**Embedded System Software**

**<Project 3>**

과목명 : 임베디드 시스템 소프트웨어

담당교수 : 서강대학교 컴퓨터공학과 박 성 용

개발자 : 20151607 정재훈

개발기간 : 2019.05.25 ~ 2019.05.31

1. 개발 목표

Module Programming, 디바이스 드라이버 구현, Interrupt 등, 실습 시간 때 배운 내용을 활용하여 간단한 Stopwatch 프로그램을 구현한다.

1. 개발 내용
2. Device Driver Module

Board의 FPGA device와 timer를 제어하는 하나의 module을 구현한다. 제어해야 할 FPGA device는 다음과 같다. FPGA\_FND

Device Driver를 module로 등록시키기 위해 init, exit 함수를 구현하며, module을 삽입시 init 함수가 호출되며 module을 삭제 시 exit 함수가 호출된다.

1. init

해당 driver를 kernel에 등록시키고 FND device를 kernel address space에 mapping 시켜준다.

또한 Stopwatch 기능을 위한 timer를 초기화 시켜준다.

1. exit

kernel에 mapping시킨 FND의 virtual space를 free해주고 사용한 timer의 list를 삭제한다. 마지막으로 등록시킨 driver를 등록해제시켜준다.

Application에서 호출하는 Device Driver의 기본기능은 open, release, write이며 각 다음과 같은 기능을 가진다.

1. open

device 파일을 open 시 작동하는 함수로 device는 하나만 open되어 있어야 하므로 이를 check 하여 여러 번 open 할 시 에러를 발생한다. 또한 전역변수를 초기화 시키고, HOME, BACK, VOL\_UP, VOL\_DOWN 키에 대한 Interrupt를 등록한다. 마지막으로 wait queue를 초기화 시킨다.

1. release

device 파일을 close시 작동하는 함수로, 열려 있는 device가 있을 때, device가 더이상 쓰지 않는다는 상태를 저장한다.

1. write

Application process를 Blocking(sleep) 시켜, 종료조건이 나오지 않는 이상 깨어나지 않도록 해준다.

이외에 FND 출력을 위한 함수를 구현하여, 초를 넣어주면 이를 분, 초로 변환해 FND에 출력해

주도록 한다.

1. Stopwatch 기능을 위한 Interrupt

해당 프로젝트에서 사용해야하는 device interrupt는 총 4가지로 Board의 HOME key, BACK key, Volume up key, Volume down key이다. 해당 버튼을 누를 시 interrupt가 발생하여 각 handler에 대한 기능이 수행되어야 한다. Board kernel에는 해당 버튼에 대한 기존 interrupt handler가 있으므로 이를 삭제한다. Device driver에서 해당 Interrupt에 대한 interrupt Handler를 새로 구현한다. 이렇게 새로 구현한 handler를 기반으로 irq\_desc[]에 등록시켜 이제부터 해당 버튼을 눌렀을 때는 내가 작성한 interrupt handler가 불리도록 한다. 각 Interrupt는 다음과 같다.

1. inter\_handler1 (HOME KEY)

Home key가 누르고 떼어졌을때 interrupt가 발생하며, 해당 함수를 호출한다. stopwatch에서 Home key의 기능은 stopwatch를 시작하는 것이다.

1. inter\_handler2 (BACK KEY)

Back key가 누르고 떼어졌을때 interrupt가 발생하며, 해당 함수를 호출한다. stopwatch에서 Back key의 기능은 stopwatch를 pause하는 것이다. 이때, 시간은 소수단위까지 저장하며 pause를 다시 누를 시 stopwatch가 기존의 정보를 가지고 다시 작동한다.

1. inter\_handler3 (VOL\_UP KEY)

Volume up key의 기능은 stopwatch를 reset하는 것이다. reset시 시간은 다시 0분 0초에서 시작하며 start 중이었으면 reset후 계속 시간이 흘러야 하고 pause 중이었다면 시간은 멈춰 있어야한다.

1. inter\_handler4 (VOL\_DOWN KEY)

Volume down key는 키가 눌렀을 때, 떼어졌을때 모두 interrupt가 발생하며, 이는 stopwatch를 종료하는 기능을 담당한다. 해당 버튼을 3초 이상 누르고 있다면 wait queue에 넣어둔 application process를 깨우며 application이 종료한다.

1. Stopwatch를 위한 Timer interrupt 구현

해당 stopwatch는 1초마다 FND가 1초씩 오르게 해야 하는데 이는 timer interrupt를 이용한다. Start 버튼인 home key를 누르면 0.1초마다 뜨는 timer interrupt를 구현하여, timer interrupt가 뜰때마다 변수 stopwatch\_sec을 1씩 올린다. 해당 변수는 0.1초 단위로 설계되어 있으며, 따라서 10번씩 뜰때마다 1초가 지났다고 판단하여 해당 값에 따른 시각을 fnd을 출력한다.

