TI DSP, MCU 및 Xilinx Zynq FPGA 프로그래밍 전문가 과정

1 개월차 시험 (18.03.16)

학생 – 이우석
colre99@naver.com

<카페에 올려진 시험문제 57 문항>

1. 이것이 없으면 사실상 C 언어를 사용할수 없다.
   C 언어에서 함수를 호출하기 위해서도 이것은 반드시 필요하다.
   이와 같은 이유로 운영체제의 부팅코드 에서도 이것을 설정하는
   작업이 진행되는데 이것은 무엇일까?
: 헤더파일

2.
3.

4. goto 를 도대체 왜 사용해야만 할까?
: 특정한 상황을 아주 간결하게 해결할 수 있고, 특정위치에 옮겨주는 역할을 하는 함수이기에 프로그램의 실행을 특정위치로 이동시키기에 괜찮다.

5. 포인터 크기에 대해 알고 있는대로 기술하시오.

: 포인터의 크기는 하드웨어가 몇 비트를 지원하는냐에 따라 달라진다. 보통은 4 비트. 즉, 32 비트 플랫폼에서의 포인터는 4 바이트 정수, 64 비트 플랫폼에서의 포인트는 8 바이트이다.

6.
7.
8.
9.
10.
11.
12.
13.
14.
15.
16.
17.
18.
19.

20.
21.
22.
23.
24.
25.
26.
27.
28.
29.
30.

31. 변수의 정의를 기술하시오.

: 변수란 함수내에서 데이터를 저장하는 메모리공간

32. 포인터의 정의를 기술하시오.

:

간단하게는, 특정변수에 대한 메모리 주소를 담을수 있는 변수

(= 메모리의 주소를 가르키는 자료형 or 주소를 저장하는 변수)

33. 함수 포인터의 정의를 기술하시오.

:

간단하게는, 특정 함수에 대한 메모리 주소를 담을수 있는 포인터.

데이터 값을 가리키는 대신, 함수 포인터는 메모리 내에서 실행 가능한 코드를 가리킨다. 역참조될 때, 함수 포인터는 가리키는 함수를 보통의 함수 호출처럼 작동시키고 인자를 보내는데 사용될 수 있다.

34.

35.

36. 메모리 계층구조에 대해 기술하시오.

:

구조는 속도순서로 Reister, Cache, Memory, Disk 가 있다.

용량순서로는 속도순서와 반대로 Disk, Memory, Cache, Register 가 있다.

37. C 언어의 기본 메모리 구조에 대해 기술하시오.

: 메모리 구조에는 Stack, Heap, Data, Text 가 있다.

지금 순서로는 메모리상의 높은 구역, 낮은구역 순이다.

Stack 에는 기본자료형 (int, char, double, void, float, 지역변수), 표준버퍼 , 시스템 (os) 임시데이터 등이 있다.

Heap 에는 배역, 클래스, 컬렉션 등의 동적변수가 있다.

Data 에는 전역변수, 정적변수가 있다.

Text 에는 함수코드, 함수 구현부 등이 있다.

38.

39.

40. 리눅스에서 디버깅을 수행하기 위한 프로그램 컴파일 방법을 기술하시오.

(임베디드 애플리케이션 30 번문항)

: C 코드를 짠 파일이 있다면(ex) test.c) 명령어 gcc -g -o debug test.c 를 입력하면 debug 파일이 생기고, debug 파일을 확인(디버깅)하려면 명령어 gdb 를 입력하면 된다.

41.
42.
43.
44.
45.
46.
47.
48.
49.
50.

51. gdb 를 사용하는 이유를 기술하라.

: gdb 를 사용하는 이유는 c 코드를 컴파일 한후, 컴파일 에서 오류가 있는지 없는지 확인하려는 이유와 메모리 레이아웃을 그리기 편하고 스택에 있는 값을 확인하기

편하기 때문이다. 그리고 esp, ebp 의 위치와 가 변수가 얼마나 떨어져 있는지 등등 확인하기 편하기 때문이다.

