

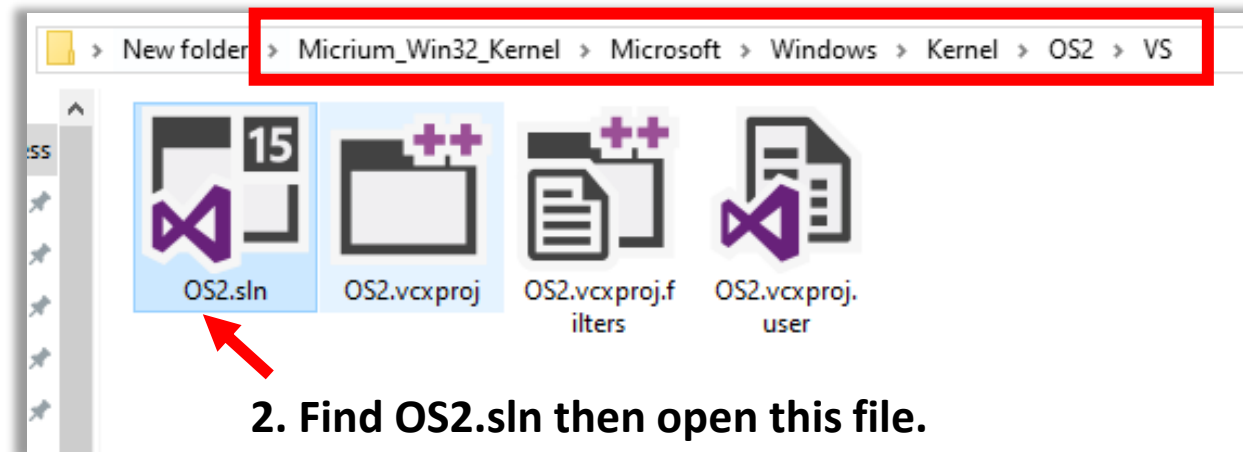
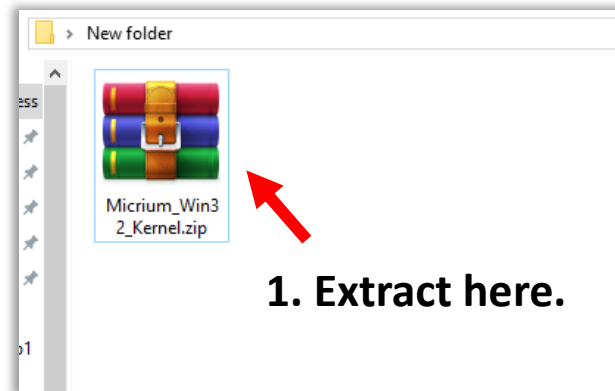
# Run $\mu\text{C}/\text{OS-II}$

2025/03/06

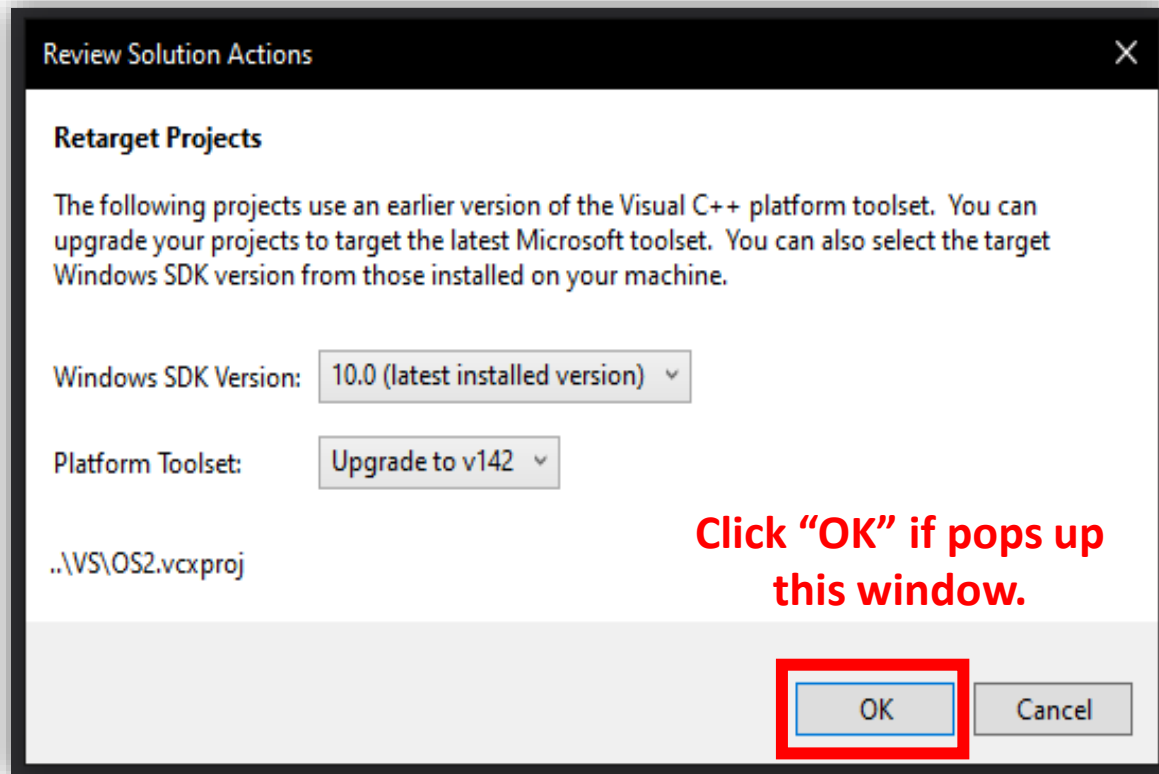
# Outline

- Open example project
- Find  $\mu$ C/OS-II source code
- Run example project
- Modify example project
- Create the initial tasks of HW1
- Debug mode

