총 100점 만점
arm-linux-gnueabi architecture를 위한 코드를 작성하라.
단, 여기서 제시되지 않은 사항은 본인이 가정하고 해도 좋으나, 답안 제출시 주석으로 명시할 것

1. 첨부된 라이브러리 (및 헤더파일) 를 이용하여 다음의 일을 하는 프로그램을 작성하시오. (25점)

- 111111 에서부터 0 까지 매 0.001초마다 카운트 다운을 하여 FND에 표시할 것
- 0초가 되면 카운트 다운을 멈추고
- GPIO LED를 모두 켤 것
- 도-미-솔-(높은)도 를 각 1초의 간격으로 버저가 울리게 할 것 (단 이때 옥타브는 440 Hz 기준 옥타브로 할 것)
- 단, 프로그램 안에서 open 함수를 쓰면 안됨 (open함수를 쓰면 0점 처리함)

채점 기준
- make 로 build가 될 것 (5점, 단 make clean 이 안되면 2.5점)
- build 된 프로그램이 키트에서 실행 될 것 (5점 - 실행이 안되면 아래 모두 0점 처리함)
- 카운트 다운이 될 것 (5점)
- GPIO LED가 처음에는 모두 꺼져 있다가 카운트다운이 완료 된 후 모두 켜질 것 (5점)
- 버저가 울리게 할 것 (5점)

제출 방법
- 최종 실행파일, makefile, header file, source code file, library file 모두 학번_exam1.tar.gz 로 압축하여 업로드 할 것

2. GPIO 버튼 입력을 받아 지정된 MSG ID로 전달하는 라이브러리를 작성할 것. (25점)

- 키를 손에서 뗐을 때만 메시지가 오도록 할 것
- 메시지로 전달되는 정보는 “키의 종류" 만 long int 형으로 전달할 것
- 메시지 ID는 999 로 할 것
- 메시지 박스가 생성되지 않았다면 먼저 생성부터 하게 할 것
- 이전에 미처 가져가지 않은 메시지가 있다면 몇개의 메시지가 전달이 안되었는지 출력하고 삭제할 것
- 라이브러리 시작 함수 이름은 startbutton(); 으로 할 것
- library는 static으로 빌드 할 것

채점 기준
- 메시지 Box가 생성되지 않았다면 먼저 생성하는가? (5점)
- 이전에 미처 가져가지 않은 메시지 처리가 정확하게 되는가? (5점)
- make 로 build가 될 것 (5점, 단 make clean 이 안되면 2.5점)
- 라이브러리가 main.c와 함께 빌드되어 실행될 것 (5점, 단 실행이 안되면 아래 모두 0점 처리함)
- GPIO 버튼을 눌렀다 뗐을 때 메시지가 정확하게 전달되어 확인이 되는가? (5점)

제출 방법
- 최종 makefile, header file, source code file, library file 모두 학번_exam2.tar.gz 로 압축하여 업로드 할 것

3. 첨부한 데이터시트를 확인하여 UART를 제어하는 모듈형 디바이스 드라이버를 작성하시오.
(50점, 어플리케이션은 작성할 필요 없다.)

- Load시 “Exam Serial Driver Loaded!” 라고 출력할 것
- Unload시 “Exam Serial Driver Unloaded!” 라고 출력할 것
- Device Node의 Major 넘버는 888, Minor 넘버는 0 으로 할 것
- Open시 “Exam Serial Driver Opened!” 라고 출력할 것
- Application이 N-Byte를 Write를 하면 N-Byte Data를 UART를 통해 출력하도록 할 것
- Application이 ioctl 명령을 통해 UART를 끌 수 있도록 할 것 (CMD = 10), 다른 값은 바뀌면 안됨
- Application이 ioctl 명령을 통해 UART를 켤 수 있도록 할 것 (CMD = 11), 다른 값은 바뀌면 안됨
- Magic Number는 199로 할 것
- IOW 만 사용할 것
- kernel의 위치는 ~/linux_kernel 로 할 것
- 소스코드가 어디있던 간 make 로 빌드되게 할 것

채점 기준
- make 로 빌드하여 ko파일이 작성될 것 (10점, 단, Build가 안되면 아래 모두 0점 처리함)
- insmod 가 될 것 (5점)
- rmmod 가 될 것 (5점)
- application에서 장치 open이 될것 (5점)
- application에서 write를 할 때 성공적으로 UART로 출력될 것 (10점)
- application에서 ioctl을 할 때 성공적으로 UART가 disable 될 것 (7.5점)
- application에서 ioctl을 할 때 성공적으로 UART가 enable 될 것 (7.5점)

제출 방법
- 최종 makefile, header file, source code file, ko file 모두 학번_exam3.tar.gz 로 압축하여 업로드 할 것

힌트

- UART에 1 Byte를 출력하는 로직의 Pseudo code는 다음과 같다.

```
while(1)
{
        if (TX_FIFO is not full)      break;
}
TX_DATA = data;
```

- Memory mapped IO 방식에서 특정 주소 (ex:0x1234)로 매핑된 레지스터의 값은 다음처럼 Access 할수 있다.

```
(*((volatile unsigned int *)(0x1234)))
```

- UART EN/Disable 시에는 bitwise 연산을 활용할 것.

- 데이터시트의 필요한 부분은 모두 highlight 되어 있고, 필요 없는 부분은 취소선 처리 되어 있음.