Ex16-LCD-Ana

1

Generated by Doxygen 1.7.5

Wed Jun 20 2012 10:36:36

Contents

Chapter 1

File Index

1.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 2

File Documentation

2.1 00readme.c File Reference

Introduction.

2.1.1 Detailed Description

Introduction. This project toggles the LEDs on the timer and displays multiple messages on the LCD.

Unlike the earlier incarnation, the LCD routines are no longer included in the project but instead are in a separate library. In this way those routines may be used by other projects by simply referencing the library and header file in the new project.

The application first sets the processor speed. In main.c, there are a number of configuration fuses set. By default, these work reasonably well on the Explorer 16, but it is preferable to be explicit about what they are doing.

The first configuration line:

```
_FOSCSEL( FNOSC_PRIPLL & IESO_OFF );
```

says to use the primary oscillator (i.e. the cystal), with the PLL system, and to start up with the user selected oscillator. An alternative is to start with a default internal RC oscillator, and then switch to the primary oscillator under program control.

The next line:

```
_FOSC( POSCMD_XT & FCKSM_CSECMD );
```

tells the dsPIC that the primary oscillator is an XT crystal. This basically affects the amount of power delivered to the crystal. EC is for very low power crystals, typically

watch crystals, XT is for "normal" crystals, and HS for high speed, typically >10MHz, crystals. It also says that it is permissible to switch clocks under program control, but should the selected oscillator fail, do not automatically switch to the fallback oscillator.

The third configuration line

```
_FWDT( FWDTEN_OFF );
```

disables the watchdog timer. If this were not done, the program would periodically reset, unless the program constantly resets the watchdog timer.

The next:

```
_FPOR( FPWRT_PWR64 );
```

holds off processor reset for 64 milliseconds after power has been applied. The idea is to give external circuitry an opportunity to stabilize before the program starts.

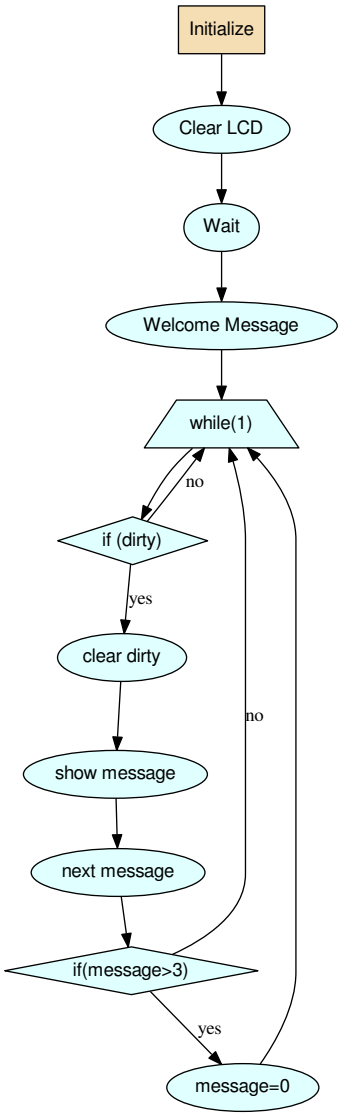
The final configuration line

```
_FICD( ICS_PGD1 & JTAGEN_OFF );
```

turns off the JTAG interface, and establishes PGD1/PGC1 as the pins for debug communication. There are three sets of programming pins on the dsPIC33FJ256GP701, so the developer may select a pair of pins that does not interfere with peripheral use for the selected circuit.

In Initialize(), two registers are set which determine how the PLL is configured. The CLKDIV register sets the pre- and post- PLL dividers which divide the clock before and after the PLL clock multiplier. PLLFBD sets the PLL feedback divisor which has the effect of multiplying the clock.

CLKDIV has a number of fields which allow the peripheral clock to be set slower than the instruction clock in some situations. These fields are not used, and are set to zero which essentially disables this feature.



Mainline flow chart

Definition in file 00readme.c.

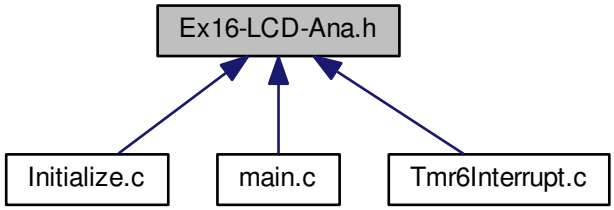
2.2 00readme.c

```
00001
```

2.3 Ex16-LCD-Ana.h File Reference

Global declarations for Ex16-LCD-Ana.

