Introduction

In this hands-on lab, we will see an example of how the low power capabilities of MQX and Kinetis MCUs can be used in an application.

Resources

PC with the following software:
- IAR IDE Version 7.20
- MQX Software, Release 4.1.0 for FRDMK22F120M
- lowpower_app3 – software provided with course materials

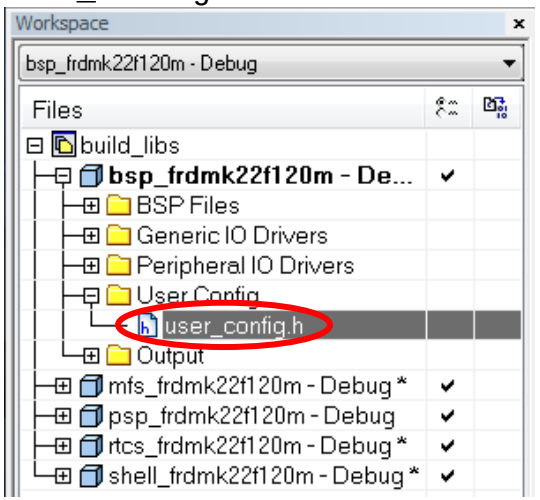
Hardware:
- FRDM-K22F
- USB Type A-micro B Cable

Lab 1: Low Power Demo

1.1 In this lab, we will take a look at an example that was created specifically for this class. It is based heavily on the lowpower example that is available with MQX 4.1 for the FRDMK22F120M. In this example, we will change the mapping between mqx low power manager power states and the power modes of the MCU, set wake up sources, and have the application determine when to transition the power mode.

1.2 In Embedded Workbench v7.20 select [File | Open | Workspace] and choose C:\Freescale\Freescale_MQX_4_1_FRDMK22F120M\build\frdmk22f120m\iar\build_libs.eww

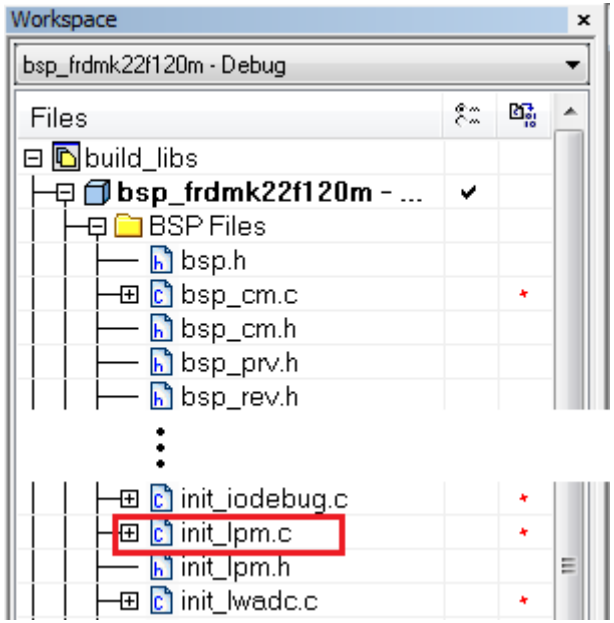
1.3 In the project "bsp_frdmk22f120m" open the folder "User Config" and open the file user_config.h



1.4 Enable the RTC (this is used to re-sync the time after periods of low power state with rtos tick disabled).
`#define BSPCFG_ENABLE_RTCDEV 1`

1.5 Also enable the low power capabilities by setting the MQX_ENABLE_LOW_POWER macro to 1
`#define MQX_ENABLE_LOW_POWER 1`

1.6 Next open `init_lpm.c` which is located in BSP_files folder.



1.7 Find the section under //LPM_OPERATION_MODE_SLEEP
Change as follows. This changes the Sleep state to be mapped to Low Leakage Stop (LLS) mode on the MCU. Also, it sets two wake up sources in the Low Leakage Wakeup Unit (LLWU) – wake up for pin 6 (PTC1) and wake up from module 0 (Low Power Timer). This was explained in the last section of the first hands-on lab in this course.

```
// LPM_OPERATION_MODE_SLEEP
{
      LPM_CPU_POWER_MODE_LLS,
      0,
      0,
      LLWU_PE2_WUPE6_MASK,
      0,
      0,
      LLWU_ME_WUME0_MASK
},
```

1.8 Next, find the section under //LPM_OPERATION_MODE_STOP
Change as follows. This changes the Stop state to be mapped to Very Low Leakage Stop 1 (VLLS1) mode on the MCU. Also, it sets two wake up sources in the Low Leakage Wakeup Unit (LLWU) – wake up for pin 6 (PTC1) and wake up from module 0 (Low Power Timer).

```
// LPM_OPERATION_MODE_STOP
{
      LPM_CPU_POWER_MODE_VLLS1,
      0,
      0,
      LLWU_PE2_WUPE6_MASK,
      0,
      0,
      LLWU_ME_WUME0_MASK
},
```

1.9 Now build the bsp project. Right click on "bsp_frdmk22f120m" and choose Make.

1.10 Now, open the example workspace we have created. Select [File | Open | Workspace].

1.11 In the pop-up dialog box, navigate to the project workspace `lowpower_lowpower_frdmk22140m.eww` in folder: `C:\Freescale\Freescale_MQX_4_1_FRDMK22F120M\mqx\examples\lowpower_dfae2014_example\build\iar\`**`lowpower_frdmk22f120m`**

