모니터 프로그램

金 惠 鎭* 李 順 珠**
高麗大學校 工科大學 電子工學科 工博*, 大學院**

Ⅰ. 序 論

모니터 프로그램의 解析은 마이크로프로세서 使用者들로서는 일찌기 관심을 두어 오지 않은 分野이다. 그래서 이번 기회에 마이크로컴퓨터의 利用度를 높이기 위하여 M6802 키이보오드 모니터 프로그램을 中心으로 모니터 프로그램의 構成을 여기에 紹介하려고 한다. 이 모니터 프로그램은 16進數 키이를 누를 때 6個의 7-分割(segment) LED 上에서 同時에 表示되도록 되어 있다. 키이보오드는 그림 1과 같이, 0~F까지의 16進數와 制御 키이로서는 "AUTO", "BACKWARD", "FORWARD", "EXAMINE", "DO", "CHANGE" 등으로 構成되어져 있다.

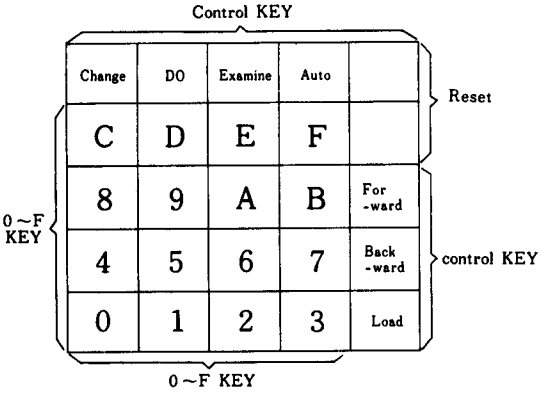


그림 1. 키이보오드의 配列

Ⅱ. 하드웨어의 構成

마이크로프로세서는 여러 가지 方法으로 하드웨어를 構成할 수 있으나 여기서는 M6802 MPU칩을 使用한 마이크로컴퓨터의 개략적인 回路圖를 그림 2에 나타내었다. 즉, M6802의 8 bit MPU칩과 2개의 6820 PIA칩과 2716 ROM칩(2 k byte)과 4개의 2114 RAM 칩(1 K×4 bit)으로 構成되어 있다. 먼저, 6802 MPU 칩에

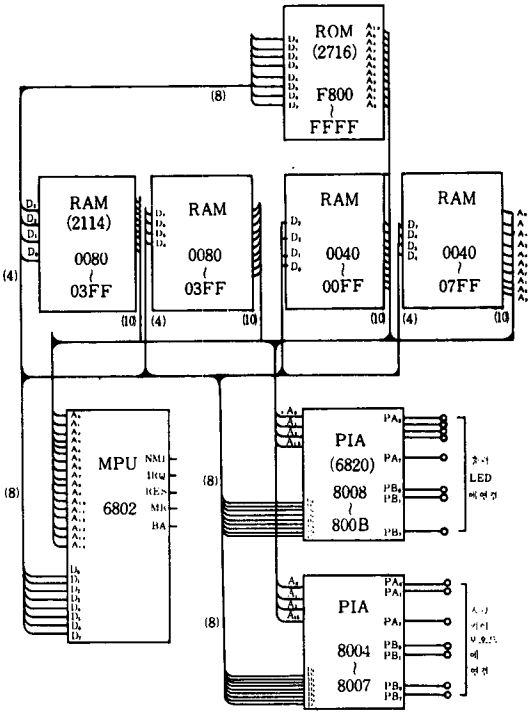


그림 2. 하아드웨어 시스템 構成

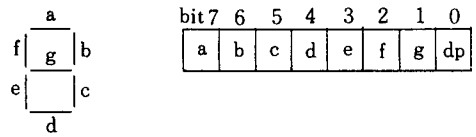
내장된 128 byte의 RAM 領域을 보면, 0000에서 005A는 使用者의 領域이고, 005B에서 006B는 모니터 프로그램을 위한 스택(stack)領域이고, 0061에서 0067 은 MPU 레지스터의 스택영역이며, 0068에서 007F는 임시저장 領域이다. 外部 RAM의 0080에서 FDFF는 확장시킬 수 있는 領域을 나타낸다.

