**Embedded System Software 과제 2**

**(과제 수행 결과 보고서)**

**과목명: [CSE4116] 임베디드시스템소프트웨어**

**담당교수: 서강대학교 컴퓨터공학과 박 성 용**

**학번 및 이름: 20181625, 남현준**

**개발기간: 2024. 05. 03. -2024. 05. 17.**

**최 종 보 고 서**

**I. 개발 목표**

- 각 과제마다 주어지는 주제를 바탕으로 본 과제에서 추구하는 개발 목표를 설정하고 그 내용을 기술할 것.

과제 2에서는 수업에서 배운 내용을 바탕으로 실습에서 사용하는 Achorimx 보드의 FPGA 모듈을 활용해 타이머의 디바이스 드라이버 및 관련 테스트 응용 프로그램의 개발을 목표로 한다.

개발 과정에서, 응용 프로그램은 디바이스 드라이버를 기존의 방식이 아닌 ioctl을 통해 타이머 디바이스의 제어를 하게 된다. Ioctl을 통해 전달된 구동 옵션을 이용해 input으로 주어지는 초기값을 기준으로 타이머의 초기화를 진행하고, 그 후 추가적인 ioctl을 통해 타이버 디바이스가 명세서의 세부기능에 맞게 구동될 수 있도록 구현을 진행한다.

**II. 개발 범위 및 내용**

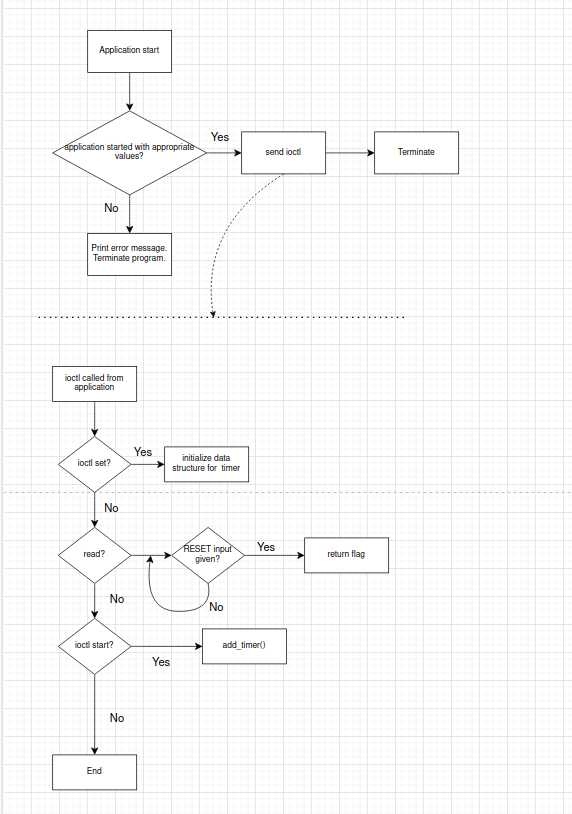
- 자신들이 설계한 개발 목표를 달성하기 위하여 어떠한 내용의 개발을 수행할 지 그 범위와 개발 내용을 기술할 것.

1. **개발 범위**
   1. **응용 프로그램**

타이머 디바이스의 구동에 사용될 초기값들을 입력받고, ioctl을 통해 device driver에 특정 요청을 전달하는 응용 application의 개발을 진행한다. Ioctl을 통해 전달되는 명령은 두 가지로, 초기값을 전달하여 타이머 디바이스의 initialization을 진행하는 명령과 타이머 디바이스에 설정된 값을 이용한 실제 구동 명령을 내리는 ioctl이 구현될 것이다.

* 1. **타이머 디바이스 드라이버**

타이머 디바이스의 제어를 하게 되는 device driver를 개발한다. 응용 프로그램으로부터 전달받는 값으로 임베디드 보드에 있는 FPGA 모듈들의 초기화를 진행하거나, 타이머 디바이스의 구동 명령을 받아 정의한 로직대로 타이머 디바이스가 구동될 수 있도록 한다.