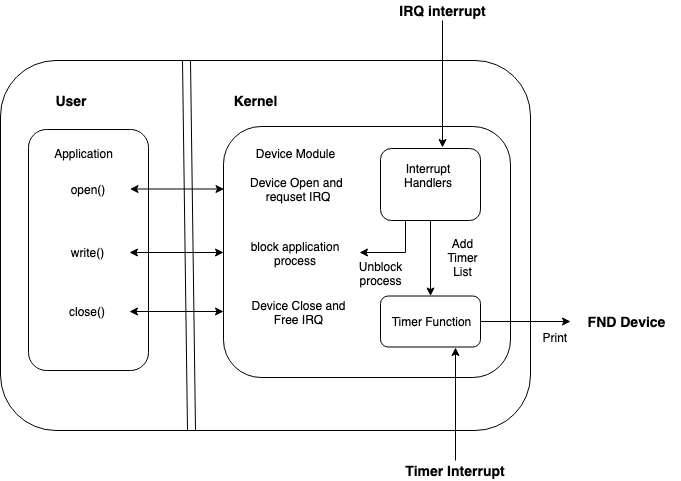
stopwatch 작동 중에 pause 버튼을 누르면 timer interrupt가 등장하지 않아야 하므로 해당 timer interrupt를 종료시키고, pause 버튼을 다시 눌렀을 때 timer interrupt가 start button process와 마찬가지로 0.1초마다 뜨는 timer interrupt를 추가시킨다.

마지막으로 종료조건에서도 timer interrupt를 이용해야 하는데, 종료버튼을 3초이상 누르면 stopwatch를 종료시켜야한다. 따라서 종료버튼을 누를 때, 0.1초 간격으로 뜨는 timer interrupt를 발생시키며, 종료버튼을 때면, 해당 timer interrupt를 timer list에서 삭제한다. 만약 3초 동안 떼는 동작이 발생하지 않으면 이를 종료조건으로 판단하고 Block 시킨 application process를 깨워 프로세스를 종료시킨다.

1. 응용프로그램 구현

응용 프로그램은 해당 device를 open후 write를 호출하여 block상태가 된다. 만약 종료조건이 성립하여 blocking에서 풀리게 된다면, 해당 device를 close하고 프로그램을 종료한다.

1. 구성



1. 제작 내용
2. Device Driver Module
3. init

먼저 register\_chrdev\_region()를 통해 Major Number 242와 device name을 Kernel에 등록시킨다. 그 후 ioremap() 함수를 이용하여 필요한 FPGA device를 kernel address space에 mapping 시킨다. 마지막으로 init\_timer()를 통해 timer 구조체를 초기화 시키고 0을 return한다.

1. exit

iounmap function을 통해 mapping 시킨 FPGA device를 mapping 해제시켜준다. del\_timer\_sync()로 timer List에 add된 timer 들을 전부 삭제 후 device를 unregister한다.

1. open

dev\_driver\_port\_usage라는 변수를 유지하고 open시에 1로 저장한다. 만약 open하려는데 1이 저장된 상태라면 여러 번 open하려는 시도로 간주하고 error 를 리턴 한다. 또한 open에서는 다음과 같은 전역 변수들을 초기화 시켜주어야 한다

* stopwatch\_cont : 1이면 stopwatch가 계속 진행 0이면 stopwatch가 정지
* stopwatch\_time : 0.1초 주기로 1씩 올라 현재 stopwatch의 시간을 알려준다
* push\_flag : VOL\_DOWN 크기가 눌려져 있다면 1 떼어졌다면 0을 저장한다.

그 후 FND 화면을 0으로 초기화 시켜주고 request\_irq()를 이용하여 HOME, BACK, VOL\_UP, VOL\_DOWN를 해당하는 interrupt handler와 함께 등록시킨다. 마지막으로 Blocking IO를 위한 wait queue를 초기화 시켜준다.

1. close

dev\_driver\_port\_usage를 0으로 만들어주며 등록시킨 irq들을 모두 free시킨다.

1. wrtie

Process를 Block시킨다.

1. fpga\_fnd\_print

FND device에 parameter로 받은 second를 가지고 분 초로 변환하여 출력시켜준다.

들어오는 second는 0~3600범위인데 이를 60으로 나눠 minute와 second로 분리하고 fnd value format에 맞추어 encoding한다. 이를 outw 함수를 통해 fnd device에 출력시킬 수 있다.

