1. 개요

타이머 카운터와 스위치 인터럽트를 사용하여 스톱워치를 구현해본다.

2. 프로그램 구조 설명

2.1 함수에 대한 설명

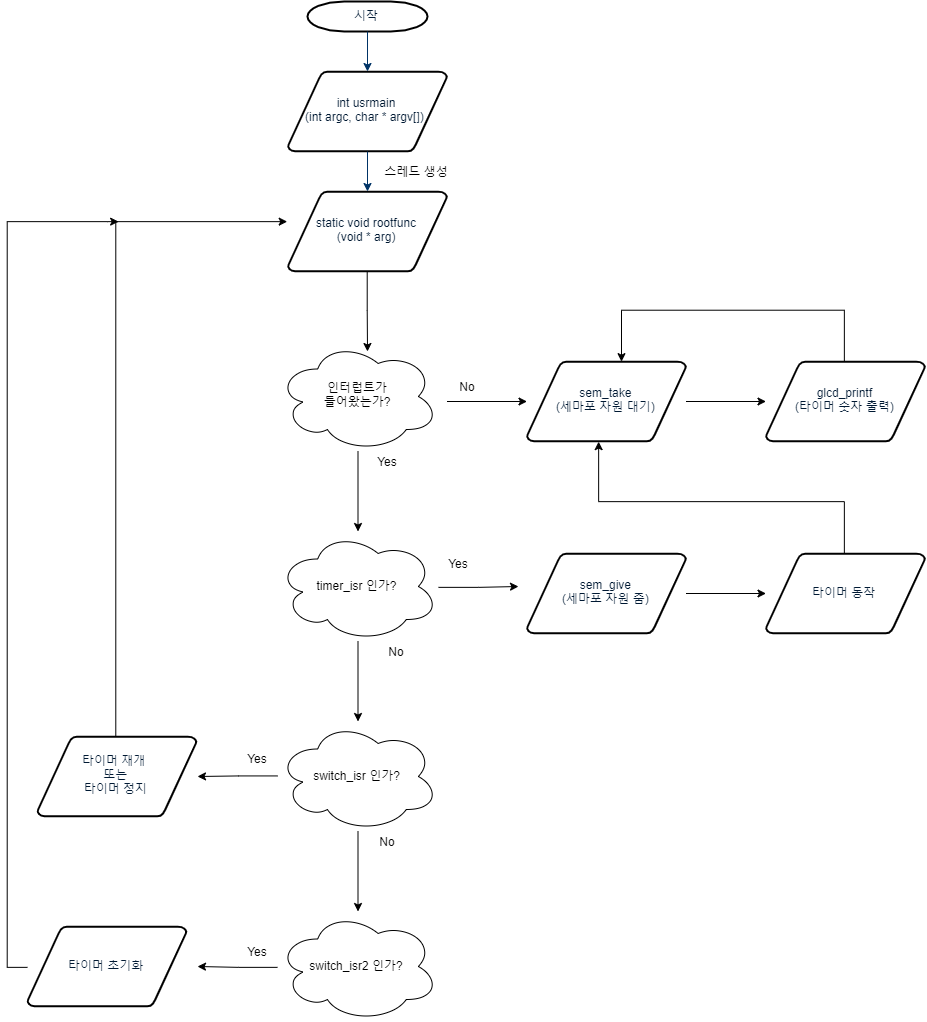
static void rootfunc(void \*arg) { // 항상 돌고 있는 함수(lcd에 print를 맡고있다)

static void timer\_isr(void) { // 1초마다 한번씩 인터럽트 함(타이머 역할)

static void switch\_isr(void) { // SW1 스위치 눌렀을때 인터럽트 함수(타이머 정지,재개)

static void switch\_isr2(void) { // SW2 스위치 눌렀을때 인터럽트 함수(타이머 초기화)

2.2 다이어그램



3. 실행 결과

동영상 결과를 같이 첨부합니다.

