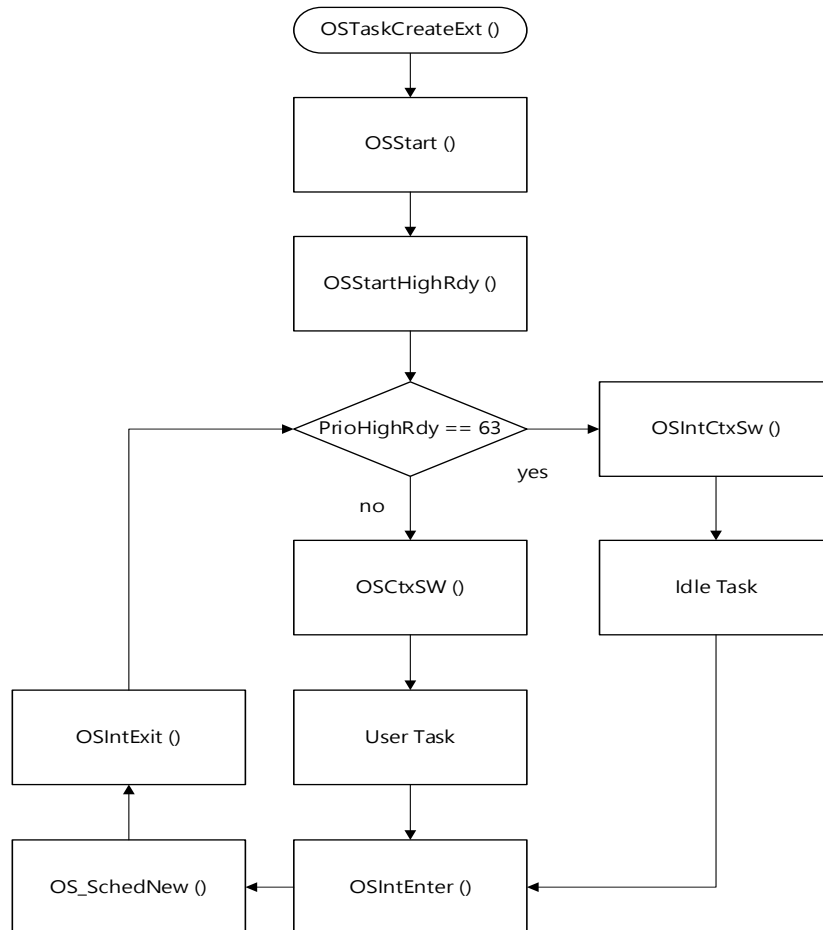


1. The system flow and the explanation of the process (functions). (45%)



在 HW1 中總共有兩個任務：Task1 和 Task2，必須要在 `OSStart ()` 之前呼叫 `OSTaskCteate ()` 建立任務，接著開始執行 `OSStart ()`，去尋找最高優先權的 Task，透過 `OSStartHighRdy ()` 去執行 Task。

透過判斷 `PrioHighRdy` 要去執行哪項 Task，如果沒有其他 Task（等於 63），就會透過 `OSIntCtx ()` 切換到 `Idle Task`；反之則會透過 `OSCtxSW ()` 切換到優先權最高的 `User Task`。

在 Task 執行結束後，將會進行 `OSIntEnter ()` 與 `OSIntExit ()`，透過 `OS_SchedNew ()` 尋找下一個 `PrioHightRdy`，反覆進行 Context Switch 切換 Task。

