```
#include <hidef.h>      /* common defines and macros */
#include <mc9s12dp512.h>     /* derivative information */
#pragma LINK_INFO DERIVATIVE "mc9s12dp512"

#include "LCD.h"
#include <stdio.h>
#include "OC.h"
#include "switches.h"

#define PROCEDURE 1 // 1 = clock; 2 = LCD test code

#if PROCEDURE == 1
void main(void) {
  char buffer[10];  // stores time for LCD printing
  unsigned short hrs, mins, secs, hrs2, mins2, secs2, error;
  OC_Init0();       // enables 1 Hz clock interrupt
  OC_Init1();       // enables 800 Hz alarm interrupt
  switchInit();     // enables switch interrupts
  LCD_Open();       // initializes LCD
  LCD_Clear();      // clears LCD screen
  asm cli           // enables interrupts
  for(;;) {
    error = LCD_ErrorCheck(); // gets LCD error code for debugging
    LCD_GoTo(0,0);  // cursors to home

    // samples globals twice to prevent critical section
    hrs = hours;
    mins = minutes;
    secs = seconds;

    hrs2 = hours;
    mins2 = minutes;
    secs2 = seconds;

    if(hrs == hrs2 && mins == mins2 && secs == secs2) {
      // if critical section avoided
      if(sprintf(buffer, "%02d:%02d:%02d", hrs,mins,secs)) {  // formats the time
        LCD_OutString(buffer);  // prints time

        // if alarm is set or alarm setting button is pressed,
        //formats the alarm time
        if((alarmSet || PTP & 0x40) &&
            sprintf(buffer, "   %02d:%02d", alarmHours, alarmMinutes)) {
          LCD_GoTo(1,0);          // goes to row 1 (second 8 characters)
          LCD_OutString(buffer);  // prints alarm time
        }
        else {
          // if alarm isn't set and button isn't pressed, clear last 8 characters
          LCD_GoTo(1,0);
          LCD_OutString("        ");
        }
      }
      if(alarmSet && hrs == alarmHours && mins == alarmMinutes) {
        alarmOn = 1;  // sounds alarm if needed
      }
    }
  }
}
#endif

#if PROCEDURE == 2

//---------------------mwait----------------------
```

```
// wait specified number of msec
// Input: number of msec to wait
// Output: none
// assumes TCNT timer is running at 16 us
void mwait(unsigned short msec){
unsigned short startTime;
  for(; msec>0; msec--){
    startTime = TCNT;
    while((TCNT-startTime) <= 63){}
  }
}

void main(void) {
  unsigned short error;
  OC_Init0();   // arms debugging interrupt (flashing PP7)
  LCD_Open();   // opens LCD
  LCD_Clear();  // clears LCD
  asm cli       // allows debugger to run
  for(;;) {
    error = LCD_ErrorCheck();   // gets error code for debugging

    // tests LCD_OutString which tests LCD_OutChar
    LCD_OutString("ABCDEFGH");
    // tests LCD_GoTo
    LCD_GoTo(1,0);
    LCD_OutString("IJKLMNOP");
    mwait(2000);  // pauses display
    LCD_Clear();
    LCD_OutString("01234567");
    LCD_GoTo(1,0);
    LCD_OutString("890,./<>");
    mwait(2000);
    LCD_Clear();
    LCD_OutString("abcdefgh");
    LCD_GoTo(1,0);
    LCD_OutString("ijklmnop");
    mwait(2000);
    LCD_Clear();
    LCD_OutString("!@#$%^&*");
    LCD_GoTo(1,0);
    LCD_OutString("()_+-=[]");
    mwait(2000);
    // tests LCD_Clear
    LCD_Clear();
  }
}
#endif
```