# Open example project



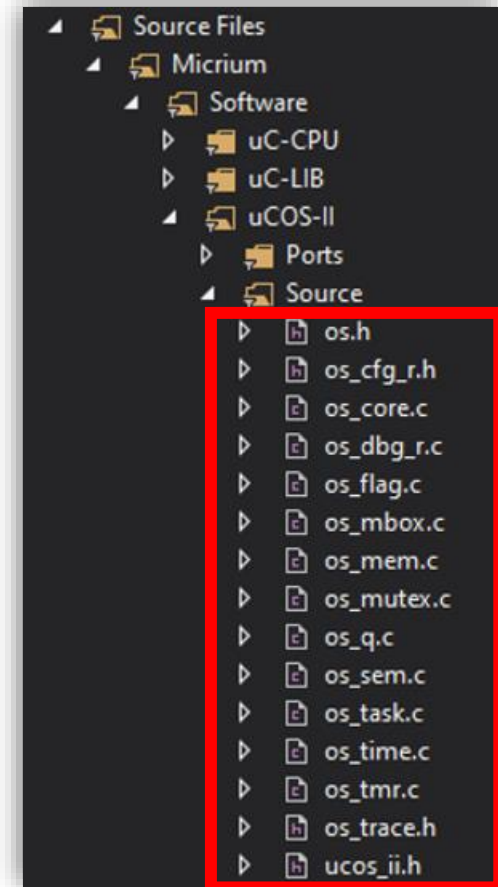
# Open example project



# Find $\mu$ C/OS-II source code

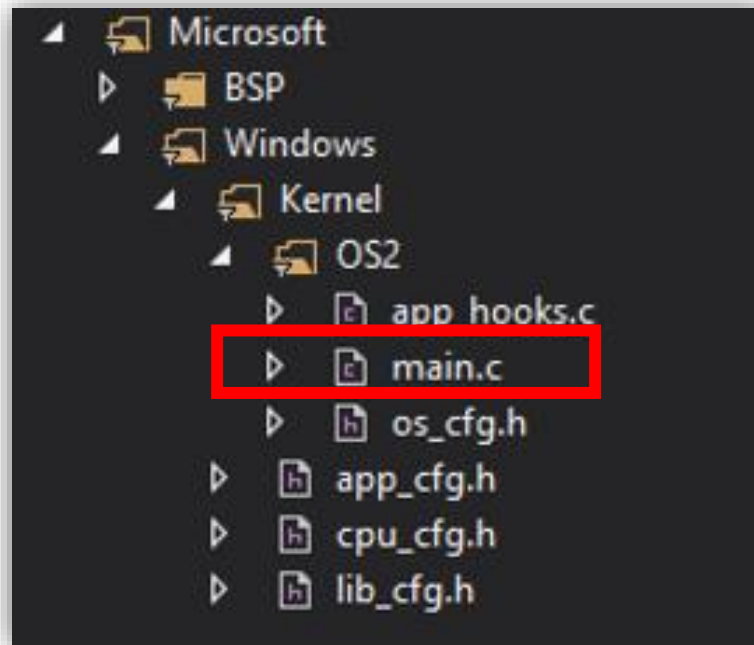
- Source code path:

Source Files\Micrium\Software\uCOSII\  
Source



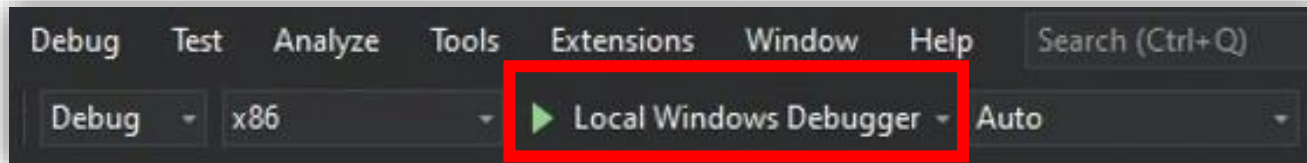
# Find $\mu$ C/OS-II source code

- Main.c path: Microsoft\Windows\Kernel\OS2



# Run example project

- Open `main.c` and then click “Local Windows Debugger” or press F5.



# Run example project

- You can see the **tasks information** in command prompt if the project has been run successfully.

```
OSTick    created, Thread ID 24144
Task[ 63] created, Thread ID 10180
Task[ 62] created, Thread ID 24488
Task[ 61] created, Thread ID 14892
Task[  3] created, Thread ID 14728
Task[  3] 'Startup Task' Running
uCOS-III is Running...
Task[ 61] 'uC/OS-II Tmr' Running
Task[ 62] 'uC/OS-II Stat' Running
Task[ 63] 'uC/OS-II Idle' Running
Time: 100
Time: 200
Time: 300
Time: 400
Time: 500
```