This graph shows which files directly or indirectly include this file:



Functions

- void Initialize (void)

  Initialization for Ex16-LCD-Ana.

Variables

- EXTERN int dirty

  Dirty flag - if non-zero display is updated.
- EXTERN int message

  Current message number to display.

2.3.1 Detailed Description

Global declarations for Ex16-LCD-Ana. File: Ex16-LCD-Ana.h Author: jjmcd

Created on June 19, 2012, 9:28 AM

Definition in file Ex16-LCD-Ana.h.

2.3.2 Function Documentation

2.3.2.1 void Initialize (void)

Initialization for Ex16-LCD-Ana.

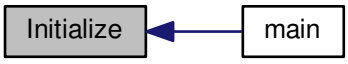
- Sets the processor clock to 40 MHz
- Initializes the ports
- Initializes timer 6
- Initializes the dirty flag and message number

Definition at line 39 of file Initialize.c.

```
{
    // Set the instruction clock speed
    //
    // Fcy 40 MIPS
    // DOZE = Fcy/8 = 011
    // DOZEN = 1
    // PLLPRE 2 = 00000
    // PLLDIV 40 = .38 = 0x26 = 0 0010 0110
    // PLLPOST 2 00
    //ROI    DOZE  DOZEN FRCDIV PLLPOST X   PLLPRE
    // 15 14 13 12  11   10 9 8    7 6    5   4 3 2 1 0
    //  0  0  0  0   0    0 0 0    0 0    0   0 0 0 0 0 0

    CLKDIV = 0x0000;
    PLLFBD = 0x0026;

    // Fcy 20 MIPS
    // PLLPRE 2 = 00000
    // PLLDIV 40 = .38 = 0x26 = 0 0010 0110
    // PLLPOST 4 01
    //ROI    DOZE  DOZEN FRCDIV PLLPOST X   PLLPRE
    // 15 14 13 12  11   10 9 8    7 6    5   4 3 2 1 0
    //  0  0  0  0   0    0 0 0    0 1    0   0 0 0 0 0 0
/*
    CLKDIV = 0x0008;
    PLLFBD = 0x0026;
*/

    TRISA = 0;              // All PORTA pins outputs
    LATA = 0x0001;          // Right LED on

    // Set timer 6 for right LED
    // Explanation ...
```

```
    //   Timer 6 will increment every 128 instruction cycles
    //   Once the count reaches 50,000, the timer 6 interrupt will fire
    //   and the count will be reset
    PR6 = 50000;              // Timer 6 counter to 50,000
    TMR6 = 0;                 // Clear timer 6
    T6CON = 0x8030;           // 1:256 prescale, timer on, Clock Fcy
    IEC2bits.T6IE = 1;        // Enable Timer 6 interrupt

    // Initialize the LCD
    LCDinit();

    // Initialize global variables
    dirty = 0;                // Message dirty flag
    message = 0;              // Current message number

}
```

Here is the caller graph for this function:



2.3.3 Variable Documentation

2.3.3.1 EXTERN int dirty

Dirty flag - if non-zero display is updated.

Definition at line 19 of file Ex16-LCD-Ana.h.

2.3.3.2 EXTERN int message

Current message number to display.

Definition at line 21 of file Ex16-LCD-Ana.h.

2.4 Ex16-LCD-Ana.h

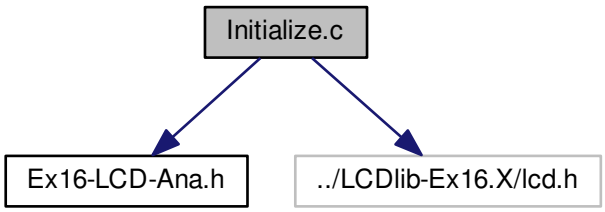
```
00001
00011 #ifndef EX16_LCD_ANA_H
00012 #define EX16_LCD_ANA_H
00013
```

```
00014 #ifdef  __cplusplus
00015 extern "C" {
00016 #endif
00017
00019 EXTERN int dirty;
00021 EXTERN int message;
00022
00024 void Initialize( void );
00025
00026
00027 #ifdef  __cplusplus
00028 }
00029 #endif
00030
00031 #endif  /* EX16_LCD_ANA_H */
00032
```

2.5 Initialize.c File Reference

Initialization for Ex16-LCD-Ana.

