



Real-Time Operating System (Day 2 Lab)

Jong-Chan Kim

Graduate School of Automotive Engineering



국민대학교
KOOKMIN UNIVERSITY

09. ButtonISR (06 복사해서 수정)

- bsw.cpp 수정: 버튼 인터럽트 받을 수 있도록
- OIL 수정: ButtonISR 추가

```
bsw.c  
  
int main(void)  
{  
    osEE_tc_stm_set_clockpersec();  
    osEE_tc_stm_set_sr0( 1000000U, 1U);  
  
    UART_init();  
    initADC();  
    initPeripheralsAndERU();  
  
    .  
    .  
    .  
}
```

```
ISR ButtonISR {  
    CATEGORY = 2;  
    SOURCE = "SCUERU0";  
    PRIORITY = 10;  
};
```

09. ButtonISR

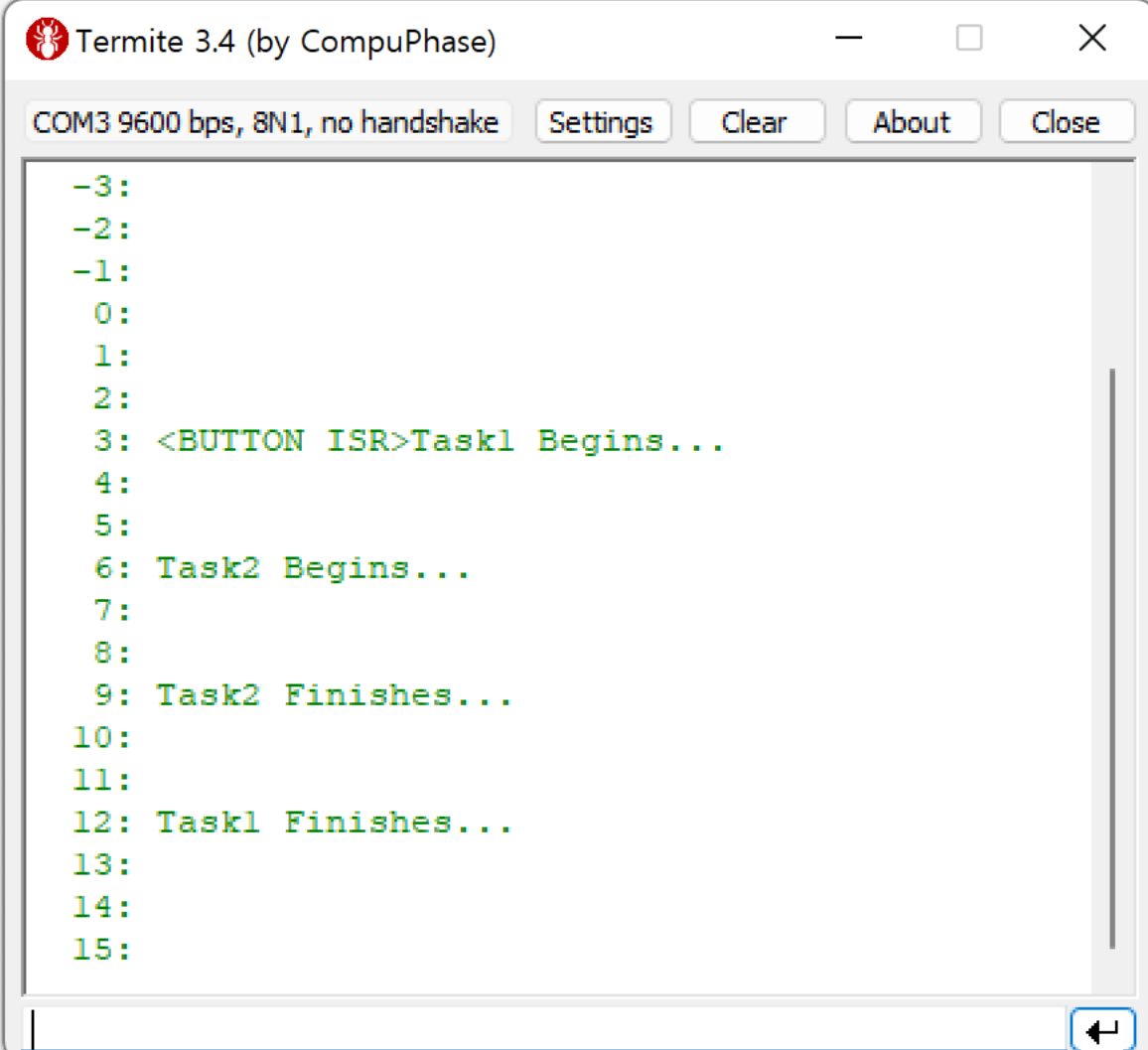
```
ISR2(TimerISR)
{
    static long c = -4;
if (c == 0)
    ActivateTask(Task1);
    printfSerial("\n%4ld: ", c++);
}
```

```
ISR2(ButtonISR)
{
    DisableAllInterrupts();
    osEE_tc_delay(5000);
    printfSerial("<BUTTON ISR>");
    // read ADC value
    unsigned int a0 = readADCValue(3);
    if (a0 < 50) { // UP
        ActivateTask(Task1);
    } else if (a0 < 200) { // DOWN
        ActivateTask(Task2);
    } else if (a0 < 380) { // LEFT
    } else if (a0 < 520) { // RIGHT
    }
    osEE_tc_delay(3000);
    EnableAllInterrupts();
}
```

asw.c

09. ButtonISR

- 중복 Activation의 경우?
 - ACTIVATION = 1; 수정 필요
- ISR에서 mdelay 실행하면?
- Nested Interrupt
 - ButtonISR 도중 TimerISR 실행



