you are required to have at least one timer interrupt to handle this functionality.

For up to 5% extra credit opportunity, implement an alarm feature for your kitchen timer. Thus, when the count reaches zero while the timer is in RUN mode then the alarm is activated, by toggling the zero display (**00:00**) on-and-off once per second. The alarm feature should be turned off when the user returns to the SET mode.

Checkout and report:

Demonstrate your working program to the lab instructor and hand in a lab report that includes the following:

- Title page: course #, course title, Lab# & title, date, professor's name, names of group members
- This lab handout.
- High-level program design, using flowcharts, for the major functional modules.
- Printout of the C source code.