

실험.실습 보고서

(3) 반

조원: (학번:12131489,이름:김영훈)

1. 제목

3주차 실습과제 - MicroC/OS-II 프로그램 시작 및 코드 작성

2. 목적

MicroC/OS-II 프로그램을 Dos와 컴파일러를 이용하여 컴퓨터에서 미리 동작을 시켜본다.

MicroC/OS-II에서 사용되는 언어를 익히고, 여러 Task를 만들어 실시간으로 어떻게 동작되는지 원리를 익힌다.

3. 실습에 필요한 기초지식

OS에 대한 기본 지식이 필요하다. Task, scheduling 등 여러 개념들이 필요하다. 또한 c언어로 작성되어 있기 때문에 c언어에 대해 알아야 한다. 그리고 MicroC/OS-II에서만 사용되는 여러 함수들에 대해 어떻게 쓰는지에 대해 익힐 필요가 있다.

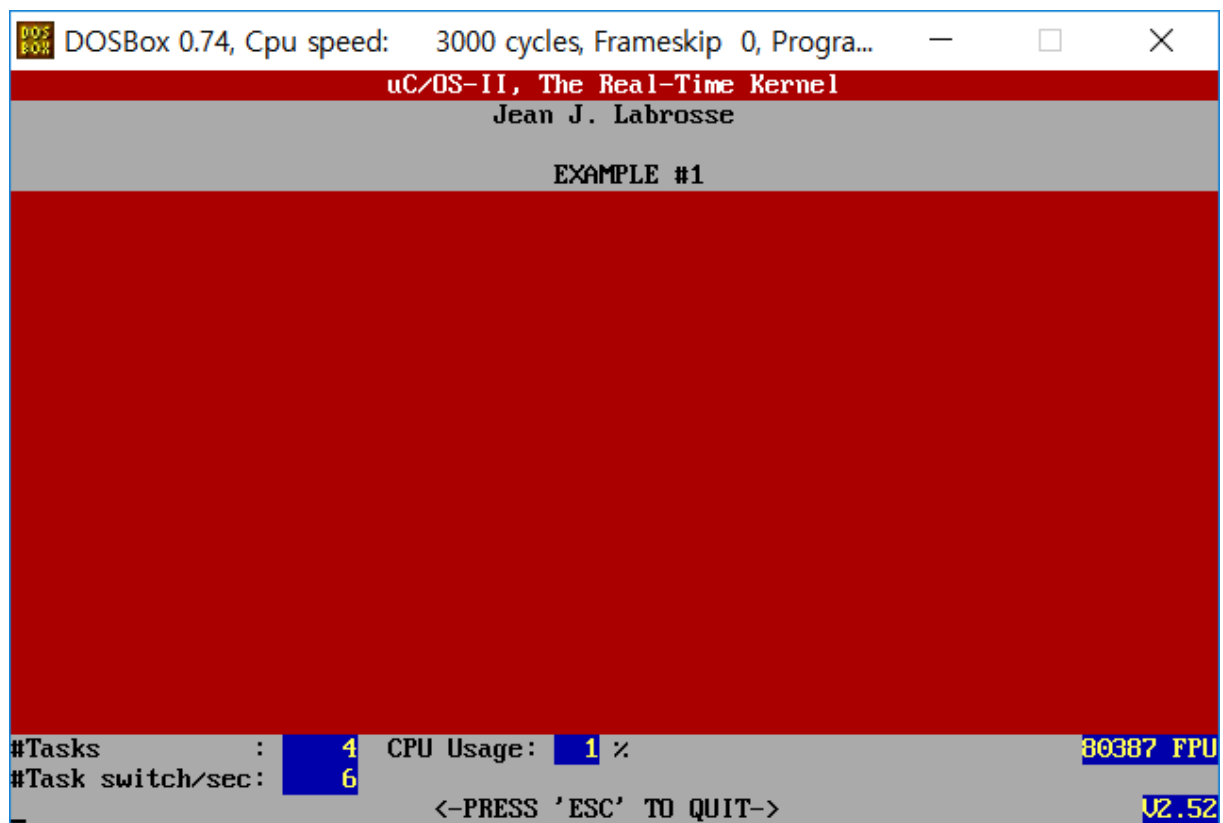
4. 실습 절차, 내용 및 결과

프로그램 실행 시 main 함수부터 시작된다. main 함수에서 TaskStart 함수의 Task를 생성한다. TaskStart 함수에서는 TaskStartCreateTasks 함수를 호출하고 user로부터 esc 키값을 입력 받았을 때 종료되도록 대기한다.