The screenshot shows a terminal window titled "Termit 3.4 (by CompuPhase)". The window has a menu bar with "COM3 9600 bps, 8N1, no handshake", "Settings", "Clear", "About", and "Close". The main text area displays a sequence of events, each preceded by a line number from -3 to 15. The events are as follows:

```
-3:
-2:
-1:
0:
1:
2:
3: <BUTTON ISR>Task1 Begins...
4:
5:
6: Task2 Begins...
7:
8:
9: Task2 Finishes...
10:
11:
12: Task1 Finishes...
13:
14:
15:
```

The text is displayed in a monospaced font, with the ISR event on line 3 and the task events on lines 6 and 12. The window has a scrollbar on the right side and a cursor at the bottom right.

10. Alarm

- OIL에 COUNTER와 ALARM 추가

```
COUNTER mycounter {  
    MINCYCLE = 1;  
    MAXALLOWEDVALUE = 127;  
    TICKSPERBASE = 1;  
};
```

```
ALARM alarm1 {  
    COUNTER = mycounter;  
    ACTION = ACTIVATETASK {  
        TASK = Task1;  
    };  
    AUTOSTART = TRUE {  
        ALARMTIME = 5;  
        CYCLETIME = 10;  
    };  
};
```

```
ALARM alarm2 {  
    COUNTER = mycounter;  
    ACTION = ACTIVATETASK {  
        TASK = Task2;  
    };  
    AUTOSTART = TRUE {  
        ALARMTIME = 5;  
        CYCLETIME = 20;  
    };  
};
```

10. Alarm

- TimerISR에서 counter1 증가


```
ISR2(TimerISR)
{
    osEE_tc_stm_set_sr0_next_match
(1000000U);
    static long c = -4;
    IncrementCounter(mycounter);
    printfSerial("\n%4ld: ", c++);
}
```

```
TASK(Task1)
{
    printfSerial("Task1 Begins...");
    mdelay(3000);
    printfSerial("Task1 Finishes...");
    TerminateTask();
}
```

```
TASK(Task2)
{
    printfSerial("Task2 Begins...");
    mdelay(3000);
    printfSerial("Task2 Finishes...");
    TerminateTask();
}
```

10. Alarm

- Alarm을 이용한 주기적 Task 실행



The screenshot shows a terminal window titled "Termite 3.4 (by CompuPhase)". The window has a menu bar with "Settings", "Clear", "About", and "Close". The status bar at the top indicates "COM3 9600 bps, 8N1, no handshake". The terminal displays a log of task execution over 28 lines. The log shows two tasks, Task1 and Task2, executing in a periodic manner. Task2 begins at lines 0, 3, and 20, and finishes at lines 3, 6, and 23. Task1 begins at lines 3, 10, and 23, and finishes at lines 6, 13, and 26. The log is as follows:

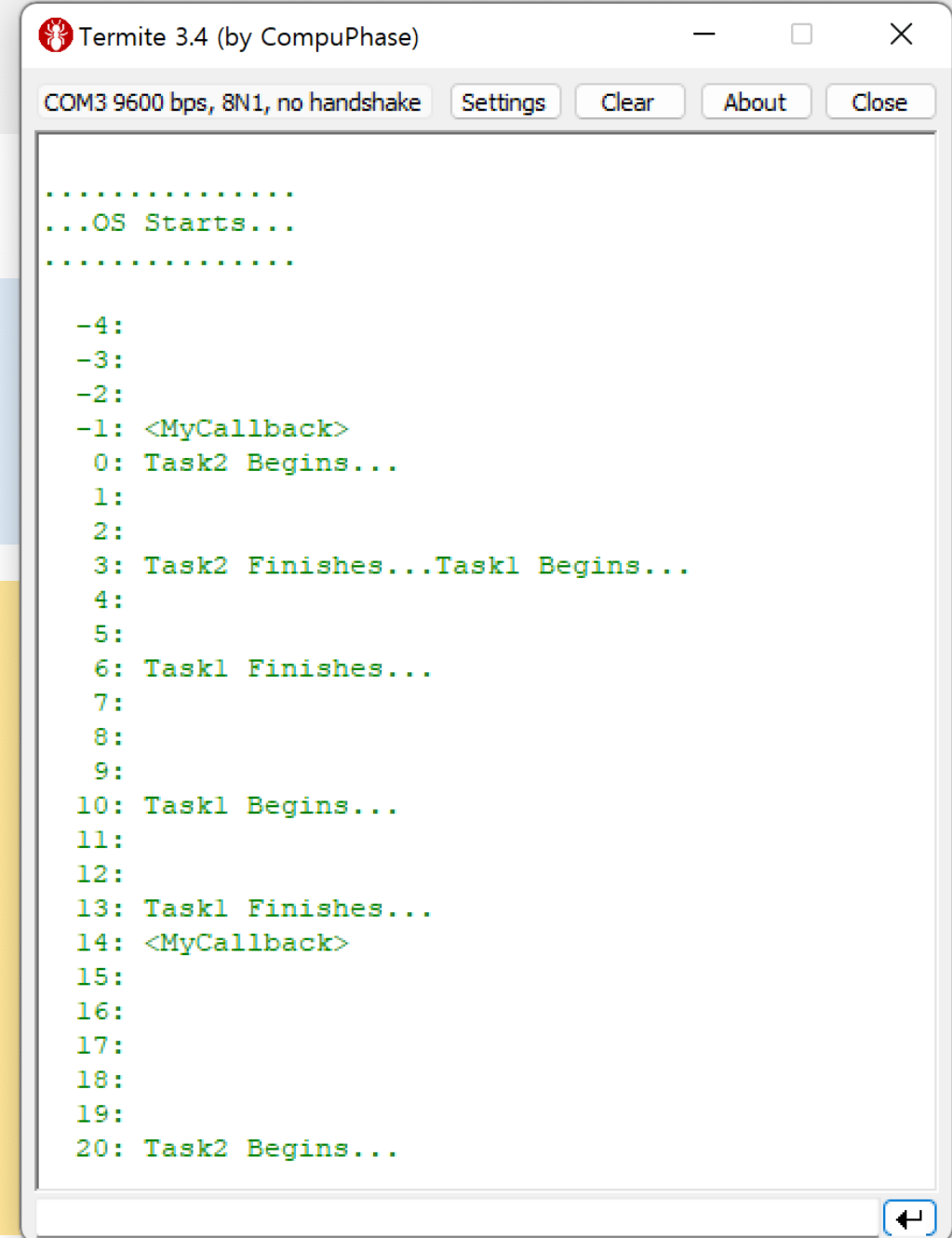
```
0: Task2 Begins...
1:
2:
3: Task2 Finishes...Task1 Begins...
4:
5:
6: Task1 Finishes...
7:
8:
9:
10: Task1 Begins...
11:
12:
13: Task1 Finishes...
14:
15:
16:
17:
18:
19:
20: Task2 Begins...
21:
22:
23: Task2 Finishes...Task1 Begins...
24:
25:
26: Task1 Finishes...
27:
28:
```

