Real-Time Operating System (Day 2 Lab)

Jong-Chan Kim

Graduate School of Automotive Engineering

KMU 국민대학교 KOOKMIN UNIVERSITY

09. ButtonISR

- ButtonISR 추가 (Button 입력에 반응하여 Task Activate)

```
ISR2(ButtonISR)
{
    unsigned int a0;
    DisableAllInterrupts();
    osEE_tc_delay(5000);
    a0 = readADCValue(3);
    if (a0 < 500) {          /* TOP */
        printfSerial("<BUTTON:T>");
        ActivateTask(Task1);
    } else if (a0 < 1200) { /* DOWN */
        printfSerial("<BUTTON:D>");
        ActivateTask(Task2);
    } else if (a0 < 1600) { /* LEFT */
        printfSerial("<BUTTON:L>");
    } else if (a0 < 2200) { /* RIGHT */
        printfSerial("<BUTTON:R>");
    } else {
        printfSerial("<BUTTON:?>");
    }
    osEE_tc_delay(3000);
    EnableAllInterrupts();
}
```

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

09. ButtonISR

- Task1, Task2 원상복귀

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

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

09. ButtonISR

```
..............
...OS Starts...
..............

 -4:
 -3:
 -2: <BUTTON:T>Task1 Begins...
 -1:
  0:
  1: Task1 Finishes...
  2:
  3:
  4: <BUTTON:D>Task2 Begins...
  5:
  6:
  7: Task2 Finishes...
  8:
  9:
 10:
 11: <BUTTON:L><BUTTON:L>
 12:
 13:
 14: <BUTTON:R>
 15:
 16:
```

Top 버튼 신호에 따라
Task1 실행

Down 버튼 신호에 따라
Task2 실행

09. ButtonISR

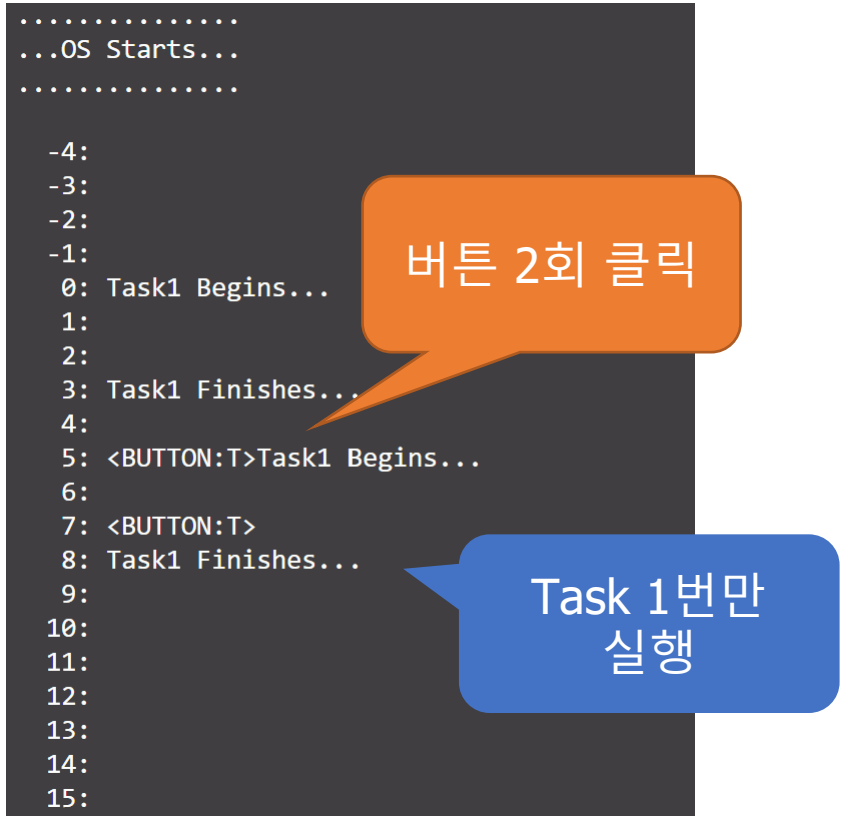
- ButtonISR에서 30초 mdelay 실행하면?

