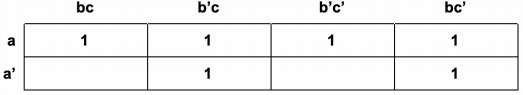
**Week 5 Assignment**

**1. (10점) 주어진 부울 식 E = a’b’c + a’bc’ + ab’c’ + ab’c + abc’ + abc를 카르노(Karnaugh) 맵을 이용하여 간편화하는 과정을 설명해주세요.**

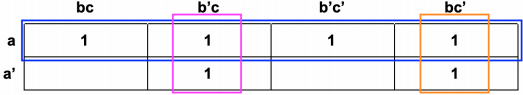
(1) 주어진 부울 식에 대해 카르노 맵을 채워주세요.

(답안)



(2) 식을 간편화하는 과정을 적어주세요.

(답안)



카르노맵에서 주어진 부울 식 E의 각 term을 적어주고 나면, 2의 거듭제곱(2, 4, 8, 16, ..) 개수로 연속한 최대의 직사각형을 만들어 줍니다. 첫번째로는 노란색 직사각형인 a가 만들어집니다. 두번째로는 초록색 직사각형인 bc’이 만들어집니다. 세번째로는 파란색 직사각형인 b’c가 만들어집니다. 작은 2칸짜리 직사각형으로 모두 분리하셔도 되나, 그렇게 하면 결국 부울 대수로 합치는 과정에 다 합쳐져서 4칸짜리 직사각형 1개가 됩니다.

(3) 최종 결과를 적어주세요.

(답안)

**a + b'c + bc'**

**2. (20점) 다음과 같은 메모리 상태 예제가 주어지고, R1 = 101, R2 = 202이 저장되어 있습니다. R1에 202값을 저장하는 마 이크로 명령을 작성해주세요.**

- 마이크로 명령은 총 4가지 종류로 “직접 값 모드” , “레지스터 모드”, “직접 모드”, “메모리 간접 모드”가 있습니다. 4가지 모드 각각에 대해 답안을 작성해주세요.

|  |  |
| --- | --- |
| **Add** | **Con** |
| 50 | 201 |
| 51 | 350 |
| ... | ... |
| 201 | 202 |
| 202 | 700 |

(답안)

(1) 직접 값 모드

R1에 직접 값을 저장해줍니다. #을 이용하여 값을 표시합니다.

**MOV R1, #202**

(2) 레지스터 모드

다른 레지스터 R2에 저장된 값을 저장합니다.

**MOV R1, R2**

(3) 직접 모드

메모리 주소를 주고, 해당 메모리 주소에 있는 값을 직접 읽어와서 저장합니다.

예시 메모리 영역에는 201번지에 202값이 저장되어 있으므로, 201을 주소값으로 주면 됩니다.

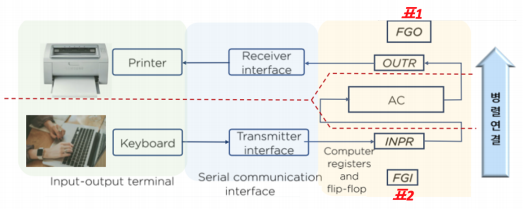
**MOV R1, 201**

(4) 메모리 간접 모드

입력한 주소값이 가진 상수값을 메모리 주소로 하여 값을 불러옵니다.

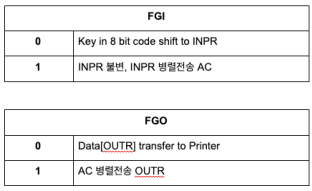
즉, @50명령어를 통해서 가져온 주소값의 상수인 201을 메모리 주소로 인지하여 직접 모드를 실행합니다. **MOV R1, @50**

**3. (20점) A의 컴퓨터는 입출력 처리를 위해 플래그를 사용한 프로그램 제어 전송 방식을 사용한다. 프로그램 제어 전송 방 식의 입출력 구성을 아래 그림과 같다.**

****

(1) 해당 그림의 FGI와 FGO에서 flag bit의 의미에 대해 아래 표에 작성해주세요.