다음으로 ROM 領域은 실제의 키이보오드 모니터 프로그램이 FE00에서 FFFF까지의 번지에 기억되어 있으며 그 容量은 0.5K byte(=512 byte)이다.

또한, PIA칩들中의 하나는 LED세그멘트에 값을 나타내 주기 위한 것이고, 다른 것은 入力키이를 받아 들

이는데 使用된다. 즉, LED 세그멘트에 값을 나타내는 데 使用하는 8004번지의 채널0은 入力 키이 0～7 을 담당하고, 8005번지의 채널1은 入力키이 8～F를 담당한다. 또, 채널2와 3은 出力을 나타내기 위한 것으로 前者는 出力 7-세그멘트 LED에 데이타를 표시하는 役割을 하며, 後者는 어느 디지트(digit)를 켜야 할지를 決定하는 役割을 담당한다.

마지막으로, 7-세그멘트 LED에 표시하기 위한 데이타는 FFE8에서 FFF7번지에 기억되어 있으며 그 表示 方法은 다음과 같다.



(a) 1 디지트를 7분할한 모양 (b) 기억장소의 8 bit 데이타

만일, 7분할 LED에 'E'를 표시하고자 하면 a, f, e, d, g가 불이 켜진 상태가 되어야 한다. 그러나 하드웨어 構成으로 불이 켜질 部分을 "0"으로 하고 小數點은 항상 "1"인 상태를 유지한다. 그러므로 8 bit 데이타는 01100001(61_{16})이 된다. 'E'가 디지트上에 나타나기 위해서는 內部的으로 61이란 데이타가 連結되어진다. 以上과 같은 方式으로 모든 0～F까지의 數字를 記憶할 데이타들로 바꾸어 저장해 둔다. 그러므로 이 모니터 프로그램에는 ASCII 코오드 같은 변환표가 포함되지 않고 7分割 LED에 나타날 8 bit 데이타가 바로 들어가 있다.

Ⅲ. 모니터 프로그램

모니터 프로그램은 初期狀態로 만드는 것부터 시작하여야 하므로 임시 저장 領域내의 모든 部分을 初期化(initialize)시킨다. 그리고, 그 初期化 過程이 끝난 狀態에서 주 프로그램인 'CONTROL' 루우틴으로부터 記述 하겠다(그림 3의 플로우차아트 참조).

No-write 플래(flag)을 1로 하고 DES(do-examine-step) 플래을 0으로 만들어 키이를 받아 들일 準備를 한다. 다음에 불려지는 서브루우틴은 'INPUT' 루우틴이고 그 것은 바로 'DISPLAY' 루우틴으로 連結되어진다.

1. LED 表示를 위한 'DISPLAY' 서브루우틴

먼저, 그림 4에는 플로우차아트를 나타내 주고 실제로 LED에 어떻게 作用하는가에 對해서는 그림 5에 보여주고 있다. 플로우차아트를 따라 가면서 이 서브루우틴을 解析 해 보자.