ISR2는 OS와 상호작용, ISR 안에서
오래 걸리는 동작을 하면 Task도 멈추고 버튼도 멈춘다. 시스템 전체가 정지 상태에 빠진다.

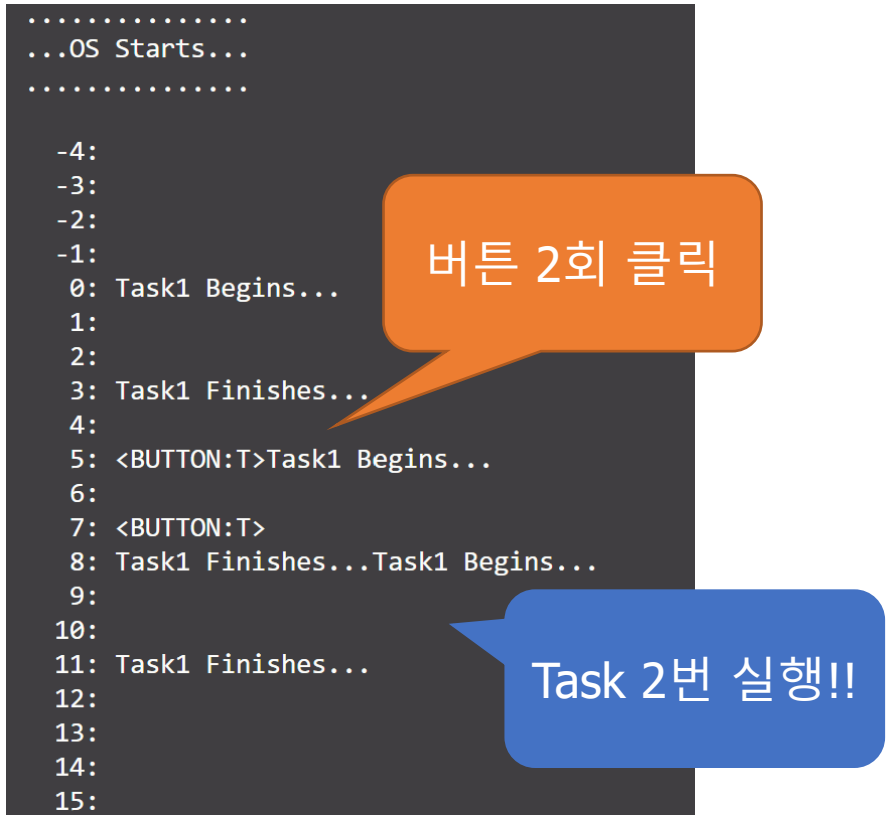
09. ButtonISR

- 중복 Activation 하려면?
  - ACTIVATION = 1 → 2로 수정 필요

- ACTIVATION = 1



- ACTIVATION = 2



10-1. 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;
    };
};
```

10-1. Alarm

- TimerISR에서
  - ActivateTask(Task1) 삭제
  - mycounter 증가

