



Real-Time Operating System (Day 3 Lab)

Jong-Chan Kim

Graduate School of Automotive Engineering



국민대학교
KOOKMIN UNIVERSITY

17. Loss

- 공유 자원 접근으로 인한 Data Loss 실험
- Task1: Low priority, AUTOSTART
- Task2: High priority, 1 ms 주기 실행

bsw.cpp

```
4  
5 #define TIMER1_US    1000U    /* 1 msec */  
6
```

Task1

```
2  
3 volatile unsigned long shared = 0;  
4
```

- 공유 전역 변수

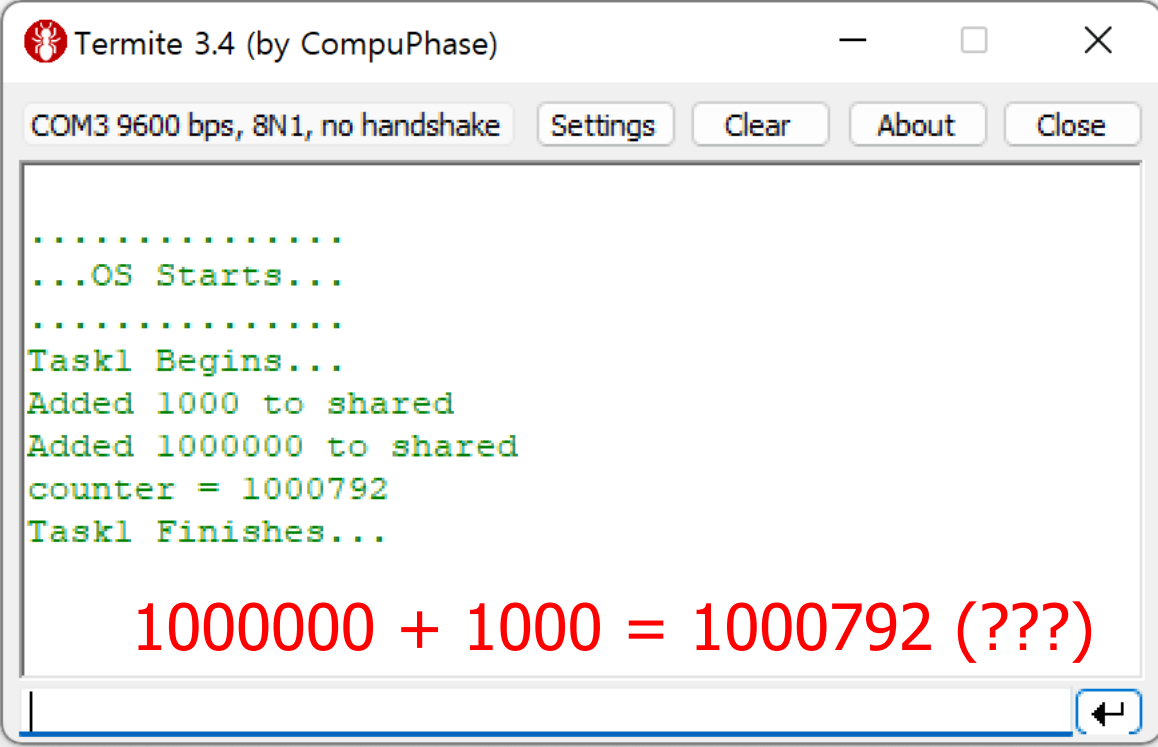
- Task1에서 100만번 shared++

```
12 unsigned long i;  
13 printfSerial("Task1 Begins...\n");  
14 for (i = 0; i < 1000000; i++) {  
15     shared++;  
16 }
```

- Task1이 한 번 실행되는 동안 Task2 반복 실행하며 shared ++
- 양 쪽 태스크에서 더한 숫자가 모두 유지가 되는지 확인

17. Loss

- Resource를 이용해서 Integrity Loss 문제 해결 필요



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 terminal output shows a sequence of events: "...OS Starts...", "Task1 Begins...", "Added 1000 to shared", "Added 1000000 to shared", "counter = 1000792", and "Task1 Finishes...". Below the terminal output, a red text overlay reads "1000000 + 1000 = 1000792 (???)".

```
.....  
...OS Starts...  
.....  
Task1 Begins...  
Added 1000 to shared  
Added 1000000 to shared  
counter = 1000792  
Task1 Finishes...
```

1000000 + 1000 = 1000792 (???)