2. The screenshot of the result. (10%)

Screenshot

```
D:\ntust\EmbeddingOS\2022.1005\M11107309_RTOS_HW1\micrium_win32_kernel\
OSTick created, Thread ID 4604
Task[ 63] created, Thread ID 4756
The file 'TaskSet.txt' was opened
Task[ 1] created, Thread ID 4136
Task[ 2] created, Thread ID 4828
Tick CurrentTask ID NextTask ID Caller
0 ***** task( 1)( 0) OSStart ()
0 task( 1)( 0) task( 2)( 0) OSCtxSW ()
0 task( 2)( 0) task( 63) OSCtxSW ()
3 task( 63) task( 1)( 1) OSIntCtxSw ()
3 task( 1)( 1) task( 63) OSCtxSW ()
5 task( 63) task( 2)( 1) OSIntCtxSw ()
5 task( 2)( 1) task( 63) OSCtxSW ()
6 task( 63) task( 1)( 2) OSIntCtxSw ()
6 task( 1)( 2) task( 63) OSCtxSW ()
9 task( 63) task( 1)( 3) OSIntCtxSw ()
9 task( 1)( 3) task( 63) OSCtxSW ()
10 task( 63) task( 2)( 2) OSIntCtxSw ()
10 task( 2)( 2) task( 63) OSCtxSW ()
12 task( 63) task( 1)( 4) OSIntCtxSw ()
12 task( 1)( 4) task( 63) OSCtxSW ()
15 task( 63) task( 1)( 5) OSIntCtxSw ()
15 task( 1)( 5) task( 2)( 3) OSCtxSW ()
15 task( 2)( 3) task( 63) OSCtxSW ()
18 task( 63) task( 1)( 6) OSIntCtxSw ()
18 task( 1)( 6) task( 63) OSCtxSW ()
20 task( 63) task( 2)( 4) OSIntCtxSw ()
20 task( 2)( 4) task( 63) OSCtxSW ()
21 task( 63) task( 1)( 7) OSIntCtxSw ()
21 task( 1)( 7) task( 63) OSCtxSW ()
24 task( 63) task( 1)( 8) OSIntCtxSw ()
24 task( 1)( 8) task( 63) OSCtxSW ()
25 task( 63) task( 2)( 5) OSIntCtxSw ()
25 task( 2)( 5) task( 63) OSCtxSW ()
27 task( 63) task( 1)( 9) OSIntCtxSw ()
27 task( 1)( 9) task( 63) OSCtxSW ()
30 task( 63) task( 1)(10) OSIntCtxSw ()
30 task( 1)(10) task( 2)( 6) OSCtxSW ()
30 task( 2)( 6) task( 63) OSCtxSW ()
```

Output.txt

```
Output.txt - 記事本
檔案 編輯 檢視
| 0 ***** task( 1)( 0) OSStart ()
| 0 task( 1)( 0) task( 2)( 0) OSCtxSW ()
| 0 task( 2)( 0) task( 63) OSCtxSW ()
| 3 task( 63) task( 1)( 1) OSIntCtxSw ()
| 3 task( 1)( 1) task( 63) OSCtxSW ()
| 5 task( 63) task( 2)( 1) OSIntCtxSw ()
| 5 task( 2)( 1) task( 63) OSCtxSW ()
| 6 task( 63) task( 1)( 2) OSIntCtxSw ()
| 6 task( 1)( 2) task( 63) OSCtxSW ()
| 9 task( 63) task( 1)( 3) OSIntCtxSw ()
| 9 task( 1)( 3) task( 63) OSCtxSW ()
| 10 task( 63) task( 2)( 2) OSIntCtxSw ()
| 10 task( 2)( 2) task( 63) OSCtxSW ()
| 12 task( 63) task( 1)( 4) OSIntCtxSw ()
| 12 task( 1)( 4) task( 63) OSCtxSW ()
| 15 task( 63) task( 1)( 5) OSIntCtxSw ()
| 15 task( 1)( 5) task( 2)( 3) OSCtxSW ()
| 15 task( 2)( 3) task( 63) OSCtxSW ()
| 18 task( 63) task( 1)( 6) OSIntCtxSw ()
| 18 task( 1)( 6) task( 63) OSCtxSW ()
| 20 task( 63) task( 2)( 4) OSIntCtxSw ()
| 20 task( 2)( 4) task( 63) OSCtxSW ()
| 21 task( 63) task( 1)( 7) OSIntCtxSw ()
| 21 task( 1)( 7) task( 63) OSCtxSW ()
| 24 task( 63) task( 1)( 8) OSIntCtxSw ()
| 24 task( 1)( 8) task( 63) OSCtxSW ()
| 25 task( 63) task( 2)( 5) OSIntCtxSw ()
| 25 task( 2)( 5) task( 63) OSCtxSW ()
| 27 task( 63) task( 1)( 9) OSIntCtxSw ()
| 27 task( 1)( 9) task( 63) OSCtxSW ()
| 30 task( 63) task( 1)(10) OSIntCtxSw ()
| 30 task( 1)(10) task( 2)( 6) OSCtxSW ()
| 30 task( 2)( 6) task( 63) OSCtxSW ()
```

3. A report that describes your implementation (please attach the screenshot of the code and MARK the modified part). (45%)

Consider two periodic tasks (τ_1 , τ_2) and their delay time are 3 ticks and 5 ticks, respectively. Task priority of two tasks (τ_1 , τ_2) are 1 and 2, respectively. Please add some code to the $\mu\text{C}/\text{OS-II}$ scheduler in the kernel level to observe how the CPU is switched among tasks by means of context switches.

