Exercise RTOS-08P. FreeRTOS real time operating system – communication with queues

REAL TIME OPERATING SYSTEMS IN CONTROL APPLICATIOS
LABORATORIUM SYSTEMÓW STEROWNIA PRZEMYSŁOWEGO I AUTOMATYKI BUDYNKÓW

KATEDRA ENERGOELEKTRONIKI I AUTOMATYKI SYSTEMÓW PRZETWARZANIA ENERGII
WWW.KANIUP.AGH.EDU.PL

AKADEMIA GÓRNICZO-HUTNICZA
WWW.AGH.EDU.PL

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

Introduction.

To use the queues, attach the queue.h header file to the project. Creating a queue can be done using the *xQueueCreate* function, which returns the value of the handle (identifier) to the queue in the form of a *QueueHandle_t* variable that must be declared as global. The arguments of the function are:

- queue size (in items)
- size of a single element.

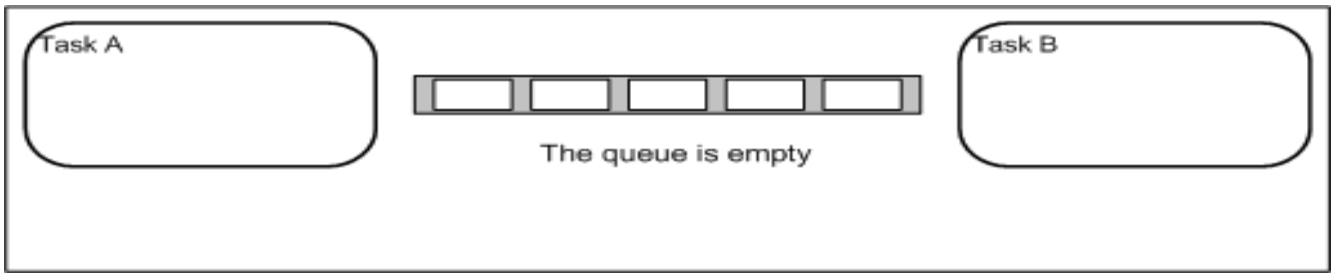
Sending (adding) items to the queue can be done using the *xQueueSend* function, whose arguments are:

- handle (identifier) to the queue that was returned by the *xQueueCreate* function
- pointer to the buffer (variable) with data to be sent (added) to the queue,
- the number of clock ticks for which the function call will lock the thread while waiting for the queue (in case the queue is full).

Receiving (removing) items from the queue is done using the *xQueueReceive* function, whose arguments are:

- handle (identifier) to the queue that was returned by the *xQueueCreate* function
- buffer (variable) pointer to which data retrieved from the queue are to be saved,
- the number of clock ticks for which the function call will lock the thread while waiting for the item to appear in the queue (in case the queue is empty).

Queue between two *tasks*:



Exercise RTOS-08P. FreeRTOS real time operating system – communication with queues

Purpose of the exercise.

The purpose of the exercise is to become familiar with the mechanisms of communication between tasks using queues.

Exercise program.

1. Use the FreeRTOS_lab_1 project from the previous exercise.
2. Implement a communication queue between two tasks.
3. Test sending and receiving elements of various types (simple variables, arrays, strings) to the queue.
4. Test the queue operation if the receive is performed by a non-blocking function (xTicksToWait = 0).
5. Test the operation of the queue if the waiting time for free space in the queue is longer than the set blocking time (xTicksToWait) of the xQueueSend function