17. Loss

- OSEK의 RESOURCE 기능을 이용하여 Data Loss 문제 해결

```
30      GetResource(S1);  
31      shared++;  
32      ReleaseResource(S1);
```

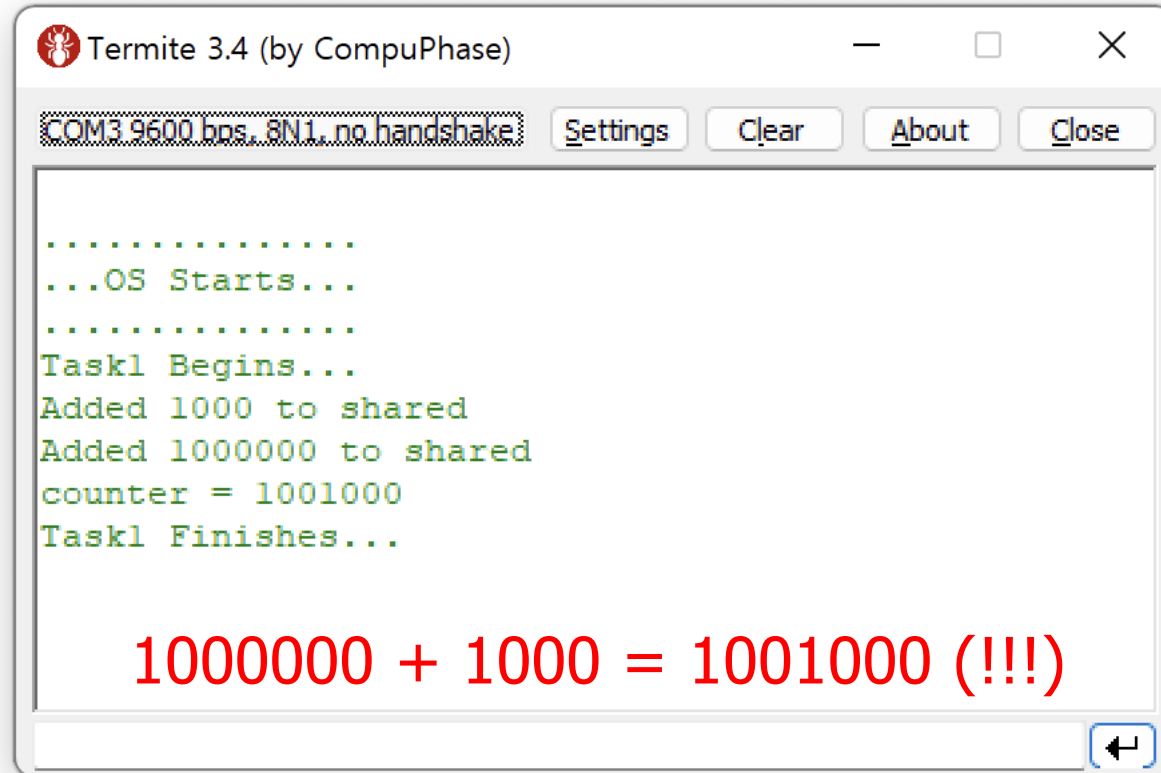
```
RESOURCE S1 {  
    RESOURCEPROPERTY = STANDARD;  
};
```

...

```
TASK Task1 {  
    PRIORITY = 1;  
    STACK = SHARED;  
    SCHEDULE = FULL;  
    AUTOSTART = TRUE;  
    ACTIVATION = 1;  
    RESOURCE = S1;  
};
```

17. Loss

- Data Integrity 문제 해결



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 terminal text is as follows:

```
.....  
...OS Starts...  
.....  
Task1 Begins...  
Added 1000 to shared  
Added 1000000 to shared  
counter = 1001000  
Task1 Finishes...
```

At the bottom of the terminal, the equation $1000000 + 1000 = 1001000 (!!!)$ is displayed in red text, highlighting a discrepancy or error in the data integrity check.