根據題目要求，需要輸出當下執行的 Task 以及下個 Task 的 ID 與次數。首先建立一個 global variable 用來計數 Task 的執行次數。

```
92
93 int TASK_NUMBER;           //Number of the input tasks
94 int TaskCtr[2];           //Task counter
95 /*Task structure*/
```

修改 InputFile ()，使 Task 的 priority 可以被自定義，透過修改 TaskSet.txt 即可針對 Task 進行修改。

```
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             /*Initial Priority*/
135             TaskParameter[j].TaskPriority = TaskInfo[i];
136         }
137         else if (i == 1)
138             TaskParameter[j].TaskArriveTime = TaskInfo[i];
139         else if (i == 2) {
140             TaskParameter[j].TaskExecutionTime = TaskInfo[i];
141         }
142         else if (i == 3)
143             TaskParameter[j].TaskPeriodic = TaskInfo[i];
144
145         i++;
146     }
147
148
149     j++;
150 }
151 fclose(fp);
152 /*read file*/
```

由於想要觀察到 Task 交換的過程，可透過 Context Switch 去進行觀察。在 OSCtrSW () 中透過讀取 OSPrioCur 與 OSPrioHighRdy 兩個變數，即可判斷現在的 Task 與下個 Task，並將其結果輸出到 Output.txt。

```
800 OSCtrCur = OSCtrHighRdy;
801
802 if (OSPrioHighRdy == 63) {
803     if (OSPrioCur == TaskParameter[0].TaskPriority) {
804         printf("%2d\ttask(%2d)(%2d)\ttask(%2d) \tOSCtrSW ()\n", OSTime, TaskParameter[0].TaskID, TaskCtr[0], OSPrioHighRdy);
805         if ((Output_err = fopen_s(&Output_fp, ".\\Output.txt", "a")) == 0) {
806             fprintf(Output_fp, "%2d\ttask(%2d)(%2d)\ttask(%2d) \tOSCtrSW ()\n", OSTime, TaskParameter[0].TaskID, TaskCtr[0], OSPrioHighRdy);
807             fclose(Output_fp);
808             TaskCtr[0]++;
809         }
810     }
811     else if (OSPrioCur == TaskParameter[1].TaskPriority) {
812         printf("%2d\ttask(%2d)(%2d)\ttask(%2d) \tOSCtrSW ()\n", OSTime, TaskParameter[1].TaskID, TaskCtr[1], OSPrioHighRdy);
813         if ((Output_err = fopen_s(&Output_fp, ".\\Output.txt", "a")) == 0) {
814             fprintf(Output_fp, "%2d\ttask(%2d)(%2d)\ttask(%2d) \tOSCtrSW ()\n", OSTime, TaskParameter[1].TaskID, TaskCtr[1], OSPrioHighRdy);
815             fclose(Output_fp);
816             TaskCtr[1]++;
817         }
818     }
819     else if (OSPrioHighRdy == TaskParameter[0].TaskPriority) {
820         printf("%2d\ttask(%2d)(%2d)\ttask(%2d)(%2d)\tOSCtrSW ()\n", OSTime, TaskParameter[1].TaskID, TaskCtr[1], TaskParameter[0].TaskID, TaskCtr[0]);
821         if ((Output_err = fopen_s(&Output_fp, ".\\Output.txt", "a")) == 0) {
822             fprintf(Output_fp, "%2d\ttask(%2d)(%2d)\ttask(%2d)(%2d)\tOSCtrSW ()\n", OSTime, TaskParameter[1].TaskID, TaskCtr[1], TaskParameter[0].TaskID, TaskCtr[0]);
823             fclose(Output_fp);
824             TaskCtr[1]++;
825         }
826     }
827     else if (OSPrioHighRdy == TaskParameter[1].TaskPriority) {
828         printf("%2d\ttask(%2d)(%2d)\ttask(%2d)(%2d)\tOSCtrSW ()\n", OSTime, TaskParameter[0].TaskID, TaskCtr[0], TaskParameter[1].TaskID, TaskCtr[1]);
829         if ((Output_err = fopen_s(&Output_fp, ".\\Output.txt", "a")) == 0) {
830             fprintf(Output_fp, "%2d\ttask(%2d)(%2d)\ttask(%2d)(%2d)\tOSCtrSW ()\n", OSTime, TaskParameter[0].TaskID, TaskCtr[0], TaskParameter[1].TaskID, TaskCtr[1]);
831             fclose(Output_fp);
832             TaskCtr[0]++;
833         }
834     }
835     OSCtrCur = OSPrioHighRdy;
836 }
```

