Exercise 5 Joinable threads

In this exercise we will create a thread which in turn spawns two joinable threads. The initial thread will then call osThreadJoin() to wait until each of the joinable threads has terminated.

In the Pack Installer select "Ex 4 Join " and copy it to your tutorial directory.

Open main.c

In main.c we create a thread called worker_Thread and define it as joinable in the thread attribute structure.

When the RTOS starts we create the led_thread() as normal.

```
__NO_RETURN void led_thread1 (void *argument) {

for (;;) {

        worker_ID1   = osThreadNew(worker_thread,(void *) LED1_ON, &ThreadAttr_worker);

        LED_On(2);

        osThreadJoin(worker_ID1);

        ……………………….
```

In this thread we create an instance of the worker thread and then call osJoin() to join it. At this point the led_thread enters a waiting state and the worker thread runs.

```
void worker_thread (void *argument) {

if((uint32_t)argument == LED1_ON) {

LED_On(1);

}

else if ((uint32_t)argument == LED1_OFF){

LED_Off(1);

}

delay(500);

osThreadExit();

}
```

When the worker thread runs it flashes the led but instead of having an infinite loop it calls osExit(); to terminate its runtime which will cause led_thread1 to leave the waiting state and enter the ready state and in this example then enter the run state.

Build the code

Start the debugger

Open the View\watch\RTOS window

Run the code and watch the behavior of the threads