EE458 - Embedded Systems
Timers and Clocks

- Outline
  - Timers
  - Clocks

- References
  - RTC: Chapters 11
  - CUG: Chapters 7, 8
Timers and Clocks

Timers

- A timer allows the scheduling of an event according to a predefined time value in the future, similar to setting an alarm clock.

- Hard timers are derived from physical timer chips that directly interrupt the processor. Soft timers are software events that are scheduled through software. Operations with demanding precision requirements should use ______ ______.
Timers and Clocks

Timers

- Soft timers allow for efficient scheduling of non-high-precision software events. Applications requiring timeouts with coarse granularity may use soft timers. (Many timeouts in network protocols are an example where course granularity is sufficient.)

- Soft timers can also reduce the system-interrupt ________ incurred by multiple hard timers.
Timers and Clocks
Real-Time and System Clocks

- A real-time clock is a part of many embedded systems. It uses powered DRAM to keep track of the date and time even when system power is off.

- The job of the system clock is identical to that of the real-time clock. It keeps track of the time and date while the system is running. It is usually initialized from the real-time clock at system start up.
A programmable interval timer (PIT) or timer chip functions as an event counter, elapsed time indicator, or periodic event generator.

The PIT is usually initialized at system start up: the timer interrupt rate is set, the timer operation mode is set, the timer ISR is installed, and the timer interrupt is enabled.

Each timer interrupt is known as a _____. The RTOS performs task scheduling at each ____. 
Timers and Clocks
Programmable Interval Timers

- The timer ISR typically performs these duties:
  - The system clock is updated, both absolute time (wall-clock time and date) and elapsed time are updated.
  - A kernel routine is called to indicate the passing of a tick.
  - The timer chip is reinitialized (if necessary).
Timers and Clocks
RTEMS Clock Directives

- The system clock is set by a call to the `rtems_clock_set()` directive:
  
  ```c
  rtems_status_code rtems_clock_set
  (rtems_time_of_day *time_buffer);
  ```

- The `time_buffer` is a ____________ with year, month, day, hour, minute, second, and ticks fields. (See next slide and CUG Chapter 7).

- To use the real-time clock define the macro: CONFIGURE_APPLICATION_NEEDS_RTC_DRIVER
struct rtems_tod_control {
    uint32_t year;   /* greater than 1987 */
    uint32_t month;  /* 1 - 12 */
    uint32_t day;    /* 1 - 31 */
    uint32_t hour;   /* 0 - 23 */
    uint32_t minute; /* 0 - 59 */
    uint32_t second; /* 0 - 59 */
    uint32_t ticks;  /* elapsed between seconds */
};

typedef struct rtems_tod_control rtems_time_of_day;
Timers and Clocks
RTEMS Clock Directives

- The system date and time can be obtained using the `rtems_clock_get()` directive:

  ```c
  rtems_status_code
  rtems_clock_get(
    rtems_clock_get_options option,
    void *time_buffer);
  ```

- The data type of the `time_buffer` structure depends on the value of the __________ parameter. (See next slide & CUG Chap 7).
Timers and Clocks
RTEMS Clock Directives

- The data type expected for `time_buffer` is based on the value of ____________ as indicated below:
  - RTEMS_CLOCK_GET_TOD:
    (rtems_time_of_day *)
  - RTEMS_CLOCK_GET_SECONDS_SINCE_EPOCH:
    (rtems_interval *)
  - RTEMS_CLOCK_GET_TICKS_SINCE_BOOT:
    (rtems_interval *)
  - RTEMS_CLOCK_GET_TICKS_PER_SECOND:
    (rtems_interval *)
  - RTEMS_CLOCK_GET_TIME_VALUE:
    (struct timeval *)
Timers and Clocks
RTEMS Clock Directives

- The `timeval` struct is a POSIX defined data type with members:

  ```c
  struct timeval {
    time_t     tv_sec;     /* seconds */
    suseconds_t  tv_usec;    /* microseconds */
  }
  ```

  where the values are the times since the ___________ (Jan 1, 1970). Refer to the `time`, `localtime` and `gettimeofday/settimeofday` POSIX documentation.
Timers and Clocks

RTEMS Timers

- RTEMS timers are objects and must be created like other kernel objects (tasks, semaphores, etc.) You must also configure the CONFIGURE_MAXIMUM_TIMERS parameter correctly.

- RTEMS provides two types: task-based (TB) and non-task-based (NTB) timers. They are created and deleted using the same directives. The directives used to fire the timers are different.
Timers and Clocks

RTEMS Timers

• NTB timers run as part of the timer interrupt service routine and should not use any directive that can not be used by an ISR.

• Task-based timer routines are more flexible than NTB routines. The Timer Server can be configured to support a floating point context.

• Task-based timer service routines are executed by the ______ ________ task. The timer server task must be accounted for when configuring the system.
Timers and Clocks
Soft (Task-Based) Timers

- The Timer Server task runs at a higher priority than any application task.
- The Timer Server must be initiated (started) before any task-based timer directives are used. The `rtems_timer_initiate_server()` directive is used for this. The directive takes stack size and _________ arguments. These arguments have the same meaning as they do for the `rtems_task_create()` directive.
Timers and Clocks
Firing a Timer

- There are four directives that can be used to schedule a timer routine to be run:
  - `rtems_timer_fire_after()`: NTB timer that fires after a specified number of ticks.
  - `rtems_timer_fire_when()`: NTB timer that fires at a specified wall clock time.
  - `rtems_timer_server_fire_after()`: task-based timer that fires after a specified number of ticks.
  - `rtems_timer_server_fire_when()`: task-based timer that fires at a specified wall clock time.
Timers and Clocks
Other Timer Directives

- There are, of course, timer directives to create and delete timers. Deleting an unexpired timer cancels the timer.
- The cancel directive can be used to halt a timer. It will not fire unless re-initiated by a reset or fire directive.
- The ______ directive can restart an unexpired timer initiated by a “fire after” directive.
- Refer to CUG Chapter 8 for timer details.