먼저, 인덱스 레지스터(index register)에 디지트 1

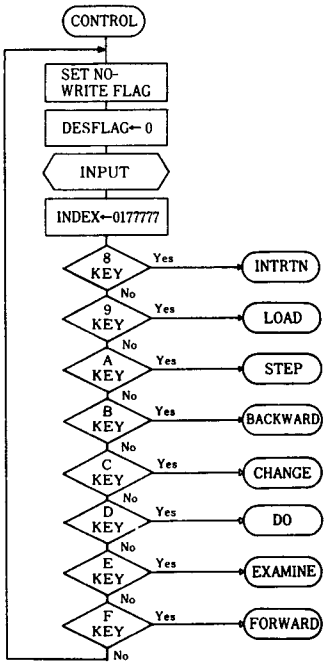


그림 3. 'CONTROL' 주 프로그램의 플로우차아트

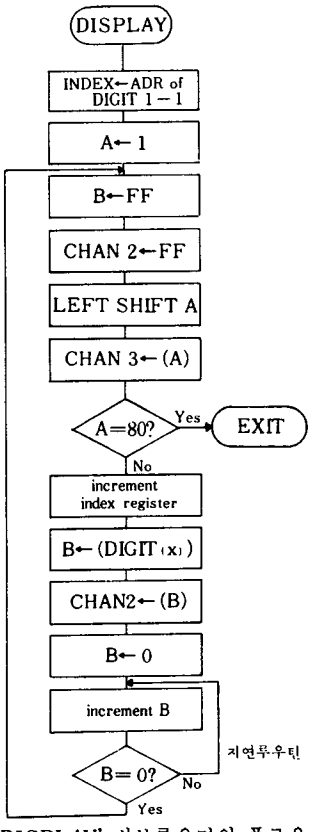


그림 4. 'DISPLAY' 서브루우틴의 플로우차아트

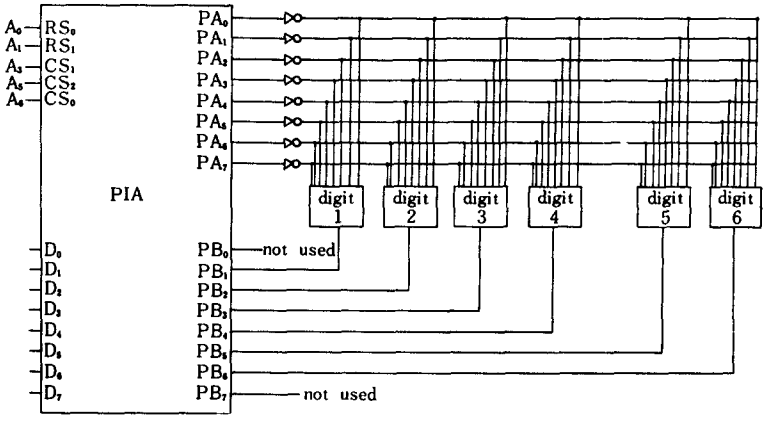


그림 5. PIA칩과 LED 표시장치와의 회로도

의 番地보다 하나 적은 값을 넣고 A레지스터에는 1을 代入한다. A 레지스터 값을 왼쪽으로 한번 이동한 후 그 값을 채널3에 넣어 줌으로써 몇 번째 디지트를 선택할 지를 決定한다. 다시 B레지스터에 FF값을 넣고 채널 2에 B 레지스터 값을 代入함으로써 채널2가 FF로 된다. 그러므로써 6개의 LED를 꺼진 狀態로 만든다. 다시 A 레지스터 값이 80_{16}인지를 비교해서 80_{16}이면 모든 6개의 LED 선택이 끝난 경우이므로 'INPUT' 서브루우틴에서 'DISPLAY'서브루우틴을 불렀던 자리로 돌아 가게 된다. 그러나 80_{16}이 되지 않은 경우에는 아직 6개의 디지트가 모두 선택되지 않았으므로 인덱스 레지스터를 增加시키고 그 內容을 B레지스터에 넣어 준다. 이 B 레지스터 값이 채널2에 들어 가게 된다. 즉, 채널2에는 채널3에 의해 지정된 디지트의 LED 데이타가 들어 가게 된다.

