과목 명: 임베디드 시스템 소프트웨어

담당 교수 명: 박성용

<<HW 3>>

**서강대학교 컴퓨터공학과**

**[학번] 20151543**

**[이름] 민지우**

# 프로그램 개요

Module programming, 디바이스 드라이버 구현, interrupt 등, 실습 시간 때 배운 내용을

활용하여 간단한 stopwatch 프로그램을 작성한다.

1. Fpga\_fnd device driver와 timer module, interrupt을 포함한 stopwatch 기능을 가진 모듈을 구현한다.
2. Module을 실행시키는 application을 구현한다.

# 프로그램 설명

## 프로그램 흐름도

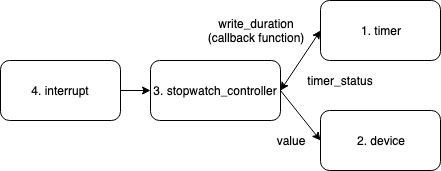


그림 1> 프로그램 흐름도

해당 모듈들의 설명은 아래 참고

# 모듈 설명

## 모듈 이름 : timer

Stopwatch의 데이터와 method을 모아둔 module  
stopwatch을 초기화하고, 시작/일시 멈춤/종료 와 같은 기능을 하기 위해 stopwatch 구조체의 데이터를 제어한다.   
이때 stopwatch의 경과 시간을 더하기 위해서 linux/timer기능을 이용했다. timer의 완료 시간을 1초 후로 설정해 1초마다 duration을 더하고, 다시 timer을 설정함으로써 stopwatch 기능을 구현하였다.

그리고 stopwatch의data가 바뀌는 것을 다른 module에게도 알리기 위해, event handler 구조를 사용하였다. Stopwatch의 duration을 바꾸게 하는 method마다 callback함수를 받아 duration을 변경한 후 해당 duration을 callback 함수로 인자로 넘겨줘 실행하게 하였다.

### set\_timeout

#### 기능

Linux timer을 설정하는 함수

(clocks)뒤에 (func)을 실행

#### 사용 변수

Long clocks: timer가 종료될 clock의 수, clocks 후에 timer가 완료된다

Void (\*func) (unsigned long): timer가 종료되고 실행될 callback함수

### stopwatch\_blink

#### 기능

Stopwatch 기능 중에서 시간을 측정하는 기능을 구현하는 함수

Stopwatch을 시작하면 해당 함수가 1초마다 실행된다.

크게 세가지 작업을 수행한다. Duration이 변경되므로,

변경된 후에 미리 저장해둔 callback함수를 실행해 변경된 duration을 인자로 넘겨준다

1. Duration을 1씩 증가
2. Duration이 변경되었으므로 blink\_handler (callback 함수)을 실행
3. Set\_timeout을 실행해, 1초 뒤에 다시 blink함수가 실행되도록 설정

#### 사용 변수

Unsigned long timeout: linux timer에서 stopwatch\_blink을 실행했을 때 넘겨주는 인자. Stopwatch의 주소를 저장

### Stopwatch\_init

#### 기능

Stopwatch의 데이터를 초기화하는 함수

Linux timer또한 초기화 한다. Stopwatch의 data가 바뀌므로 인자로 받는 after\_handler라는 callback 함수를 실행해 변경된 duration인 0을 넘겨준다.

#### 사용 변수

Void (\* after\_handler) (unsigned long): duration이 초기화된 후에 초기화된

duration을 받아서 실행되는 함수

### Stopwatch\_pause

#### 기능

Stopwatch을 중지시키는 함수

Linux timer을 종료시키고, timer의 남은 clock 수를 stopwatch.rest에 저장한다.

#### 사용 변수

사용 변수 없음

### Stopwatch\_start

#### 기능

Stopwatch을 시작시키는 함수

입력으로 받은 blink\_handler을 미리 저장해 stopwatch\_blink될 때마다 실행 되도록 한다. Stopwatch가 이전에 중지 되었다가 시작하는 것이면 남은 clocks뒤에, 그것이 아니면 1초 뒤에 stopwatch\_blink가 실행되게끔 설정한다.기능

#### 사용 변수

Void (\* blink\_handler) (unsigned long): stopwatch\_blink에서 duration이 변경될 때마다 실행될 함수

### Stopwatch\_reset

#### 기능

Stopwatch을 reset하는 함수

Stopwatch을 중지시키고, stopwatch의 data을 초기화한다.

#### 사용 변수

Void (\*after\_handler) (unsigned long): stopwatch\_init에서 duration이 변경될 때 실행될 함수

### Stopwatch\_get\_status

#### 기능

현재 stopwatch의 상태를 return하는 함수

#### 사용 변수

사용 변수 없음

## 모듈 이름 : device

Device을 제어하는 모듈

Device을 초기화하고, 읽거나 쓰는 기능을 포함한다. 해당 프로젝트에서는 fnd device만을 사용하기 때문에 해당 device와 관련된 기능만을 가지고 있다.