11. Alarm Callback

- 콜백 함수 등록

```
ALARMCALLBACK(MyCallback) {  
    printfSerial("MyCallback Begins...");  
    printfSerial("MyCallback Finishes...");  
}
```

```
ALARM alarm3 {  
    COUNTER = mycounter;  
    ACTION = ALARMCALLBACK {  
        ALARMCALLBACKNAME = "MyCallback";  
    };  
    AUTOSTART = TRUE {  
        ALARMTIME = 5;  
        CYCLETIME = 15;  
    };  
};
```



The screenshot shows a terminal window titled "Termite 3.4 (by CompuPhase)". The status bar at the top indicates "COM3 9600 bps, 8N1, no handshake" and includes buttons for "Settings", "Clear", "About", and "Close". The terminal output displays a sequence of events:

```
.....  
...OS Starts...  
.....  
  
-4:  
-3:  
-2:  
-1: <MyCallback>  
  0: Task2 Begins...  
  1:  
  2:  
  3: Task2 Finishes...Task1 Begins...  
  4:  
  5:  
  6: Task1 Finishes...  
  7:  
  8:  
  9:  
 10: Task1 Begins...  
 11:  
 12:  
 13: Task1 Finishes...  
 14: <MyCallback>  
 15:  
 16:  
 17:  
 18:  
 19:  
 20: Task2 Begins...
```


12. Event

```
TASK(Task2)
{
    EventMaskType mask;
    printfSerial("Task2 Begins...");
    printfSerial("Task2 Waits...");
    WaitEvent(Event1 | Event2);
    printfSerial("Task2 Wakes Up...");
    GetEvent(Task2, &mask);
    if (mask & Event1) {
        printfSerial("[Event1]");
        ClearEvent(Event1);
    }
    if (mask & Event2) {
        printfSerial("[Event2]");
        ClearEvent(Event2);
    }
}
```

```
    printfSerial("Task2 Finishes...");
    TerminateTask();
}
```