52. 기계어 레벨에서 스택을 조정하는 명령어는 어떤것들이 있는가?

: push, pop, add, callq 가 있다.

53.

54. 가위 바위 보 게임을 만들어보자. 프로그램을 만들고 컴퓨터랑 배틀한다. (임베디드 애플리케이션 문제 36 번)

:

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <time.h>
4
5 int main(void)
```

```
6 {
7    int user;
8    int computer;
9    int result[3] = {0,};
10    int cnt = 0;
11
12    srand((unsigned)time(NULL));
13
14    while(1)
15    {
16       computer = rand()%3+1;
17       printf("\n");
18       printf("ㅁㅁㅁㅁㅁㅁㅁㅁ가위 바위 보 게임ㅁㅁㅁㅁㅁㅁㅁㅁㅁㅁㅁㅁ\n");
19       printf("\n");
20       printf("1. 가위  2. 바위  3. 보       0. 종료\n\n");
21       printf("\n");
22       printf("뭐 낼건지 입력하세요: ");
23       scanf("%d", &user);
24       printf("\n");
25       printf("\n");
26       printf("\n");
```

```
27      printf("\n");
28      printf("\n");
29        if(user > 0 && user < 4)
30        {
31          cnt++;
32
33          printf("\n 컴퓨터는 : %s\n", (computer == 1 ? "가위" : computer == 2 ? "바위" : "보" ));
34          printf("\n");
35          printf("\n");
36          printf("사용자는 : %s\n\n", (user == 1 ? "가위" : user == 2 ? "바위" : "보" ));
37
38          if(computer == user)
39          {
40            printf("\n");
41            printf("\n");
42            printf("앗!..비겼잖아ㅠ.");
43            result[1]++;
44          }
45
46          else if(((computer == 1) && (user == 3)) ||
47                  ((computer == 2) && (user == 1)) ||
```

```
48              ((computer == 3) && (user == 2)))
49          {
50              printf("\n");
51              printf("\n");
52              printf("졌다.....졌다아아아아아아아아\n\n");
53              result[2]++;
54          }
55
56          else
57          {
58              printf("\n");
59              printf("\n");
60              printf("내가 이기는건 당연하지 하하하하하.\n\n");
61              result[0]++;
62          }
63              printf("\n");
64              printf("\n");
65              printf("\n");
66              printf("\n");
67              printf("이번판 결과는 %d 전 %d 승 %d 무 %d 패\n\n", cnt, result[0], result[1], result[2]);
68      }
```