### Fnd\_write

#### 기능

Fnd device에 입력으로 받은 숫자 리스트를 출력하는 함수

#### 사용 변수

Char\* val: fnd에 출력할 숫자 리스트

### Fnd\_init

#### 기능

Fnd device을 제어하기 위해 초기화하는 함수

Fnd device의 주소를 mapping 한다

#### 사용 변수

사용 변수 없음

## 모듈 이름: stopwatch\_controller

Stopwatch와 device model을 control하는 모듈. 이번 프로젝트에서 명세한 기능들을 구현한 모듈. Stopwatch의 상태를 받아와 stopwatch method을 호출하고 callback 함수로 device method을 넘겨준다.

### Stopwatch\_ctrl\_start

#### 기능

Stopwatch을 시작하는 함수

Stopwatch model의 start함수를 실행시킨다. 이때 stopwatch의 상태가 TIMER\_INIT인 상태에만 동작한다.

#### 사용 변수

사용 변수 없음

### Stopwatch\_ctrl\_pause

#### 기능

Stopwatch을 중지시키는 함수

Stopwatch의 상태에 따라서 동작이 달라진다. TIMER\_RUNNING일 때는 stopwatch model의 pause을 실행시킨다. TIMER\_PAUSE일 때는 stopwatch model의 start을 실행시킨다. 그 외 상태에서는 아무 동작도 하지 않는다.

#### 사용 변수

사용 변수 없음

### Stopwatch\_ctrl\_reset

#### 기능

Stopwatch을 reset하는 함수

Stopwatch의 상태에 따라서 stopwatch model을 reset하고 난 뒤의 행동이 달라진다. 이전의 상태가 TIMER\_RUNNING이었으면, stopwatch model을 다시 시작한다.

#### 사용 변수

사용 변수 없음

### Stopwatch\_ctrl\_exit

#### 기능

Stopwatch을 종료하는 함수

Stopwatch model의 reset함수를 실행한다. 즉, stopwatch의 데이터를 초기화한다.

#### 사용 변수

사용 변수 없음

### Stopwatch\_ctrl\_init

#### 기능

Stopwatch을 초기화하는 함수

Stopwatch model의 init함수와 device에서 fnd init함수를 실행한다.

즉 stopwatch controller에서 제어하는 다른 module들을 초기화한다.

#### 사용 변수

사용 변수 없음

### Write\_duration

#### 기능

입력으로 받는 duration을 fnd에 출력될 숫자 리스트로 변환하고 이를 fnd\_write에게 넘겨준다

Stopwatch model에서 duration이 달라질 때 실행되는 callback 함수

#### 사용 변수

Unsigned long duration: stopwatch model의 duration

## 모듈 이름: interrupt

Module의 main이 되는 모듈

Module의 기능들을 file operation과 연결해준다.

Stopwatch controller의 기능들을 interrupt handler로 등록한다.

write가 실행될 시 wait queue에 등록해 block시킨다.

Linux timer을 사용해 3초 동안 vol- 버튼이 눌러졌을 때 wake up 할 수 있도록 하였다.

이번 프로젝트와 관련된 method들에 대해서만 서술하겠음

### Handler\_home

#### 기능

Home button이 눌렸다 떨어질 때마다 발생하는 interrupt을 다루는 interrupt\_handler  
stopwatch controller의 start기능을 실행한다

#### 사용 변수

Int irq: interrupt handler에게 넘겨지는 인자. 해당 handler에서는 사용되지 않음

Void dev\_id: interrupt handler에게 넘겨지는 인자. 해당 handler에서는 사용되지 않음

Struct pt\_regs\* reg: interrupt handler에게 넘겨지는 인자. 해당 handler에서는 사용되지 않음

### Handler\_back

#### 기능

back button이 눌렸다 떨어질 때마다 발생하는 interrupt을 다루는 interrupt\_handler  
stopwatch controller의 pause기능을 실행한다

#### 사용 변수

Int irq: interrupt handler에게 넘겨지는 인자. 해당 handler에서는 사용되지 않음

Void dev\_id: interrupt handler에게 넘겨지는 인자. 해당 handler에서는 사용되지 않음

Struct pt\_regs\* reg: interrupt handler에게 넘겨지는 인자. 해당 handler에서는 사용되지 않음

### Handler\_vol\_up

#### 기능

Vol+ button이 눌렸다 떨어질 때마다 발생하는 interrupt을 다루는 interrupt\_handler  
stopwatch controller의 reset기능을 실행한다

#### 사용 변수

Int irq: interrupt handler에게 넘겨지는 인자. 해당 handler에서는 사용되지 않음

Void dev\_id: interrupt handler에게 넘겨지는 인자. 해당 handler에서는 사용되지 않음

Struct pt\_regs\* reg: interrupt handler에게 넘겨지는 인자. 해당 handler에서는 사용되지 않음

### Handler\_vol\_down

#### 기능

Vol- button이 눌렸을 때와 떨어질 때 마다 발생하는 interrupt을 다루는 interrupt\_handler

Interrupt가 발생할 때마다 stopwatch 기능을 실행시킬 timer을 종료시킨다.