TaskStartCreateTasks 함수에서는 N_TASKS 변수의 값만큼 task를 생성한다. 또한 각 task가 생성될 때 마다 다음 task는 1초의 delay를 가지고 생성된다.

각 task의 번호에 따라 화면의 색을 다르게 바꾸는데, 1번은 red, 2번은 blue, 3번은 brown, 3번은 green으로 바꾼다. 화면을 바꾼 후 N_TASKS의 값만큼 지연을 시켜 waiting queue에 들어간다.

(1) Red만 칠하기

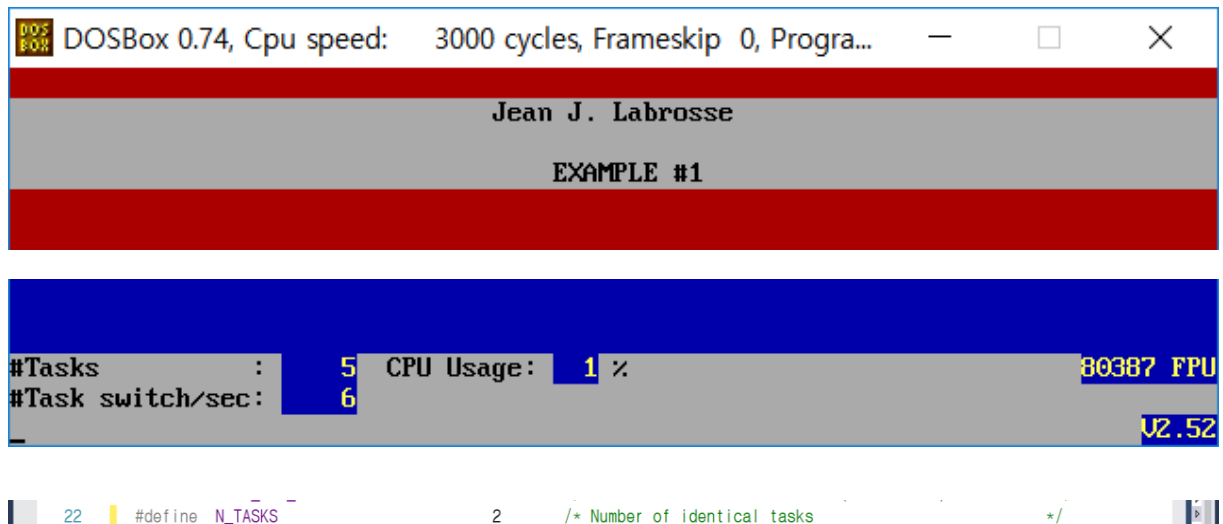


실행결과 Red를 칠하는 task가 생성되어 Task의 총 개수는 4개가 되었다.

```
22  #define N_TASKS 1 /* Number of identical tasks */
```

N_TASKS의 값을 1로 하여 화면을 칠하는 Task가 1개만 만들어지도록 하였다.

