**Embedded 3rd HW. Interrupt & Device\_Driver (설계 프로젝트 수행 결과)**

**과목명: 임베디드 시스템 프로그래밍**

**담당교수: 서강대학교 컴퓨터공학과 박 성 용**

**학번 및 이름: 20141542, 심찬양**

**개발기간: 2019. 05. 26. - 2019. 05. 30.**

**최 종 보 고 서**

**I. 개발 목표**

Module Programming, 디바이스 드라이버 구현, interrupt 등, 실습 시간 때 배운 내용을 활용하여 간단한 stopwatch 프로그램을 작성한다.

**II. 개발 범위 및 내용**

**가. 개발 범위**

1. Stop watch Device Driver 개발

1-a. Timer를 이용한 Device Driver 개발

1-b. Interrupt를 이용한 Device Driver 개발

1-c. wait\_queue를 이용한 process block 제어

1-d. memory mapped 방식을 이용한 fpga\_fnd device control Driver 개발

2. 1-a, 1-b, 1-c, 1-d 를 종합한 User Application Program 제작

**나. 개발 내용**

1. Stop watch Device Driver 개발

- 기존에 사용하던 home, back, vol+, vol- 버튼에 연결되어 있는 interrupt를 가려 놓고 (주석처리) 이번 프로젝트에서 요구하는 기능을 수행하는 interrupt device driver module을 새롭게 구현 한다.

2. User Application Program 제작

- 간단한 user level단계의 응용프로그램으로 device driver module을 작성하고 kernel에 등록한 다음 정해진 /dev/stopwatch 라는 이름의 device driver를 open 하고 write() 함수를 사용하여 stopwatch의 기능을 수행하는 application program을 제작 한다.

**III. 추진 일정 및 개발 방법**

**가. 추진 일정**

**- 5/26 (일) : 이번 과제를 하기 위해 실습한 5,6주차 Timer,interrupt 실습내용을 복습한 뒤 App 폴더에 있는 user application program을 작성하고 프로젝트 명세서를 정독함.**

**- 5/27-29 (월-수) : module 폴더안에 stopwatch\_driver.c 를 구현하고 테스트 해보면 서 코드를 완성함.**

**- 5/30 (목) : document 및 read.me 등을 작성하고 최종 테스트를 완료하였다.**

**나. 개발 환경 및 방법**

**개발 환경 :** Ubuntu Linux 16.04 Terminal에서 vi 편집기를 이용하였으며

arm-none-linux-gnueabi-gcc로 Cross-컴파일을 하였습니다. Adb push 명령으로 usb을 통해 전달한 바이너리 파일을 실행 함으로 결과를 확인 할 수 있었습니 다.

**개발 방법 :**

1.Module (Device Driver) 개발

1.a : 5번째 실습이었던 timer 실습을 다시 복기 해보며 코드를 작성했다. Struct timer\_list 구조체를 사용했다. 초기timer 함수를 세팅하는 함수에서 call back 함수를 호출하고 정해진 만료시간마다 Call back 함수를 호출 하여 1초( = HZ) 마다 fnd의 값을 변경하는 timer 함수를 작성한다. 종료를 하 기 위한 조건으로 voldown 버튼을 3초이상 누를 때 종료하게 되는데 이 3초라는 시간을 재기위해 end\_timer를 추가로 사용한다.

1.b : 6번째 실습이었던 interrupt 실습을 참고 하면서 코드를 작성했다. 그래서 driver.c 의 코드가 실습 때 사용한 코드의 형태와 유사하다. Irq 번호를 얻는 함수와 얻은 irq번호를 등록하는 request\_irq() 함수 역시 실습 때 사용한 코드를 참조했다. Flag 역시 IRQF\_TRIGGER\_FALLING 과 RISING flag를 사용하여 누를때와 뗄때 interrupt 신호가 가게끔 했다. 버튼마다 다른 irq번호로 들어오는 interrupt 신호를 통해 각각의 함수가 실행되게 하였다.