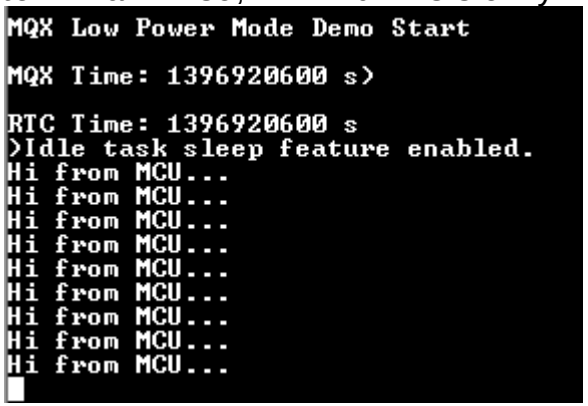
1.12 Make certain that you select build target `Int Flash Debug` so that the project is linked to run in internal flash memory.

1.13 Make the project by clicking the Make button.

1.14 Flash the image to the board using the Download and Debug button.

1.15 Open Tera Term. Click `Setup->Serial Port` and choose [`COM`*xx*] for the port (the serial port number will vary from system to system). Then, select 115200 for the baud rate, 8 bit for the data configuration, no parity, 1 stop bit, and no flow control. Observe the output from the `printf` statements.

1.16 Example Start: In the Tera Term window, you will see that the application starts up, the time is synced between the mqx time and real time clock time, and it enables the sleep while idle feature for automatic power savings. Every 1 second, it prints a friendly message to the terminal. Also, LED2 blinks slowly.



1.17 Start the Time-out: Press SW2. The LED blinks faster and starts a countdown of 10 seconds. If you press SW2 the countdown is reset back to the start. This simulates a user interacting with the application. If no interaction in 10 seconds, the countdown expires.

```
Hi from MCU...
Hi from MCU...
Timing out in 10 seconds...
Timing out in 9 seconds...
Timing out in 8 seconds...
Timing out in 7 seconds...
Timing out in 6 seconds...
Timing out in 5 seconds...
Going to Stop Soon...Blinking LED Very Fast
Timing out in 4 seconds...
Going to Stop Soon...Blinking LED Very Fast
```

1.18 Enter Low Power Mode: If 10 seconds expires with no button pushes, the application puts the system in to the Sleep state which we mapped to Low Leakage Stop (LLS). It can be woken up at any time by pressing SW2
Note – bug to be investigated – it requires you to press SW3 twice to wake up.

```
Stopping Now...

Timer wakeup set to 10 seconds.

Setting operation mode to LPM_OPERATION_MODE_SLEEP ...
Info: In this Demo SLEEP operation mode is mapped to LLS power mode.
      Core and most peripherals are inactive in this mode, reacting only t
      specified wake up events. The events can be changed in BSP (init_lpm
      Serial line is turned off in this mode. The core will wake up from
      LPT interrupt or SW3 button press.
```

1.19 Notice that when the system is woken up after a period of time in LLS mode (system ticks automatically disabled) the mqx time gets out of sync. This is easily remedied by re-syncing the time with the Real Time Clock on Kinetis MCUs.

1.20 Open the Source folder and `main.c` file. Set the DEMO_LOW_POWER2 define to a 1 (shown here).

```
#define DEMO_LOW_POWER2  1
```

This enables the demo to enter a second low power mode after a second period of no activity.

1.21 Make [F7], Flash [Ctrl + D], and run [F5] the demo with this new setting.

1.22 Low Power Stage 2: The Low Power Timer (LPT) continues to run in all low power modes of Kinetis MCUs. In this example it is used to trigger a second low power stage. If the button was not pressed after 10 additional seconds, the timer wakes up the system briefly and the application puts the system into Stop state which we mapped to Very Low Leakage Stop 1. Press SW2 any time to wake up.

```
wakeup from timer
going to stage 2 low power state

Setting operation mode to LPM_OPERATION_MODE_STOP ...
Info: In this Demo STOP operation mode is mapped to VLLS1 power mode.
      Core and peripherals are inactive in this mode, reacting only to
      specified wake up events. The events can be changed in BSP (init_lpm.c).
      Serial line is turned off in this mode. The core will wake up from
      LPT interrupt or SW3 button press.
```

1.23 Press SW2 again, the example starts over.

Code Analysis

1.24 Open the Source folder and `main.c` file. This example shows all the features that the previous lowpower example had.

1.25 The application example creates three tasks:

- **main**: the main task initializes the system and then sits in a loop, processing state changes as indicated

by demo flags that get set after timeout periods expire.

- **for_loop_led_task**: sets up the LED and toggles its state with a frequency given by current clock configuration. You will see a change of the clock configuration as a change of LED blink rate. When the LED2 is not blinking the core is in the low power stop mode and is not executing any task.
- **Timeout task:** once the timeout has started this task runs every 1 second and decrements a counter.

1.26 If the timout sequence has been started and the timeout value is 0 `if(Demo.TimeOut <= 0)`, then it goes into the low power mode.

1.27 Before entering low power mode it sets the wakeup source to be the Low Power Timer with a wakeup of 10 seconds.

```
/* Set time to wake up */
set_timer_wakeup(10);
```

1.28 Then it calls `_lpm_set_operation_mode()` to pu the device into sleep mode, which we have mapped to LLS.

```
_lpm_set_operation_mode (LPM_OPERATION_MODE_SLEEP);
```

1.29 SW3 is also set to be a wakeup source so that if SW3 is pressed, it will wake up at any time.

1.30 More!