```
ISR2(TimerISR)
{
    static long c = -4;
    osEE_tc_stm_set_sr0_next_match(1000000U);
    if (c == 0)
        ActivateTask(Task1);
    IncrementCounter(mycounter);
    printfSerial("\n%4ld: ", c++);
}
```

10-1. Alarm

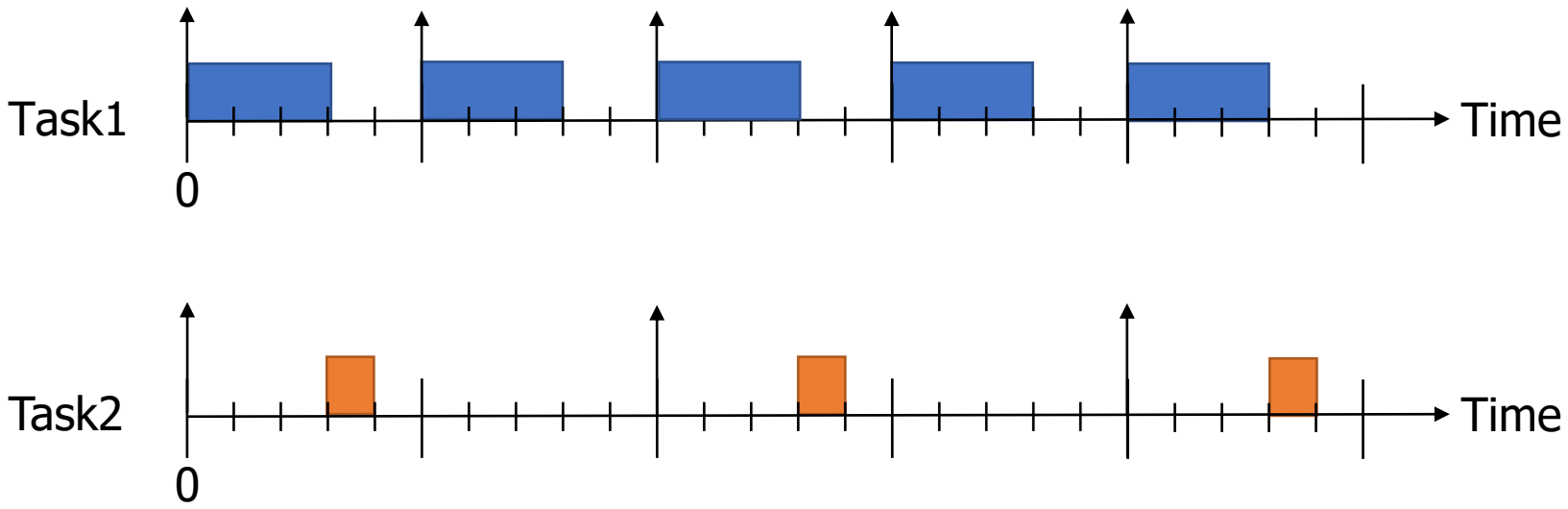
- Alarm1: Task1 실행
  - OS 시작 후 5ms 뒤 첫 실행, 이후 10ms 주기
- Alarm2: Task2 실행
  - OS 시작 후 5ms 뒤 첫 실행, 이후 20ms 주기

우선순위에 따라,
Task2 먼저 실행 후
Task1 실행

```
..............
...OS Starts...
..............

 -4:
 -3:
 -2:
 -1:
  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:
 29:
```

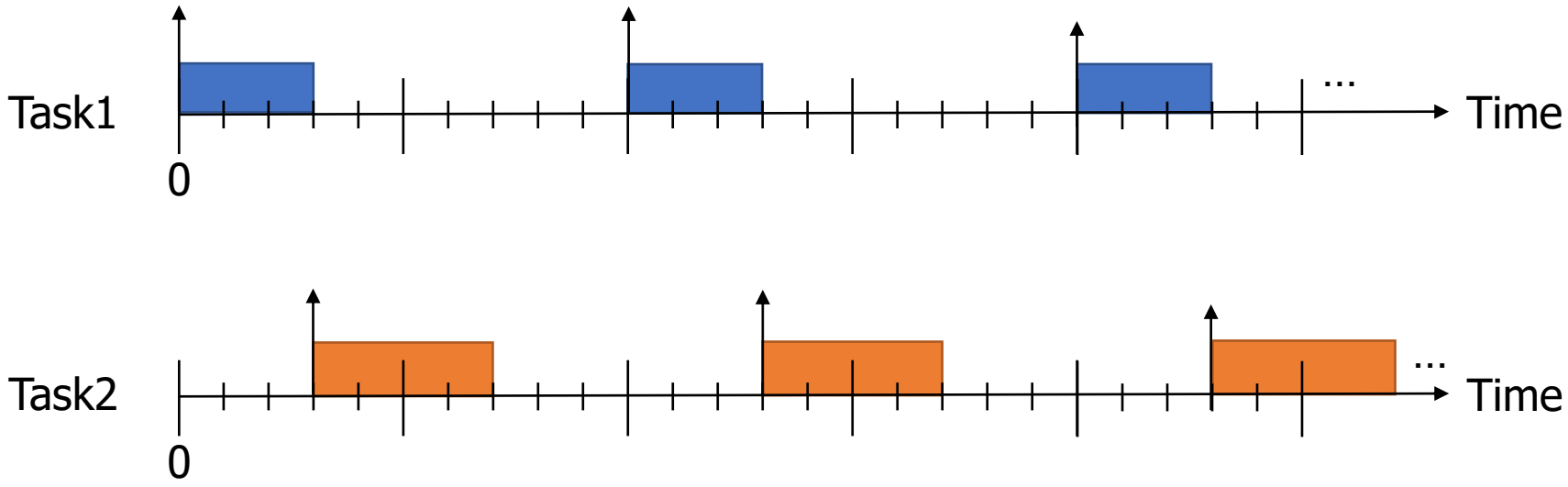
10-2. Alarm

- [예제] 알람을 활용하여 아래 Timing diagram 구현하기



10-3. Alarm

- [예제] 06-3. Periodic Tasks
  - Task1: 높은 우선순위, 실행 시간 3초
  - Task2: 낮은 우선순위, 실행 시간 4초
  - AUTOSTART = False로 수정하고 동일하게 구현
  - 타이머 인터럽트가 아닌 알람을 활용하여 10초마다 주기적으로 실행



11. Alarm Callback

- 콜백 함수 등록

