1. 개요

색상에 따라 모터의 속도가 달라지는 HW02의 코드에서 task를 분리하여 재구현해본다.

2. 프로그램 구조 설명

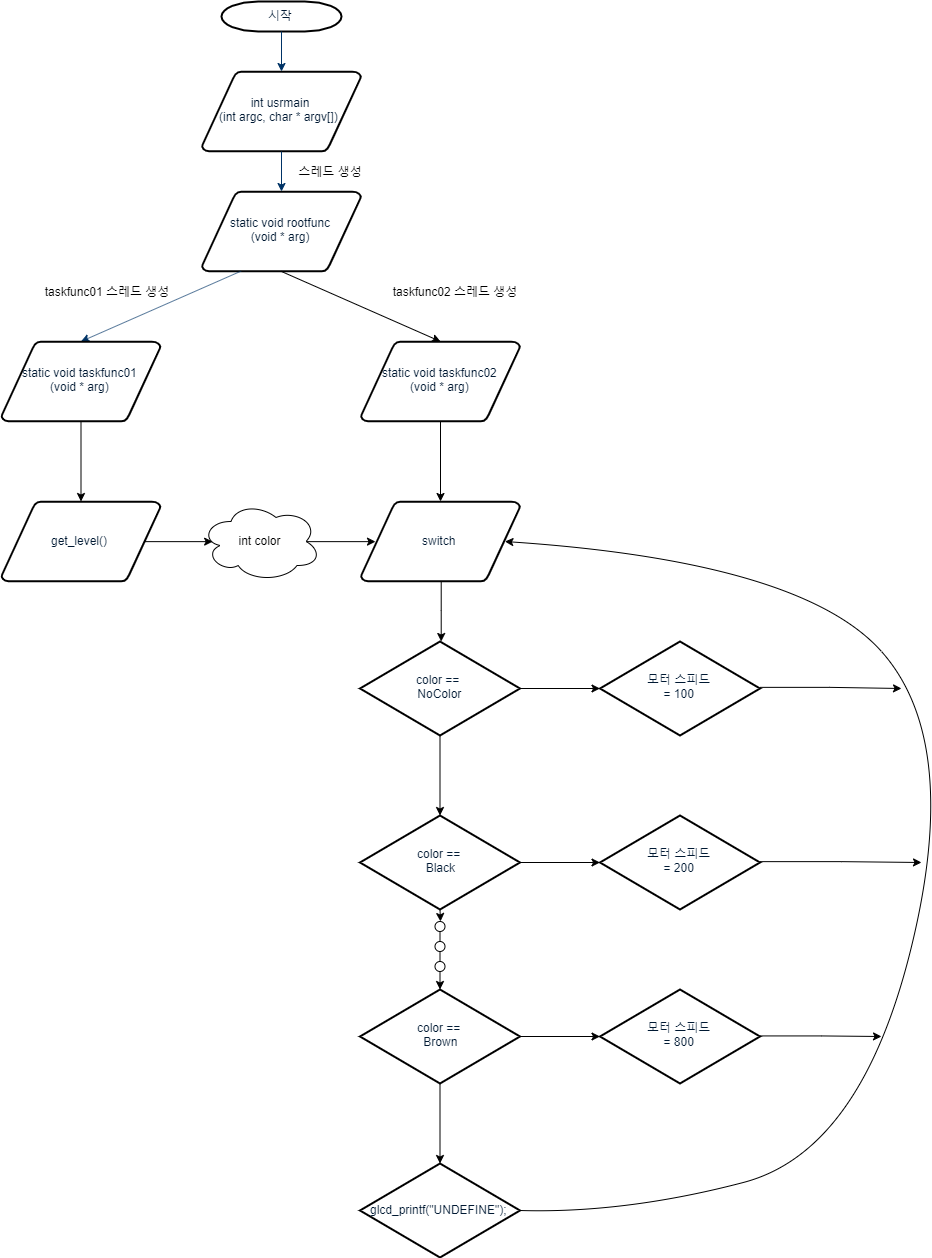
2.1 함수에 대한 설명

static void taskfunc01(void \* arg) { // 색상을 체크해서 color에 저장하는 테스크 1

static void taskfunc02(void \* arg) { // 색상에 따라 모터를 제어하는 테스크 2

task\_create(………) //테스크를 하나 생성합니다.

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)

1. task\_create 함수와 변수의 의미

task\_create 함수는 테스크를 생성하는 함수입니다.

task\_pt task\_handle01; => 생성된 테스크의 주소를 저장할 포인터입니다.

2. task\_create에서 priority를 변경하면 어떻게 되는지

Ready상태에 있는 테스크 중에서 priority가 높은 순서대로 먼저 실행됩니다.

같은 priority라면 Round-Robin으로 실행됩니다.