1. Stopwatch 기능을 위한 Interrupt
2. inter\_handler1 (HOME KEY)

Start button이 눌렀을 때, stopwatch\_cont 변수를 1로 바꾸고, 0.1초마다 불러지는 Timer를 timer list에 넣어야 한다. 따라서 현재 timer\_list를 지우고 expire가 현재로부터 HZ/10 만큼 더한 timer element를 timer list에 삽입한다. 이때 불려지는 timer\_function은 stopwatch\_timer\_function()이며 해당 함수는 0.1초마다 stopwatch\_time을 1씩 늘리고 1초가 지나면 fnd 값을 갱신해주는 역할을 한다.

1. inter\_handler2 (BACK KEY)

Back key가 눌렸을 때, stopwatch\_cont를 0으로 바꾸어 start에 의해 생겨난 timer가 더이상 동작되지 못하고 return되게해준다. 따라서 현재 stopwatch는 멈춘 상태가 된다. 만약 다시 한번 눌렀을 때, stopwatch\_cont가 1로 바뀌며, 이때는 start와 마찬가지로 0.1초마다 stopwatch\_timer\_function()를 호출하는 timer element를 timer list에 추가한다.

1. inter\_handler3 (VOL\_UP KEY)

Volume up key를 눌렀을 때, 현재 저장된 stopwatch\_time 변수를 0으로 초기화 시키고 이를 FND에 출력한다.

1. inter\_handler4 (VOL\_DOWN KEY)

Volume down key를 눌렀을 때, push\_flag를 1로 바뀌고, 0.1초마다 exit\_timer\_function을 호출하는 timer element를 timer list에 추가한다. 해당 exit\_timer\_function은 push\_flag가 1일때만 동작하며 총 3초가 되었다고 판단했을 때, block한 process를 unblock 시키고 return 한다. 만약 Volume down key를 떼었을 때, push\_flag가 0으로 바뀌어 exit\_timer\_function이 더이상 동작하지 못하게 한다.

1. Stopwatch를 위한 Timer interrupt 구현
2. stopwatch\_timer\_function

이 stopwatch\_timer\_function은 start button이나 pause button에 의해 timer element가 timer list에 추가되었을 때 0.1초마다 불러지는 함수이다. 만약 pause button에 의해 stopwatch\_cont변수가 0이 되었을 경우 더이상 timer list에 새로운 timer를 추가하지 않고 종료한다. 만약 stopwatch\_cont가 1인경우, 현재 stopwatch\_time 변수를 1씩 올리고 만약 stopwatch\_timer % 10이 0이라면 1초가 지났다고 판단하여 1초가 지난 시각을 FND에 출력한다. 마지막으로 0.1초 후에 stopwatch\_timer\_function을 호출하는 timer를 timer list에 새롭게 추가한다.

1. exit\_timer\_function

이 exit\_timer\_function은 volume down키가 눌러지면 timer element가 timer list에 추가되는데 이 timer element에 의해 0.1초마다 불러지는 함수이다. push\_flag 변수를 보고 현재 volume down 키가 눌러져 있는 상태(1)인지 눌러져 있지 않은 상태(0)인지 확인한다. 만약 현재 눌러져 있는 상태이고, pushed\_time이 30이 넘으면, 총 3초동안 눌러져 있다고 판단한다. 이 때 block 되어 있는 process를 깨우며 del\_timer\_sync를 통해 stopwatch\_timer\_function()과 관련된 타이머를 지우고, FND에 0을 띄우고 return한다.

만약 push\_flag가 0이면 해당 더이상 눌러져있지않다고 판단하여 return한다.

push\_flag가 1이지만 pushed\_time이 3초가 지나지 않았을 때는 pushed\_time을 1씩 증가시키고 해당 함수를 부르는 새로운 timer를 timer list에 추가시켜준다.

1. Application

open 함수를 통해 해당 device open하고, write함수를 호출한다. 그러면 해당 process는 block되며 IRQ interrupt를 통해 특정 종료 조건이 만족되야 unblock 된다. 그 후 device 를 close하고 종료한다.

1. 기타

**느낀 점**  
수업시간 때 배운 Device Driver Module, IRQ interrupt, Timer interrupt를 직접 실습해 보는 경험을 하면서 Timer Device와 외부 Device에서 Interrupt가 발생되는 과정을 보다 깊게 이해할 수 있었다. 또한 Sleep 과 Wake 메커니즘을 통해 어떻게 Blocking IO device driver가 구현되는지 직접 체험할 수 있어 값진 경험이 된 것 같다.