18. Mutex

Waiting/Wakeup 을 위
해 Event 지정 필요

• mutex.c

```
#include "ee.h"  
#include "bsw.h"  
#include "mutex.h"
```

```
void InitMutex(MutexType *mutex, EventMaskType event)  
{  
    mutex->flag = UNLOCKED;  
    mutex->waiting_task = 0;  
    mutex->event = event;  
}
```

```
void GetMutex(MutexType *mutex)  
{  
    if (mutex->flag == LOCKED) {  
        printfSerial("  --> BLock");  
        GetTaskID(&(mutex->waiting_task));  
        WaitEvent(mutex->event);  
    }  
    mutex->flag = LOCKED;  
}
```

```
void ReleaseMutex(MutexType *mutex)  
{  
    if (mutex->flag == LOCKED) {  
        mutex->flag = UNLOCKED;  
        if (mutex->waiting_task != 0) {  
            SetEvent(mutex->waiting_task, mutex->event);  
        }  
    }  
}
```

**PCP 없이 Mutex 사용할 경우 문제점을
확인하기 위한 Dummy 구현**

18. Mutex

- mutex.h

```
#ifndef MUTEX_H_
#define MUTEX_H_

#define LOCKED    1
#define UNLOCKED  0

typedef struct _MutexType {
    int flag;
    EventMaskType event;
    TaskType waiting_task;
} MutexType;

void InitMutex(MutexType *mutex, EventMaskType event);

void GetMutex(MutexType *mutex);

void ReleaseMutex(MutexType *mutex);

#endif /* MUTEX_H_ */
```

18. Mutex

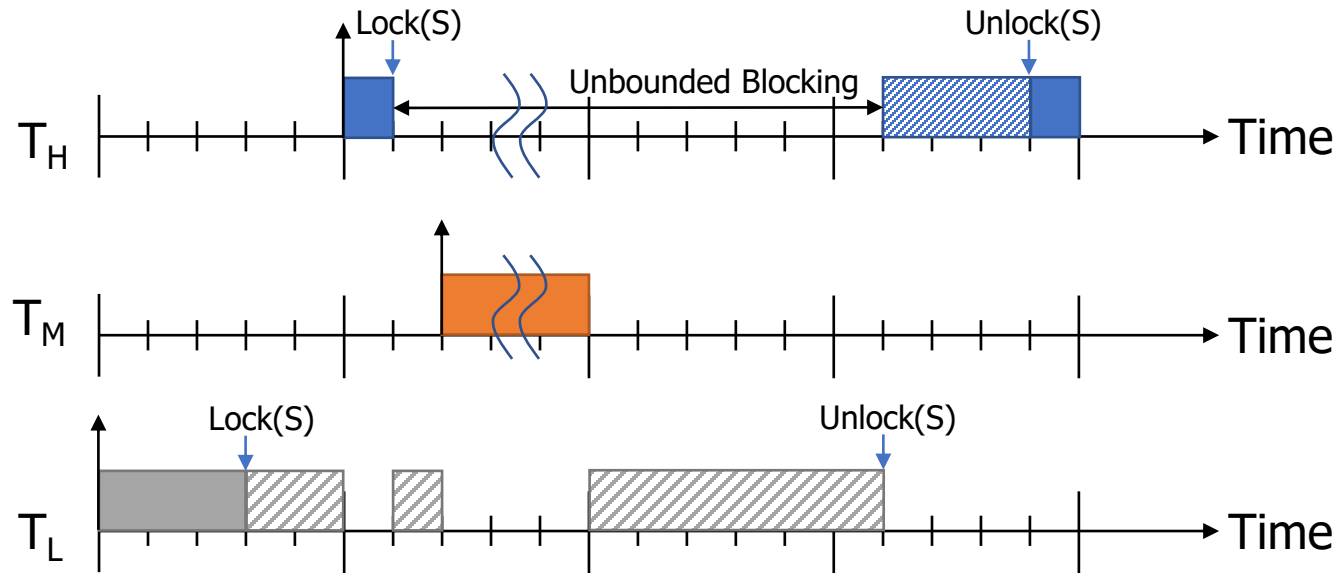
- Mutex 선언
- Timer ISR 이용
 - Mutex 초기화
 - Task Activation
- Mutex의 동작 확인

```
MutexType s1;

ISR2(TimerISR)
{
    static long c = -5;
    printfSerial("\n%4ld: ", ++c);
    if(c == -4) {
        InitMutex(&s1, Event1);
    }
    else if (c == 0) {
        ActivateTask(TaskL);
    }
    else if (c == 5) {
        ActivateTask(TaskH);
    }
}
```