而沒有任務的時候，將會執行 Idle Task，因此可透過 OSIntCtxSw () 進行 Idle Task 的觀察，並將結果輸出到 Output.txt。

```
954 void OSIntCtxSw (void)
955 {
956     OSTaskHook();
957
958     OSCtrCur = OSCtrHighRdy;
959
960     if (OSPrioHighRdy == TaskParameter[0].TaskPriority) {
961         printf("%2d\ttask(%2d) \ttask(%2d)(%2d)\tOSIntCtxSw ()\n", OSTime, OSPrioCur, TaskParameter[0].TaskID, TaskCtr[0]);
962         if ((Output_err = fopen_s(&Output_fp, ".\\Output.txt", "a")) == 0) {
963             fprintf(Output_fp, "%2d\ttask(%2d) \ttask(%2d)(%2d)\tOSIntCtxSw ()\n", OSTime, OSPrioCur, TaskParameter[0].TaskID, TaskCtr[0]);
964             fclose(Output_fp);
965         }
966     }
967     else if (OSPrioHighRdy == TaskParameter[1].TaskPriority) {
968         printf("%2d\ttask(%2d) \ttask(%2d)(%2d)\tOSIntCtxSw ()\n", OSTime, OSPrioCur, TaskParameter[1].TaskID, TaskCtr[1]);
969         if ((Output_err = fopen_s(&Output_fp, ".\\Output.txt", "a")) == 0) {
970             fprintf(Output_fp, "%2d\ttask(%2d) \ttask(%2d)(%2d)\tOSIntCtxSw ()\n", OSTime, OSPrioCur, TaskParameter[1].TaskID, TaskCtr[1]);
971             fclose(Output_fp);
972         }
973     }
974     OSCtrCur = OSPrioHighRdy;
975 }
```

根據題目要求，OSStart () 也要進行輸出，所以也針對 OSStart () 進行修改，一樣透過讀取 OSPrioHighRdy 去判斷下個執行的 Task。

```
870 void OSStart (void)
871 {
872     OSRunning = OS_FALSE;
873     OS_SchedNew();
874     /* Find highest priority's task priority number */
875     if (OSPrioHighRdy == TaskParameter[0].TaskPriority) {
876         printf("%2d\t***** \ttask(%2d)(%2d)\tOSStart ()\n", OSTime, TaskParameter[0].TaskID, TaskCtr[0]);
877         if ((Output_err = fopen_s(&Output_fp, ".\\Output.txt", "a")) == 0) {
878             fprintf(Output_fp, "%2d\t***** \ttask(%2d)(%2d)\tOSStart ()\n", OSTime, TaskParameter[0].TaskID, TaskCtr[0]);
879             fclose(Output_fp);
880         }
881     }
882     else if (OSPrioHighRdy == TaskParameter[1].TaskPriority) {
883         printf("%2d\t***** \ttask(%2d)(%2d)\tOSStart ()\n", OSTime, TaskParameter[1].TaskID, TaskCtr[1]);
884         if ((Output_err = fopen_s(&Output_fp, ".\\Output.txt", "a")) == 0) {
885             fprintf(Output_fp, "%2d\t***** \ttask(%2d)(%2d)\tOSStart ()\n", OSTime, TaskParameter[1].TaskID, TaskCtr[1]);
886             fclose(Output_fp);
887         }
888     }
889
890     OSPrioCur = OSPrioHighRdy;
891     OSCtrHighRdy = OSCtrCur[OSPrioHighRdy]; /* Point to highest priority task ready to run */
892     OSCtrCur = OSCtrHighRdy;
893     OSStartHighRdy(); /* Execute target specific code to start task */
894 }
```