(usrmain 실행동영상.mp4)

**(실행 결과는 자신의 컴퓨터환경에서 실행한 결과 캡쳐 및 보드 동작 사진을 문서에 첨부해서 제출합니다. 실행결과가 성공한 경우 뿐만 아니라 실패한 경우도 포함하시기 바랍니다.**

**올린 코드를 실행한 결과와 사진으로 첨부한 결과가 다를 경우 감점합니다.)**

4. 고찰 (자세히 쓸 것)

이번 프로그래밍은 인터럽트를 이용한 프로그래밍이었습니다. 프로그래밍 부분에 있어서는 어려 운 점이 없었지만 타이머 인터럽트를 구현하는 부분을 이해하는 것이 어려웠습니다.

또한 인터럽트에 대한 지식을 알고 프로그래밍 하는 것이기 때문에 인터럽트 핸들러

함수에 최대한 적은 코드를 넣어서 메인 함수가 인터럽트 함수 때문에 돌지 못하는 시간을

최대한 줄이려고 노력했고, 사용한 인터럽트 프로그래밍이 앞으로 어떻게 사용될 수 있을지 고 민하며 프로그래밍했습니다. 좀 더 인터럽트에 익숙해지도록 노력하겠습니다.

5. 프로그램 소스 파일 (주석을 꼭 달 것)

/\*

Copyright (C) 2009 Sung Ho Park

Contact: ubinos.org@gmail.com

This file is part of the exe\_helloworld component of the Ubinos.

GNU General Public License Usage

This file may be used under the terms of the GNU

General Public License version 3.0 as published by the Free Software

Foundation and appearing in the file license\_gpl3.txt included in the

packaging of this file. Please review the following information to

ensure the GNU General Public License version 3.0 requirements will be

met: http://www.gnu.org/copyleft/gpl.html.

GNU Lesser General Public License Usage

Alternatively, this file may be used under the terms of the GNU Lesser

General Public License version 2.1 as published by the Free Software

Foundation and appearing in the file license\_lgpl.txt included in the

packaging of this file. Please review the following information to

ensure the GNU Lesser General Public License version 2.1 requirements

will be met: http://www.gnu.org/licenses/old-licenses/lgpl-2.1.html.

Commercial Usage

Alternatively, licensees holding valid commercial licenses may

use this file in accordance with the commercial license agreement

provided with the software or, alternatively, in accordance with the

terms contained in a written agreement between you and rightful owner.

\*/

/\* -------------------------------------------------------------------------

Include

------------------------------------------------------------------------- \*/

#include "../ubiconfig.h"

// standard c library include

#include <stdio.h>

#include <sam4e.h>

// ubinos library include

#include "itf\_ubinos/itf/bsp.h"

#include "itf\_ubinos/itf/ubinos.h"

#include "itf\_ubinos/itf/bsp\_fpu.h"

// chipset driver include

#include "ioport.h"

#include "pio/pio.h"

// new estk driver include

#include "lib\_new\_estk\_api/itf/new\_estk\_led.h"

#include "lib\_new\_estk\_api/itf/new\_estk\_glcd.h"

#include "lib\_switch/itf/lib\_switch.h"

#include "lib\_sensor/itf/lib\_sensor.h"

#include "lib\_EV3\_sensor/itf/lib\_EV3\_sensor.h"

#include "lib\_sensorcalib/itf/lib\_sensorcalib.h"

#include "lib\_motor\_driver/itf/lib\_motor\_driver.h"

// custom library header file include

//#include "../../lib\_default/itf/lib\_default.h"

#include "../../lib\_new/itf/lib\_new.h"

#include "lib\_new\_estk\_api/itf/new\_estk\_led.h"

#include "lib\_new\_estk\_api/itf/new\_estk\_glcd.h"

#include "lib\_motor\_driver/itf/lib\_motor\_driver.h"

#include "lib\_switch/itf/lib\_switch.h"

#include "itf\_ubinos/itf/bsp\_intr.h"

//chipset driver include

#include "ioport.h"

#include "pio/pio.h"

/\*global variables \*/

sem\_pt \_g\_sem; //api용 세마포어 포인터 형 정의

int count = 0;

#define TIMER\_STOP 0 //sw1이 눌러졌을때

#define TIMER\_RUN 1 //sw1이눌러져있지 않을때

static char g\_state = TIMER\_RUN;