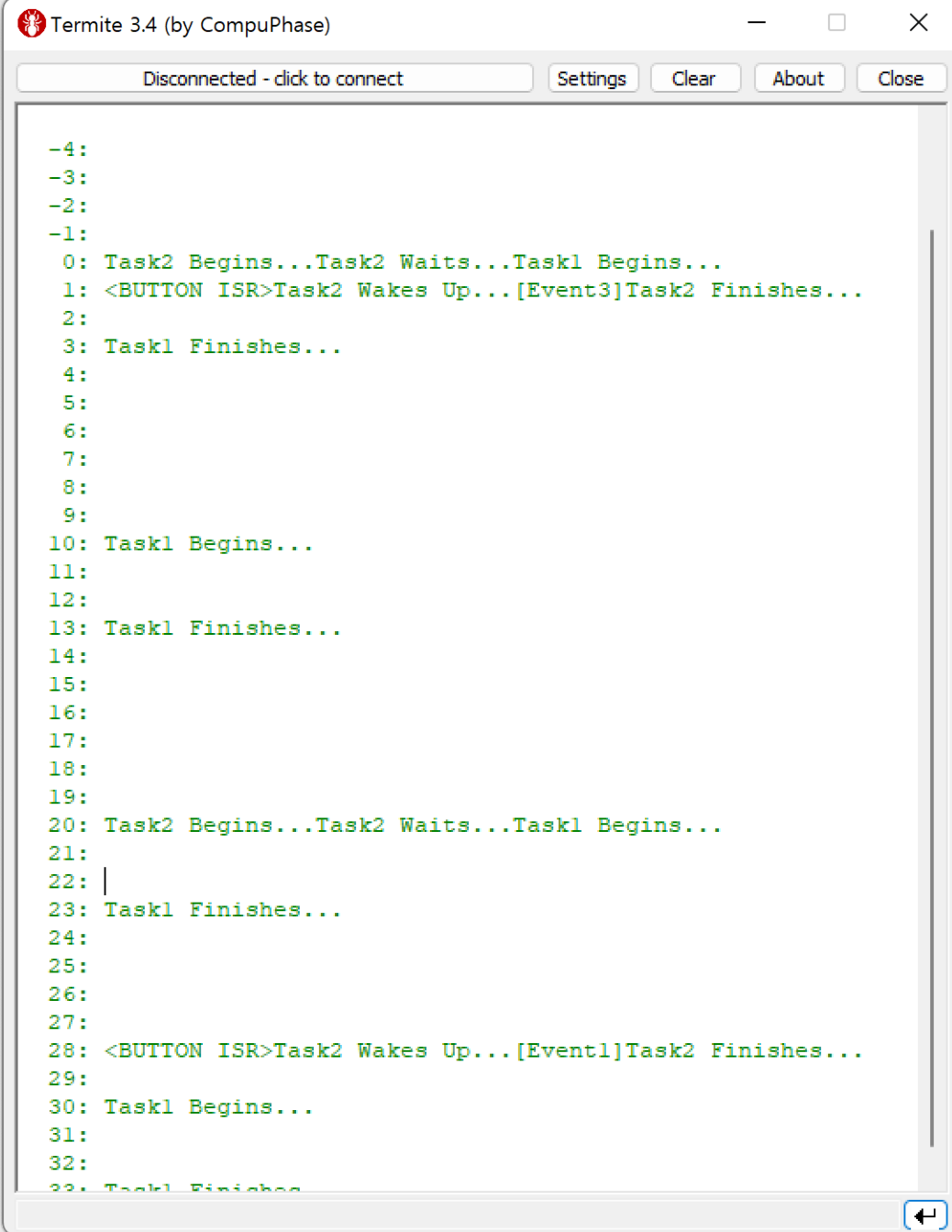
12. Event

```
ISR2(ButtonISR)
{
    DisableAllInterrupts();
    osEE_tc_delay(5000);
    printfSerial("<BUTTON ISR>");
    // read ADC value
    unsigned int a0 = readADCValue(3);
    if (a0 < 50) { // UP
        SetEvent(Task2, Event1);
    } else if (a0 < 200) { // DOWN
        SetEvent(Task2, Event2);
    } else if (a0 < 380) { // LEFT
    } else if (a0 < 520) { // RIGHT
    }
    osEE_tc_delay(3000);
    EnableAllInterrupts();
}
```

```
CPU_DATA = TRICORE {
    ID = 0x0;
    CPU_CLOCK = 200.0
    MULTI_STACK = TRUE;
};
...
EVENT Event1 { MASK = AUTO; };
EVENT Event2 { MASK = AUTO; };
...
TASK Task2 {
    PRIORITY = 2;
    STACK = PRIVATE {
        SIZE = 512;
    };
    SCHEDULE = FULL;
    EVENT = Event1;
    EVENT = Event2;
...
}
```