`#include "Ex16-LCD-Ana.h"` `#include "../LCDlib-Ex16.X/lcd.-h"` Include dependency graph for Initialize.c:



Defines

- #define **EXTERN** extern

Functions

- void Initialize (void)

  Initialization for Ex16-LCD-Ana.

2.5.1 Detailed Description

Initialization for Ex16-LCD-Ana.

Definition in file Initialize.c.

2.5.2 Function Documentation

2.5.2.1 void Initialize (void)

Initialization for Ex16-LCD-Ana.

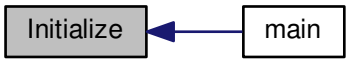
- Sets the processor clock to 40 MHz
- Initializes the ports
- Initializes timer 6
- Initializes the dirty flag and message number

Definition at line 39 of file Initialize.c.

```
{
    // Set the instruction clock speed
    //
    // Fcy 40 MIPS
    // DOZE = Fcy/8 = 011
    // DOZEN = 1
    // PLLPRE 2 = 00000
    // PLLDIV 40 = .38 = 0x26 = 0 0010 0110
    // PLLPOST 2 00
    //ROI   DOZE  DOZEN FRCDIV PLLPOST X   PLLPRE
    // 15 14 13 12  11  10 9 8   7 6   5  4 3 2 1 0
    //  0  0  0  0   0   0 0 0   0 0   0  0 0 0 0 0

    CLKDIV = 0x0000;
    PLLFBD = 0x0026;

    // Fcy 20 MIPS
    // PLLPRE 2 = 00000
    // PLLDIV 40 = .38 = 0x26 = 0 0010 0110
    // PLLPOST 4 01
    //ROI   DOZE  DOZEN FRCDIV PLLPOST X   PLLPRE
    // 15 14 13 12  11  10 9 8   7 6   5  4 3 2 1 0
    //  0  0  0  0   0   0 0 0   0 1   0  0 0 0 0 0
/*
    CLKDIV = 0x0008;
    PLLFBD = 0x0026;
*/

    TRISA = 0;              // All PORTA pins outputs
    LATA = 0x0001;          // Right LED on

    // Set timer 6 for right LED
    // Explanation ...
    //   Timer 6 will increment every 128 instruction cycles
    //   Once the count reaches 50,000, the timer 6 interrupt will fire
```

```
    //   and the count will be reset
    PR6 = 50000;              // Timer 6 counter to 50,000
    TMR6 = 0;                 // Clear timer 6
    T6CON = 0x8030;           // 1:256 prescale, timer on, Clock Fcy
    IEC2bits.T6IE = 1;        // Enable Timer 6 interrupt

    // Initialize the LCD
    LCDinit();

    // Initialize global variables
    dirty = 0;                // Message dirty flag
    message = 0;              // Current message number

}
```

Here is the caller graph for this function:



2.6 Initialize.c

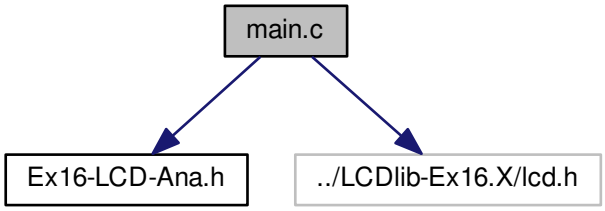
```
00001
00007 #if defined(__PIC24E__)
00008 #include <p24Exxxx.h>
00009
00010 #elif defined (__PIC24F__)
00011 #include <p24Fxxxx.h>
00012
00013 #elif defined(__PIC24H__)
00014 #include <p24Hxxxx.h>
00015
00016 #elif defined(__dsPIC30F__)
00017 #include <p30Fxxxx.h>
00018
00019 #elif defined (__dsPIC33E__)
00020 #include <p33Exxxx.h>
00021
00022 #elif defined(__dsPIC33F__)
00023 #include <p33Fxxxx.h>
00024
00025 #endif
00026
00027 #define EXTERN extern
00028 #include "Ex16-LCD-Ana.h"
00029
00030 #include "../LCDlib-Ex16.X/lcd.h"
00031
00033
00039 void Initialize( void )
```

```
00040 {
00041     // Set the instruction clock speed
00042     //
00043     // Fcy 40 MIPS
00044     // DOZE = Fcy/8 = 011
00045     // DOZEN = 1
00046     // PLLPRE 2 = 00000
00047     // PLLDIV 40 = .38 = 0x26 = 0 0010 0110
00048     // PLLPOST 2 00
00049     //ROI    DOZE  DOZEN FRCDIV PLLPOST X   PLLPRE
00050     // 15 14 13 12  11  10 9 8   7 6   5  4 3 2 1 0
00051     //  0  0  0  0   0   0 0 0   0 0   0  0 0 0 0 0
00052
00053     CLKDIV = 0x0000;
00054     PLLFBD = 0x0026;
00055
00056     // Fcy 20 MIPS
00057     // PLLPRE 2 = 00000
00058     // PLLDIV 40 = .38 = 0x26 = 0 0010 0110
00059     // PLLPOST 4 01
00060     //ROI    DOZE  DOZEN FRCDIV PLLPOST X   PLLPRE
00061     // 15 14 13 12  11  10 9 8   7 6   5  4 3 2 1 0
00062     //  0  0  0  0   0   0 0 0   0 1   0  0 0 0 0 0
00063 /*
00064     CLKDIV = 0x0008;
00065     PLLFBD = 0x0026;
00066 */
00067
00068     TRISA = 0;              // All PORTA pins outputs
00069     LATA = 0x0001;          // Right LED on
00070
00071     // Set timer 6 for right LED
00072     // Explanation ...
00073     //   Timer 6 will increment every 128 instruction cycles
00074     //   Once the count reaches 50,000, the timer 6 interrupt will fire
00075     //   and the count will be reset
00076     PR6 = 50000;             // Timer 6 counter to 50,000
00077     TMR6 = 0;               // Clear timer 6
00078     T6CON = 0x8030;         // 1:256 prescale, timer on, Clock Fcy
00079     IEC2bits.T6IE = 1;      // Enable Timer 6 interrupt
00080
00081     // Initialize the LCD
00082     LCDinit();
00083
00084     // Initialize global variables
00085     dirty = 0;              // Message dirty flag
00086     message = 0;            // Current message number
00087
00088 }
```

2.7 main.c File Reference

Mainline for Ex16-LCD-Ana.

`#include "Ex16-LCD-Ana.h" #include "../LCDlib-Ex16.X/lcd.-h"` Include dependency graph for main.c:



Functions

- _FICD (ICS_PGD1 &JTAGEN_OFF)

  Communicate on PGC1/EMUC1 and PGD1/EMUD1, JTAG is Disabled.
- _FOSC (POSCMD_XT &FCKSM_CSECMD)

  XT Oscillator Mode, Clock switching is enabled, Fail-Safe Clock Monitor is disabled.
- _FOSCSEL (FNOSC_PRIPLL &IESO_OFF)

  Primary Oscillator (XT, HS, EC) w/ PLL, Start up with user-selected oscillator.
- _FPOR (FPWRT_PWR64)

  Power-on reset timer 64 ms.
- _FWDT (FWDTEN_OFF)

  Watchdog timer enabled/disabled by user software.
- int main (void)

  Mainline for Ex16-LCD-Ana.

Variables

- char szMessage [4][17]

  Table of messages to be displayed.

2.7.1 Detailed Description

Mainline for Ex16-LCD-Ana. This application is intended to show use of the timer and the LCD. A flag is passed from the ISR to the mainline to indicate time to update the display.

A second line of the display contains the message number, to demonstrate LCD cursor positioning.

File: main.c Author: jjmcd

Created on June 19, 2012, 9:27 AM

Definition in file main.c.

2.7.2 Function Documentation

2.7.2.1 _FICD (ICS_PGD1 & *JTAGEN_OFF*)

Communicate on PGC1/EMUC1 and PGD1/EMUD1, JTAG is Disabled.

2.7.2.2 _FOSC (POSCMD_XT & *FCKSM_CSECMD*)

XT Oscillator Mode, Clock switching is enabled, Fail-Safe Clock Monitor is disabled.

2.7.2.3 _FOSCSEL (FNOSC_PRIPLL & *IESO_OFF*)

Primary Oscillator (XT, HS, EC) w/ PLL, Start up with user-selected oscillator.