/\* ----------------

\* PROTOTYPES

\* \*/

static void rootfunc(void \*arg);

static void switch\_isr(void);

static void switch\_isr2(void);

static void timer\_isr(void);

//static void print\_lcd(void \*arg);

/\*function Definitions \*/

int usrmain(int argc, char \* argv[]) {

int r;

printf("\n\n\n\r");

printf("==========");

printf("exe\_intr\_test(build time: %s%s) \n\r", \_\_TIME\_\_, \_\_DATE\_\_);

//r = task\_create(NULL, print\_lcd, NULL, task\_getmiddlepriority(), 256, "print\_lcd");

r = task\_create(NULL, rootfunc, NULL, task\_getmiddlepriority(), 256, "root");

if (0 != r) {

logme("fail at task\_create\r\n");

}

PMC->PMC\_PCER0 = 1 << ID\_TC3;

/\*Timer Configuration \*/

TC1->TC\_CHANNEL[0].TC\_CCR = TC\_CCR\_CLKDIS; //counter clock disable command

TC1->TC\_CHANNEL[0].TC\_IDR = 0xFFFFFFFF; //timer interrupt disable command

/\*Channel Mode (TCLK1 Select(0) and RC Compare Trigger Enable \*/

TC1->TC\_CHANNEL[0].TC\_CMR = (TC\_CMR\_TCCLKS\_TIMER\_CLOCK5 | TC\_CMR\_CPCTRG);

TC1->TC\_CHANNEL[0].TC\_CCR = TC\_CCR\_CLKEN; //counter clock enable command

TC1->TC\_CHANNEL[0].TC\_IER = TC\_IER\_CPCS; //RC compare interrupt enable

/\*interrupt handler \*/

intr\_connectisr(TC3\_IRQn, timer\_isr, 0x40, INTR\_OPT\_\_LEVEL);

/\*enable interrupt\*/

intr\_enable(TC3\_IRQn); // NVIC interrupt enable command register

/\*Timer Interrupt period configuration \*/

TC1->TC\_CHANNEL[0].TC\_RC = 32768; //set timer for about 1s interval

/\*Start timer\*/

TC1->TC\_CHANNEL[0].TC\_CCR = TC\_CCR\_SWTRG; //software trigger is performed

ubik\_comp\_start();

return 0;

}

static void rootfunc(void \*arg) { // 항상 돌고 있는 함수

switch\_init(switch\_isr, switch\_isr2);

glcd\_init();

//기본 lcd 표시

glcdGotoChar(0, 0);

glcd\_printf("SEC : %3d", count);

for (;;) {

sem\_take(\_g\_sem); // 세마포 가져감(기다림), 1초마다 glcd\_printf 합니다.

glcdGotoChar(0, 0);

glcd\_printf("SEC : %3d", count);

task\_sleep(100);

}

}

static void timer\_isr(void) { // 1초마다 한번씩 인터럽트 함

unsigned int reg;

sem\_give(\_g\_sem); //세마포 자원을 줌

if (g\_state == TIMER\_STOP) { // 초기값이 TIMER\_RUN이기 때문에 초기 상태는 타이머가 동작하지않음

count++;

reg = TC1->TC\_CHANNEL[0].TC\_SR;

printf("HW\_TIMER [TC:%d] \r\n", reg);

}

}

static void switch\_isr(void) { // SW1 스위치 눌렀을때 인터럽트 함수

if (g\_state == TIMER\_STOP) //처음 시작 및 스위치가 홀수로 눌러졌을 때

{

g\_state = TIMER\_RUN; //다음 스위치가 눌러지면 인터럽트가 발생하여 switch\_isr함수로 들어오게 되고 미리 변수를 바꿔줌

}

else if (g\_state == TIMER\_RUN)

{

g\_state = TIMER\_STOP;

}

}

static void switch\_isr2(void) { // SW2 스위치 눌렀을때 인터럽트 함수

count = 0; //SW2가 눌러지면 초기화

glcdGotoChar(0, 0);

glcd\_printf("SEC : %3d", count);

}

6. 자료 출처 (어떤 책을 참고했으며 어떤 웹사이트를 참고했는지 출처를 적습니다)

교수님의 ppt를 대부분 참고했습니다.