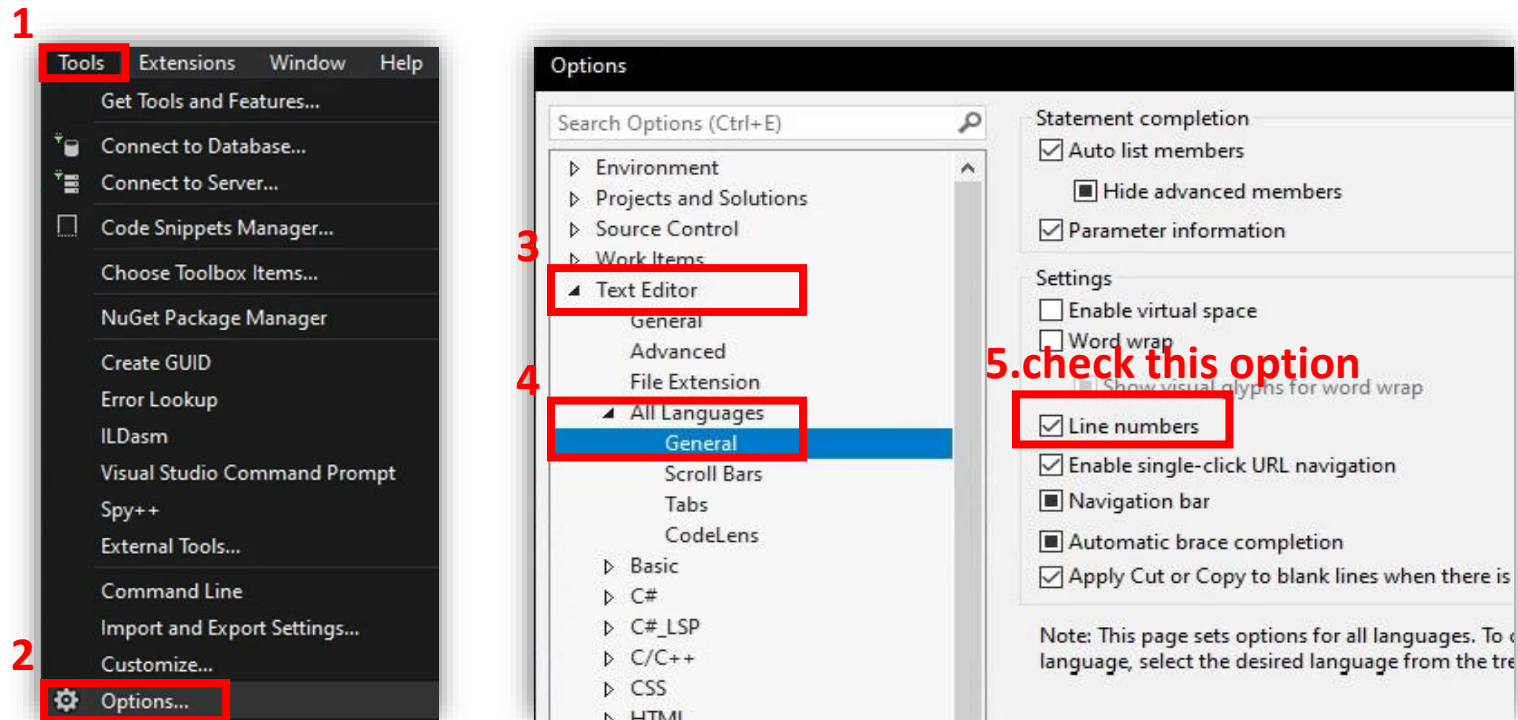
# Modify example project

- You need to disable two tasks and messages **before submitting your project.**

```
OSTick    created, Thread ID  452
Task[ 63] created, Thread ID 10768
Task[ 62] created, Thread ID  1548
Task[ 61] created, Thread ID 10696
Task[  3] created, Thread ID 17848
Task[  3] 'Startup Task' Running
uCOS-III is Running...
Task[ 61] 'uC/OS-II Tmr' Running
Task[ 62] 'uC/OS-II Stat' Running
Task[ 63] 'uC/OS-II Idle' Running
Time: 100
Time: 200
Time: 300
Time: 401
Time: 501
Time: 601
Time: 701
```

# Modify example project

- First, open the line numbers.



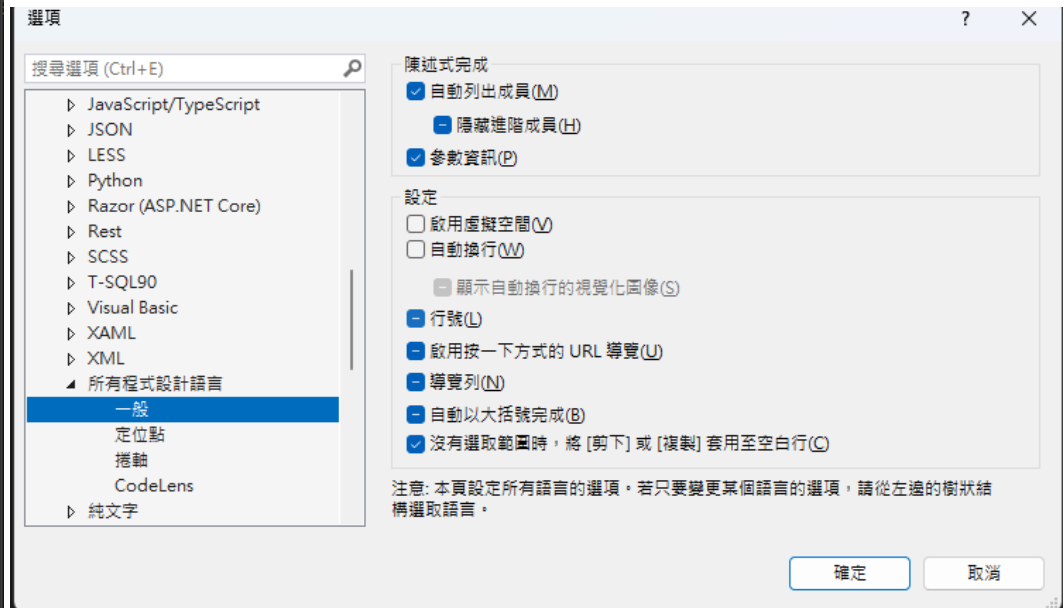
# Modify example project

- First, open the line numbers.