1. **개발 내용**
   1. 응용 프로그램

타이머 디바이스에 명령을 내리게 될 ioctl()와 관련된 개발을 진행한다. 과제 명세서에서 타이머 디바이스는 /dev/dev\_driver라는 이름을 가진다고 명시되어 있고, 이에 따라 /dev/dev\_driver에 대해 open(), ioctl(), read(), close()를 수행할 수 있도록 개발을 진행한다. 이 과정에서 ioctl을 통해 전달되는 명령 중, 타이머 디바이스의 초기화와 관련된 부분에서 TIMER\_INTERVAL이 1 ~ 100, TIMER\_CNT가 1 ~ 200, TIMER\_INIT가 0001 ~ 8000 (입력 값이 00X0과 같이 0이 3개, 1 ~ 8 범위의 숫자 1개로 구성)의 범위를 가짐을 유의하고 해당 부분에 대해 예외 처리가 진행될 수 있도록 한다.

* 1. 타이머 디바이스

명령을 받게 되면, 타이머 디바이스는 다음과 같은 FPGA 모듈에 대해 제어를 하게 된다 :

* + - 1. DOT
      2. FND
      3. LCD
      4. LED
      5. RESET

과제 명세서와 함께 주어진 과제2\_driver의 내용을 참고하여 ioremap(), iounmap()과 같은 함수들을 이용해 FPGA 모듈을 memory mapped I/O 방식으로 활용될 수 있도록 physical address에 매핑을 진행하고, outw() 함수를 통해 모듈에서 사용될 내용이 write될 수 있도록 한다.

FPGA 모듈의 제어를 하는 과정에서 다음과 같이 작동될 수 있도록 한다.

* + - 사용하는 타이머 디바이스는 /dev/dev\_driver, dev\_driver.ko를 통해 driver의 설치, 제거 및 사용될 수 있도록 한다.
    - 타이머 디바이스의 register 사용하는 device type은 character device, Major number는 242로 설정하도록 한다.
    - Ioctl을 통해서 전달되는 초기값에서 TIMER\_INTERVAL은 1 ~ 200, TIMER\_CNT는 1 ~ 200, TIMER\_INIT은 0001 ~ 8000 사이의 범위에서만 전달될 수 있도록 하고 범위를 벗어나거나 조건에 어긋나는 경우 디바이스를 초기화하는 것이 아닌 에러를 출력하고 응용 프로그램이 종료될 수 있도록 한다.
    - TIMER\_INTERVAL이 0이 되기 전까지, 매 interval마다 FPGA 모듈을 제어하고 갱신된 값이 모듈을 통해 출력될 수 있도록 한다.
    - FND에 출력되는 값은 매 interval마다 현재 0이 아닌 값에서 1 증가된 값으로 갱신될 수 있도록 한다. 만약, 현재 digit의 값이 8이라면 다음 값은 9가 아닌 1이 되도록 설정하도록 한다. 초기에 입력 받았던 값을 기준으로 digit이 증가하며 기존 값과 같은 갑이 되는 경우 digit의 위치를 한 칸 오른쪽으로 이동시킬 수 있도록 한다. 가령, 기존 값이 8이었던경우 1 ~ 7까지 1바퀴를 돈 다음 8로 갱신되는 순간, shift right와 유사하게 FND의 4 digit을 기준으로 1칸 오른쪽으로 이동될 수 있도록 한다. 현재의 0이 아닌 digit의 위치가 000X, 즉 가장 오른쪽이었던 경우 X000이 될 수 있도록 한다.
    - DOT에 출력되는 값은 현재 FND에 출력되는 00X0과 같은 4 digit 중, 0이 아닌 값과 같은 값을 DOT 모듈 상에 출력될 수 있도록 한다. 가령, 0020이 현재 FND에 출력된 상태라면 DOT 모듈에는 2가 출력되어야 한다.
    - LCD는 총 32 length이며 첫 16 character가 L CD의 첫 번째 줄, 다음 16 character가 LCD의 두 번째 줄에 출력되는 내용을 담게 된다. 첫 번째 줄에는 본인의 학번을 좌측 정렬, 응용 프로그램을 통해 전달받게 되는 TIMER\_CNT의 값을 우측 정렬로 표시될 수 있도록 한다. 가령, 본인의 학번이 “2018XXXX”, 전달 받은 TIMER\_CNT가 100인 경우 “2018XXXX 100”이 될 수 있도록 하며, 이외의 칸은 ‘ ‘, 즉 공백으로 채워지도록 하여 가독성에 지장이 가지 않도록 한다. 이 중, TIMER\_CNT의 경우, ioctl을 통해 구동 명령이 주어지는 경우, 매 interval마다 값이 감소하게 되며 감소한 값에 맞게 적절히 right-align하고 갱신된 값을 LCD의 첫 번째 줄에 반영할 수 있도록 한다.
    - LCD의 두 번째 줄에는 본인의 이름 이니셜 3자가 들어가게 된다. 이니셜의 경우 매 interval마다 오른쪽/왼쪽으로 1칸씩 이동될 수 있도록 한다. 초기에 이동하는 방향은 오른쪽으로 마지막 이니셜이 16번째 index에 오는 순간 이동 방향을 바꾸게 된다. 마찬가지로 이니셜이 왼쪽으로 1칸씩 이동하는 경우, 첫 번째 이니셜이 1번째 index에 오는 경우 이동 방향을 바꾸게 된다.
    - LED의 경우, DOT에 표현된 digit에 해당되는 값과 같은 번호인 LED가 turn on될 수 있도록 한다.
    - TIMER\_CNT회만큼 타이머 디바이스가 적절하게 수행되며 FPGA 모듈을 제어한 다음, FND를 0000, DOT와 LED를 turn off 할 수 있도록 한다. LCD의 경우 “Time’s up! 0\nTurn off in x…”을 출력하도록 하며 이 중, x는 3으로 설정되어 3초 후에 LCD의 내용을 clear하게 된다. X의 경우 interval (= 1 second)마다 값이 감소하며 0이 되는 순간 LCD가 clear out될 수 있도록 timer를 설정한다.