(2) Red와 Blue만 칠하기



```
DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Progra...
Jean J. Labrosse
EXAMPLE #1

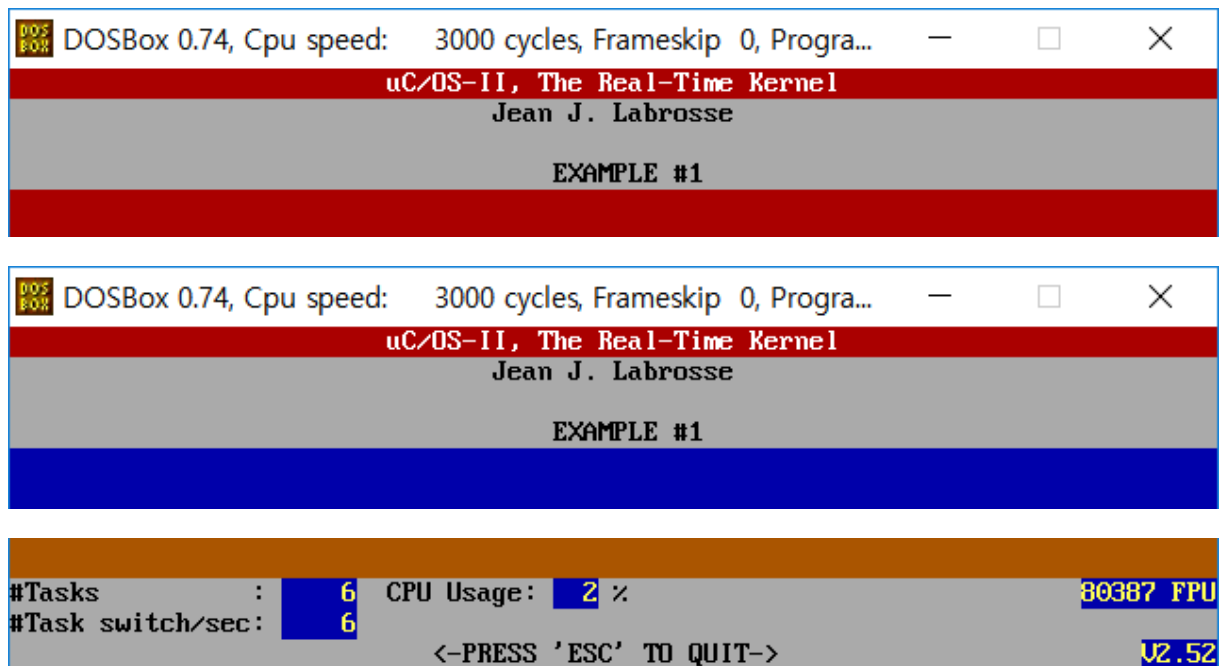
#Tasks : 5 CPU Usage: 1 % 80387 FPU
#Task switch/sec: 6 U2.52

22 #define N_TASKS 2 /* Number of identical tasks */
```

N_TASKS의 값을 2로 하여 화면을 칠하는 Task가 2개 만들어지도록 하였다.

첫번째 task는 빨간색을 표시하고 task 개수 만큼인 2초 동안 waiting queue에 들어간다. 1초 후에 두번째 task가 실행되어 파란 화면으로 바꾸고 2초 동안 waiting queue에 들어간다. 이 작업이 반복되면 1초 마다 빨강, 파랑이 반복되어 화면의 색을 바꾼다.

