**Embedded 3rd HW.**

**(설계 프로젝트 수행 결과)**

**과목 명: [CSE4116] Embedded Systems**

**담당 교수: 서강대학교 컴퓨터 공학과 박 성 용**

**학번 및 이름: 20091648, 이 준 호**

**개발 기간: 2014. 05. 27. - 2014. 06. 08.**

**최종 보고서**

**I. 개발 목표**

본 프로젝트에서는 module programming, device driver 구현, interrupt 등을 활용하여 achro 4210 Exynos 보드 위에서 작동하는 간단한 stopwatch program을 구현한다.

**II. 개발 범위 및 내용**

**가. 개발 범위**

Stopwatch Driver - gpio\_fnd device driver와 timer module, interrupt를 포함하는 하나의 module을 구현한다. 각 device는 최초 한번 insmod 시 initialize가 되고, user level에서 write를 호출하면 interrupt를 이용하여 stopwatch 기능이 작동되도록 한다.

Application - 위에서 구현한 device driver를 이용하여 stopwatch 기능을 사용할 수 있도록 하는 application을 구현한다.이 application은 device 파일을 open 하여, write를 통해 출력을 시작해 준 뒤, close 하고 종료된다.

**나. 개발 내용**

1) FND device의 앞의 두 자리는 분 (60분) 단위를 출력하고, 뒤의 두 자리는 초 (60초) 단위를 표시한다. FND device의 초기 상태는 0000으로 하며, 각각의 키 입력은 interrupt를 이용하여 수행한다.

SW2 : start -> 1초마다 fnd의 정보를 갱신 (timer 사용)

SW3 : pause -> 일시 정지

SW4 : reset -> 초기 상태로 돌아감 (0000)

SW6 : quit -> 3초 이상 누르고 있을 시 application을 종료하고, fnd를

꺼준다.

2) Device driver의 이름은 /dev/stopwatch로 통일하고, major number는 245로 설정한다.

3) 응용 프로그램 (application)

device 파일을 open하여, write를 통해 출력을 시작해준 뒤, close하고 종료된다.

**III. 추진 일정 및 개발 방법**

**가. 추진 일정**

|  |  |
| --- | --- |
| 2014. 05. 27. | Assignment #3의 요구 사항 분석 및 프로젝트 진행 계획 |
| 2014. 05. 29. | 내부적으로 Stopwatch의 기본 기능 구현 |
|  | Application의 기본 frame 구현 |
| 2014. 06. 02. | FND device에 stopwatch timer 출력 |
| 2014. 06. 03. | 알려진 bug 수정 |
|  | time counting 알고리즘 수정 |
| 2014. 06. 04. | Code 최적화 및 정리 |
| 2014. 06. 05. | Document 초안 작성 |
| 2014. 06. 08. | Document 수정 및 제출 |

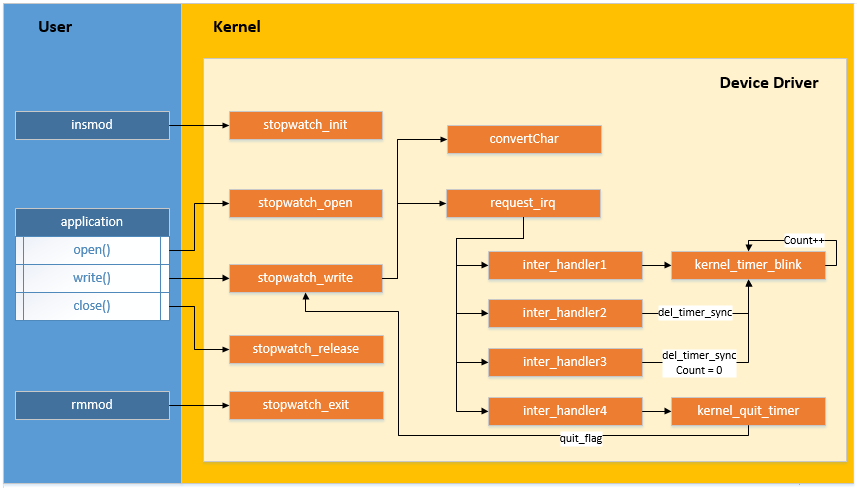
**나. 개발 방법**

Module programming : fnd device와 timer module 그리고 interrupt를 stopwatch module에서 initialize 한 후 application에서 write가 호출 되면 interrupt를 이용하여 각각의 switch에 맞는 기능을 실행시켜 준다. start switch가 press 되면 timer를 이용하여 1초마다 count를 1 증가 시켜 그에 맞는 시간을 FND device에 출력하여 주고, pause를 할 경우 timer를 일시적으로 제거하여 더 이상 count가 증가하는 것을 막아준다. Reset을 press 하면 timer structure에 있는 count의 값을 0으로 초기화 해주고, FND device에 write를 한다. 종료 switch를 3초 동안 누르고 있으면 timer를 제거하고 FND device를 off 시키고, device를 release하고, application을 종료한다.

Application : device를 open하고 write를 한다. 이때, user가 application을 종료하기를 기다리기 위해 while 문을 돌려 종료하기 위한 return value가 오기를 대기하고, 종료 조건의 return value를 받으면 device driver를 close하고 application을 종료한다.