(답안)

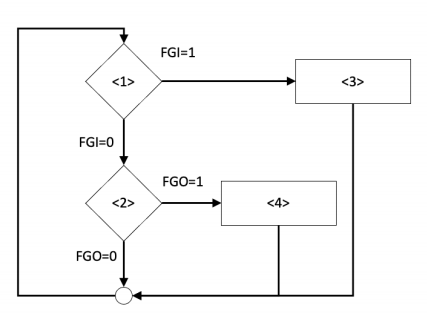


(2) 이 컴퓨터의 프로세스 처리 속도는 fetch-and-execute cycle time이 20μs이고 (하나의 instruction을 처리하는데 20μs가 걸림), 입출력 장치 처리 속도는 100 문자/초입니다. (0.01초마다 1 문자가 입/출력됨.) CPU가 수행할 수 있 는 instruction은 아래의 4가지입니다.

|  |  |
| --- | --- |
| <1> | FGI 를 체크하는 명령어 |
| <2> | FGO 를 체크하는 명령어 |
| <3> | INPR 에서 데이터를 읽는 명령어 (FGI 재설정 포함) |
| <4> | OUTR 에 데이터를 내보내는 명령어 (FGO 재설정 포함) |

이 CPU가 아래의 순서도대로 입출력을 수행한다고 할 때, (a) 문자 하나를 입력받은 후 (INPR에서 데이터를 한번 읽은

후) 문자 하나를 입력받기까지 <1> 명령어는 몇 회 실행되어야하는지 풀이 과정을 포함하여 작성해주세요. (b) 문자 하 나를 출력한 후 (OUTR에 데이터를 쓴 후) 문자 하나를 출력하기까지 <2> 명령어는 몇 회 실행되어야하는지 풀이 과정 을 포함하여 작성해주세요.

(답안)

**(a) 250회**

**(b) 250회**

프로세스 처리속도: 20μs / 1 Instruction

입출력 처리속도: 1문자 / 0.01초 = 1문자 / 10000μs

따라서 10000μs/20μs = 500회.

FGI와 FGO에 대해 번갈아 확인하므로 500회를 양분하여 각각 250회 실행함.

**4. (20점) A는 홀수 패리티를 활용하여 “HELLOWORLD”라는 문자열을 B에게 전송했습니다. (각 문자는 8비트의 아스키 코드를 사용하며 이중 MSB가 패리티 비트로 활용되고, 문자는 모두 대문자를 사용합니다. 예를 들어, 대문자 A는 가장 왼쪽의 패리티 비트를 포함하여 11000001로 표현됩니다.) 이때 B가 A로부터 받은 비트 스트림 (문자열을 비트열로 표 시한 결과)은 다음과 같습니다.**

11001000 11000101 01001100 01001100 01001111

01010111 01001111 11010010 01001100 11000101

위의 비트 스트림에 대해 각 문자가 제대로 전송되었는지 잘못 전송되었는지 작성하고 (O,X로 표시), 잘못 전송된 문자의 경 우 패리티 비트나 아스키 코드 중 어느 부분이 잘못 전송되었는지 작성해주세요. (**패리티 비트 또는 아스키 코드로 표시**. 문자 전송에 성공했다면 잘못 전송된 부분은 빈 칸으로 남겨주세요.)

(답안)

**패리티 비트 / 아스키 코드**

HELLO WORLD중 **E, R**은 패리티 비트가 틀렸고, **D**는 아스키 코드가 틀림.

D의 경우 아스키 코드의 오류를 의도하였으나 주어진 비트 스트림에 대해 패리티 비트가 틀린 것도 맞기 때문에 아래의 기준에 따라 정답 처리.

① 패리티 비트 & 아스키 코드: 정답

② 아스키 코드: 정답

③ 패리티 비트: 오답 (패리티 비트만 고치면 여전히 아스키 코드는 01000101(E) 이기에 오답 처리)

11001000 **11000101** 01001100 01001100 01001111

01010111 01001111 **11010010** 01001100 **11000101**

E 정답 = **0**1000101

R 정답 = **0**1010010

D 정답 = 1**1000100**