

Freescalé MQX RTOS Example Guide

RTC example

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

The example

The RTC example synchronizes the RTC module time with MQX RTOS system time, sets the second alarm interrupt and handler, shows the IRTC counter up demo, the IRTC standby RAM demo.

Running the example

Before building MQX libraries it is necessary to enable BSPCFG_ENABLE_RTCDEV in the user_config.h file:

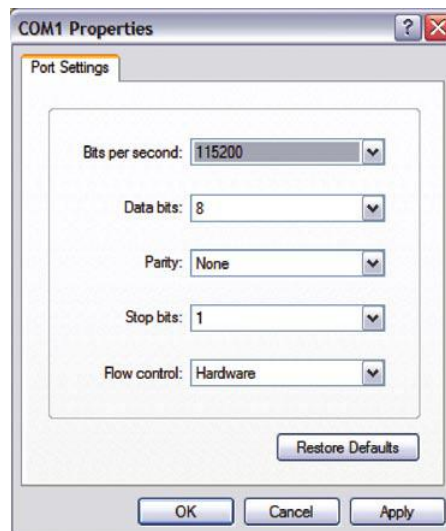
```
#define BSPCFG_ENABLE_RTCDEV 1
```

Run HyperTerminal on the PC (Start menu->Programs->Accessories->Communications).

Make a connection to the serial port which is connected to the board (usually will be COM1).



Set it with 115200 baud, no parity, 8 bits and click OK.

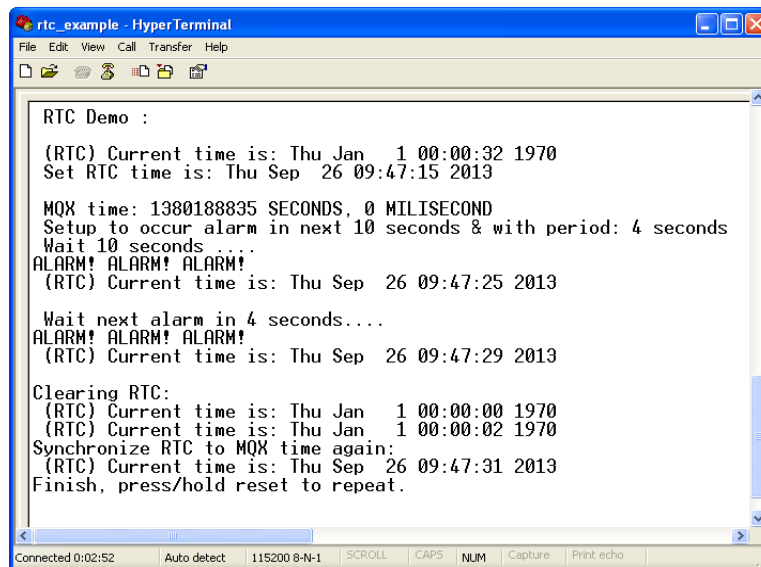


At the beginning, the serial terminal will show the current RTC time and the time default (26-09-2013 9:47:15). Time default is used to set for both MQX RTOS system and RTC.

RTC alarm interrupt is installed in next 10 seconds alarm with period 4 seconds. And then clear RTC time to 0.

The demo sets 10 seconds alarm. Therefore, the microcontroller must wait until ALARM occurs, after 10 seconds the serial terminal shows "ALARM! ALARM! ALARM! ". Then after next 4 seconds (wait for next ALARM interrupt), the serial terminal shows "ALARM! ALARM! ALARM!" again.

