TI DSP,Xilinx zynq FPGA,MCU 및 Xilinx zynq FPGA 프로그래밍 전문가 과정

강사-INNOVA LEE(이상훈)

Gccompil3r@gmail.com

학생-윤지완

Yoonjw7894@naver.com

```
Void init_rand_arr(int *arr, int size)
{
int i;

for(i = 0; i < size; i++)
 {
redo:
 arr[i] = rand() % 10 + 1;

if(is_dup(arr, i))
 {
printf("%d dup! redo rand()₩n", arr[i]);
goto redo;
 }
 }
}
```

Rand()난수를 이용해 나누기 10을 나눈 값에서 1씩더하면서 arr[i]배열에 넣고 있는 부분이다

그 다음부분은 goto명령어가 특정부분에 동작하는건 아는데 위에 부분의 명령어의 해석을 잘못 하겠습니다.

```
Void print_arr(int *arr, int size)
{
int i;
```

```
for(i = 0; i < size; i++)
printf("arr[%d] = %d₩n", i, arr[i]);
}
```

이 부분은 위에 rand()를 이용해 넣은 값을 배열 몇번에 값이 얼마있다 알려주는 부분이다.

```
queue *get_queue_node(void)
                                        {
                                          queue *tmp;
                                          tmp = (queue *)malloc(sizeof(queue));
                                          tmp->link = NULL;
                                        return tmp;
                                        }

                                        voidenqueue(queue **head, int data)
                                        {
                                        if(*head == NULL)
                                          {
                                          *head = get_queue_node();
                                          (*head)->data = data;
                                        return;
                                          }

                                        enqueue(&(*head)->link, data);
                                        }
```

queue를 동작하는 부분이고 그전에 한 방법과 동일하다.

```
voidextract_even(queue *head, int *extract)
{
  queue *tmp = head;

while(tmp)
  {
if(!(tmp->data % 2))
  extract[extract_idx++] = tmp->data;
  tmp = tmp->link;
  }
}
```

Extract_even은 2로 나누어 0이 되면 if문이 발동해extract 배열에 head->data에 들어있던 짝수 난수들이 값을 넣는다.

main에서 extract[11]로 선언했다 그래서 함수 호출에서 포인터로 받은 것이다.

```
intmain(void)
        {
        int i, fd, len, sum = 0;
        char *convert[10] = {0};
        int arr[11] = {0};
```

```
char tmp[32] = {0};
int extract[11] = {0};
int size = sizeof(arr) / sizeof(int) - 1;
 queue *head = NULL;

srand(time(NULL));

init_rand_arr(arr, size);
print_arr(arr, size);

for(i = 0; i < size; i++)
enqueue(&head, arr[i]);

extract_even(head, extract);
```

이제 메인에서 arr,tmp,extract부분을 배열로 선언하고 함수 호출을 하고 있다. 그럼 위에 있던 함수들이 동작을해서 값을 arr 배열에 extract 배열에 값이 들어가는 것이다.

```
printf("₩nExtract:₩n");
                print_arr(extract, extract_idx);

                 fd = open("log.txt", O_CREAT | O_WRONLY | O_TRUNC, 0644);

                for(i = 0; i < extract_idx; i++)
                 sum += extract[i];

                sprintf(tmp, "%d", sum);
```

write(fd, tmp, strlen(tmp));

close(fd);

fd = open("log.txt", O_CREAT | O_WRONLY | O_TRUNC, 0644);

파일을 log.txt의 이름으로 만들고 파일이 없으면 만들고 갱신 쓰는 것이 가능하다.

그리고 for문을 이용해 extract의 짝수 난수들을 sum+로 짝수들의 합을 동작하였다.

그리고

sprintf는 버퍼로 사용될 변수로 출력이 됩니다. 즉, char * 로 선언된 변수안에 입력된 내용이 문자열로써 담기게 되는 것이죠. 짝수를 다 더한 값을 모니터 화면에 출력을 하는 것이고 tmp가 sum을 문자열로 저장을한다.그리고 write를 통해서 출력을 하고 strlen는 string length로 아까 sum의 정수를 tmp가 문자열로 받아 저장하고 strlen()를 통해서 문자열의 길이를 반환해 주는것이고,출력부분에서 fd를 넣었다는 것은 log.txt파일에 그 값을 출력시키겠다는 것이다.