201711395 컴퓨터공학과 박성준

사용한 구조체 목록

1. msgget의 ioctl에 넘겨줄 인자에 대한 구조체

```
struct qid {
        int key;
        int msgflg;
};
```

2. msgbuf 구조체

```
struct msgbuf {
        long type;
        char text[128];
};
```

3. msgrcv, msgsnd의 ioctl에 넘겨줄 인자에 대한 구조체

```
struct msg_config {
        int key;
        struct msgbuf *msg;
        int msgsz;
        long msgtyp;
        int msgflg;
};
```

4. 메시지 큐 구조체

```
struct msg_queue {
        struct list_head list;
        struct msgbuf *msg;
};
```

ku_ipc.c의 중요 전역 변수

1. int queue_reference_counter[QUEUE_NUM];

해당큐를 접근하는 프로세스의 개수가 몇 개인지 저장하는 변수

2. static struct msg_queue msg_queue_head[QUEUE_NUM];

메시지 큐의 첫 번째 메시지를 가르키는 list_head를 가진 메지 큐의 헤드 변수

구현

ku_ipc_lib에서는 함수마다 디바이스 파일을 열고 닫으며 cmd에 따라 필요한 ioctl을 호출하게 하였다. ku_ipc_lib에서는 일부 플래그의 예외처리와 해당 프로세스가 어떤 큐를 접근하고 있는지에 대한 예외처리가 되어있다.

ku_ipc.c는 커널 모듈로써 필요한 함수들과 기능들이 들어있으며 rcv의 경우 NO_ERROR에 대한 예외처리를 담고 있다. 큐의 구현은 커널의 링크드 리스트를 활용했다. 메시지에 대한 메모리는 동적 할당 해주었다. spin_lock으로 레이스 컨디션을 방지했다.

Function in ku_ipc_lib.c

ku_msgget	Functionality	queue_use로 먼저 해당 프로세스가 사용중인지 확인하고 사용중이면 -1을 리턴하고 미사용중이면 진행한다. key와 msgflg값을 qid에 넣는다. qid변수는 ku_ipc ioctl 호출시에 인자로 넘겨준다. KU_IPC_EXCL로 호출되었는데 이미 사용중인 큐이거나 해당 프로세스가 이미 사용중인 큐이면 -1을 리턴한다. 시작시 디바이스파일을 열고 함수 종료직전 닫아준다.
	Parameters	int key, int msgflg
	Return Value	int
ku_msgclose	Functionality	queue_use로 먼저 해당 프로세스가 사용중인지 확인하고 사용중이지 않으면 -1을 리턴한다. 사용중이면 계속 진행한다. msqid 변수를 ku_ipc ioctl호출시 넘겨준다. 시작시 디바이스파일을 열고 함수 종료직전 닫아준다.
	Parameters	int msqid
	Return Value	int
ku_msgsnd	Functionality	queue_use로 먼저 해당 프로세스가 사용중인지 확인하고 사용중이지 않으면 -1을 리턴한다. 사용중이면 계속 진행한다. msg_config구조체 변수를 선언하여 함수 파라미터값을 넣어주고 해당 변수를 ku_ipc ioctl호출시 인자로 넘겨준다. msgflg 와 KU_IPC_NOWAIT와의 비트연산을 통해 해당 비트가 0이면 메시지가 전송안되는 상황임으로 반복적으로 ioctl이 성공할때까지 호출한다. KU_IPC_NOWAIT 비트가 1이면 전송 실패시 바로 -1을 리턴한다. 시작시 디바이스파일을 열고 함수 종료직전 닫아준다.
	Parameters	int msqid, void *msgp, int msgsz, int msgflg
	Return Value	int

ku_msgrcv	**Functionality**	queue_use로 먼저 해당 프로세스가 사용중인지 확인하고 사용중이지 않으면 -1을 리턴한다. 사용중이면 계속 진행한다. msg_config구조체 변수를 선언하여 함수 파라미터값을 넣어주고 해당 변수를 ku_ipc ioctl호출시 인자로 넘겨준다. msgflg 와 KU_IPC_NOWAIT와의 비트연산을 통해 해당 비트가 0이면 메시지가 받아지지 않는 상황으로 반복적으로 ioctl이 성공할때까지 호출한다. KU_IPC_NOWAIT 비트가 1이면 ioctl의 리턴값이 -1혹은 -2인 경우 -1을 리턴하고 종료한다. -1인 경우 메시지가 없어서 종료되는 ioctl리턴값이고 -2인 경우는 KU_MSG_NOERROR비트 마스크 부분이 0이고 메시지가 받는 사이즈가 큐 메세지보다 작을 때 이루어진다. 시작시 디바이스파일을 열고 함수 종료직전 닫아준다.
	Parameters	int msqid, void *msgp, int msgsz, long msgtyp, int msgflg
	Return Value	int

Function in ku_ipc.c

ku_ipc_ioctl	**Functionality**	파라미터 cmd에 따라 다른 그에 해당되는 맞는 함수를 호출한다. arg인자로 받은 파라미터는 각각 함수로 넘겨줄 인자에 맞는 형태로 형변환 해준다.
	Parameters	struct file *file, unsigned int cmd, unsigned long arg
	Return Value	static long
ku_ipc_exit	**Functionality**	기본적인 디바이스 드라이버 제거함수와 함께 할당된 모든 큐 메시지들을 삭제하고 메모리를 프리해준다.
	Parameters	void
	Return Value	static long
ku_ipc_msgget	**Functionality**	queue_reference_counter를 참조하여 KU_IPC_EXCL인 경우 아무도 참조 안할때만 성공(0)을 리턴하고 아닌경우는 항상 성공적인 값을 리턴한다. 실패시에는 -1을 리턴한다. 성공한 경우 queue_reference_counter을 하나 증가시킨다. spin_lock으로 레이스 컨디션을 방지한다.
	Parameters	struct qid *my_qid
	Return Value	static long

ku_ipc_msgclose	**Functionality**	queue_reference_counter를 참조하여 사용중이지 않으 큐에 대해 호출되었거나 잘못된 큐번호에 접근한 경우 실패(-1)을 리턴한다. 성공한 경우 queue_reference_counter을 하나 감소시킨다. spin_lock으로 레이스 컨디션을 방지한다.
	Parameters	int msqid
	Return Value	static long
ku_ipc_msgsnd	**Functionality**	메시지를 전송할 큐를 list_for_each_entry를 통해 순회하며 메시지 개수와 메시지 총 크기를 계산하고 유휴공간이 없어 넣을 수 없으면 -1을 리턴한다. 유휴 공간이 있다면 메시지를 커널로 copy_from_user로 받아와 큐의 뒷부분에 넣어준다.
	Parameters	struct msg_config *my_msg
	Return Value	static long
ku_ipc_msgrcv	**Functionality**	메시지를 받을 큐를 list_for_each_entry를 통해 순회하며 받고싶은 타입과 같은 타입의 메시지가 큐에 있는지 확인한다. 같은 타입의 메시지가 없는 경우 -1을 리턴한다. 같은 타입의 메시지가 있는 경우 KU_MSG_NOERROR여부를 확인하여 비트마스크가 0이고 큐의 메시지 사이즈가 읽을 메시지 사이즈 보다 큰 경우 -2를 리턴한다. 해당 비트마스크가 1인 경우 읽을 수 있는 만큼 copy_to_user로 읽어온다. 읽어온 경우 해당 메시지를 삭제하고 해당 메모리를 해제해준다.
	Parameters	struct msg_config *my_msg
	Return Value	static long