Freescale MQX RTOS Example Guide

KLOG example

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

The example

The example shows the usage of the kernel log component of RTOS MQX. The kernel log allows user's application program to record the information about the context switch as tasks being re-scheduled and interrupt happened in the application program. The kernel log API is used to enable log for specific component and to display log information to terminal.

Running the example

The MQX_USE_LOGS and MQX_KERNEL_LOGGING macros 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

The application example creates only one task called main_task.
The main_task creates the kernel log component as it is optional component of RTOS MQX.
The kernel log is then activated over specific components using command _klog_control(), for example

_klog_control(KLOG_ENABLED | KLOG_CONTEXT_ENABLED | LOG_INTERRUPTS_ENABLED| KLOG_SYSTEM_CLOCK_INT_ENABLED | KLOG_FUNCTIONS_ENABLED | KLOG_TIME_FUNCTIONS | KLOG_INTERRUPT_FUNCTIONS, TRUE);

The information of function switching, interrupt occurrence, task switching are displayed over the output terminal.

An example of the output is shown in the following figure.

```
Kernel log contains:
1. 0x00000000:2D6A2 -> FUN       _time_delay_ticks 0x0 0x0 0x0 0x0 0x0
2. 0x00000000:2DC4A -> NEW TASK TD 0x1FFF25FC ID 0x10003 STATE 0x2 STACK 0x1FFF2BE4
3. 0x00000000:2DF46 -> XFUN      _time_delay_ticks 0x0 0x0 0x0 0x0 0x0
4. 0x00000000:2E22C -> FUN       _time_delay_ticks 0x5 0x0 0x0 0x0 0x0
5. 0x00000000:2E634 -> NEW TASK TD 0x1FFF1AEC ID 0x10002 STATE 0x2 STACK 0x1FFF1CAC
6. 0x00000000:02B5 -> INT   0xF
7. 0x00000001:071F -> INT   0xF END
8. 0x00000001:023C -> INT   0xF
9. 0x00000002:05DF -> INT   0xF END
10. 0x00000002:021F -> INT   0xF
11. 0x00000003:05BA -> INT   0xF END
12. 0x00000003:021F -> INT   0xF
13. 0x00000004:05BA -> INT   0xF END
14. 0x00000004:021F -> INT   0xF
15. 0x00000005:06BC -> INT   0xF END
16. 0x00000005:0939 -> NEW TASK TD 0x1FFF25FC ID 0x10003 STATE 0x2 STACK 0x1FFF2BD4
17. 0x00000005:0BE7 -> XFUN      _time_delay_ticks 0x0 0x0 0x0 0x0 0x0
18. 0x00000005:0EC0 -> FUN       _time_delay_ticks 0xA 0x0 0x0 0x0 0x0
19. 0x00000005:12C2 -> NEW TASK TD 0x1FFF1AEC ID 0x10002 STATE 0x2 STACK 0x1FFF1CA4
20. 0x00000005:021F -> INT   0xF
21. 0x00000006:05B5 -> INT   0xF END
22. 0x00000006:0224 -> INT   0xF
23. 0x00000007:05C6 -> INT   0xF END
24. 0x00000007:0224 -> INT   0xF
25. 0x00000008:05C6 -> INT   0xF END
26. 0x00000008:0224 -> INT   0xF
27. 0x00000009:05C6 -> INT   0xF END
28. 0x00000009:0224 -> INT   0xF
29. 0x0000000A:05C6 -> INT   0xF END
30. 0x0000000A:0224 -> INT   0xF
31. 0x0000000B:05C6 -> INT   0xF END
32. 0x0000000B:0224 -> INT   0xF
33. 0x0000000C:05C6 -> INT   0xF END
34. 0x0000000C:0224 -> INT   0xF
35. 0x0000000D:05C6 -> INT   0xF END
36. 0x0000000D:0224 -> INT   0xF
37. 0x0000000E:05C6 -> INT   0xF END
38. 0x0000000E:0224 -> INT   0xF
39. 0x0000000F:06B1 -> INT   0xF END
40. 0x0000000F:0928 -> NEW TASK TD 0x1FFF25FC ID 0x10003 STATE 0x2 STACK 0x1FFF2BD4
41. 0x0000000F:0BD4 -> XFUN      _time_delay_ticks 0x0 0x0 0x0 0x0 0x0
42. 0x0000000F:0E93 -> FUN       _time_delay_ticks 0xF 0x0 0x0 0x0 0x0
```