(3) Red와 Blue, Brown만 칠하기



```
DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Progra...
uC/OS-II, The Real-Time Kernel
Jean J. Labrosse
EXAMPLE #1

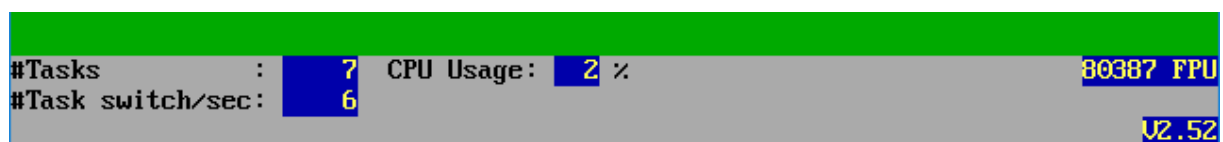
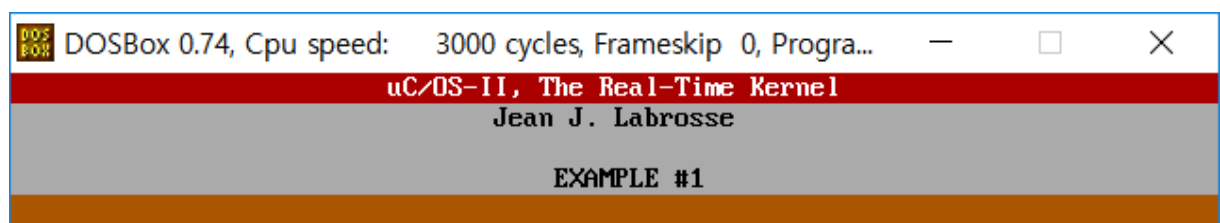
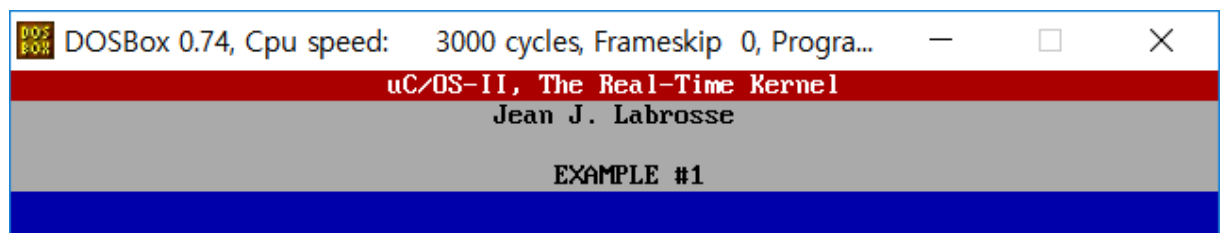
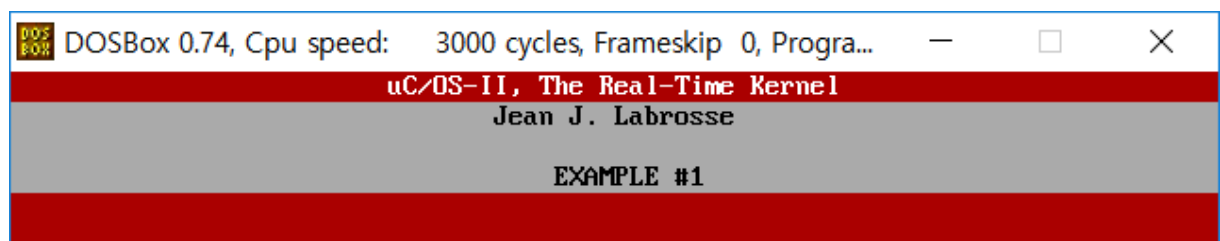
#Tasks : 6 CPU Usage: 2 % 80387 FPU
#Task switch/sec: 6 <-PRESS 'ESC' TO QUIT-> U2.52
```

```
22  #define N_TASKS 3 /* Number of identical tasks */
```

N_TASKS의 값을 3으로 하여 화면을 칠하는 Task가 3개 만들어지도록 하였다.

첫번째로 빨강 task가 실행되고 3초동안 waiting queue에 들어간다. 1초 후 파랑 task가 실행되고, 이후 브라운 task가 실행된다. 결과적으로 1초마다 빨강->파랑->브라운으로 화면이 전환된다.

(4) Red, Blue, Brown, Green 모두 칠하기



```
22  #define N_TASKS 4 /* Number of identical tasks */
```

N_TASKS의 값을 4로 하여 화면을 칠하는 Task가 4개만 만들어지도록 하였다.

각 task가 화면의 색을 바꾸고 waiting queue에 들어가는 시간은 4초이다. 빨강->파랑->브라운->연두 순으로 화면의 색이 1초마다 바뀐다.

5. 결론

화면의 색을 바꿀 때 여러 task들이 딜레이 없이 색을 바꾸면, 우리 눈에서 인지할 수 없는 시간에 한번에 색이 바뀌어서 마지막 우선순위를 가진 task의 색만 보였다. 이를 방지하기 위해 색을 바꿀 때 세마포어를 줘서 다른 task들이 몇 초동안은 색을 못바꾸도록 세마포어를 갖고 있는 방법이 있었다. 다른 방법으로 task들을 서로 다른 시간에 불러오고 waiting queue에 넣는 방법이 있었다.

결과적으로, 여러 task를 만들고 context switch을 하며 multitasking을 할 수 있음을 알았다. 이를 활용하여 각 task마다 실시간으로 interrupt를 받아 그에 대한 처리를 해줄 수 있도록 설계할 수 있음을 더욱 느꼈다.