1.c : interrupt를 사용하기 위해 수행중이던 process를 block을 시킬 필요가 있 음을 우리는 실습을 통해 알았으므로, wait\_queue\_head\_t 타입의 변수를 선언하여 init를 수행하고 interruptible\_sleep\_on() 함수와 \_\_wake\_up() 함 수를 사용하여 user process에서 시작한 process context를 block 시키고 버튼을 통한 interrupt 신호를 통해 device를 제어하는 interrupt context 의 코드로 프로그램을 진행한다.

1.d : 이전에도 driver를 module로써 등록하지 않고 fnd의 물리적 주소를 kernel 주소에 직접 지정해서 memory mapped 방식을 사용하여 직접적 인 값을 통해 fnd의 값을 control하는 방식을 사용했다.

2. User Application program 제작

- device dirver를 만들어 board에 모듈을 등록하고 얻은 major 번호를 통해 device driver의 이름을 정해서 파일을 만든다. 만들어진 파일의 이름으로 open() 함수에서 사용해서 device driver의 모든 init부분을 수행시킨다. 함수의 Return 값을 fd의 정수값으로 반환 하여 그런 다음 write() 에서 fd와 나머지 정보는 이 프로그램을 실행하는데 사용하지 않았으므로 Null, 과 0을 넘겨주어 write() 함수를 수행시켜서 stopwatch 기능이 수행되도록 한다.

target Board의 kernel에 명세서에서 요구한 새로운 system call을 추가하기 위한 작업을 수행한다. 절차는 아래와 같다 system call number를 할당한다(차례로할당)-> system call table에 새로운 system call을 등록한다. -> kernel의 소스폴더가 있는 곳에 자신이 만든 system call.c 파일을을 저장해 놓는다. -> kernel을 recompile 하여 새로운 bootimage를 만들고 board를 reboot한다. -> User Application program의 실행 파일을 만들어 board에서 실행시켜본다.

2. Device Driver

- host PC에서 device driver module을 생성 한 다음 target board로 옮기는 절차를 수행한다. 절차는 다음과 같다. 먼저 만들고자 하는 device driver의 major number를 결정한다. (겹치지 않는 적절한 번호 설정) device\_dirver.c 파일과 같은 드라이버 코드 파일을 작성하고 make를 하는데 적절한 옵션을 통해 driver.ko 등의 파일을 생성한다. -> board에서 insmod를 수행하여 device driver module를 kernel에 심는 작업을 한다 -> mknod 명령어를 통해 /dev/에 파일을 만들어 user program 과 device driver 사이에서 interface 역할을 수행한다 -> user program을 실행 시켜 device driver가 올바르게 수행되는지 확인한다.

3. user program

- syscall(), open(), write()등의 함수를 사용하여 프로그램을 작성한다. 적절한 parameter를 썼는지 프로젝트 명세서에 요구한 대로 작동하는지 return값을 확인 해 보고 device도 적절히 실행되는지 확인한다.

**IV. 연구 결과**

- 최종 연구 개발 결과를 자유롭게 기술할 것.

**1. 합성 내용:**

**스크린샷이(가) 표시된 사진

자동 생성된 설명**

**[ 그림 1 ] 전반적인 interrupt 흐름도**

설명 :

user program에서 open() 함수를 호출했을 때 driver 코드 안에 있던 itner\_open() 함수가 수행되어 각각의 버튼의 irq 번호를 얻어내고 그 irq번호를 request\_irq() 함수를 통해 irq\_desc[]에 저장되게 하여 해당하는 irq의 interrupt 신호가 들어오게 되면 그 신호에 맞는 함수를 수행하게 되는 과정을 그림으로 나타낸 것이다.

스크린샷이(가) 표시된 사진

자동 생성된 설명

**[ 그림 2 ] 전체 프로그램 흐름도**

설명 : 전체 프로그램의 수행 context를 user level과 kernel level으로 나누어 보았다. 주로 함수의 단위로 clustering 하여 좀 더 세부적인 프로그램의 흐름을 알 수 있다. 밑에 초록색인 Hardware는 interrupt context로 외부의 입력에 의한 코드의 수행이다.