Interrupt가 눌러졌을 때 발생한거라면, (timer의 expire 시간이 현재시간보다 이전이면) timer을 설정해준다.

이렇게 함으로써 interrupt가 발생하고 3초동안 해당 interrupt가 다시 발생하지 않으면 stopwatch을 종료시키고, 다시 발생하게 되면 동작중인 timer을 삭제한다

#### 사용 변수

Int irq: interrupt handler에게 넘겨지는 인자. 해당 handler에서는 사용되지 않음

Void dev\_id: interrupt handler에게 넘겨지는 인자. 해당 handler에서는 사용되지 않음

Struct pt\_regs\* reg: interrupt handler에게 넘겨지는 인자. 해당 handler에서는 사용되지 않음

### inter\_open

#### 기능

Application에서 해당 모듈 드라이버 파일을 open 할 때 실행되는 함수

해당 함수에서 실습 시간에 배운 request\_irq을 이용한다.

Vol- 버튼을 제외한 버튼들에 대해서는 button이 눌렸다 떨어졌을 때에만 실행되지만, vol- 버튼은 버튼이 눌릴 때와 버튼이 떨어질 때 둘다 실행되게끔 하였다.

크게 세가지 기능을 수행한다

1. Interrupt handler을 등록한다
2. Stopwatch\_controller을 초기화 한다
3. Vol- timer에게 실행될 함수로 stopwatch 종료 함수를 등록한다

#### 사용 변수

해당 함수에서 사용하지 않아 설명 생략

### inter\_release

#### 기능

Application에서 모듈의 드라이버 파일을 닫을 때 실행된다

1. 등록된 handler을 해제한다
2. Stopwatch 기능을 종료한다

#### 사용 변수

해당 함수에서 사용하지 않아 설명 생략

### inter\_init

#### 기능

모듈 드라이버를 등록할 때 실행된다

Vol- timer을 초기화한다

#### 사용 변수

사용 변수 없음

### inter\_end

#### 기능

1. Stopwatch controller의 exit을 실행해 stopwatch 기능을 종료한다
2. Wait queue에 있는 application을 깨운다

#### 사용 변수

해당 함수에서 사용하지 않아 설명 생략

# 전역 변수 정의

## 모듈 이름: timer



### NO\_LAP 0

Stopwatch가 중지되지 않으면 rest\_clocks에 저장될 clock 수

### Struct Stopwatch

Stopwatch 의 기능에 필요한 data을 정의하고 저장한다

Struct timer\_list timer: stopwatch가 시간을 측정하는 데 사용하는 linux timer

Unsigned long duration: stopwatch에서 측정된 시간 (초단위)을 저장

Unsigned long rest\_clocks: stopwatch 가 중단될 때 남은 clocks을 저장

TimerStatus status: stopwatch의 상태

Void (\* blink\_handler) (unsigned long): stopwatch blink에서 실행될 함수

### Static Stopwatch stopwatch

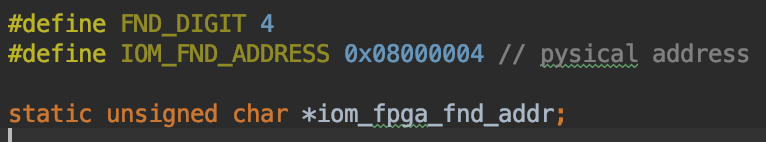
Stopwatch data을 저장한다

### Enum TimerStatus



Stopwatch의 상태를 enum으로 정의

## 모듈 이름: device



### FND\_DIGIT 4

Fnd의 digit

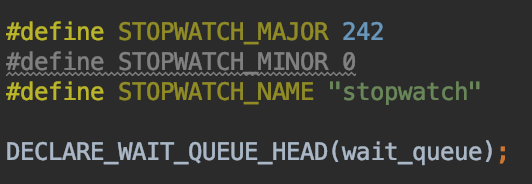
### IOM\_FND\_ADDRESS 0x 08000004

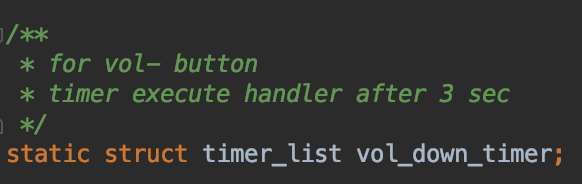
fnd device의 물리적인 주소

### Static unsigned char \*iom\_fpga\_fnd\_addr

fnd device의 address로 mapping된 주소를 저장

## 모듈 이름: interrupt.c





### STOPWATCH\_MAJOR 242

Stopwatch 디바이스 모듈의major #

### STOPWATCH\_MINOR 0

Stopwatch 디바이스 모듈의 minor#

### STOPWATCH\_NAME “stopwatch”

Stopwatch 디바이스 모듈의 이름

### Wait\_queue

Application이 block 될 때 들어갈 wait\_queue

### Static struct timer\_list vol\_down\_timer

Vol- 이 3초동안 눌리면 종료되도록 하는 timer