1



2

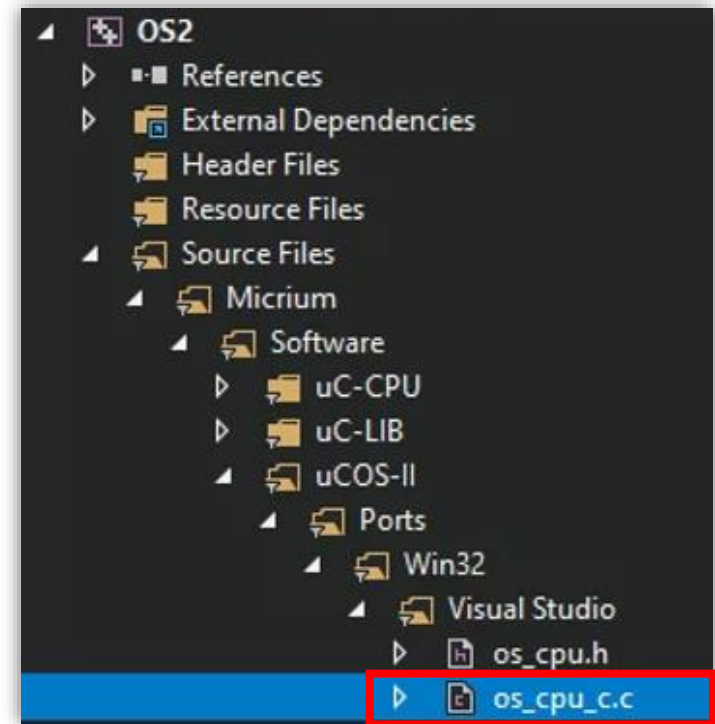


# Modify example project

- Find **os\_cpu.c.c** and then open it.

- os\_cpu.c.c path:

Micrium\Software\uCOSII\Ports\  
Win32\Visual Studio



# Modify example project

- Comment out the 1237<sup>th</sup> line.

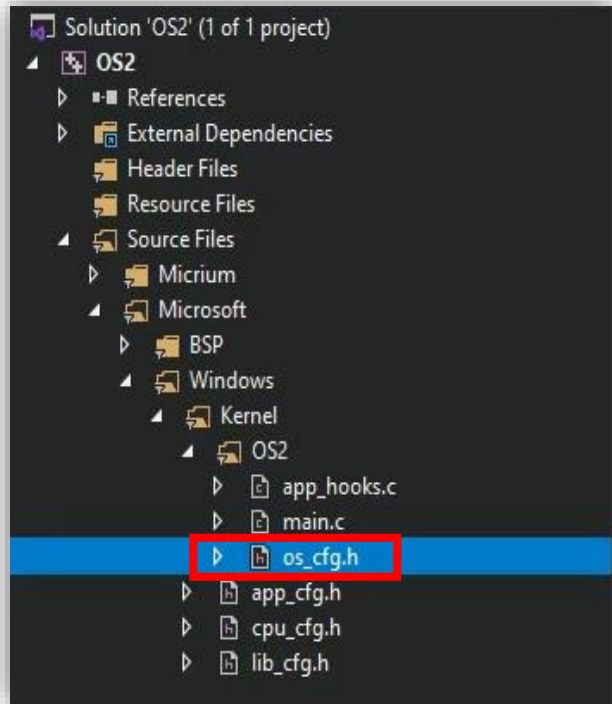
```
1236  #if (OS_MSG_TRACE > 0u)
1237      OS_Printf("Task[%3.1d] '%s' Running\n", p_tcb->OSTCBPrio, p_tcb->OSTCBTaskName);
1238  #endif
```



```
1236  #if (OS_MSG_TRACE > 0u)
1237      //OS_Printf("Task[%3.1d] '%s' Running\n", p_tcb->OSTCBPrio, p_tcb->OSTCBTaskName);
1238  #endif
```

# Modify example project

- Next, find **os\_cfg.h** and then open it.



- **os\_cfg.h** path:  
Micrium\_Win32\_Kernel\  
Microsoft\Windows\Kernel\OS2

# Modify example project

- Go to the 71<sup>th</sup> line and 139<sup>th</sup> line and then **DISABLE** them.

```
70  #define OS_TASK_REG_TBL_SIZE      1u  /* Size of task variables array (#of INT32U entries)
71  #define OS_TASK_STAT_EN Task[62]  1u  /* Enable (1) or Disable(0) the statistics task
72  #define OS_TASK_STAT_STK_CHK_EN  1u  /* Check task stacks from statistic task
```

```
138  /* ----- TIMER MANAGEMENT -----
139  #define OS_TMR_EN Task[61]  1u  /* Enable (1) or Disable (0) code generation for TIMERS
140  #define OS_TMR_CFG_MAX      16u  /* Maximum number of timers
```



```
70  #define OS_TASK_REG_TBL_SIZE      1u  /* Size of task variables array (#of INT32U entries)
71  #define OS_TASK_STAT_EN           0u  /* Enable (1) or Disable(0) the statistics task
72  #define OS_TASK_STAT_STK_CHK_EN  1u  /* Check task stacks from statistic task
```

```
138  /* ----- TIMER MANAGEMENT ----- *
139  #define OS_TMR_EN           0u  /* Enable (1) or Disable (0) code generation for TIMERS *
140  #define OS_TMR_CFG_MAX      16u  /* Maximum number of timers *
```

# Modify example project

- Finally, **rerun this project** and you can see the modified tasks information.

```
OSTick    created, Thread ID 17868
Task[ 63] created, Thread ID 18020
Task[  3] created, Thread ID 18740
uCOS-III is Running...
Time: 100
Time: 200
Time: 300
Time: 400
Time: 500
Time: 600
Time: 700
Time: 800
Time: 900
Time: 1000
Time: 1100
Time: 1200
```



# Modify example project

- Go to `os_cfg.h` and then set 1 tick in 1 second.

