TI DSP, MCU 및 Xilinx Zynq FPGA 프로그래밍 전문가 과정

강사 – Innova Lee(이상훈)
gcccompil3r@gmail.com
학생 – 최대성
c3d4s19@naver.com

2018.03.15. – 016일차 수업

---

* Memory Pool 방식의 큐(Queue)

헤더파일 및 매크로 추가

```
#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>
//필요에 따라 배열 크기 더 늘려서 사용
#define STRUCT_ARR_LEN 1000
```

데이터 타입과 Queue 구조체 정의

```
typedef int Data;
typedef struct _Queue{
    struct _Queue* next;  //다음 Queue 구조체
    unsigned int structIndex;//데이터 구조체index
    unsigned int structFirstData;//Queue데이터출구
    unsigned int structLastData;//Queue데이터입구
    Data data[STRUCT_ARR_LEN];  //데이터 배열
}Queue;
```

Queue 구조체 초기화 후 반환 함수

```
Queue* getNewQueue(){
    Queue* newQueue =
(Queue*)malloc(sizeof(Queue));
    newQueue->next = NULL;  //다음 Queue 구조체
    newQueue->structFirstData = 1;
    newQueue->structLastData = 0;
    newQueue->structIndex = 0;
}
```

Insert Queue 함수

```
void insertQueueData(Queue** Node, Data data ){
    //들어온 노드가 Null인 경우
    if(!(*Node)){
        *Node = getNewQueue();
    }
    Queue* tmp = *Node;
    unsigned int structNum = (*Node)->structIndex;
    //tmp->next 값이 NULL일 때 까지 while 루프
계속 돌림 (마지막 구조체 찾기)
    while(tmp->next){
        structNum++;
        tmp = tmp->next;
    }
    //마지막 구조체에 더이상 공간이 없을 경우 ->
tmp값을 새로운 구조체 주소값으로 넘어가도록 만듬
    if(tmp->structLastData == STRUCT_ARR_LEN -
1 ){
        tmp->next = getNewQueue();
        tmp = tmp->next;
        tmp->structIndex = structNum + 1;
    }
    //실제 insert 과정
    tmp->structLastData++;
    tmp->data[tmp->structLastData] = data;
}
```

Print Queue 함수

```
void printQueueData(Queue** Node){
    Queue* tmp = *Node;
    //들어온 노드가 Null인 경우
    if(!Node)
        return;
    while (tmp){
        for (int i = 0; i < tmp->structLastData;
i++){
            printf(" %d ", tmp->data[i + 1]);
        }
        tmp = tmp->next;
    }
}
```

Pop Queue 함수

```
Data popQueueData(Queue** Node){
    //들어온 노드가 Null인 경우
    if(!(*Node)){
        printf("Queue is empty!₩n");
        return -1;
    }
    Queue* tmp = *Node;
    Data popData;
    unsigned int structNum = tmp->structIndex;
    //구조체에 데이터가 하나밖에 없을 경우
    // -> 데이터 반환 후 구조체 삭제하고 tmp값을
이전 구조체 주소값으로 넘어가도록 만든다
    if(tmp->structFirstData == tmp-
>structLastData ){
        popData = tmp->data[tmp->structFirstData];
        //구조체가 하나밖에 없는 경우 (다음 구조체
없음) -> 구조체 초기화
        if(!tmp->next){
            tmp->structFirstData = 1;
            tmp->structLastData = 0;
        }
        //다음 구조체가 존재 할 경우
        else{
            *Node = (*Node)->next;
            free(tmp);
        }
        //tmp->next 값이 NULL일 때 까지 while 루프
계속 돌림
        tmp = *Node;
        while (tmp->next){
```

```
            tmp->structIndex--; //모든 구조체
index 1씩 감소
            tmp = tmp->next;
        }
    }
    //구조체에 데이터가 비어있을 경우
    else if(tmp->structLastData == 0){
        printf("Queue is empty!₩n");
        return -1;
    }
    //구조체에 데이터가 2개 이상 있을 경우
    else{
        popData = tmp->data[tmp->structFirstData];
        tmp->structFirstData++;
    }
    return popData;
}
```

테스트용 Main 함수

```
int main(){
    Queue* Queue1 = NULL;// getNewQueue();
    insertQueueData(&Queue1,10);
    insertQueueData(&Queue1,20);
    insertQueueData(&Queue1,40);
    insertQueueData(&Queue1,70);
    insertQueueData(&Queue1,44);

    printQueueData(&Queue1);
    printf("₩n%d₩n" ,popQueueData(&Queue1) );
    printf("₩n%d₩n" ,popQueueData(&Queue1) );
    printf("₩n%d₩n" ,popQueueData(&Queue1) );
    printf("₩n%d₩n" ,popQueueData(&Queue1) );
    printf("₩n%d₩n" ,popQueueData(&Queue1) );
    printf("₩n%d₩n" ,popQueueData(&Queue1) );
    printf("------------------------------ ");
    insertQueueData(&Queue1,70);
    insertQueueData(&Queue1,44);
    insertQueueData(&Queue1,33);
    printf("₩n%d₩n" ,popQueueData(&Queue1) );
    printf("₩n%d₩n" ,popQueueData(&Queue1) );
    printf("₩n%d₩n" ,popQueueData(&Queue1) );
    printf("₩n%d₩n" ,popQueueData(&Queue1) );
    return 0;
}
```