```
ALARMCALLBACK(MyCallback)
{
    printfSerial("<MyCallback>");
}
```

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

12. Event

```
ISR2(ButtonISR)
{
    unsigned int a0;
    DisableAllInterrupts();
    osEE_tc_delay(5000);
    a0 = readADCValue(3);
    if (a0 < 500) {
        printfSerial("<BUTTON:T>");
        SetEvent(Task2, Event1);
    } else if (a0 < 1200) {
        printfSerial("<BUTTON:D>");
        SetEvent(Task2, Event2);
    } else if (a0 < 1600) {
        printfSerial("<BUTTON:L>");
    } else if (a0 < 2200) {
        printfSerial("<BUTTON:R>");
    } else {
        printfSerial("<BUTTON:?>");
    }
    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 = 1024;
    };
    SCHEDULE = FULL;
…
    EVENT = Event1;
    EVENT = Event2;
…
}
```

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

- OS 시작 시 알람 콜백과 Task2가 먼저 실행 → Task2는 이벤트 대기 상태로 진입
- 버튼 입력으로 Event1이 발생하면 Task2가 깨어나 이벤트를 처리하고 종료됨

```
.............
...OS Starts...
.............

 -4:
 -3:
 -2:
 -1: <MyCallback>
  0: Task2 Begins...Task2 Waits...Task1 Begins...
  1:
  2:
  3: Task1 Finishes...
  4:
  5:
  6:
  7:
  8:
  9:
 10: Task1 Begins...<BUTTON:T>Task2 Wakes Up...[Event1]Task2 Finishes...
 11:
 12:
 13: Task1 Finishes...
```

우선순위에 따라,
Task2 먼저 실행 후
Task1 실행

12. Event

- 우선순위 반대의 경우 스케줄링

```
...............
...OS Starts...
...............

  -4:
  -3:
  -2:
  -1: <MyCallback>
   0: Task1 Begins...
   1:
   2:
   3: Task1 Finishes...Task2 Begins...Task2 Waits...
   4:
   5:
   6:
   7:
   8:
   9:
  10: Task1 Begins...<BUTTON:T>
  11:
  12:
  13: Task1 Finishes...Task2 Wakes Up...[Event1]Task2 Finishes...
  14: <MyCallback>
  15:
  16:
```

우선순위에 따라,
Task1 먼저 실행 후
Task2 실행

13. Alarm SetEvent

- Alarm을 이용한 주기적인 SetEvent Action

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

13. Alarm SetEvent

- Alarm을 이용한 주기적인 SetEvent Action

```
.............
...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:
```

Alarm에 의해 주기적으로 Event가 발생하여 Task2 깨어남

14. Hook

- OIL Hook 사용 설정

```
    KERNEL_TYPE = OSEK {
        CLASS = ECC2; // Default
    };

    STARTUPHOOK = TRUE;
    SHUTDOWNHOOK = TRUE;
    PRETASKHOOK = TRUE;
    POSTTASKHOOK = TRUE;
};
```

14. Hook

- Task2 원상복귀

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

- ButtonISR에 ShutdownOS 추가

```
ISR2(ButtonISR)
{
    …
    if (a0 < 500) {
        printfSerial("<BUTTON:T>");
        SetEvent(Task2, Event1);
    } else if (a0 < 1200) {
        printfSerial("<BUTTON:D>");
        SetEvent(Task2, Event2);
    } else if (a0 < 1600) {
        printfSerial("<BUTTON:L>");
    } else if (a0 < 2200) {
        printfSerial("<BUTTON:R>");
        ShutdownOS(1);
    }
    …
}
```

14. Hook

- StartupHook
- ShutdownHook