**IV. 연구 결과**

**1. 합성 내용:**



**2. 제작 내용: 개발 결과**

**static void kernel\_timer\_blink(unsigned long timeout)**

역할 : 1초마다 count를 1씩 증가 시킨다.

구현 방법 : timer module을 사용하여 1초마다 timer structure의 count variable을 1씩 증가 시킨다.

**static void kernel\_quit\_timer(unsigned long timeout)**

역할 : program이 종료될 수 있도록 flag를 설정한다.

구현 방법 : 전역 변수인 quit\_flag를 0으로 설정한다.

**irqreturn\_t inter\_handler1(int irq, void \*dev\_id, struct pt\_regs \*reg)**

역할 : timer를 추가하여 stopwatch를 시작한다.

구현 방법 : interrupt가 발생하면 timer를 add하여 kernel\_timer\_blink가 수행되도록 한다.

**irqreturn\_t inter\_handler2(int irq, void \*dev\_id, struct pt\_regs \*reg)**

역할 : timer를 제거하여 stopwatch를 멈춘다.

구현 방법 : interrupt가 발생하면 timer를 delete하여 더 이상 count가 올라가지 않도록 한다.

**irqreturn\_t inter\_handler3(int irq, void \*dev\_id, struct pt\_regs \*reg)**

역할 : timer의 count를 0으로 초기화 한다.

구현 방법 : interrupt가 발생하면 timer를 delete하여 더 이상 count가 올라가지 않도록 하고, count의 값을 0으로 설정하여 준다.

**irqreturn\_t inter\_handler4(int irq, void \*dev\_id, struct pt\_regs \*reg)**

역할 : switch가 3초 동안 눌렸는지 판단하여 프로그램을 종료 시킨다.

구현 방법 : interrupt가 발생하면 timer를 이용하여 3초 동안 interrupt가 발생 하였는지 판단하고, 3초 동안 interrupt가 발생한 경우 quit\_timer를 add하여 프로그램이 종료되도록 한다.

**int stopwatch\_open(struct inode \*minode, struct file \*mfile)**

역할 : device의 usage count를 설정하고 interrupt를 설정한다.

구현 방법 : stopwatch usage count를 설정하고, count를 0으로 초기화 한 후, request\_irq를 이용하여 각각의 switch의 interrupt 발생 조건을 설정한다.

**int stopwatch\_release(struct inode \*minode, struct file \*mfile)**

역할 : open했던 device들을 종료 시킨다.

구현 방법 : timer를 delete하고 FND device를 off 시킨 후, interrupt를 free 시켜준다.

**char convertChar(int num)**

역할 : 넘겨 받은 num을 char 형태로 변환하여 return 한다.

구현 방법 : parameter로 넘겨 받은 num을 그에 맞는 FND 출력 용 char 형태로 변환하여 해당하는 값을 return 하여 준다.

**ssize\_t stopwatch\_write(struct file \*inode, const short \*gdata, size\_t length, loff\_t \*off\_what)**

역할 : timer의 count 값을 이용하여 FND에 출력하여 준다.

구현 방법 : while문과 msleep을 이용하여 FND에 분/초에 해당하는 값을 계산하여 출력하여 준다.

**int \_\_init stopwatch\_init(void)**

역할 : fnd driver, timer module, interrupt를 초기화 한다.

구현 방법 : ioremap를 사용하여 device의 address를 mapping 하여 주고, timer를 초기화 한다.

**void \_\_exit stopwatch\_exit(void)**

역할 : stopwatch module을 unregister 한다.

구현 방법 : unregister\_chrdev를 이용하여 device module을 제거한다.

**3. 시험 및 평가 내용:**

주어진 요구 사항에 대해 만족스러운 output을 만들어 내었다. 여러 차례 시도를 하였으며, input에 있어서 많은 제약 사항을 두고 switch에서만 interrupt가 별다른 문제를 발견할 수 없었다. 지난 프로젝트에 비해 FND 출력에 있어서 좀 더 깔끔하지 못한데, 이는 새로운 방법을 이용해 FND를 출력하는 과정에서 발생한 문제인 듯 하다. Application이 종료될 시에 mapping한 memory를 free 해주기 때문에 메모리 누수를 막을 수 있었다. 지속적인 업데이트를 통해 발견되는 문제에 대해 업데이트를 통해 보완해 나갈 수 있을 것으로 기대한다. 지난 2차 과제와 비슷한 유형의 과제였기 때문에 설계에 있어서 큰 어려움이 없었고, 그로 인해 프로그램의 완성도에 있어서 많은 향상을 얻을 수 있었다.

**V. 기타**

**1. 연구조원 기여도:**

20091648 이준호 : 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.**

저번 과제에서 많은 부분 동일한 부분이 겹쳤기 때문에 크게 어려움은 없었던 것 같다. 기존과 같이 driver module을 이용하지 않고, interrupt를 이용하여 switch로 부터 input을 받는 방법이 재미있었다. Driver module을 사용하는 것보다 좀 더 효과적으로 구현할 수 있었던 것 같다. 여전히 많은 코드를 하나의 파일에 모두 구현한 것이 아쉽다.