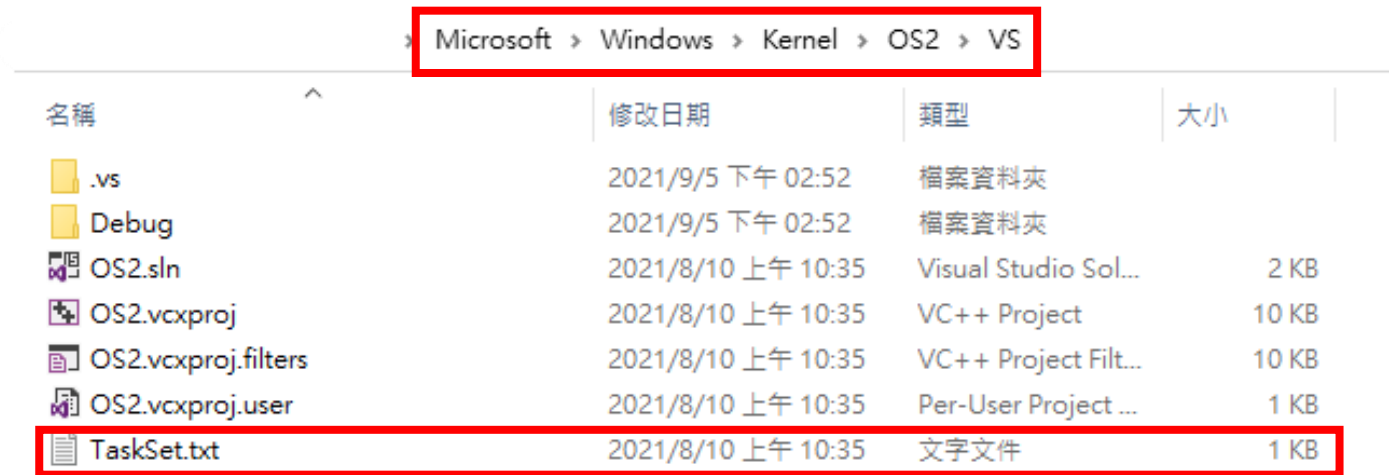
```
49  
50 #define OS_TICK_STEP_EN      1u  /* Enable tick stepping feature for uC/OS-View  
51 #define OS_TICKS_PER_SEC    100u /* Set the number of ticks in one second  
52
```



```
50 #define OS_TICK_STEP_EN      1u  /* Enable tick stepping feature for uC/OS-View  
51 #define OS_TICKS_PER_SEC    1u  /* Set the number of ticks in one second  
52  
53 #define OS_TLS_TBL_SIZE      0u  /* Size of Thread-Local Storage Table
```

# Create the initial tasks of HW1

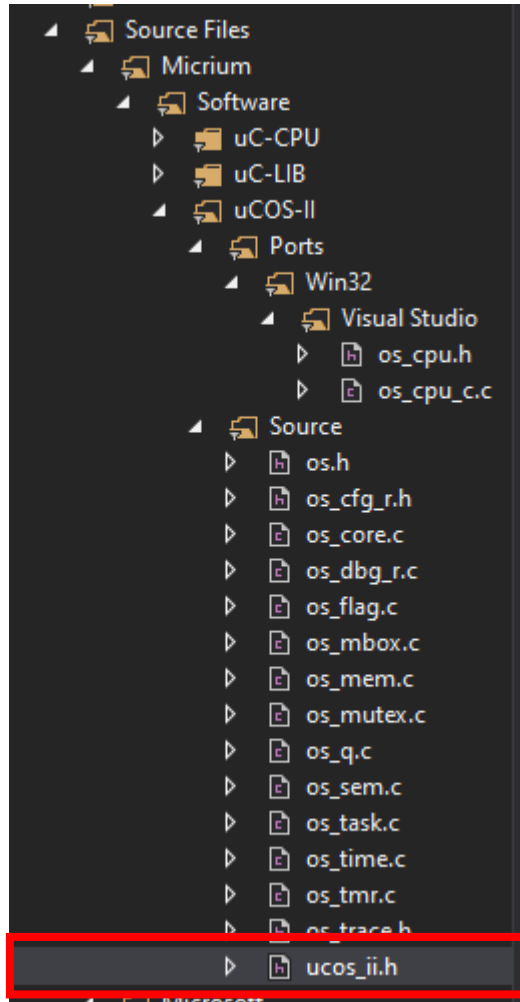
- Download input TaskSet.txt from Moodle.
- Move the file to the path  
Micrium\_Win32\_Kernel\Microsoft\Windows\Kernel\OS2