다음의 지연루우틴은 B에 0을 넣어 주고 增加시켜 比較하는 過程을 다시 0이 될 때까지 되풀이 하여 FF번만큼 遲延을 시켜 준다. 이 遲延 루우틴의 역할은 6개의 LED上에 모두 불이 들어 온 狀態로 우리가 볼 수 있게 지연시켜 주는 것이다. 결론적으로, 'DISPLAY' 서브루우틴은 디지트 1에서 6까지의 LED들 中의 어느 하나를 선택하고 또 그 디지트에 해당되는 LED 데이타를 나타내 주는 것이다.

2. 키이보오드 'INPUT' 서브루우틴

'DISPLAY' 루우틴이 끝나면 다시 'INPUT' 루우틴으로 돌아와 다음 過程을 수행한다. 이것의 플로우차아트는 그림6에 나타내었다. 그림6 'INPUT' 루우틴의 주요 役割은 入力으로 어떤 키이가 눌려져서 들어 왔을 때 그 값을 선별하는 것이다. 그래서, 構成面으로 볼때 上루우프와 下루우프로 나뉘어져 있다. 여기서 上루우프는 키이8에서 F까지의 數字를, 下루우프는

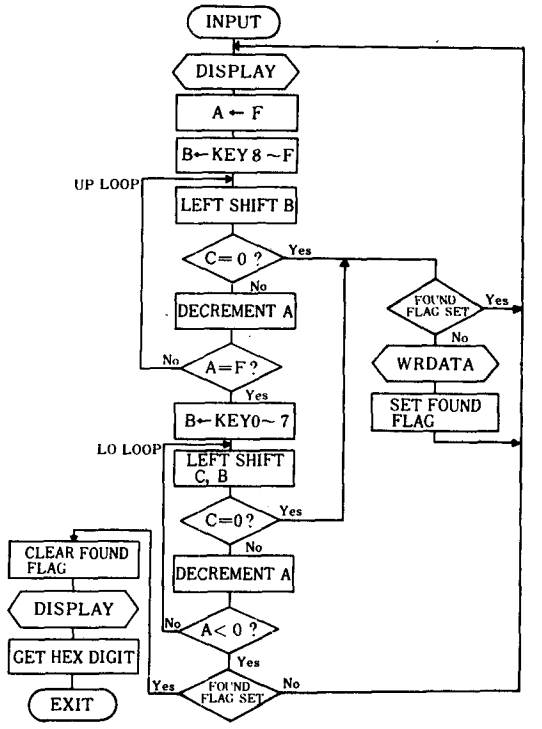


그림 6. 'INPUT' 플로우차아트

키이0에서 7까지의 數字를 선별하는 役割로 나뉜다. 上루우프에 해당되는 8에서 F까지의 숫자가 들어 왔을 때 A레지스터에 F값을 代入한다.

이 部分에서, B레지스터에 눌러진 키이8에서 F까지의 한 값이 들어 오는데 그 값들의 構成은 그림7에 나타내었다.

즉, 키이0이 눌려지면 SWA_0만이 닫힌 狀態가 되어 0V가 걸린다. 그러므로 나머지 PB_7에서 PA_1 까지는 5V가 걸려 "1"상태를 나타낸다. 또, 키이F가 눌려지