2.7.2.4 _FPOR (FPWRT_PWR64)

Power-on reset timer 64 ms.

2.7.2.5 _FWDT (FWDTEN_OFF)

Watchdog timer enabled/disabled by user software.

2.7.2.6 int main (void)

Mainline for Ex16-LCD-Ana.

Blink two LEDs and display a number of messages on the LCD

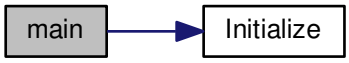
Pseudocode:

```
Initialize()
Clear the LCD display
Delay one dirty flag cycle
Display a welcome message
do forever
  if the dirty flag is set
    clear the dirty flag
    clear the display
    display the current message
    increment the message number
    display the message number
    if we are at the end of messages
      point to the first message
```

Definition at line 103 of file main.c.

```
{
    // Initialize ports and variables
    Initialize();

    // Clear the screen
    LCDclear();

    // Wait a while to pretend like we are thinking hard
    dirty = 0;
    while ( !dirty )
        ;
    dirty = 0;

    // Display a friendly welcome mesage
    LCDputs("In Principio    erat Verbum ");

    while (1)
    {
        // If the message needs to be updated
        if ( dirty )
        {
            // Remember we did it
            dirty = 0;
            // Clear the display
            LCDclear();
            // Display the current message
            LCDputs(szMessage[message]);
            // Point to the next message
            message++;
            // Position cursor to the middle of line 2
            LCDposition( 0x40+5);
            // Display the message number
            LCDletter(0x30+message );
            // If we are at the end of the messages
            if ( message > 3 )
                // point back to the firest message
                message = 0;
        }

    }
}
```

Here is the call graph for this function:



2.7.3 Variable Documentation

2.7.3.1 char szMessage[4][17]

Initial value:

```
{
    "Message One     ",
    "msg num 2       ",
    "Number three    ",
    "I am number four"
}
```

Table of messages to be displayed.

Definition at line 75 of file main.c.

2.8 main.c

```
00001
00018 /*********************************************************************************
00019  * Software License Agreement
00020  *
00021  * GPLV2+
00022  *
00023  *********************************************************************************/
00024
00025
00026 #if defined(__PIC24E__)
00027 #include <p24Exxxx.h>
00028
00029 #elif defined (__PIC24F__)
00030 #include <p24Fxxxx.h>
00031
00032 #elif defined(__PIC24H__)
00033 #include <p24Hxxxx.h>
00034
00035 #elif defined(__dsPIC30F__)
00036 #include <p30Fxxxx.h>
```

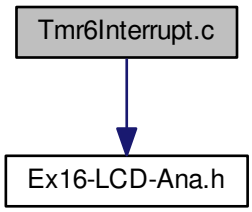
```
00037
00038 #elif defined (__dsPIC33E__)
00039 #include <p33Exxxx.h>
00040
00041 #elif defined(__dsPIC33F__)
00042 #include <p33Fxxxx.h>
00043
00044 #endif
00045
00046 /*  This is cheating
00047  *
00048  * This is sort of a trick.  Global variables must be defined once,
00049  * but anyplace they are used, they must be referenced as extern.  To
00050  * simplify keeping track, globals are declared in the header file
00051  * as EXTERN.  In the mainline, EXTERN is defined as nothing before
00052  * the header is included.  In all other files, EXTERN is declared
00053  * as extern.  This way all globals are created in the mainline but
00054  * are visible to all the other routines.
00055  */
00056 #define EXTERN
00057 #include "Ex16-LCD-Ana.h"
00058 // Notice that the LCD header file is provided by the LCD library project
00059 #include "../LCDlib-Ex16.X/lcd.h"
00060
00061 // Configuration fuses
00062 //
00064 _FOSCSEL( FNOSC_PRIPLL & IESO_OFF );
00066 _FOSC( POSCMD_XT & FCKSM_CSECMD );
00068 _FWDT( FWDTEN_OFF );
00070 _FPOR( FPWRT_PWR64 );
00072 _FICD( ICS_PGD1 & JTAGEN_OFF );
00073
00075 char szMessage[4][17] =
00076 {
00077     "Message One     ",
00078     "msg num 2       ",
00079     "Number three    ",
00080     "I am number four"
00081 };
00082
00084
00103 int main(void)
00104 {
00105     // Initialize ports and variables
00106     Initialize();
00107
00108     // Clear the screen
00109     LCDclear();
00110
00111     // Wait a while to pretend like we are thinking hard
00112     dirty = 0;
00113     while ( !dirty )
00114         ;
00115     dirty = 0;
00116
00117     // Display a friendly welcome mesage
00118     LCDputs("In Principio    erat Verbum ");
00119
00120     while (1)
00121     {
00122         // If the message needs to be updated
00123         if ( dirty )
00124         {
00125             // Remember we did it
00126             dirty = 0;
00127             // Clear the display
00128             LCDclear();
00129             // Display the current message
```

```
00130             LCDputs(szMessage[message]);
00131             // Point to the next message
00132             message++;
00133             // Position cursor to the middle of line 2
00134             LCDposition( 0x40+5);
00135             // Display the message number
00136             LCDletter(0x30+message );
00137             // If we are at the end of the messages
00138             if ( message > 3 )
00139                 // point back to the firest message
00140                 message = 0;
00141         }
00142
00143     }
00144 }
```