```
void StartupHook(void)
{
    printfSerial("[StartupHook]");
}

void ShutdownHook(StatusType Error)
{
    printfSerial("[ShutdownHook]");
}
```

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

```
..............
...OS Starts...
..............
[StartupHook]
  -4:
  -3:
  -2:
  -1:
   0: [PreTaskHook (4)]3: ready...4: running...Task2 Begins...
   1:
   2:
   3: Task2 Finishes...[PostTaskHook(4)]3: ready...4: running...[PreTaskHook (3)]3: running...4: suspended...Task1 Begins...
   4:
   5:
   6: Task1 Finishes...[PostTaskHook(3)]3: running...4: suspended...[PostTaskHook(3)]3: running...4: suspended...
   7:
   8:
   9:
  10: Task1 Begins...
  11:
  12:
  13: Task1 Finishes...[PostTaskHook(3)]3: running...4: suspended...[PostTaskHook(3)]3: running...4: suspended...
  14:
  15:
  16:
  17:
  18:
  19:
```

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

고의 에러
잘못된 ID = 30

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 OSError_ActivateTask_TaskID()\
  ((TaskType)osEE_get_api_param1().num_param)

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

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

ee_oo_api_osek.h

15. Error Handling

```
...............
...OS Starts...
...............

  -4:
  -3:
  -2:
  -1:
   0: Task2 Begins...
   1:
   2:
   3: Task2 Finishes...Task1 Begins...
   4: [ErrorHook: error = 3, service = 10, TaskID = 30]
   5:
   6: Task1 Finishes...
   7:
   8:
   9:
  10: Task1 Begins...
  11:
  12:
  13: Task1 Finishes...
  14:
  15:
  16:
  17:
  18: <BUTTON:R>
```

→ Shutdown

고의 에러
잘못된 ID = 30

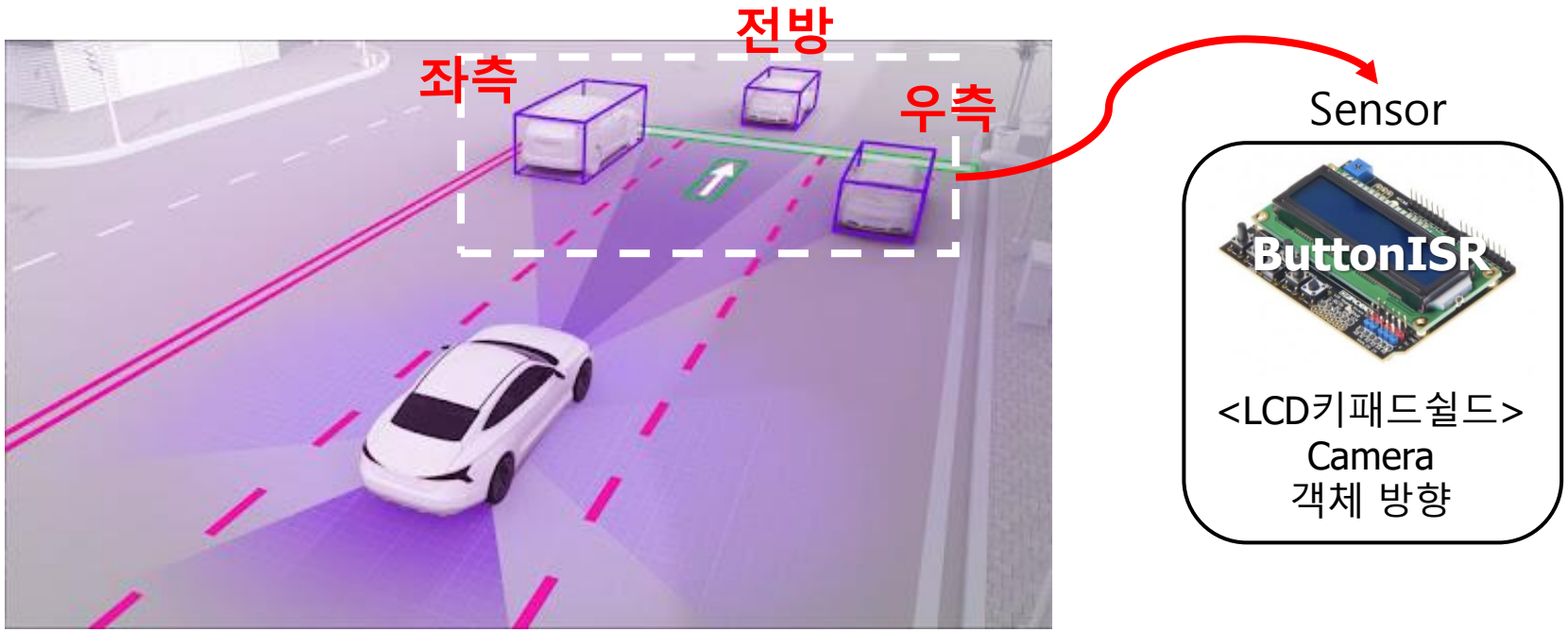
16. Deadline Miss

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

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

