Freescale MQX RTOS Example Guide

CLKAPI example

This document explains the CLKAPI basic example, what to expect from the example and a brief introduction to the API used.

The example

The clock elements of all the modules in Vybrid MCU are managed in a tree where the related elements are linked together. The example shows the property of the clock associated with every module available in Vybrid MCU and modification to some properties of clock elements for USB0, LPRC and USB1 modules.

Running the example

The BSPCFG_ENABLE_CLKTREE_MGMT macro must be set to non-zero in the user_config.h file prior to compilation of MQX libraries and the example itself.

To run the example the corresponding IDE, compiler, debugger and a terminal program are needed.

Explaining the example

In this example the clock element of every module available in Vybrid MCU is managed in a tree wherein each clock element is a node with a corresponding pointer to it. The clock associated with a module is derived from the clock of other module. This example makes use of the clock management driver.
There is one task in this example called CLK_API_DEMO_TASK. Task CLK_API_DEMO_TASK does the following.

- get the pointer to the clock node associated with QSPI0, USB1, LPRC module using function *clock_get()*.
- display the attribute of each clock element related to those modules using function *clock_dump()* and modify the frequency of the clock element by a call to function *clock_set_freq()*. The *clock_set_freq()* shows the dividers available for each clock element and the valid range of value for those dividers. This function is only effective when the number of dividers and the value of dividers are correct.
- display the attribute of the clock element of all the available modules in Vybrid MCU by invoking function *clock_dump_all()*. The output includes the frequency of each module and the relationship between clock elements of the modules.

The following picture shows the output for clock element of QSPI0 module as an example.

```
-------- QSPI Freq Change Experiment --------
-------------------------------
CLK <QSPI0> : <CLOSE> [NORMAL] 480MHz
    - PARENT              : <PLL3>
    - EN_CNT/ACTIVE_CHILD : 0/0
    - CHILDS :
        NA
    - PATH :
        (O)PLL3              (O)CLK_24M_IRC       (O)FIRC
    - ALTERNATIVE PARENT :
        PLL3 / PLL3_PFD4 / PLL2_PFD4 / PLL1_PFD4 / PLATFORM_BUS
-------------------------------
Freq Configuration Register for [QSPI0] are:
    - 1. CCM_CSCDR3.QSPI0_DIV (0~1) (Current 0)
    - 2. CCM_CSCDR3.QSPI0_X2_DIV (0~1) (Current 0)
    - 3. CCM_CSCDR3.QSPI0_X4_DIV (0~3) (Current 0)
Freq Configuration Register for [QSPI0] are:
    - 1. CCM_CSCDR3.QSPI0_DIV (0~1) (Current 0)
    - 2. CCM_CSCDR3.QSPI0_X2_DIV (0~1) (Current 0)
    - 3. CCM_CSCDR3.QSPI0_X4_DIV (0~3) (Current 0)
Freq Configuration Register for [QSPI0] are:
    - 1. CCM_CSCDR3.QSPI0_DIV (0~1) (Current 0)
    - 2. CCM_CSCDR3.QSPI0_X2_DIV (0~1) (Current 0)
    - 3. CCM_CSCDR3.QSPI0_X4_DIV (0~3) (Current 0)
Invalid div value for [QSPI0], please check
-------------------------------
CLK <QSPI0> : <CLOSE> [NORMAL] 60MHz
    - PARENT              : <PLL3>
    - EN_CNT/ACTIVE_CHILD : 0/0
    - CHILDS :
        NA
    - PATH :
        (O)PLL3              (O)CLK_24M_IRC       (O)FIRC
    - ALTERNATIVE PARENT :
        PLL3 / PLL3_PFD4 / PLL2_PFD4 / PLL1_PFD4 / PLATFORM_BUS
-------------------------------
```