2.9 Tmr6Interrupt.c File Reference

Timer 6 interrupt service routine.

`#include "Ex16-LCD-Ana.h"` Include dependency graph for Tmr6Interrupt.c:



Defines

- #define **EXTERN** extern

Functions

- void __attribute__ ((__interrupt__, auto_psv))

  Timer 6 Interrupt Service Routine.

Variables

- int delayCount

  Counter used to delay toggling dirty flag.

2.9.1 Detailed Description

Timer 6 interrupt service routine. Whenever Timer 6 expires, this routine toggles the rightmost 2 LEDs. After 5 interrupts, it sets the dirty flag causing the mainline to display a new message on the LCD.

Definition in file Tmr6Interrupt.c.

2.9.2 Function Documentation

2.9.2.1 void __attribute__ ((__interrupt__, auto_psv))

Timer 6 Interrupt Service Routine.

Gets executed whenever Timer 6 expires

Pseudocode:

```
Clear timer interrupt flag
Toggle right 2 LEDs (XOR LATA with 3)
increment delayCount
if  delayCount > 5
    Set dirty flag
    Reset delay count
```

Definition at line 50 of file Tmr6Interrupt.c.

```
{
    IFS2bits.T6IF = 0;      // Clear timer interrupt flag
                            // This is always the first order of
                            // business in an interrupt routine

    LATA ^= 0x0003;         // Toggle right 2 LEDs
    delayCount++;           // Increment delayCount
    if ( delayCount > 5 )   // Only update display every 5
    {                       // toggles of LEDs
        dirty = 1;          // Set the dirty flag
        delayCount = 0;     // Reset the delayCount
    }
}
```

2.9.3 Variable Documentation

2.9.3.1 int delayCount

Counter used to delay toggling dirty flag.

Definition at line 35 of file Tmr6Interrupt.c.

2.10 Tmr6Interrupt.c

```
00001
00011 #if defined(__PIC24E__)
00012 #include <p24Exxxx.h>
00013
00014 #elif defined (__PIC24F__)
00015 #include <p24Fxxxx.h>
00016
00017 #elif defined(__PIC24H__)
00018 #include <p24Hxxxx.h>
00019
00020 #elif defined(__dsPIC30F__)
00021 #include <p30Fxxxx.h>
00022
00023 #elif defined (__dsPIC33E__)
00024 #include <p33Exxxx.h>
00025
00026 #elif defined(__dsPIC33F__)
00027 #include <p33Fxxxx.h>
00028
00029 #endif
00030
00031 #define EXTERN extern
00032 #include "Ex16-LCD-Ana.h"
00033
00035 int delayCount;
00036
00038
00050 void __attribute__((__interrupt__, auto_psv)) _T6Interrupt( void )
00051 {
00052     IFS2bits.T6IF = 0;      // Clear timer interrupt flag
00053                             // This is always the first order of
00054                             // business in an interrupt routine
00055
00056     LATA ^= 0x0003;         // Toggle right 2 LEDs
00057     delayCount++;           // Increment delayCount
00058     if ( delayCount > 5 )   // Only update display every 5
00059     {                       // toggles of LEDs
00060         dirty = 1;          // Set the dirty flag
00061         delayCount = 0;     // Reset the delayCount
00062     }
00063 }
```