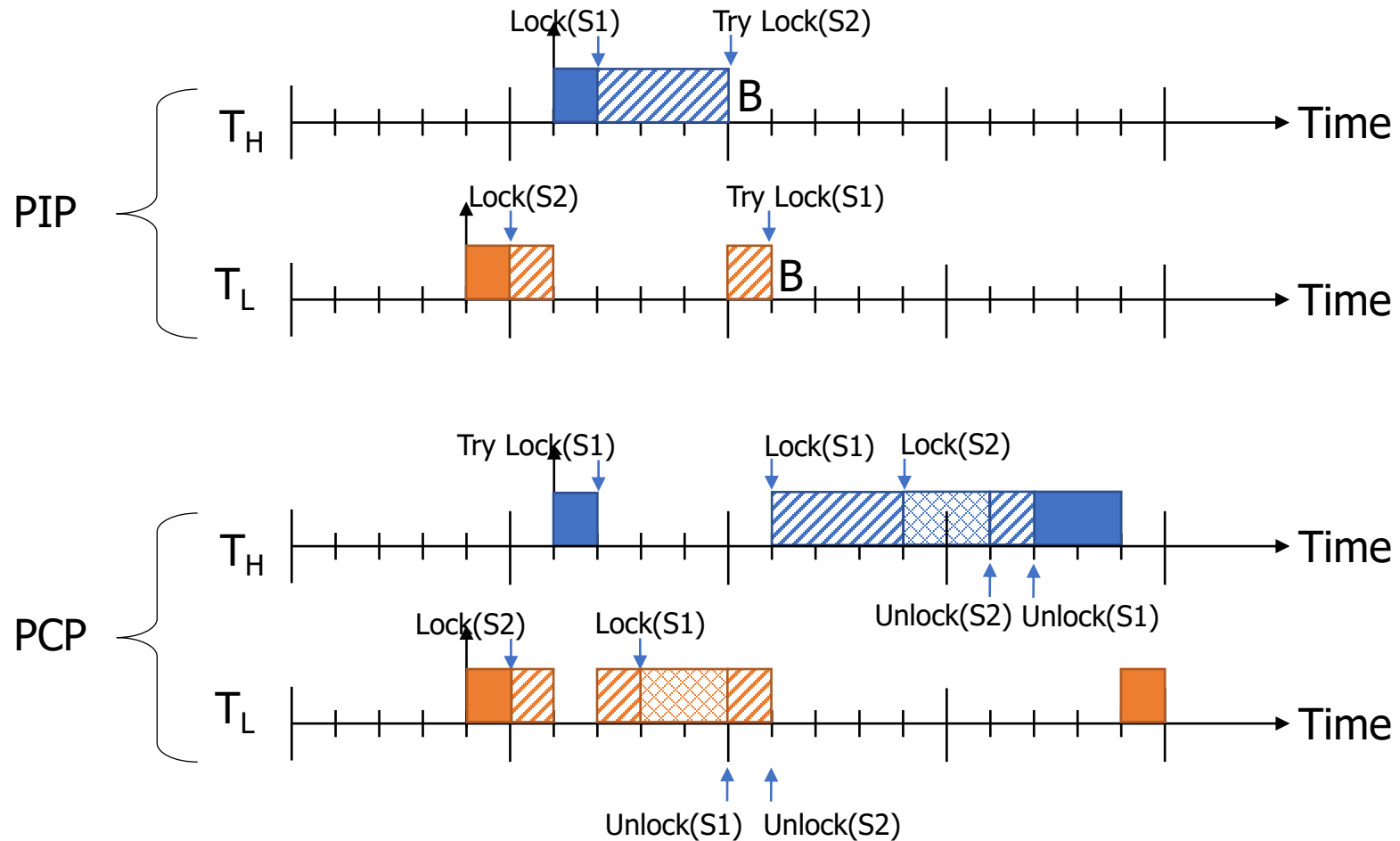

19. Priority Inversion

- 아래 스케줄 재현하기
- PCP가 적용된 RESOURCE를 이용하여 스케줄 변화 확인



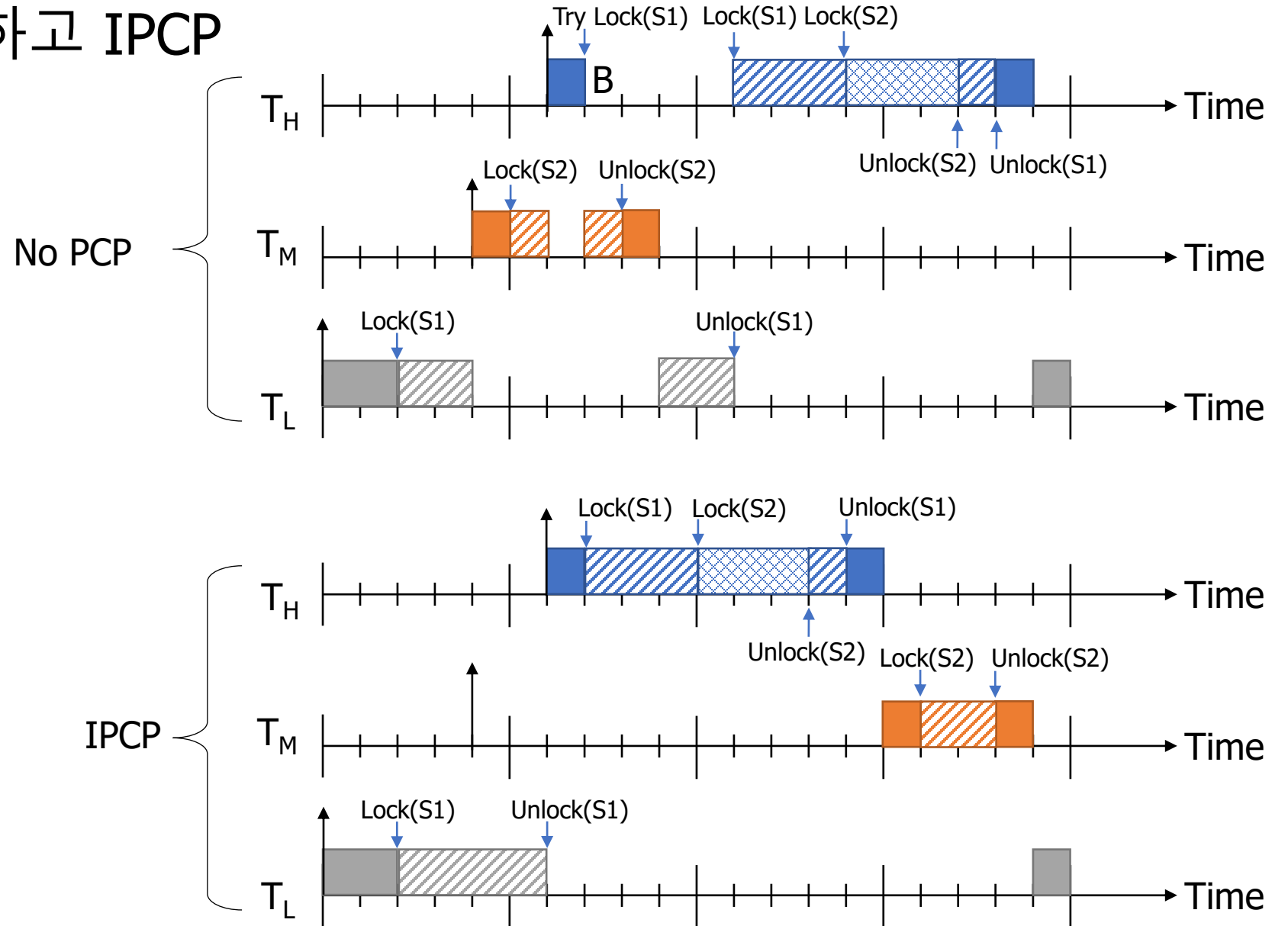
20. Deadlock

- Deadlock 재현하기
- PCP가 적용된 RESOURCE를 이용하여 Deadlock 해결



21. IPCP

- 우측 스케줄 재현하고 IPCP 동작 확인하기



Questions