Microsoft > Windows > Kernel > OS2 > VS				
名稱	修改日期	類型	大小	
.vs	2021/9/5 下午 02:52	檔案資料夾		
Debug	2021/9/5 下午 02:52	檔案資料夾		
OS2.sln	2021/8/10 上午 10:35	Visual Studio Sol...	2 KB	
OS2.vcxproj	2021/8/10 上午 10:35	VC++ Project	10 KB	
OS2.vcxproj.filters	2021/8/10 上午 10:35	VC++ Project Filt...	10 KB	
OS2.vcxproj.user	2021/8/10 上午 10:35	Per-User Project ...	1 KB	
TaskSet.txt	2021/8/10 上午 10:35	文字文件	1 KB	

# Create the initial tasks of HW1

- Go to `ucos_ii.h`



- ucos\_ii.h path:  
Micrium\_Win32\_Kernel\Micrium  
\Software\uCOS-II\Source

# Create the initial tasks of HW1

- Then, include <string.h> and add some parameter structure in ucos\_ii.h

```
58  /*read file*/  
59  #include <string.h>
```

```
67  /*End time for the simulation*/  
68  #define SYSTEM_END_TIME 30  
69  
70  /*Input File*/  
71  FILE* fp;  
72  #define INPUT_FILE_NAME "./TaskSet.txt"  
73  #define OUTPUT_FILE_NAME "./Output.txt"  
74  #define MAX 20           //Task maximum number  
75  #define INFO 4           //information of task  
76  /*Input File*/  
77  
78  /*Output file*/  
79  FILE* Output_fp;  
80  errno_t Output_err;  
81  /*Output file*/
```

**Notice:**  
Please make sure filenames are same as the figure.

**Hints:**  
Parameters may be modified by different project

# Create the initial tasks of HW1

- Then, include <string.h> and add some parameter structure in ucos\_ii.h

```
83  /*Task structure */
84  typedef struct task_para_set {
85      INT16U TaskID;
86      INT16U TaskArriveTime;
87      INT16U TaskExecutionTime;
88      INT16U TaskPeriodic;
89      INT16U TaskNumber;
90      INT16U TaskPriority;
91  }task_para_set;
92
93  int TASK_NUMBER;
94  OS_STK** Task_STK;
95  task_para_set TaskParameter[OS_MAX_TASKS];
```

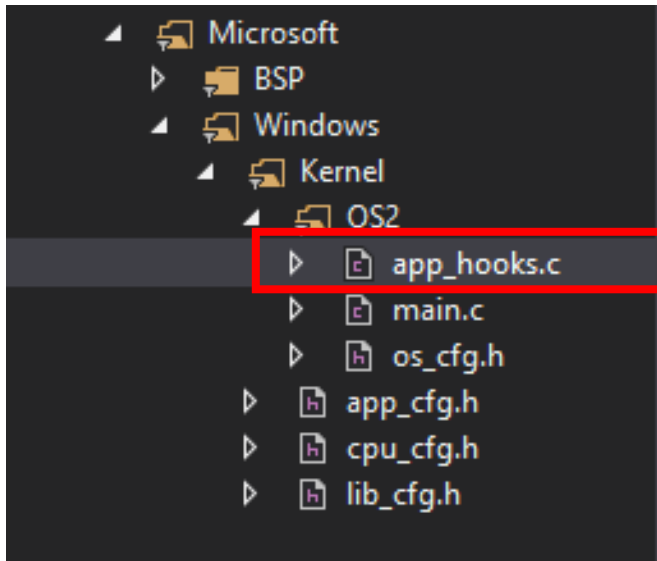
# Create the initial tasks of HW1

- Then, include <string.h> and add some parameter structure in uclos\_ii.h

```
1477 void      App_TaskReturnHook  (OS_TCB      *ptcb);
1478
1479 void      App_TaskStatHook    (void);
1480
1481 #if OS_TASK_SW_HOOK_EN > 0u
1482 void      App_TaskSwHook      (void);
1483 #endif
1484
1485 void      App_TCBInitHook      (OS_TCB      *ptcb);
1486
1487 #if OS_TIME_TICK_HOOK_EN > 0u
1488 void      App_TimeTickHook     (void);
1489 #endif
1490
1491 void      OutFileInit(void);
1492 void      InputFile(void);
1493 #endif
1494
1495 /*
1496 *****
1497 *                                     FUNCTION PROTOTYPES
1498 *
1499 * IMPORTANT: These prototypes MUST be placed in OS_CPU.H
1500 *****
1501 */
1502
1503 #if 0
1504 void      OSStartHighRdy      (void);
1505 void      OSIntCtxSw          (void);
1506 void      OSCtxSw             (void);
1507 #endif
1508
1509
```

# Create the initial tasks of HW1

- Go to `app_hooks.c` and then add two functions.



- `app_hooks.c` path:  
Micrium\_Win32\_Kernel\Microsoft\Windows\Kernel\OS2

# Create the initial tasks of HW1

- Then, add two functions in `app_hooks.c`

```
91 void OutFileInit() {
92     /*Clear the file*/
93     if ((Output_err = fopen_s(&Output_fp, OUTPUT_FILE_NAME, "w")) == 0)
94         fclose(Output_fp);
95     else
96         printf("Error to clear output file");
97 }
```

```
99 void InputFile() {
100     /*
101      * Read File
102      * Task Information:
103      * Task_ID ArriveTime ExecutionTime Periodic
104      */
105     errno_t err;
106     if ((err = fopen_s(&fp, INPUT_FILE_NAME, "r")) == 0)        /*task set 1-4*/
107     {
108         printf("The file 'TaskSet.txt' was opened\n");
109     }
110     else
111     {
112         printf("The file 'TaskSet.txt' was not opened\n");
113     }
114
115     char str[MAX];
116     char* ptr;
117     char* pTmp = NULL;
118     int TaskInfo[INFO], i, j = 0;
119     TASK_NUMBER = 0;
```