The serial terminal snapshot that shows the RTC example running on a Kinetis board, as below:



```
rtc_example - HyperTerminal
File Edit View Call Transfer Help
RTC Demo :
(RTC) Current time is: Thu Jan  1 00:00:32 1970
Set RTC time is: Thu Sep  26 09:47:15 2013

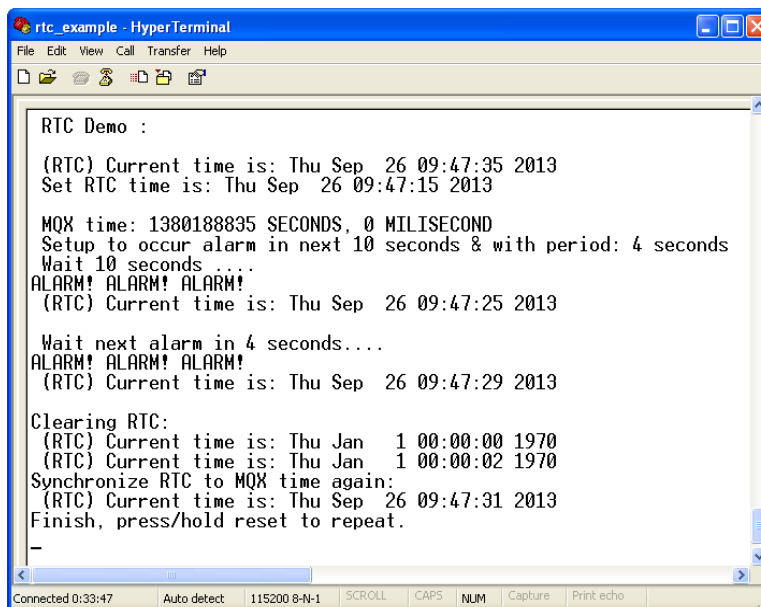
MQX time: 1380188835 SECONDS, 0 MILLISECOND
Setup to occur alarm in next 10 seconds & with period: 4 seconds
Wait 10 seconds ....
ALARM! ALARM! ALARM!
(RTC) Current time is: Thu Sep  26 09:47:25 2013

Wait next alarm in 4 seconds....
ALARM! ALARM! ALARM!
(RTC) Current time is: Thu Sep  26 09:47:29 2013

Clearing RTC:
(RTC) Current time is: Thu Jan  1 00:00:00 1970
(RTC) Current time is: Thu Jan  1 00:00:02 1970
Synchronize RTC to MQX time again:
(RTC) Current time is: Thu Sep  26 09:47:31 2013
Finish, press/hold reset to repeat.

Connected 0:02:52  Auto detect  115200 8-N-1  SCROLL  CAPS  NUM  Capture  Print echo
```

Reset board to verify that the RTC time isn't clear:



```
rtc_example - HyperTerminal
File Edit View Call Transfer Help
RTC Demo :
(RTC) Current time is: Thu Sep  26 09:47:35 2013
Set RTC time is: Thu Sep  26 09:47:15 2013

MQX time: 1380188835 SECONDS, 0 MILLISECOND
Setup to occur alarm in next 10 seconds & with period: 4 seconds
Wait 10 seconds ....
ALARM! ALARM! ALARM!
(RTC) Current time is: Thu Sep  26 09:47:25 2013

Wait next alarm in 4 seconds....
ALARM! ALARM! ALARM!
(RTC) Current time is: Thu Sep  26 09:47:29 2013

Clearing RTC:
(RTC) Current time is: Thu Jan  1 00:00:00 1970
(RTC) Current time is: Thu Jan  1 00:00:02 1970
Synchronize RTC to MQX time again:
(RTC) Current time is: Thu Sep  26 09:47:31 2013
Finish, press/hold reset to repeat.
-

Connected 0:33:47  Auto detect  115200 8-N-1  SCROLL  CAPS  NUM  Capture  Print echo
```

Explaining the example

The application example creates only one main task. The flow of the task is described in next figure.

The main task starts with the RTC clock synchronize time with MQX system time. MQX system time format is absolute second/millisecond time, RTC time format is UTC time (the number of seconds from 1/1/1970); then it calls these following functions to setup time for RTC and MQX system.

```
_time_set (&mqx_time);    /* Set time for MQX system */
_rtc_set_time (rtc_time); /* Set time for RTC */
```

RTC demo is using one source interrupt for alarm interrupt. To setup alarm interrupt, it calls function `rtc_callback` as following:

```
_rtc_callback_reg((INT_ISR_FPTR)rtc_callback, (void*)NULL);
(rtc_callback is a pointer to alarm call back function)
```

To unregister alarm callback and disable RTC alarm, please call the function as following:

```
_rtc_callback_reg(NULL, (void*)NULL);
(alarm call back function is NULL)
```

Alarm interrupt demo sets up to alarm occur in next 10 seconds with period 4 seconds as these following lines:

```
#define ALARM_NEXT_TIME      (10)
#define ALARM_PERIOD        (4)
```

```
rtc_time += (uint32_t)ALARM_NEXT_TIME;;
_rtc_set_alarm(rtc_time, (uint32_t)ALARM_PERIOD);
```

After the alarm interrupt happens, Clear RTC time and then synchronizes to MQX system time again.