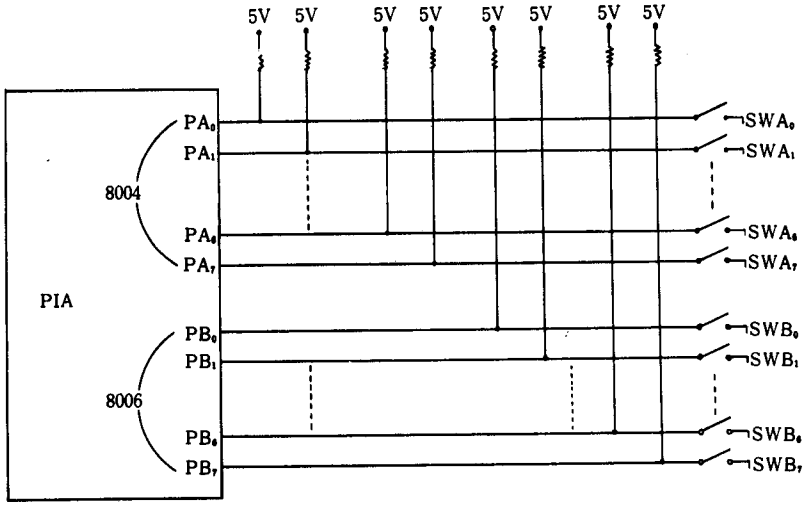


그림 7. 스윗치가 닫혔을 때 "0"가 되고 스윗치가 열렸을 때 "1"이 된다.

면 SWB7만이 닫히고 나머지는 열린 狀態가 된다. 그러므로 PB7에 "0"가 되고 나머지는 "1"이 된다.

이런 과정으로, 키이 값들을 가지고 B레지스터가 왼쪽으로 1 bit 이동되었을 때 캐리(carry)가 없으면 (즉, C=0), 그 키이는 A레지스터에 넣어 준 F값 그대로가 될 것이다. 그러나 키이 F가 아니라 다른 키이면 그때 캐리(carry)는 1이 되고, A레지스터를 하나 減少시켜 다시 上루우프를 돌게 된다. 감소된 A레지스터 값이 키이 8에서 F내의 것이 아니면 여덟 번의 減少過程을 거쳐 A=7과 비교해서 같으므로 下루우틴으로 점프한다. 下루우틴은 上루우틴과 同一한 方法으로 전개된다. 上루우틴에서 캐리가 0이 되어 키이값을 제대로 선별했을 경우에는 'FOUND'을 확인해서 1이면 처음으로 다시 돌아가 다른 키이를 기다리거나 눌려진 現狀態의 키이값을 가지고 다시 되풀이 한다. 'FOUND' 플랙이 0인 경우에는 'WRDATA' 서브루우틴을 수행하고 'FOUND' 플랙을 세트(set)한 후 처음으로 돌아간다.

위 과정에서, 마지막의 'WRDATA' 서브루우틴은 4 bit의 16진수를 8 bit의 LED 코오드로 바꾸어 주고, 다시 그것을 6개 디지트중의 하나에 써 준다. 그러므로써, 우리가 'DISPLAY' 루우틴을 통하여 선별된 키이의 숫자를 LED上에 써 줄 수 있다.

3. 其他 서브루우틴(그림 8 참조)

처음의 'CONTROL' 주 프로그램에서 'INPUT' 서브루우틴까지의 過程이 끝남으로써 마이크로프로세서의 모니터하는 基本的인 內容을 알게 되었다. 그러나 外部的으로 임의로 선택할 수 있는 여덟 개의 키이에 관한 解析 問題가 남았다.

