Title: Basic Scheduler Manual

History

Issue status (Index)	Maturity/Date (draft/invalid/valid) (dd-mmm-yyyy)	Author Department	Check/Release Department	Description
1.0	Draft 13-11-15	Edgar Escayola	Edgar Escayola	Creation of the document
1.1	Draft 15-11-15	Adrián Zacarías	Adrián Zacarías	General description.
1.2	Draft 17-11-15	Edgar Escayola	Edgar Escayola	Format

Table of Contents

Section 1. Purpose

This document was designed to aid the developer on the implementation of the basic scheduler.

Section 2. Definitions and abbreviations

Abbreviations

ms - milisecond (1E-3 seconds)

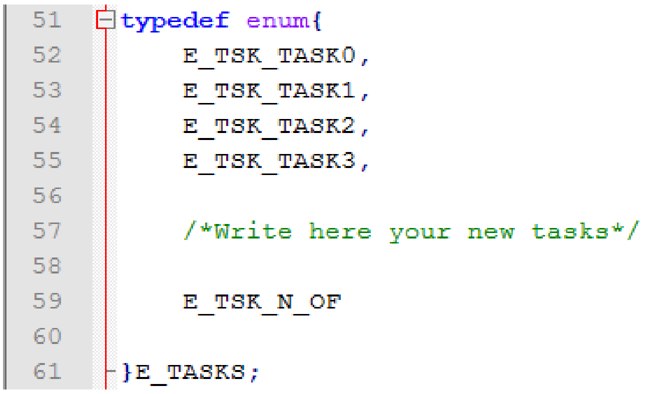
Section 3. General information

Tick period = 1 ms

Files	Description	Modifiable
Kernel.h and Kernel.c	Kernel of scheduler	No
Tasks_init.h and Tasks_init.c	Definition of tasks	Yes
Tasks.h and Tasks.c	Implementation of tasks	Yes
HAL.h and HAL.c	Hardware Application Layer	Yes

Section 4. Add a new task

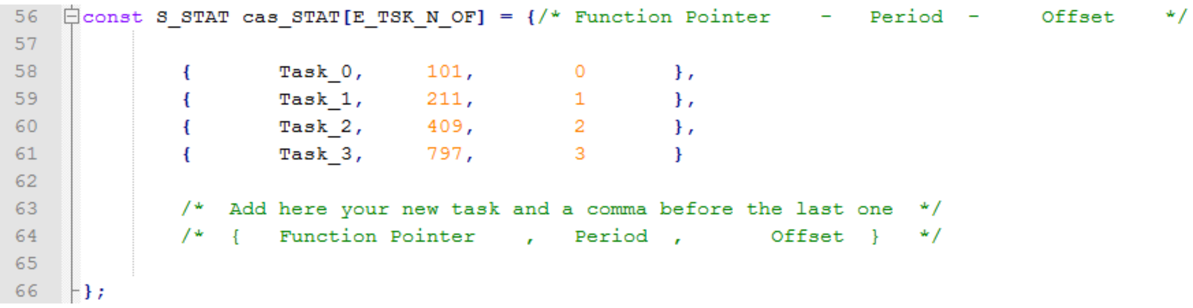
1. **Modify Init_Tasks.h**
   - The enumeration E_TASKS must be updated using the following format: E_TSK_TASKn where "n" is the number of the new TASK.



```
typedef enum{
        E_TSK_TASK0,
        E_TSK_TASK1,
        E_TSK_TASK2,
        E_TSK_TASK3,

        /*Write here your new tasks*/

        E_TSK_N_OF

}E_TASKS;
```

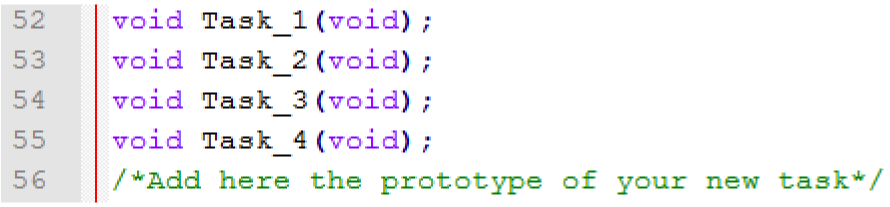
- The array cas_STAT must be updated. The set should contain the name of the function pointer, the period and the offset. The function pointer should use the following format: Task_n where “n” is the number of the task added. The period and the offset is in ms according to the Tick rate.

  Note: It shouldn’t be forgotten to add the comma before the last task.



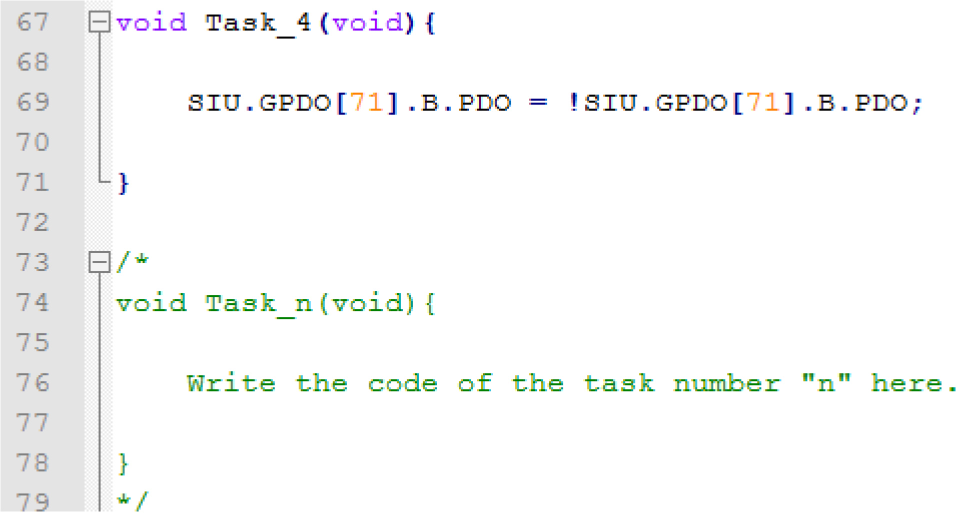
```
const S_STAT cas_STAT[E_TSK_N_OF] = {/* Function Pointer    -   Period  -     Offset    */

            {       Task_0,     101,         0       },
            {       Task_1,     211,         1       },
            {       Task_2,     409,         2       },
            {       Task_3,     797,         3       }

            /*  Add here your new task and a comma before the last one  */
            /*  {   Function Pointer    ,   Period  ,       Offset  }   */

};
```

2. **Modify Tasks.h**
   - The prototype of the new task must be added using the following format: *void Task_n (void)* where n is the where “n” is the number of the new TASK.



```
void Task_1(void);
void Task_2(void);
void Task_3(void);
void Task_4(void);
/*Add here the prototype of your new task*/
```

3. **Modify Tasks.c**
   - The implementation of the new task must be added at the end of this document following the next example:



```
void Task_4(void){

    SIU.GPDO[71].B.PDO = !SIU.GPDO[71].B.PDO;

}

/*
void Task_n(void){

    Write the code of the task number "n" here.

}
*/
```