12. Event

- ClearEvent는 왜 필요한가?
- 우선순위 반대의 경우 스케줄링



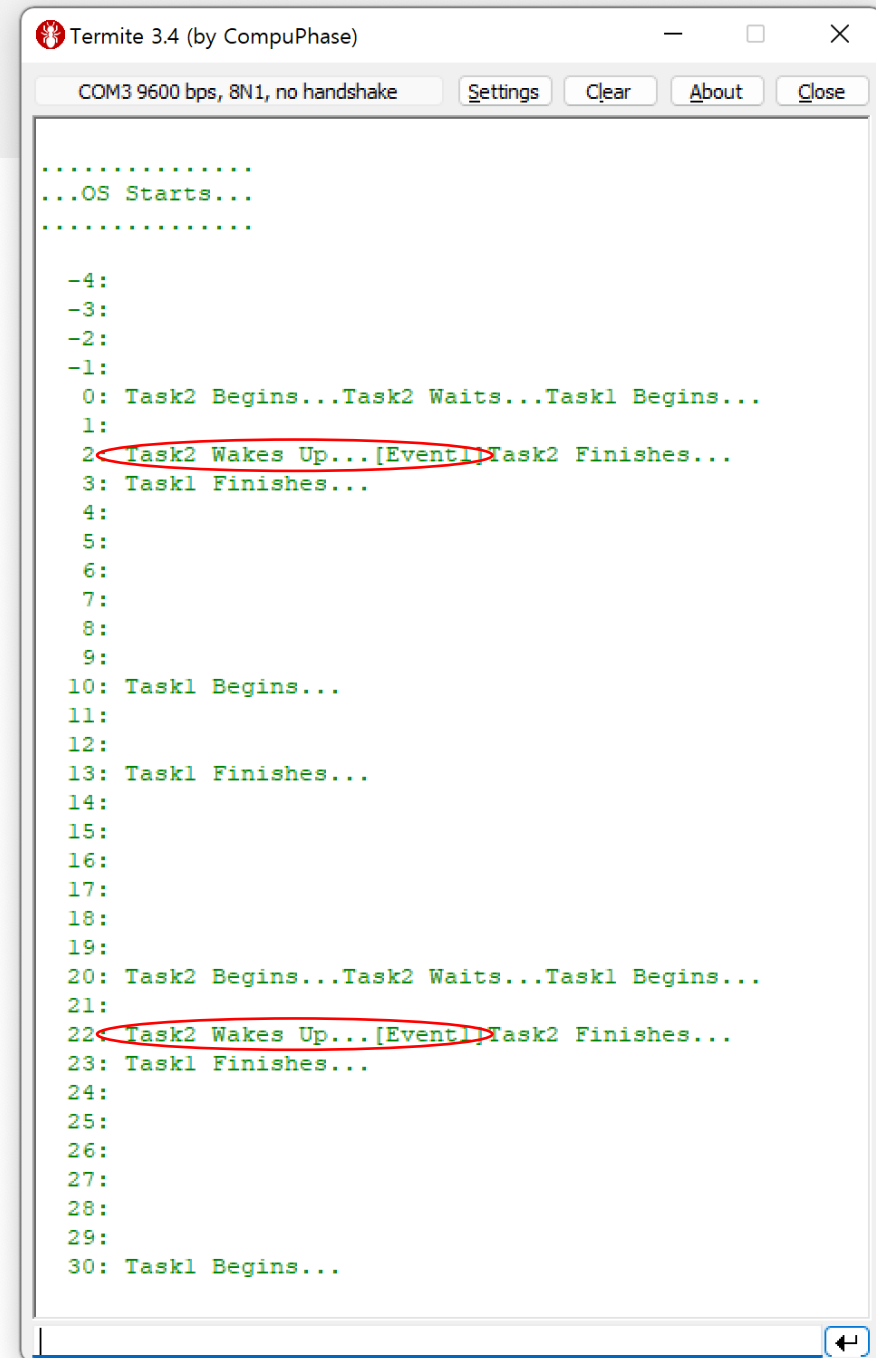
The screenshot shows a window titled "Termite 3.4 (by CompuPhase)". At the top, there is a status bar that says "Disconnected - click to connect" and four buttons: "Settings", "Clear", "About", and "Close". The main area of the window is a text field containing a log of events. The log consists of numbered lines from -4 to 33. Lines -4 to -1 are empty. Line 0 is "0: Task2 Begins...Task2 Waits...Task1 Begins...". Line 1 is "1: <BUTTON ISR>Task2 Wakes Up...[Event3]Task2 Finishes...". Lines 2 to 9 are empty. Line 10 is "10: Task1 Begins...". Lines 11 to 12 are empty. Line 13 is "13: Task1 Finishes...". Lines 14 to 19 are empty. Line 20 is "20: Task2 Begins...Task2 Waits...Task1 Begins...". Line 21 is empty. Line 22 has a vertical cursor. Line 23 is "23: Task1 Finishes...". Lines 24 to 27 are empty. Line 28 is "28: <BUTTON ISR>Task2 Wakes Up...[Event1]Task2 Finishes...". Lines 29 to 31 are empty. Line 30 is "30: Task1 Begins...". Line 31 is empty. Line 32 is empty. Line 33 is "33: Task1 Finishes...".

```
-4:
-3:
-2:
-1:
0: Task2 Begins...Task2 Waits...Task1 Begins...
1: <BUTTON ISR>Task2 Wakes Up...[Event3]Task2 Finishes...
2:
3: Task1 Finishes...
4:
5:
6:
7:
8:
9:
10: Task1 Begins...
11:
12:
13: Task1 Finishes...
14:
15:
16:
17:
18:
19:
20: Task2 Begins...Task2 Waits...Task1 Begins...
21:
22: |
23: Task1 Finishes...
24:
25:
26:
27:
28: <BUTTON ISR>Task2 Wakes Up...[Event1]Task2 Finishes...
29:
30: Task1 Begins...
31:
32:
33: Task1 Finishes...
```

13. Alarm SetEvent

- Alarm의 SetEvent Action

```
ALARM alarm3 {  
    COUNTER = mycounter;  
    ACTION = SETEVENT {  
        TASK = Task2;  
        EVENT = Event1;  
    };  
    AUTOSTART = TRUE {  
        ALARMTIME = 7;  
        CYCLETIME = 20;  
    };  
};
```



```
Termite 3.4 (by CompuPhase)  
COM3 9600 bps, 8N1, no handshake Settings Clear About Close  
.....  
...OS Starts...  
.....  
-4:  
-3:  
-2:  
-1:  
0: Task2 Begins...Task2 Waits...Task1 Begins...  
1:  
2: Task2 Wakes Up...[Event1] Task2 Finishes...  
3: Task1 Finishes...  
4:  
5:  
6:  
7:  
8:  
9:  
10: Task1 Begins...  
11:  
12:  
13: Task1 Finishes...  
14:  
15:  
16:  
17:  
18:  
19:  
20: Task2 Begins...Task2 Waits...Task1 Begins...  
21:  
22: Task2 Wakes Up...[Event1] Task2 Finishes...  
23: Task1 Finishes...  
24:  
25:  
26:  
27:  
28:  
29:  
30: Task1 Begins...
```