**III. 추진 일정 및 개발 방법**

- 자신들이 설정한 개발 목표를 달성하기 위한 개발 일정을 설정하고, 각 요소 문제를 해결하기 위해서 어떤 방법을 사용할 지 기술할 것.

**가. 추진 일정**

1. **개발 방법**
   1. 응용 어플리케이션

응용 어플리케이션에서는 target device에서 host에서 compile한 실행파일을 실행할 때 TIMER\_INTERVAL, TIMER\_CNT, TIMER\_INIT의 값을 같이 argument로 넘겨주게 된다.

이 때, argument의 수가 총 4개가 되어야 하고, TIMER\_INTERVAL은 1 ~ 100, TIMER\_CNT는 1 ~ 200, TIMER\_INIT은 0이 3개, 0이 아니고 1 ~ 8의 범위의 숫자 하나로 구성되어 있어야 한다. 해당 조건을 만족하지 못하는 경우, error message가 출력될 수 있도록 하고 응용 프로그램을 종료한다.

(사진)

입력에 문제가 없었을 경우, ioctl을 통해 입력받은 TIMER\_INTERVAL, TIMER\_CNT, TIMER\_INIT을 바탕으로 타이머 디바이스의 초기화를 진행할 수 있도록 한다.

(사진)

이후, while()문과 read()를 통해 Achroimx 보드의 Reset버튼이 눌러지는지 device driver를 통해 모니터링을 하다가 Reset 입력이 주어지는 경우 while()문을 탈출하여 ioctl을 통해 타이머 디바이스의 구동이 시작될 수 있도록 한다.

(사진)

ioctl()을 성공적으로 전송했다면 close()를 통해 driver file을 닫고 응용 프로그램이 종료될 수 있도록 한다.

* 1. 타이머 디바이스

타이머 디바이스의 작동을 제어할 디바이스 드라이버를 구현한다. 타이머 디바이스는 FND, DOT, LCD, PUSH\_SWITCH, LED를 제어할 수 있어야 한다. 이를 가능하게 하기 위해 디바이스 드라이버의 설치가 진행되는 init() 함수에서 FPGA 모듈들을 제어할 수 있도록 모듈의 physical address와의 매핑을 진행한다.

(사진)

디바이스 드라이버의 등록을 진행하며 init()에서 register\_chrdev()를 호출해 유효한 디바이스 드라이버로 등록되며 exit()에서의 unregister\_chrdeV()를 통해 유효한 디바이스 드라이버의 목록에서 삭제된다. 한편, 응용 프로그램에서 주어지는 명령에 대해 file\_operations를 정의하는 dirver\_fops 구조체에서 특정 작동에 실행되는 함수를 현재 구현하는 driver module에 구현된 함수로 지정하여 해당 함수에 정의된 작동을 할 수 있게 된다.