```
69
70
71          else
72          {
73            if(user == 0)
74            {
75              printf("이제 여기서 그만~~~");
76              break;
77            }
78              printf("숫자를 제대로 입력해 주세요\n\n");
79          }
80
81
82          system("pause");
83          system("cls");
84      }
85
86    return 0;
87 }
```

55.
56.
57.

< 임베디드 애플리케이션 분석 41 문제 >

1. 이것이 없으면 사실상 C 언어를 사용할수 없다.
C 언어에서 함수를 호출하기 위해서도 이것은 반드시 필요하다.
이와 같은 이유로 운영체제의 부팅 코드에서도 이것을 설정하는 작업이 진행되는데 이것은 무엇일까?
: 헤더파일

2.

3.

4. goto 의 정체성은? goto 를 도대체 왜 사용해야만 할까?
: 특정한 상황을 아주 간결하게 해결할 수 있고, 특정위치에 옮겨주는 역할을 하는 함수이기에 프로그램의 실행을 특정위치로 이동시키기에 괜찮다.

5. 포인터 크기에 대해 알고있는대로 기술하시오.
:포인터의 크기는 하드웨어가 몇 비트를 지원하는냐에 따라 달라진다. 보통은 4 비트. 즉, 32 비트 플랫폼에서의 포인터는 4 바이트 정수, 64 비트 플랫폼에서의 포인트는 8 바이트이다.

6.

7. 이중 배열을 함수의 인자로 입력 받아 3 by 3 행렬의 곱셈을 구현하시오.

8.

9.

10. 파이프라인이 깨지는 경우 어떠한 이점이 있는지 기술하시오.

: 파이프라인이 깨지면 컴퓨터의 프로세서는 메모리에서 첫 번째 명령어를 가지고 와서, 그것이 요구하는 연산을 수행하고, 그리고 나서 다음번 명령어를 메모리로부터 가져오는 식으로 동작한다.

11.

12.

13.

14.

15. Intel Architecture 와 ARM Architecture 의 차이점은 ?

: Arm 아키텍처가 Intel 아키텍처 보다 훨씬 저전력이고 최적의 성능을 보여준다. Intel 은 다소 복잡한 구조로 되있기에 기능과 성능이 더 높을순 있지만, 그만큼 높은 전력과 높은 발열을 낸다는 단점이 있다. 그에 반해, ARM 은 임베디드의 부분

으로서 낮은 전력으로 간소하고 최적의 성능을 낼수 있다는 장점이 있다.

16.

17.

18.

19. int *p[3] 와 int (*p)[3] 는 같은 것일까 ? 다른 것일까 ? 이유를 함께 기술.
: 다른것. 이유는 전자 p[3]은 p 를 가르키는 포인터변수. 후자(*p)[3]은 배열을 가리키는 포인터 변수.

20.

21. 변수의 정의를 기술하시오.
: 변수란 함수내에서 데이터를 저장하는 메모리공간

22. 포인터의 정의를 기술하시오.
: 간단하게는, 특정변수에 대한 메모리 주소를 담을수 있는 변수
(= 메모리의 주소를 가르키는 자료형 or 주소를 저장하는 변수)

23.

24. 함수 포인터의 정의를 기술하시오.
: 간단하게는, 특정 함수에 대한 메모리 주소를 담을수 있는 포인터.
데이터 값을 가리키는 대신, 함수 포인터는 메모리 내에서 실행 가능한 코드를 가리킨다. 역참조될 때, 함수 포인터는 가리키는 함수를 보통의 함수 호출처럼 작동시키고 인자를 보내는데 사용될 수 있다.

25.

26. 메모리 계층 구조에 대해 기술하시오.
: 구조는 속도순서로 Reister, Cache, Memory, Disk 가 있다.
용량순서로는 속도순서와 반대로 Disk, Memory, Cache, Register 가 있다.

27. C 언어의 기본 메모리 구조에 대해 기술하시오.
: 메모리 구조에는 Stack, Heap, Data, Text 가 있다.
지금 순서로는 메모리상의 높은 구역, 낮은구역 순이다.

Stack 에는 기본자료형 (int, char, double, void, float, 지역변수), 표준버퍼 , 시스템

(os) 임시데이터 등이 있다.

Heap 에는 배역, 클래스, 컬렉션 등의 동적변수가 있다.

Data 에는 전역변수, 정적변수가 있다.

Text 에는 함수코드, 함수 구현부 등이 있다.

28.

29.

30. 리눅스에서 디버깅을 수행하기 위한 프로그램 컴파일 방법을 기술하시오.
: C 코드를 짠 파일이 있다면(ex) test.c) 명령어 gcc -g -o debug test.c 를 입력하면 debug 파일이 생기고, debug 파일을 확인(디버깅)하려면 명령어 gdb 를 입력하면 된다.

31. vi 에서 코드가 정렬이 잘 안되어 있다.
이 상황에서 어떻게 해야 예쁘고 깔끔하게 정렬되는가 ?

편집모드로 전환하기 위해 일단 ‘esc’ 를 누르고 난후, 커서를 정렬하고싶은 공간맨위 맨앞 위치 시킨후 ‘v’를 클릭후, 커서를 다시 정렬공간 맨아래 맨앞까지 다시 위치시킨다. 그러면 정렬 하려는 구역만큼 드래그 되는데 그때 “=” 키를 클릭하면 정렬완료.

32.

33. gdb 를 사용하는 이유를 기술하시오.
: gdb 를 사용하는 이유는 c 코드를 컴파일 한후, 컴파일 에서 오류가 있는지 없는지 확인하려는 이유와 메모리 레이아웃을 그리기 편하고 스택에 있는 값을 확인하기

편하기 때문이다. 그리고 esp, ebp 의 위치와 가 변수가 얼마나 떨어져 있는지 등등

확인하기 편하기 때문이다.

34. 기계어 레벨에서 스택을 조정하는 명령어는 어떤것들이 있는가 ?
(직접 연산을 하는 경우를 제외하고 기술하도록 한다. (4 가지))페
:push, pop, add, callq

35.

36. 가위 바위 보 게임을 만들어보자. 프로그램을 만들고 컴퓨터랑 배틀한다.
(카페용 문제에 답안 작성함.)

37.

38.

39.

40. 한달간 적은 시간이기도 하고 많은 시간이기도 했다.
한 달간 수업을 진행하면서 본인이 느낀점을 20 줄 이상 최대한 상세하게 기술하시오.
또한 앞으로에 대한 포부 또한 기술하길 바란다.

그리고 앞으로 어떤 일을 하고 싶은지 상세히 기술하도록 한다.

: 느낀점은 지금까지와는 다르게 여유를 가지며 지내면 큰일나겠다 싶었습니다.

앞으로는 여유없이 집에 도착하면 공부해야겠습니다. 프로젝트할때 제가 꼭 해보고

싶었던 바램들을 써보기 위해 정말 열심히 할것입니다. 앞으로 하고싶은 일은

드론 같은 휴대용 무선 비행 물체를 만들어서 군사용 및 생활용 으로 만드는 일을 하고싶습니다. 다방면으로 많이 쓰이게 여러 용도로 만들고 싶습니다. 그만큼 다 배우고 싶고 응용하고 싶고 그렇습니다.

41.

< 자료구조 문제 17 문제 >

1. 값이 1 ~ 4096 까지 무작위로 할당되어 배열에 저장되도록 프로그래밍 하시오.
(배열의 크기는 100 개 정도로 잡는다)
:

```
#include

int main(void)
```

```
{
int a[100];
        for(i = 0; i < 4097; i++)
                if( i == 4096 )
                    returun 1;
                else
                    a[i] = [i];

return 0;
}
```

2. [복합문제 1.2] 앞서 만든 코드를 아래 설명을 보고 개량 하도록 한다.

(각 배열은 물건을 담을 수 있는 공간에 해당한다.

앞서서 100 개의 공간에 물건들을 담았는데 공간의 낭비가 있을 수 있다.

이 공간의 낭비가 얼마나 발생했는지 파악하는 프로그램을 작성하시오.)

:

```
#include
int main(void)
{
    int a[100];
    int b;
    for(b = 0; b < 101; b++)
     {
        if(a[b] == NULL)
       {
        printf("empty space = %d\n");
      else if (a[b] != NULL)
            continue;
```

```
        else
            return 1;
    }
    return 0;
}
```

3. [복합문제 1.3] 앞서 만든 코드를 보다 더 개량한다.

(문제에서 확장하여 공간을 보다 효율적으로 관리하고 싶어서 4096, 8192, 16384 등의 4096 배수로 크기를 확장할 수 있는 시스템을 도입했다.

이제부터 공간의 크기는 4096 의 배수이고 최소 크기는 4096, 최대 크기는 131072 에 해당한다. 발생할 수 있는 난수는 1 ~ 131072 로 설정하고 이를 효율적으로 관리하는 프로그램을 작성하시오. (사실 리눅스 커널의 Buddy 메모리 관리 알고리즘임)

:

```
#include <stdio.h>


int main(void){
   int rand_num;
   srand((unsigned)time(NULL));
   int num[4096 * a];
   int a;
   int b;
   rand_num = rand();
   for(a = 4096; a < 131072; a++)
    {
       for(b = 1; b < 131072; b++)
              rand_num = rand() % 131072 + 1;
     }
```

```
  return 0;
}
```

4.
5.
6.
7.
8.
9.
10.
11.
12.
13.
14.
15.
16.
17.
