Embedded System Software1

과제 #2 KU\_PIR



컴퓨터공학과

201411292

이민환

Basic design

프로세스에서 open을 하게 되면 각자의 자료구조를 생성하기 위해 커널영역에서 해당 프로세스의 pid 값을 확인하여 새로운 pid인 경우 새 자료구조를 만듭니다. 다른 프로세스에서 open을 하게 되면 pid 값이 다르기 때문에 새 자료구조를 만들지만, 같은 프로세스에서 open을 여러 번 해도 pid 값이 같기 때문에 자료구조가 새로 생기지 않고, fd 값만 증가합니다. close를 하면 해당 pid를 확인하여 일치하는 자료구조를 제거하고, read를 하게 되면 가장 오래된 데이터부터 하나씩 읽고 삭제합니다. 만약 데어터가 하나도 없다면, 데이터가 들어올 때까지 대기를 하기 위해 waitqueue에 프로세스를 넣습니다. 데이터가 존재하는지 확인하기 위하여 count 변수를 선언 후, insertData를 할 때마다 증가시켜주고, read 할 때 마다 감소하고, flush를 하면 0으로 만듭니다. flush를 하게 되면 close와 비슷하게 pid를 확인하고 안의 데이터를 모두 제거합니다. insertData를 하게 되면 모든 자료구조에 데이터를 넣습니다. 만약 데이터가 꽉 차 있다면 가장 오래된 데이터를 지우고 새 데이터를 넣어줍니다. 그 후, wake up() 함수를 사용하여 대기중인 프로세스를 깨워줍니다.

센서가 감지를 하게 되었을 때 flag 값으로 rising, falling 두개를 구분하여 rising인 경우이면 rf\_flag = ‘1’, falling인 경우에는 rf\_flag = ‘0’으로 하고 timestamp는 jiffies 값으로 하여 insertData를 호출해 자료구조에 데이터를 추가합니다.

Description for important functions

ku\_ipc\_lib

|  |  |  |
| --- | --- | --- |
| ku\_pir\_open() | Functionality | 파일 오픈을 하고, fd 값을 받아와 ioctl을 호출한다. |
| Parameters |  |
| Return Value | 성공하면 fd, 실패하면 -1 |

|  |  |  |
| --- | --- | --- |
| ku\_pir\_close() | Functionality | 파일을 닫는다. |
| Parameters | int fd |
| Return Value | 성공하면 0, 실패하면 -1 |

|  |  |  |
| --- | --- | --- |
| ku\_pir\_read() | Functionality | 데이터를 저장할 구조체 주소와 함께 ioctl을 호출한다. |
| Parameters | int fd, struct ku\_pir\_data \*data |
| Return Value |  |

|  |  |  |
| --- | --- | --- |
| ku\_pir\_flush() | Functionality | Flush ioctl을 호출한다. |
| Parameters | int fd |
| Return Value |  |

|  |  |  |
| --- | --- | --- |
| ku\_pir\_insertData() | Functionality | 추가할 데이터를 저장한 구조체와 함께 ioctl을 호출한다 |
| Parameters | long unsigned int ts, char rf\_flag |
| Return Value | 성공하면 0, 실패하면 -1 |

ku\_ipc\_c

|  |  |  |
| --- | --- | --- |
| ku\_pir\_open() | Functionality | 해당 프로세스의 pid를 확인하여 새로운 프로세스인 경우 새 자료구조를 만들어 list에 연결해준다. |
| Parameters | int fd |
| Return Value | 성공하면 fd, 실패하면 -1 |

|  |  |  |
| --- | --- | --- |
| ku\_pir\_close() | Functionality | 해당 프로세스의 pid를 확인하여 일치하는 pid를 가지고 있는 자료구조를 삭제한다. |
| Parameters | int fd |
| Return Value | 성공하면 0, 실패하면 -1 |

|  |  |  |
| --- | --- | --- |
| ku\_pir\_read() | Functionality | 해당 프로세스의 pid를 확인하여 일치하는 pid를 가지고 있는 자료구조의 첫번째 데이터를 불러오고, 데이터가 없을 경우 프로세스를 넣어 재워준다. |
| Parameters | int fd, struct ku\_pir\_data \*data |
| Return Value |  |

|  |  |  |
| --- | --- | --- |
| ku\_pir\_flush() | Functionality | 해당 프로세스의 pid를 확인하여 일치하는 pid를 가지고 있는 자료구조의 데이터를 모두 삭제한다. |
| Parameters | int fd |
| Return Value |  |

|  |  |  |
| --- | --- | --- |
| ku\_pir\_insertData() | Functionality | Parameter로 받아온 데이터를 모든 자료구조에 넣어주고, 데이터가 가득 차 있는 자료구조인 경우 가장 오래된 데이터를 지우고 새로운 데이터를 저장한다. 그 후 잠들어 있는 프로세스를 깨워 준다. |
| Parameters | int fd |
| Return Value | 성공하면 0, 실패하면 -1 |

|  |  |  |
| --- | --- | --- |
| sensor\_irq() | Functionality | 센서가 감지되면 실행되는 함수로 flag 값을 확인하여 rising, falling을 구분하고 jiffies를 timestamp로 하여 자료구조에 데이터를 넣어준다. |
| Parameters | int irq, void \*dev\_id |
| Return Value | IRQ\_HANDLED |