14. Hook

- OIL 파일 Hook 설정
- Alarm3 삭제
- 8번 예제의 printState() 추가

```
...  
    KERNEL_TYPE = OSEK {  
        CLASS = ECC2; // Default  
    };  
    STARTUPHOOK = TRUE;  
    SHUTDOWNHOOK = TRUE;  
    PRETASKHOOK = TRUE;  
    POSTTASKHOOK = TRUE;  
};  
...
```

14. Hook

- Task2 변경
- 8번 예제의 printState() 사용. 추가

```
TASK(Task2)
{
    printfSerial("Task2 Begins...");
    mdelay(3000);
    printfSerial("Task2 Finishes...");
    TerminateTask();
}
```

14. Hook

- StartupHook
- ShutdownHook

```
void StartupHook(void)
{
    printfSerial("..StartupHook..\n");
}

void ShutdownHook(StatusType Error)
{
    printfSerial("ShutdownHook...\n");
    printState(Task1);
    printState(Task2);
}
```

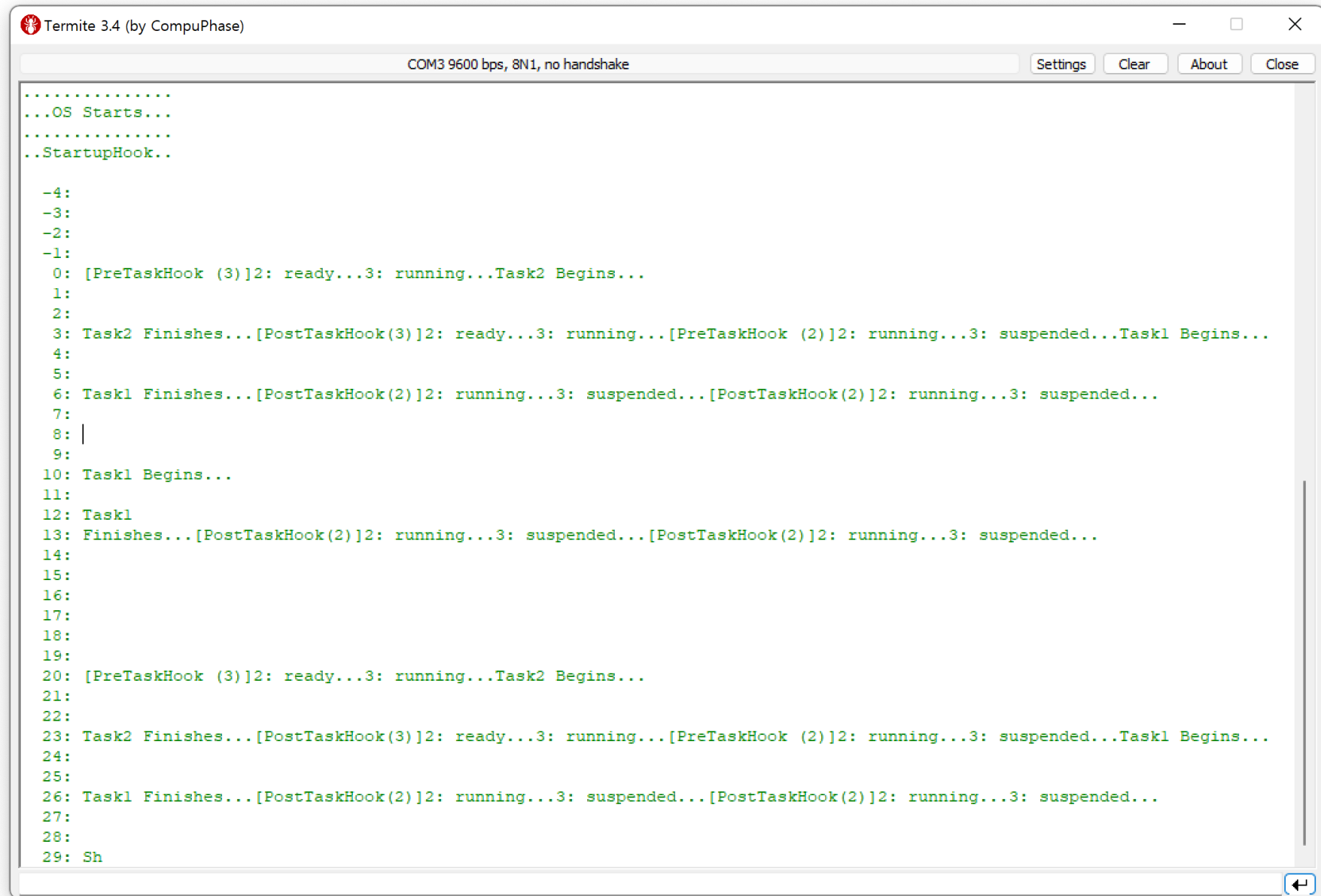
14. Hook

- PreTaskHook
- PostTaskHook

```
void PreTaskHook(void)
{
    TaskType id;
    GetTaskID(&id);
    printfSerial("[PreTaskHook(%d)]", id);
    printState(Task1);
    printState(Task2);
}

void PostTaskHook(void)
{
    TaskType id;
    GetTaskID(&id);
    printfSerial("[PostTaskHook(%d)]", id);
    printState(Task1);
    printState(Task2);
}
```


