프로세스 동기화 과제

20185309

황명원

소스코드:

#include <stdio.h>  
#include <stdlib.h>  
#include <pthread.h>  
  
int cnt;  
int flag[2] = {0,0};  
int turn = 0;  
  
void \*func0() {  
 for (int i = 0; i < 100000; i++) {  
 flag[0] = 1;  
  
 turn = 1;  
  
 while (flag[1] && turn==1) {}  
 cnt++;  
 printf("cnt1 :: %d\n", cnt);  
 flag[0] = 0;  
 }  
}  
void \*func1() {  
 for (int i = 0; i < 100000; i++) {  
 flag[1] = 1;  
  
 turn = 0;  
  
 while (flag[0]&&turn==0) {}  
 cnt++;  
 printf("cnt2 :: %d\n", cnt);  
 flag[1] = 0;  
 }  
}  
int main()  
{  
 pthread\_t thread1, thread2;  
 int iret1, iret2;  
  
 iret1 = pthread\_create( &thread1, NULL, func0, NULL);  
 iret2 = pthread\_create( &thread2, NULL, func1, NULL);  
  
 pthread\_join( thread1, NULL);  
 pthread\_join( thread2, NULL);  
  
 printf("Thread error: %d\n", 200000-cnt);  
 exit(0);  
}

분석내용:

임계 영역에 진입하려면 먼저 flag[i] =j로 하여 임계 영역에 들어가게 합니다.

turn = j 로 설정하고 내부 while문을 수행하고

내부 while 문에서, 프로세스 j가 임계구역에 들어갈 의사표시를 하지 않았다면 임계 영역에 들어갈 수 있습니다.

만약, 두 개의 프로세스가 동시에 임계구역에 진입하려고 한다면 둘 중에 하나만을 선택해야 하므로, 변수 turn이 역할을 발휘하고 조금이라도 turn변수가 늦게 수행된 프로세스가 내부의 while문에서 기회를 양보합니다.

임계 영역에서 나오는 프로세스는 flag[i]를 i로 함으로써, 다른 프로세스가 임계 영역에 들어가도록 허용합니다.

이렇게 하게 되면 두개의 프로세스는 겹쳐서 실행이 되지 않아 문제에 의도한 대로 cnt를 증가 시킬수 있습니다.

실행결과:

텍스트이(가) 표시된 사진

자동 생성된 설명파일 생성

텍스트이(가) 표시된 사진

자동 생성된 설명실행(1)

(중략)

텍스트이(가) 표시된 사진

자동 생성된 설명 실행(2)

정상적으로 실행 됨을 확인 할수 있습니다.