‘tasks and queues‘ -- «задачи и очереди»

task 2, with normal priority, задача 2, с обычным приоритетом,

the entry function is Task2\_init.. функция входа - Task2\_init..

functions for the tasks are defined \_- функции для задач определены

void StartDefaultTask(void const \* argument)

{

/\* USER CODE BEGIN 5 \*/

/\* Infinite loop \*/

for(;;)

{

HAL\_GPIO\_TogglePin(GPIOA, GPIO\_PIN\_0);

osDelay(1);

}

/\* USER CODE END 5 \*/

}

/\* USER CODE BEGIN Header\_Task2\_init \*/

/\*\*

\* @brief Function implementing the Task2 thread.

\* @param argument: Not used

\* @retval None

\*/

/\* USER CODE END Header\_Task2\_init \*/

void Task2\_init(void const \* argument)

{

/\* USER CODE BEGIN Task2\_init \*/

/\* Infinite loop \*/

for(;;)

{

HAL\_GPIO\_TogglePin(GPIOA, GPIO\_PIN\_1);

osDelay(1);

}

/\* USER CODE END Task2\_init \*/

}

**Creating a Task \_\_** Создание задачи

In order to create a new Task, we have to follow some set of steps, and they are as follows:-

1. Define a **ThreadID** for the task. This variable will store the unique ID of the task, once created. Later, all the operations will require this ID.

Определите ThreadID для задачи. В этой переменной будет храниться уникальный идентификатор задачи после ее создания. Позже для всех операций потребуется этот идентификатор. os**ThreadID** Task3Handle;

ooooo Task3Handle;

Task3Handle;

2.) Define the **entry function** for the task. This is the main function of the task. Your program will be written inside it. Remember that the tasks in the Free RTOS, are not designed to handle any return value. So, the entry function should always have an infinite loop, inside which, your whole program should be written.

void Task3\_init (void const \* argument)

{

while (1)

{

// do something

osDelay (1000); // 3 sec delay

}

}

3.) Inside our main function, we need to **define the task** first and than **create** it.

// define thread

osThreadDef(Task3, Task3\_init, osPriorityBelowNormal, 0, 128);

//create thread

Task3Handle = osThreadCreate(osThread (Task3), NULL);

* **osThreadDef** takes the parameters as the name of the task, the entry function, the priority, instance, and the stack size.
* After the task is defined, we can create it using **osThreadCreate**, and assign the ID to the **Task3Handle**