14. Hook



The screenshot shows a Termite 3.4 terminal window with a title bar that includes a red icon, the text "Termite 3.4 (by CompuPhase)", and standard window controls. Below the title bar is a status bar displaying "COM3 9600 bps, 8N1, no handshake" and four buttons: "Settings", "Clear", "About", and "Close". The main area of the terminal contains a series of green text lines representing system events and task states. The text is as follows:

```
.....  
...OS Starts...  
.....  
..StartupHook..  
  
-4:  
-3:  
-2:  
-1:  
 0: [PreTaskHook (3)]2: ready...3: running...Task2 Begins...  
 1:  
 2:  
 3: Task2 Finishes...[PostTaskHook(3)]2: ready...3: running...[PreTaskHook (2)]2: running...3: suspended...Task1 Begins...  
 4:  
 5:  
 6: Task1 Finishes...[PostTaskHook(2)]2: running...3: suspended...[PostTaskHook(2)]2: running...3: suspended...  
 7:  
 8: |  
 9:  
10: Task1 Begins...  
11:  
12: Task1  
13: Finishes...[PostTaskHook(2)]2: running...3: suspended...[PostTaskHook(2)]2: running...3: suspended...  
14:  
15:  
16:  
17:  
18:  
19:  
20: [PreTaskHook (3)]2: ready...3: running...Task2 Begins...  
21:  
22:  
23: Task2 Finishes...[PostTaskHook(3)]2: ready...3: running...[PreTaskHook (2)]2: running...3: suspended...Task1 Begins...  
24:  
25:  
26: Task1 Finishes...[PostTaskHook(2)]2: running...3: suspended...[PostTaskHook(2)]2: running...3: suspended...  
27:  
28:  
29: Sh
```

A vertical scrollbar is visible on the right side of the terminal window, and a blue arrow icon is located at the bottom right corner of the window frame.

15. Error Handling

- OIL 파일 설정

오류가 발생한 Service ID와
Parameter 정보 접근

```
...  
    KERNEL_TYPE = OSEK {  
        CLASS = ECC2; // Default  
    };  
    STARTUPHOOK = FALSE;  
    SHUTDOWNHOOK = FALSE;  
    PRETASKHOOK = FALSE;  
    POSTTASKHOOK = FALSE;  
    ERRORHOOK = TRUE;  
    USEGETSERVICEID = TRUE;  
    USEPARAMETERACCESS = TRUE;  
};  
...
```

15. Error Handling

```
ISR2(TimerISR)
{
    osEE_tc_stm_set_sr0_next_match( 1000000U );
    static long c = -4;
    TaskStateType s;
    if (c == 5) {
        GetTaskState(30, &s);
    }
    IncrementCounter(mycounter);
    printfSerial("\n%4ld: ", c++);
}

void ErrorHook(StatusType error)
{
    printfSerial("[ErrorHook: error = %d, service = %d, TaskID = %d]",
        error,
        OSErrorGetServiceId(),
        OSError_GetTaskState_TaskID());
}
```

Parameter 정보 접근 매크로
(ee_oo_api_osek.h)

15. Error Handling

```
E_OK,                /* ((StatusType)0) */
E_OS_ACCESS,         /* ((StatusType)1) */
E_OS_CALLEVEL,       /* ((StatusType)2) */
E_OS_ID,             /* ((StatusType)3) */
E_OS_LIMIT,          /* ((StatusType)4) */
E_OS_NOFUNC,         /* ((StatusType)5) */
E_OS_RESOURCE,       /* ((StatusType)6) */
E_OS_STATE,          /* ((StatusType)7) */
E_OS_VALUE,          /* ((StatusType)8) */
E_OS_SERVICEID,      /* ((StatusType)9) */
E_OS_ILLEGAL_ADDRESS, /* ((StatusType)10) */
...
```

ee_api_types.h

```
OSServiceId_ActivateTask      = (0),
OSServiceId_TerminateTask     = (2),
OSServiceId_ChainTask         = (4),
OSServiceId_Schedule          = (6),
OSServiceId_GetTaskID         = (8),
OSServiceId_GetTaskState      = (10)
,
OSServiceId_DisableAllInterrupts = (12)
,
OSServiceId_EnableAllInterrupts = (14)
,
OSServiceId_SuspendAllInterrupts = (16)
,
OSServiceId_ResumeAllInterrupts = (18)
,
OSServiceId_SuspendOSInterrupts = (20)
,
OSServiceId_ResumeOSInterrupts = (22)
...
```

15. Error Handling

```
/**
 * \brief This macro returns the TaskID parameter passed to ActivateTask().
 * \ingroup primitives-hook
 */
#define OSErrror_ActivateTask_TaskID()\
    ((TaskType)osEE_get_api_param1().num_param)

/**
 * \brief This macro returns the TaskID parameter passed to ChainTask().
 * \ingroup primitives-hook
 */
#define OSErrror_ChainTask_TaskID()\
    ((TaskType)osEE_get_api_param1().num_param)

/**
 * \brief This macro returns the TaskID parameter passed to GetTaskID().
 * \ingroup primitives-hook
 */
#define OSErrror_GetTaskID_TaskID()\
    ((TaskRefType)osEE_get_api_param1().p_param)
```