(사진)

타이머의 작동에 필요한 자료구조를 정의한다. 해당 과제에서는 timer의 interval, 작동 횟수등이 주어지며 이를 interval, cnt에 저장하도록 한다. 타이머가 interval이 지남에 따라 FPGA 모듈의 값도 갱신되는데 이 때, FND의 경우 현재 값이 00X0이고 (X+1) % 8 ~ (X-1) % 8의 범위로 갱신되었다면 다음 갱신에는 000X로 갱신되어야 한다. 이를 수월하게 관리할 수 있도록 init을 선언했고 해당 변수에서 TIMER\_INIT으로 전달되는 값을 관리하게 된다.   
그 외에도 LCD의 제어에 필요한 본인의 학번, 이니셜을 관리하기 위해 st\_id와 name\_initial을 선언했고 first\_initial, last\_initial, dir을 통해 현재 LCD가 타이머의 interval이 지남에 따라 어느 방향으로 갱신될 지 관리할 수 있도록 했다. (사진)  
driver\_ioctl()함수에서는 응용 프로그램에서 ioctl() 명령이 주어지는 경우, 전달된 parameter 중 cmd의 값에 맞게 작동이 처리될 수 있도록 한다. 명령에는 두 가지 종류가 있고 각각 IOCTL\_SET과 IOCTL\_START이다.

(#define 사진)  
IOCTL\_SET은 응용 프로그램을 통해 전달된 TIMER\_INTERVAL, TIMER\_CNT, TIMER\_INIT을 값을 이용해 앞서 언급한 타이머에서 사용하는 자료구조의 초기화를 진행한다. 응용 프로그램에서 전달되는 string에 해당 정보들이 저장되어 있고, 적절히 3, 3, 4의 길이를 토대로 자료구조에 저장될 수 있도록 한다.

(사진)

TIMER\_CNT로 전달되는 값을 통해 해당 문자열의 길이를 구한다. 해당 정보는 후에 interval이 지나며 LCD에 표시되는 TIMER\_CNT의 정보를 갱신하는 것에 사용된다.  
(사진)

학번, 이니셜 등을 LCD의 갱신에서 사용될 수 있도록 자료구조의 초기화를 진행하며 해당 작업을 마쳤다면 타이머 디바이스에서 사용할 fpga 모듈들의 초기화와 timer\_init()을 통해 타이머의 초기화 진행될 수 있도록 한다.   
(사진)  
IOCTL\_START의 경우, 응용 프로그램에서 명령이 주어질 때 실제로 구동이 진행되도록 하며 driver\_set()을 호출하는 것으로 timer를 등록해 interval이 지날 때마다 timer에 등록된 function의 작동이 수행될 수 있도록 한다.   
(사진)  
Timer에 등록되는 function인 timer\_blink()에서는 매 interval마다 FPGA 모듈의 값을 갱신할 수 있도록 한다. 각각 fpga\_fnd\_blink(), fpga\_lcd\_blink(), fpga\_dot\_blink(), fpga\_led\_blink()로 모듈의 제어를 실행하며 이는 timer\_cnt의 값이 1 이상일 때까지 진행된다.   
(사진)  
다만, timer\_cnt가 0이 되는 경우, 질의응답란에 답변을 참조하여 해당 timer의 interval이 지난 다음 “Time’s up!…”와 같은 종료 안내 메시지가 출력되는 것이 아닌, timer\_cnt가 만료된 직후 종료 안내 메시지가 출력될 수 있도록 예외처리를 해주었다.

(사진)  
이후의 timer는 새로운 timer인 timer2로 해당 timer를 timer list에 등록 후, function으로 timer\_after\_timer\_cnt()를 선언하여 실행할 수 있도록 했다.

(사진)  
timer\_after\_timer\_cnt()에서는 LCD에서 종료 되기 x초 전이라는 메시지가 갱신될 수 있도록 했고 x는 3부터 시작하여 interval인 1초가 지날 때마다 1씩 감소하게 되고 timer\_cnt가 0이 된다면 LCD를 blank line으로 초기화하며 타이머 디바이스의 작동이 종료된다.  
(사진)

**IV. 연구 결과**

- 최종 연구 개발 결과를 자유롭게 기술할 것.

**V. 기타**

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