Simple menu system and Watchdog

Exercise 1

Study example.cpp and the associated classes in the menu.zip file. You need a working LiquidCrystal class to test the example.

To operate the menu, you need to send four types of events to the menu system:

- Up
- Down
- Ok
- Back

Implement a menu system where the menu is operated with three push buttons (up, down, ok). Pressing a button sends an event to the menu handler. Note that you must implement filtering that prevents multiple events from a single key press. “Back” event is sent when no button is pressed in 10 seconds.

Add upper and lower limits to IntegerEdit class. The limits are given in the constructor. The class then prevents the user from setting a value that is outside of the given limits. For example, if the limits are set to 0 and 100 then user can’t decrement the value below 0 or increment above 100.

Make a menu that has three items:

1. Time (0 – 10)
2. Blank (100 – 200)
3. Light (0 – 2)

When a value is changed the program prints all three values. You can choose to print to either ITM console or the debug UART.

Exercise 2

Add third parameter to Integer edit constructor. The parameter determines increment / decrement step per up/down event.

Implement a class called “DecimalEdit” with functionality that is similar to IntegerEdit except that DecimalEdit displays the value with one decimal precision. For example, temperature could be 21.5. The constructor must take three parameters: upper limit, lower limit, and increment/decrement step.

Make a menu that has three items:

1. Time (0 – 200), step 20, Decimal
2. Blank (0 - 1), step 0.1, Decimal
3. Light (0 – 3), step 1, Integer

Time determines how long a led is ON. **Blank** determines how long a led is OFF. **Light** determines which led is controlled: 0 = none (all leds off all the time), 1 = red, 2 = green, 3 = blue.

For example:

- Time = 100, Blank = 0.5, led = 1 → red led blinks (100 ms ON, 500 ms OFF, 100 ms ON, …)
- Time = 200, Blank = 0.3, led = 3 → blue led blinks (200 ms ON, 300 ms OFF, 200 ms ON, …)
- Time = 200, Blank = 1.0, led =0 → all leds are off

Blinking must be implemented so that navigating around in the menu does not affect blinking rate.

Exercise 3

Use Exercise 1 as a starting point for this exercise.

Study periph_wwdt in lpcxpresso_nxp_lpcxpresso_1549.zip and Chapter 17 in the processor user manual.

Note that in this exercise you don't need a watchdog interrupt and you don't need to configure deep sleep and wakeup. You just need reset.

Estimate worst case delay of one round in your menu handling software main loop. Set the watchdog timer to reset the processor when the watchdog times out. Add watchdog feed at the end of your main loop. Set the timeout time to 10 x mainloop worst case delay.

Implement a class called SleepEdit that sleeps for 10 seconds after the menu value is saved. This allows you to stop the main loop and make WDT reset the CPU.

Implement a class called KillerEdit that makes an invalid watchdog feed sequence when the menu value is saved. This allows you to perform an immediate reset with software.

When the program starts it must check the watchdog status flag and set the green led on if the reset was not caused by watchdog and the red led if reset was caused by the watchdog.

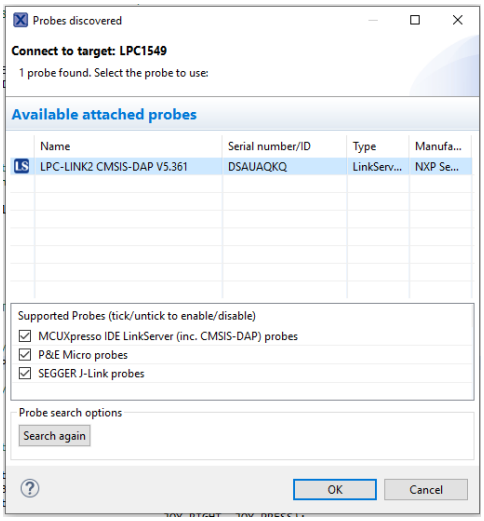
Add an instance of both classes to your code to test watchdog. Test that you can generate both types of watchdog resets.

If you can't communicate with your board see next page for recovery instructions.

If you can't communicate with your board do the following:

1. Press and hold SW2 and SW3 at the same time
2. Press reset while holding SW2 and SW3
3. Release reset while holding SW2 and SW3
4. Release SW2 and SW3
5. Start GUI Flash tool

6. Select your probe and press OK



7. Select erase tab and run mass erase