3. bsp\_busywaitms의 의미(어떤 방식으로 동작하는지)

task\_sleep은 running 상태의 테스크를 block 상태로 보내버리기 때문에 running 상태

에 다른 테스크를 올릴 수 있습니다. 하지만 bsp\_busywaitms의 경우 테스크를 running

상태에서 일정시간 동안 잡고 있는 것이기 때문에 그 시간동안 다른 테스크가 돌아 갈

수 없습니다.

4. 2개의 task 중 어느 task의 우선순위를 더 높게 주어야 할지 고민해 볼 것

저의 생각에는 모터를 제어하기 위해서는 먼저 광센서를 이용해 색상을 체크하는 것이

먼저이므로 컬러센서 task가 우선순위가 더 높아야 할 것 같습니다.

고찰 :

이번 프로그래밍은 ubinos의 스레드인 테스크를 알아보았습니다. 운영체제 시간에 배웠던

Pthread와 사용법이 거의 유사했고, 개념도 거의 같았기 때문에 코딩하는데는 어려움이 없었고

C언어의 스레드를 다시 한번 복습해보는 시간이 되어서 유익했습니다.

또한 bsp\_busywaitms라는 새로운 개념도 배웠고 ubinos에서 멀티스레드 프로그래밍을 하는

방법을 알게 되어서 좋았습니다. 앞으로 멀티스레드 처리를 어떻게 하느냐가 굉장히 중요할

것 같은데 멀티 스레드를 어떻게 구현해야 더 효율적이고 알맞은 코딩인지 스스로 코드를

짜보며 공부하도록 하겠습니다.

5. 프로그램 소스 파일 (주석을 꼭 달 것)

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

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"

// user header file include

#define IDLE\_PRIORITY 0

**// 색깔을 변수로 define합니다.**

#define NoColor 0

#define Black 1

#define Blue 2

#define Green 3

#define Yellow 4

#define Red 5

#define White 6

#define Brown 7

task\_pt task\_handle01;

task\_pt task\_handle02;

int color; **// 컬러를 입력받아 저장하는 변수**

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

Global variables

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

static void rootfunc(void \* arg);

static void taskfunc01(void \* arg) { **// 색상을 체크해서 color에 저장합니다.**

while (1) {

color = ev3\_sensor\_get(0);

task\_sleep(50);

}

}

static void taskfunc02(void \* arg) { **// 색상에 따라 모터를 제어합니다.**

glcd\_clear();

while (1) {

glcdGotoChar(0, 0);

switch (color)

{

case NoColor:

glcd\_printf("No color VAL : %3d ", ev3\_sensor\_get(0));

motor\_set(0, 100);

break;

case Black:

glcd\_printf("Black VAL : %3d ", ev3\_sensor\_get(0));

motor\_set(0, 200);

break;

case Blue:

glcd\_printf("Blue VAL : %3d ", ev3\_sensor\_get(0));

motor\_set(0, 300);

break;

case Green:

glcd\_printf("Green VAL : %3d ", ev3\_sensor\_get(0));

motor\_set(0, 400);

break;

case Yellow:

glcd\_printf("Yellow VAL : %3d ", ev3\_sensor\_get(0));

motor\_set(0, 500);

break;

case Red:

glcd\_printf("Red VAL : %3d ", ev3\_sensor\_get(0));

motor\_set(0, 600);

break;

case White:

glcd\_printf("White VAL : %3d ", ev3\_sensor\_get(0));

motor\_set(0, 700);

break;

case Brown:

glcd\_printf("Brown VAL : %3d ", ev3\_sensor\_get(0));

motor\_set(0, 800);

break;

default:

glcd\_printf("UNDEFINE ");

break;

}

task\_sleep(50);

}

}

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

Prototypes

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

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

Function Definitions

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

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

int r;

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

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

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

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

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

if (0 != r) {

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

}

ubik\_comp\_start();

/\* code below ubik\_comp\_start function never be executed \*/

return 0;

}

static void rootfunc(void \* arg) {

**//lcd와 모터 초기화**

glcd\_init();

motor\_init();

**//lcd clear**

glcd\_clear();

**//컬러센서 초기화**

ev3\_sensor\_init(0, COL\_COLOR);

**//테스크 2개 생성**

**// 색상체크 테스크**

if (task\_create(&task\_handle01, taskfunc01, NULL, IDLE\_PRIORITY + 2, 256, "task\_01") != 0) {

printf("테스크 생성실패");

}

**// 색상에 따른 모터제어 테스크**

if (task\_create(&task\_handle02, taskfunc02, NULL, IDLE\_PRIORITY + 1, 256, "task\_02") != 0) {

printf("테스크 생성실패");

}

}

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

교수님의 ppt를 대부분 참고했습니다.