# Create the initial tasks of HW1

- Then, add two functions in `app_hooks.c`

```
120 while (!feof(fp))
121 {
122     i = 0;
123     memset(str, 0, sizeof(str));
124     fgets(str, sizeof(str) - 1, fp);
125     ptr = strtok_s(str, " ", &pTmp);
126     while (ptr != NULL)
127     {
128         TaskInfo[i] = atoi(ptr);
129         ptr = strtok_s(NULL, " ", &pTmp);
130         /*printf("Info: %d\n", task_inf[i]);*/
131         if (i == 0) {
132             TASK_NUMBER++;
133             TaskParameter[j].TaskID = TASK_NUMBER;
134         }
135         else if (i == 1)
136             TaskParameter[j].TaskArriveTime = TaskInfo[i];
137         else if (i == 2) {
138             TaskParameter[j].TaskExecutionTime = TaskInfo[i];
139         }
140         else if (i == 3)
141             TaskParameter[j].TaskPeriodic = TaskInfo[i];
142
143         i++;
144     }
145
146     /*Initial Priority*/
147     TaskParameter[j].TaskPriority = j; //just an example
148
149     j++;
150 }
151 fclose(fp);
152 /*read file*/
153 }
```

Notice:  
Here has a space in  
strotok\_s function

Hints:  
Initial priority can be  
fixed when use  
different scheduling.

# Create the initial tasks of HW1

- You need to declare the task **stack size** and as **GLOBAL** variables in main.c.

```
51  /*
52  ****
53  *                                LOCAL GLOBAL VARIABLES
54  ****
55  */
56
57  #define TASK_STACKSIZE          2048
58
```

# Create the initial tasks of HW1

- Initial the output file and read the input file in main function.
- Create Stack Size for every task in main function.

```
99      /*Initialize Output File*/
100     OutFileInit();
101
102     /*Input File*/
103     InputFile();
```

```
105     /*Dynamic Create the Stack size*/
106     Task_STK = malloc(TASK_NUMBER * sizeof(int*));
107
108     /* for each pointer, allocate storage for an array of ints */
109     int n;
110     for (n = 0; n < TASK_NUMBER; n++) {
111         Task_STK[n] = malloc(TASK_STACKSIZE * sizeof(int));
112     }
```

# Create the initial tasks of HW1

- Declare and define task function.

```
65 static void task1(void* p_arg);  
66 static void task2(void* p_arg);  
67
```

```
141  
142 void task1(void* p_arg) {  
143     task_para_set* task_data;  
144     task_data = p_arg;  
145     while (1)  
146     {  
147         printf("Tick: %d, Hello from task%d\n", OSTime, task_data->TaskID);  
148         OSTimeDly(task_data->TaskPeriodic);  
149     }  
150 }  
151  
152 void task2(void* p_arg) {  
153     task_para_set* task_data;  
154     task_data = p_arg;  
155     while (1)  
156     {  
157         printf("Tick: %d, Hello from task%d\n", OSTime, task_data->TaskID);  
158         OSTimeDly(task_data->TaskPeriodic);  
159     }  
160 }
```

# Create the initial tasks of HW1

- Add this part when you print the output information.
- Then, you can write the output in Output.txt.

```
printf("Tick: %d, Hello from task%d\n", OSTime, task_data->TaskID);
```



```
printf("Tick: %d, Hello from task%d\n", OSTime, task_data->TaskID);  
if ((Output_err = fopen_s(&Output_fp, "./Output.txt", "a")) == 0)  
{  
    fprintf(Output_fp, "Tick: %d, Hello from task%d\n", OSTime, task_data->TaskID);  
    fclose(Output_fp);  
}
```

Output.txt - 記事本

檔案(F) 編輯(E) 格式(O) 檢視(V)

```
Tick: 0, Hello from task1  
Tick: 0, Hello from task2  
Tick: 3, Hello from task1  
Tick: 6, Hello from task1  
Tick: 6, Hello from task2  
Tick: 9, Hello from task1  
Tick: 12, Hello from task1  
Tick: 12, Hello from task2  
Tick: 15, Hello from task1  
Tick: 18, Hello from task1  
Tick: 18, Hello from task2  
Tick: 21, Hello from task1  
Tick: 24, Hello from task1  
Tick: 24, Hello from task2  
Tick: 27, Hello from task1  
Tick: 30, Hello from task1  
Tick: 30, Hello from task2
```

# Create the initial tasks of HW1

- Call *OSTaskCreateExt(...)* in main function to create a new task.

```
114  /*Creat Task Set*/
115  OSTaskCreateExt(task1,                                /* Create the task1 */
116      &TaskParameter[0],
117      &Task_STK[0][TASK_STACKSIZE - 1],
118      TaskParameter[0].TaskPriority,
119      TaskParameter[0].TaskID,
120      &Task_STK[0][0],
121      TASK_STACKSIZE,
122      &TaskParameter[0],
123      (OS_TASK_OPT_STK_CHK | OS_TASK_OPT_STK_CLR));
124
125  OSTaskCreateExt(task2,                                /* Create the task2 */
126      &TaskParameter[1],
127      &Task_STK[1][TASK_STACKSIZE - 1],
128      TaskParameter[1].TaskPriority,
129      TaskParameter[1].TaskID,
130      &Task_STK[1][0],
131      TASK_STACKSIZE,
132      &TaskParameter[1],
133      (OS_TASK_OPT_STK_CHK | OS_TASK_OPT_STK_CLR));
```

You can find its definition by right-click its name.