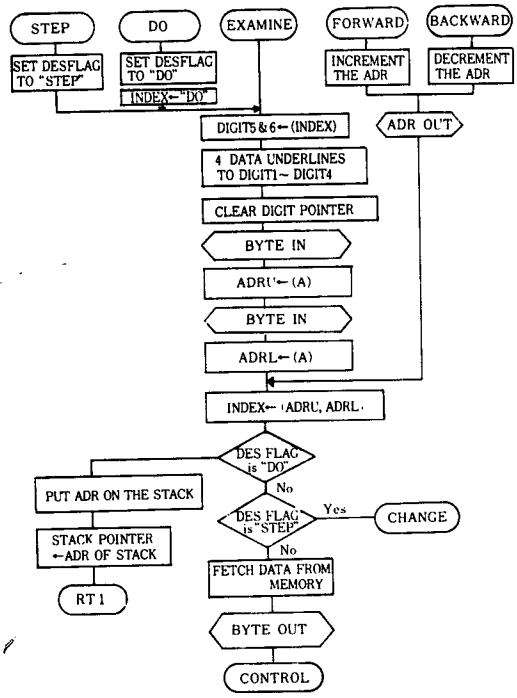


그림 8. 외부 키이에 관한 플로우차아트

첫째, 'EXAMINE' 서브루우틴은 入力되어진 데이타를 다시 조사해 보는 것이다. 먼저, 인덱스레지스터의 內容을 디지트5,6에 넣어 주고 디지트1에서 4까지는 밑줄(-)을 써 준다. 다음 過程의 'BYTEIN' 서브루우틴은 2개의 키이보오드 코오드에서 읽혀 지고 다시 그것들을 8 bit 숫자로 變換한다. 그래서, 그것들의 7-分割코오드는 'DISPLAY' 서브루우틴에 관한 것이고 그것은 다음의 2디지트로 나타난다. 그 過程은 入力

에서 받은 키이 데이타를 4 bit만큼 왼쪽으로 이동하여 任意의 場所에 저장한다. 다시 새로운 데이타를 入力에서 받아 A레지스터에 저장한다. 예전의 임의의 場所에 저장된 값을 A 레지스터 값에 더해 준다. 이리하여 처음 'BYTEIN' 서브루우틴은 上部番地의 內容으로 되고 다음의 것은 下部番地로 채워 준다. 이 上部番地와 下部番地는 現在의 번지수를 나타내 주며 그 값은 인덱스레지스터로 옮겨진다. 다음으로, 'DES' 플랙의 狀態가 'DO'나 'STEP'의 어느 것도 아닐때는 위에서 얻은 인덱스레지스터 값 (즉, 현재의 번지)에 따라 記憶장치에서 데이타를 끄집어 낸다.

다음의 'BYTEOUT' 서브루우틴은 'BYTEIN' 서브루우틴에 對應되는 것으로 比較할 수 있다. 즉, 8 bit 데이타를 받아 들여 두 개의 8 bit LED 코오드로 變換시키고 다시 그것을 2개의 디지트에 써 준다. 다시, 그 構成을 보면 A레지스터의 內容을 B레지스터로 옮긴 후 오른 쪽으로 4 bit를 이동하여 上部 디지트의 內容을 A레지스터로 옮긴다. 그리고 'WRDATA' 서브루우틴의 實行으로 4 bit의 16진수를 8 bit LED 코오드로 바꾼다. 이것을 6개의 디지트중의 5번째 디지트에 써 주어 上部 디지트를 얻는다. 다시 A레지스터에 下部 디지트의 內容을 옮겨 위와 같이 얻을 수 있다. 'BYTEOUT' 서브루우틴이 끝나면 처음의 'CONTROL' 서브루우틴의 시작점으로 돌아가 새로운 상태를 기다린다.

둘째, 'STEP'이나 'DO' 서브루우틴일 경우

DES 플랙만을 변화시키며 나머지 루우틴은 처음의 'EXAMINE'과 同一하게 構成한다. 물론, 'DO' 서브루우틴은 인덱스 레지스터에 'DO'에 해당하는 데이타를 넣어 주어 7-分割 LED上에서 'DO'를 表示할 수 있게 한다. 'STEP' 서브루우틴은 過程中에서 'CHANGE' 서브루우틴으로 빠지게 되어 있다.

셋째, 'CHANGE' 서브루우틴

이 서브루우틴은 記憶裝置內의 데이타를 文字 그대로 다른 값으로 바꾸어 주는 것이다. 어떤 記憶된 內容을 바꾸어 주려면 'EXAMINE'을 實行해 몇 번지에 記憶된 內容을 바꾸어 줄지를 決定한 후 디지트 5, 6에서 바꿔 준다(그림 9 참조). 구성은 디지트 5, 6에 밑줄을 나타내고 디지트포인터가 디지트 5를 가리키도록 한다. 'BYTEIN' 서브루우틴을 實行함으로써 入力 데이타를 받아 들여 2개의 디지트를 얻는다. 현재의 번지를 인덱스 레지스터로 옮긴 후 記憶장치로 데이타를 저장한다. 기억장치에 데이타가 잘 저장 되었는지를 확인하여 잘 되지 않았으면 디지트 5, 6에 밑줄을 표시