**2. 제작 내용:**

1. device driver 개발

- 명세서에서 요구한대로 stopwatch를 fnd를 통해 시간을 나타내는 프로그램을 작성한다. 먼저 Interrupt를 사용하기 전에 arch/arm/mach-mx6/board-achroimx.c 수정을 해야한다.

스크린샷, 텍스트이(가) 표시된 사진

자동 생성된 설명

기존 사용하던 gpio interrupt를 가려 놓는다.

- 사용법은 다음과 같다.

1. home 버튼 입력시 timer(1초 단위)를 시작하게 된다.

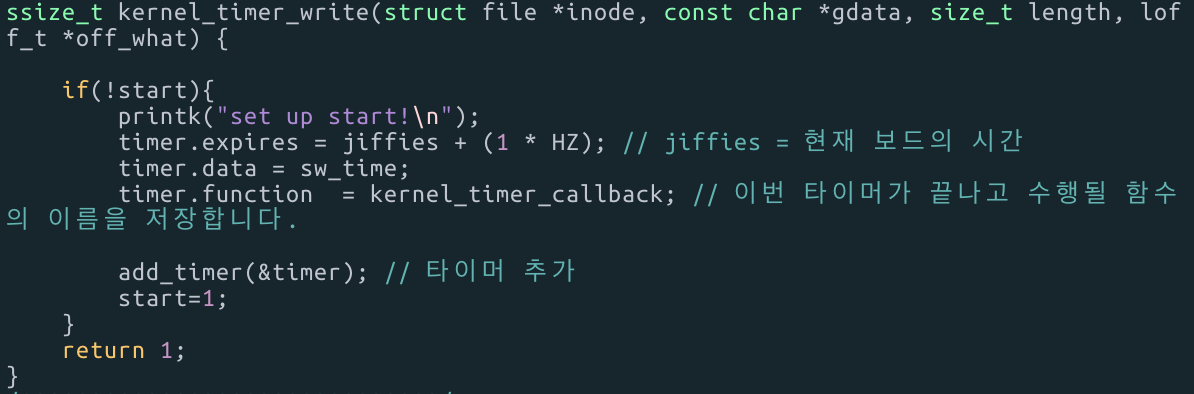
2. back 버튼을 누르면 진행 중이던 timer를 중단하고 시간을 소수점자리까지 유지한다.

3. vol+ 버튼을 누르면 진행 중이든 back버튼을 눌러 중지 중이든 timer를 0으로 reset한 다.

4. vol- 버튼을 3초이상 눌렀을 경우 수행중이던 timer를 삭제하고 block중인 process를 깨워(wake\_up) userprogram을 종료한다.

먼저 밝히기를 파라미터를 상당수 사용을 안 한부분이 있는데 형식을 유지하기 위해 사용하였고 실질적으로 사용하지 않았습니다.

- home (start 기능)



Home 버튼을 누르면 위 함수가 처음으로 실행된다. 그런 다음 kernel\_timer\_callback 함수가 실행이 되는데 1초마다 반복 호출 되어 fnd에 값을 sw\_time의 값을 fnd의 형식에 맞게 %, /, shift 계산을 통해 fnd에 출력을 하게 한다.

텍스트이(가) 표시된 사진

자동 생성된 설명

- back 버튼 (pause 기능)