**Hints:**  
Create the task in the for loop will be suitable next project.

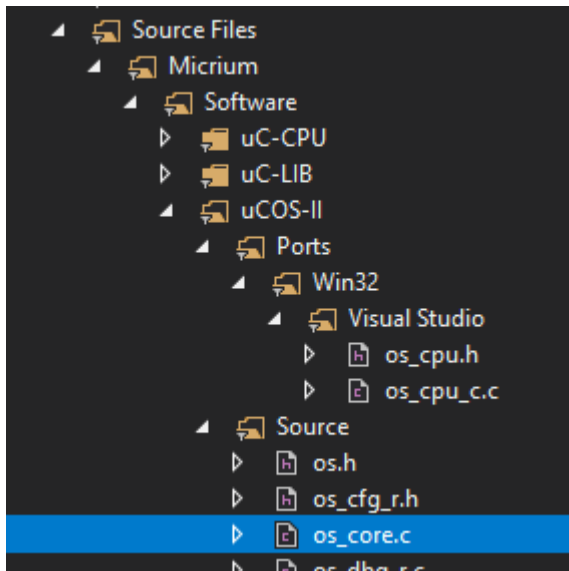
# Create the initial tasks of HW1

- Comment out or delete “*create the startup task*” and “*OSTaskNameSet(...)*”

```
130  /*
131  OSTaskCreateExt( StartupTask,                      /* Create the startup task
132                  0,
133                  &StartupTaskStk[APP_CFG_STARTUP_TASK_STK_SIZE - 1u],
134                  APP_CFG_STARTUP_TASK_PRIO,
135                  APP_CFG_STARTUP_TASK_PRIO,
136                  &StartupTaskStk[0u],
137                  APP_CFG_STARTUP_TASK_STK_SIZE,
138                  0u,
139                  (OS_TASK_OPT_STK_CHK | OS_TASK_OPT_STK_CLR));
140
141  #if OS_TASK_NAME_EN > 0u
142      OSTaskNameSet(      APP_CFG_STARTUP_TASK_PRIO,
143                      (INT8U *)"Startup Task",
144                      &os_err);
145  #endif
146  */
```

# Create the initial tasks of HW1

- Open the file **os\_core.c**
- Finally, add the system end time in the OSTimeTick()



- os\_core.c path:  
Micrium\_Win32\_Kernel\Micrium\  
Software\uCOS-II\Source

OSTimeTick():

```
961      /*Setting the end time for the OS*/  
962      if (OSTimeGet() > SYSTEM_END_TIME) {  
963          OSRunning = OS_FALSE;  
964          exit(0);  
965      }  
966      /*Setting the end time for the OS*/
```

This function  
will stop  
project



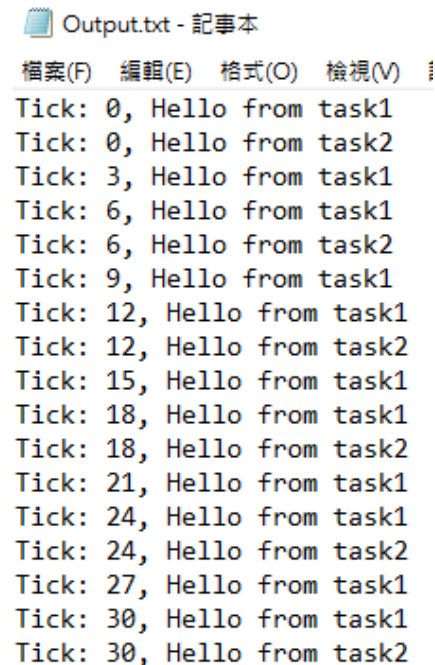
# Create the initial tasks of HW1

- Then rerun this project, you can see two tasks information in command prompt.

```
OSTick    created, Thread ID 13816
Task[ 63] created, Thread ID  3592
The file 'TaskSet.txt' was opened
Task[  0] created, Thread ID  6108
Task[  1] created, Thread ID  1904
Tick: 0, Hello from task1
Tick: 0, Hello from task2
Tick: 3, Hello from task1
Tick: 6, Hello from task1
Tick: 6, Hello from task2
Tick: 9, Hello from task1
Tick: 12, Hello from task1
Tick: 12, Hello from task2
Tick: 15, Hello from task1
Tick: 18, Hello from task1
Tick: 18, Hello from task2
```

# Create the initial tasks of HW1

- Please make sure the output formation is exactly same as the answer we provide.
- We will use our checker to see if the answer is correct or not in HW through the output of your project.



Output.txt - 記事本

檔案(F) 編輯(E) 格式(O) 檢視(V) ⋮

Tick: 0, Hello from task1  
Tick: 0, Hello from task2  
Tick: 3, Hello from task1  
Tick: 6, Hello from task1  
Tick: 6, Hello from task2  
Tick: 9, Hello from task1  
Tick: 12, Hello from task1  
Tick: 12, Hello from task2  
Tick: 15, Hello from task1  
Tick: 18, Hello from task1  
Tick: 18, Hello from task2  
Tick: 21, Hello from task1  
Tick: 24, Hello from task1  
Tick: 24, Hello from task2  
Tick: 27, Hello from task1  
Tick: 30, Hello from task1  
Tick: 30, Hello from task2

***Now you can use this project to do HW1!***

# Disclaimer

Use of this documentation is your acknowledgment that you agree to and will comply with the following notices and disclaimers. You are advised to seek appropriate legal, engineering, and other professional advice regarding the use, interpretation, and effect of this documentation.

This document only provides students who take this course (Embedded OS Implementation).

Any modification or spreading of this document without permission is not allowed.

Copyright (C) Embedded System Software Lab.

All Right Reserved.