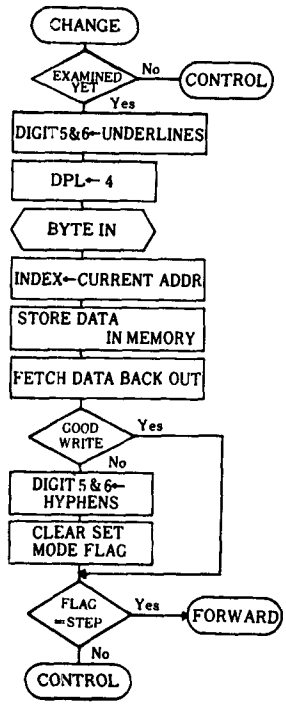


그림 9. 'CHANGE' 플로우차아트

하게 된다. 'DES' 플랙의 狀態가 'STEP'이면 'FORWARD' 서브루우틴으로 가서 다음 기억할 場所에 表示할 準備를 한다.

네째, 'FORWARD'와 'BACKWARD' 서브루우틴

이 경우는 番地數를 변화시켜 주면서 그에 따른 데이타를 볼 수 있는 것이다. 'FORWARD'는 番地數의 增加와 그것에 따른 데이타를 보는 데 반해, 'BACKWARD'는 번지수의 減少에 따른 데이타를 본다. 그러므로, 'FORWARD' 서브루우틴만을 記述하겠다. 먼저, 番地數를 하나 增加시킨 다음 'ADROUT' 서브루우틴을 實行한다.

'ADROUT' 서브루우틴은 上, 下部 番地에 있는 數字들을 4개의 7-分割 코오드로 變換시킨다. 그것을 다시 디지트 1에서 4까지에 써 준다. 이것은 番地數를 LED上으로 끄집어 내어 보기 위한 것이므로 임시의 場所에 貯藏된 番地數를 디지트 1에서 4까지로 끄집어 낸다. 그러므로 A레지스터에 上部 番地數의 內容을 옮기고 'BYTEOUT'를 實行함으로써 8 bit 數字의 上部 번지수를 2개의 8 bit LED 코오드로 變換시킨 후 다시 디지트 1, 2에서 보여 준다. 다시 下部 番地數를 A레지스터로 옮겨 위 過程을 되풀이 하여 디지트 3, 4도 나타내 지게 된다. 그 다음의 過程은 'EXAMINE' 서브루우틴을 實行하는 것과 同一하다.

4. 外部的으로 조절이 가능한 키이의 使用列

우리는 모든 서브루우틴에 대한 說明을 마쳤으나, 실제로 그 키이가 눌러졌을 때, 그들의 作動 狀態를 LED와 관련시켜 알아 보기로 하자.

1)"AUTO" 키이가 눌러지면,

이것은 위에서 말한 'STEP' 서브루우틴과 同一한 動作을 하므로,

———— —— 으로

LED上에 나타나고, 'DES' 플래은 'STED' 으로 세트된다. MPU칩의 0071番地內의 'DES' 플래의 狀態 變化는 'DO' 일때 FF, 'EXAMINE' 일때 00, 'STEP' 일때 01으로 나타난다. 그 다음의 過程은 'EXAMINE'을 거쳐 番地數를 손으로 키이를 직접 눌러 나타내 주고 브랜치 命令(branch instruction)에서 'CHANGE' 서브루우틴으로 간다. 데이타는 손으로 키를 눌러 기억시킨 후 나타낸다.

