Exercise RTOS-10P. FreeRTOS –timers

REAL TIME OPERATING SYSTEMS
LABORATORIUM SYSTEMÓW STEROWNIA PRZEMYSŁOWEGO I AUTOMATYKI BUDYNKÓW

KATEDRA ENERGOELEKTRONIKI I AUTOMATYKI SYSTEMÓW PRZETWARZANIA ENERGII AKADEMIA GÓRNICZO-HUTNICZA
WWW.KANIUP.AGH.EDU.PL WWW.AGH.EDU.PL

Subject:	**FreeRTOS real time operating system – time manipulating, timers**
Tools:	**Visual Studio Express, FreeRTOS sources**
Required skills:	**Basic knowledge of issues related to programming in C language and real-time systems**

Introduction.

Timer - a system mechanism that realizes precise timekeeping that can be used to:

- measuring time intervals with high accuracy and repeatability
- unblocking ("waking") the process at a specific time
- cyclical stimulation of the process to work
- securing locking operations against excessive blocking

the timer can result in the generation of:

- events
- signals
- deposites
- callbacks

Timer features functions:

```
TimerHandle_t xTimerCreate
              ( const char * const pcTimerName,
                const TickType_t xTimerPeriod,
                const UBaseType_t uxAutoReload,
                void * const pvTimerID,
                TimerCallbackFunctićon_t pxCallbackFunction );
```

pcTimerName	– timer name (string)
xTimerPeriod	– timer period
uxAutoReload	– auto-reload timer (when True), or one-shot (when False)
pvTimerId	– timer identifier (most often used in the callback function to identify the timer)
pcCallbakFunction	– function called after the timer has expired

Exercise RTOS-10P. FreeRTOS –timers

Purpose of the exercise.

The purpose of the exercise is to learn about the possibilities of measuring time in the real-time operating system.

Exercise program.

1. Use project 013-Timers_1 and test the operation in each of the following cases:
   1.1. Run two tasks (Callbacks) triggered from two timers counting with a) the same time, b) different times.
   1.2. Run two time driven tasks (Callbacks). One with auto-reload timer, the other with one-shoot timer.

2. Use project 014-Timers_2 and test the operation in each of the following cases:
   2.1. Run three timer driven tasks (Callback) with different period times. In one callback calculate the time between successive calls of two other callbacks function.