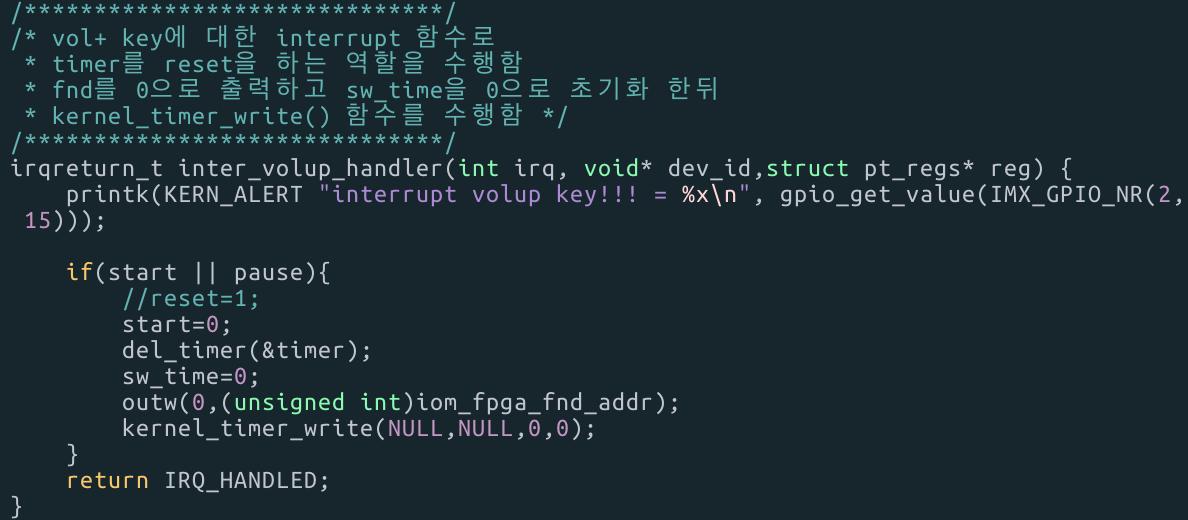
Back버튼을 누르게 되면 아래 함수가 실행되며, pause의 값을 0,1로 바꾸면서 수행의 분기를 구현했다. 진행중일때 pause를 하게 되면 timer를 삭제하고 남은 시간을 remain\_time에 저장한다.

다시 시작버튼을 눌렀을 때 남은 시간(remain\_time)에 원래의 값(sw\_time)을 더한 값을 fnd에 출력함으로써 이전에 수행중이던 timer의 값을 유지할 수 있다.

텍스트이(가) 표시된 사진

자동 생성된 설명

- vol+ 버튼 (reset 기능)



Vol+ 버튼을 누르면 현재 수행중이던 시간 sw\_timed을 0으로 초기화 하고 timer를 삭제한 뒤 fnd에 0을 출력해 reset이 된 상태로 돌린다. 그리고 마지막으로 Kernel\_timer\_write()을 수행시킨다.

- vol- 버튼 (종료 기능)

Voldown 버튼을 3초이상 누르고 있을 경우 프로그램이 종료가 되는 기능을 수행한다. 버튼을 눌렀을 때 3초라는 시간을 계산하기 위해서 추가로 하나의 end\_timer를 선언하고 초기화 하여 사용한다. 이 버튼에서 다른 버튼과 다른점은 rising (뗄때) 할때도 interrupt가 되게 하여 3초라는 시간을 제어해서 종료할 수 있는 점이다.

텍스트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

2. user program 제작

텍스트이(가) 표시된 사진

자동 생성된 설명

* 비교적 간단한 코드로 작성된다. 중요한 점은 device driver의 이름과 open(), write()를 사용함으로써 kernel level의 module 함수를 call하여 사용하는 것이다.

**3. 시험 및 평가 내용:**

- 평가 방법:

명세서에서 요구한 대로 Device의 이름을 정하고 프로그램을 수행 시켰을 때, 버튼의 입력을 잘 받는지, 명세서의 요구대로 stop watch가 작동하는가에 따라 여러 번 수행시켜 가면서 평가하였다.

**V. 기타**

**1.** **연구 조원 기여도**:

심찬양 : 100%

**2.** **기타 본 설계 프로젝트를 수행하면서 느낀 점을 요약하여 기술하라. 내용은 어떤 것이든 상관이 없으며, 본 프로젝트에 대한 문제점 제시 및 제안을 포함하여 자유롭게 기술할 것.**

이번 프로젝트로 인해서 timer의 사용법과 비교적 간단한 interrupt의 사용을 직접 코딩 할 수 있어서 좋았다.