1. 개요

교차 컴파일에 익숙해지며 malloc으로 동적할당을 하여 memory overflow를

유도한다.

2. 프로그램 구조 설명

2.1 함수에 대한 설명

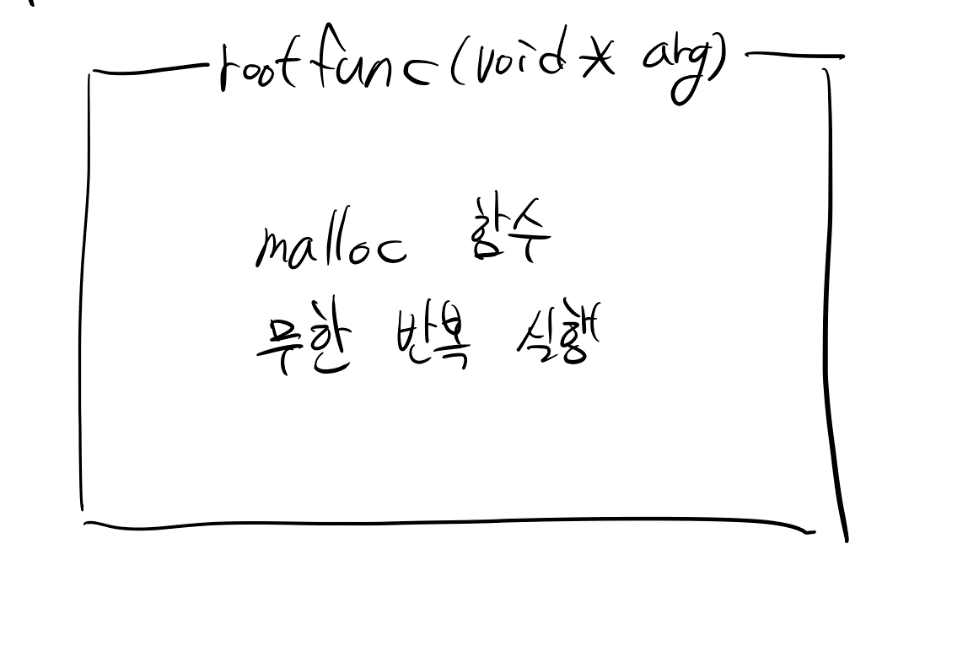
(포인터유형\*)malloc(동적할당 size);

: 동적할당을 위한 malloc 함수입니다.

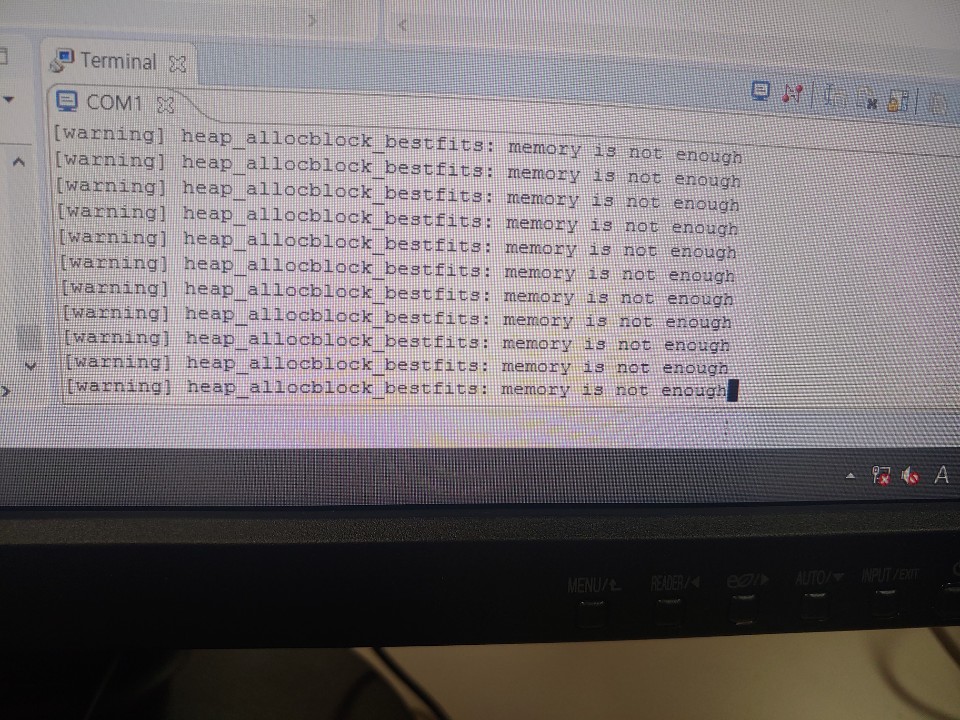
할당된 메모리를 free 해주지 않고 반복적으로

Malloc을 사용하면 memory overflow 현상이 일어납니다.

2.2 다이어그램



3. 실행 결과



**(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 과제는 타겟 시스템은 ESTK 보드, 호스트 시스템은 데스크탑으로 하는

교차 컴파일에 대한 실습이었습니다. 코딩 난이도는 c언어를 복습하는 정도로

어렵지 않았지만, 교차 컴파일을 많이 접해보지 않았기 때문에 익숙해지는데

시간이 걸렸습니다. 특히 처음에는 memory overflow 경고 창이 스스로 뜰 것으로

알고 있었는데 경고 창이 뜨지 않아 헤맸었습니다. 결국 COM1포트 시리얼 창을

따로 띄워 줘야 경고창을 볼 수 있음을 알게 되었습니다.

앞으로 좀 더 교차 컴파일에 익숙해지는 것이 필요할 것 같습니다.

5. 프로그램 소스 파일 (주석을 꼭 달 것)

//필요한 라이브러리들을 include 합니다.

#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"

// custom library header file include

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

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

// user header file include

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

Global variables

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

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

Prototypes

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

static void rootfunc(void \* arg);

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

Function Definitions

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

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

int r;

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

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

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

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

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

if (0 != r) {

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

}

ubik\_comp\_start();

return 0;

}

static void rootfunc(void \* arg) {

int\* b = (int\*)malloc(sizeof(5000)); //처음 int 포인터 b를 선언하고 멜록을 시도합니다.

for (;;) {

// for문 안에서 반복적으로 사이즈가 큰 메모리를 동적할당합니다.

b = (int\*)malloc(sizeof(5000));

}

}

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

대부분 교수님의 ppt를 참고했습니다.