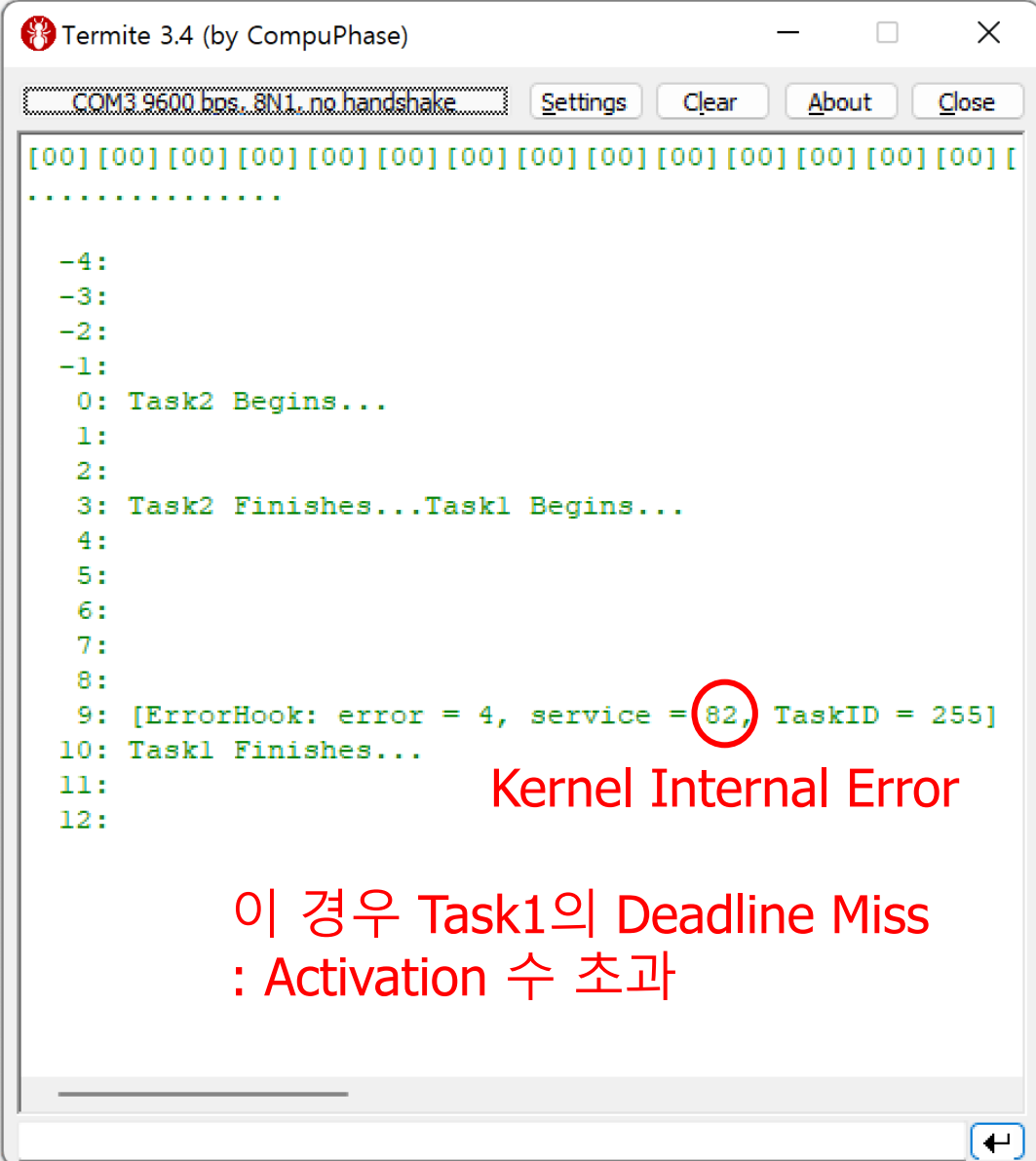
ee_oo_api_osek.h

16. Deadline Miss

```
ISR2(TimerISR)
{
osEE_tc_stm_set_sr0_next_match( 1000000U );
    static long c = -4;

    IncrementCounter(counter1);
    printfSerial("\n%4ld: ", c++);
}

TASK(Task1)
{
    TaskType id;
    printfSerial("Task1 Begins...");
    mdelay(7000);
    printfSerial("Task1 Finishes...");
    TerminateTask();
}
```



The screenshot shows the Termite 3.4 terminal window. The title bar reads "Termite 3.4 (by CompuPhase)". The status bar at the top indicates "COM3 9600 bps, 8N1, no handshake". The terminal output shows a sequence of events: lines -4 to -1 are blank; line 0 says "Task2 Begins..."; line 1 is blank; line 2 is blank; line 3 says "Task2 Finishes...Task1 Begins..."; line 4 is blank; line 5 is blank; line 6 is blank; line 7 is blank; line 8 is blank; line 9 shows an error: "[ErrorHook: error = 4, service = 82, TaskID = 255]"; line 10 says "Task1 Finishes..."; line 11 is blank; line 12 is blank. The number 82 in the error message is circled in red. Below the terminal output, the text "Kernel Internal Error" is written in red. At the bottom right, there is a red Korean text annotation: "이 경우 Task1의 Deadline Miss : Activation 수 초과".

```
COM3 9600 bps, 8N1, no handshake Settings Clear About Close

[00] [00] [00] [00] [00] [00] [00] [00] [00] [00] [00] [00] [00] [00] [
.....

-4:
-3:
-2:
-1:
 0: Task2 Begins...
 1:
 2:
 3: Task2 Finishes...Task1 Begins...
 4:
 5:
 6:
 7:
 8:
 9: [ErrorHook: error = 4, service = 82, TaskID = 255]
10: Task1 Finishes...
11:
12:
```

Kernel Internal Error

이 경우 Task1의 Deadline Miss
: Activation 수 초과

Questions