```
.............
...OS Starts...
.............

 -4:
 -3:
 -2:
 -1:
  0: Task2 Begins...
  1:
  2:
  3: Task2 Finishes...Task1 Begins...
  4:
  5:
  6:
  7:                    Kernel Internal Error
  8:
  9: [ErrorHook: error = 4, service = 82, TaskID = -1]
 10: Task1 Finishes...
 11:     이 경우 Task1의 Deadline Miss
 12:     : Activation 수 초과
 13:
```

Team Project

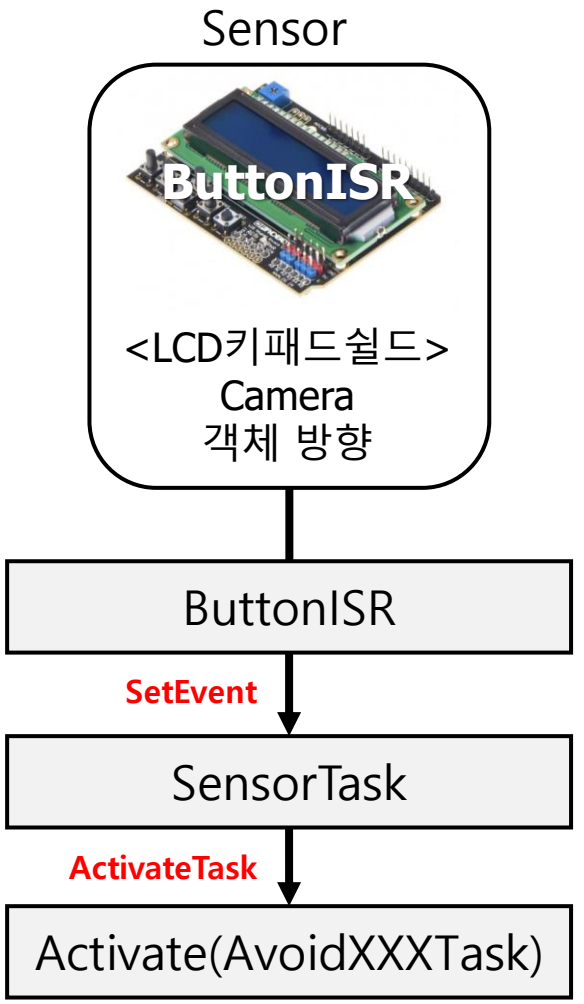
- RTOS 기반 자율주행 장애물 감지 및 회피 시스템
  - ButtonISR로 전방 객체의 위치를 파악한 후 적절한 Task를 실행
  - ✓ RTOS에서 ISR, Task, Event 구조 이해
  - ✓ 자율주행 시스템처럼 Event 기반 판단 및 회피 전략 구성



Team Project

- 시스템 시나리오
  1. 자율주행 차량이 주행 중
  2. 전방 / 좌측 / 우측에 장애물이 감지
  3. 감지 Event에 따라 적절한 Task 실행

Task 이름	우선순위	기능 설명
SensorTask	3	방향 판단, AvoidTask 결정
AvoidFrontTask	4	전방 회피
AvoidLeftTask	2	좌측 회피
AvoidRightTask	2	우측 회피



Questions