2)"BACKWARD"키이가 눌러질 때

番地數를 하나 減少시키고 그 番地數에 記憶된 데이타를 LED上에 나타낸다. 즉, 'B' 키이를 누르기 전의 狀態가

8004 15 일때,

"B"키이를 누르면,

8003 43 으로

LED上에 變化가 일어난다(이때 디지트5,6의 '43'은 8003番地에 記憶되어 있던 데이타 이다).

3)'CHANGE' 키이를 누를 때

이 키이는 記憶된 데이타를 바꿔 주기 위한 것이다. 앞의 서브루우틴에서 언급한 바 대로 'EXAMINE' 키이를 먼저 눌러 준 후 CHANGE 키이를 使用한다. 'EXAMINE' 키이에 의해 番地數가 LED上에 나타난 후 'CHANGE' 키이를 누르고 디지트5, 6에 키이를 눌러 새로운 데이타를 바꿔준다. 즉, 'EXAMINE' 키이를 누르면, LED上에는

8004 15 가 되고

'CHANGE' 키이를 누르면

8004 —— 가 된다.

만일, 바꿔 줄 데이타가 '25' 이면

8004 25 로 나타내 준다. (이때 '25'는 직접 키이를 눌러서 나타난 데이타이다).

4)'DO' 키이를 누르면,

프로그램을 實行시키라는 命令으로 프로그램의 實行이 몇 番地에서 시작될지를 알리는 役割을 한다. 먼저, 'DES' 플래을 'DO' 로 세트 시킨 후 인덱스 레지스터에 'DO'에 해당하는 16진수 데이타를 넣는다. 'EXAMINE' 서브루우틴을 거쳐 스택에 貯藏된 번지수에서 프로그램을 實行하도록 RTI(return interrupt) 命令을 使用한다. 番地數 8000을 키이를 눌러 나타낸 후 'DO' 키이를 누르면 8000番地부터 프로그램이 實行된다.

5)'EXAMINE' 키이를 누를 때

앞에서 說明한 서브루우틴과 同一한 內容으로 動作한다. 이 키이가 눌러지면 番地數 部分인 디지트1에서 4까지가 밑줄(-)로 나타난다. 그리고, 키이(0~F)를 눌러줌으로써 디지트1에서 4에 나타난 番地數의 內容을 試驗해 볼 수 있다. 즉, 'EXAMINE' 키이를 누르면 ———— —— 으로 나타나고 8004番地를 키이를 누르면 그 番地의 데이타까지 나타난다.

즉, 8004 15 로,

이 作用은 이미 'CHANGE' 키이에서 보여 준 바와 同一하게 쓰인다.

6)'FORWARD' 키이를 누를 때

'BACKWARD' 키이와 同一한 構成이나 番地數를 增加시켜 주는 部分이 다르다. 즉, 원래 8004 43으로 되어 있던 番地數와 데이타가 F키이를 누름으로써, 8004 15의 狀態로 番地數가 增加되고 그에 따른 데이타가 나타난다.

Ⅳ. 結 論

以上으로 모니터 프로그램의 전반적인 作用과 실제로 키이의 使用法에 따른 LED 變化를 알아 보았다. 우리가 一般的으로 마이크로프로세서의 모니터 프로그램으로부터 자세한 解析과 그에 따른 應用을 해 나감으로써 새로운 프로세서로의 開發에 도달할 수 있을 것이다. 실제로 0.5K byte 밖에 미치지 못하는 이 모니터 프로그램도 처음으로 開發하는 데에는 많은 時間과 努力이 必要한 것이다.

그러나, 모니터 프로그램을 바로 理解하지 않고서는 마이크로프로세서 間의 병렬 처리 問題나 한글 處理 시스템등 여러 應用 등을 擴大해 나갈 수 없을 것이다.

앞으로 이 部分에 관한 많은 硏究를 우리 나라에서는 積極的으로 해 나아 가야만 컴퓨터의 國產化를 이룰 수 있을 것이다.

參 考 文 獻

[1] Ron Bishop, "BASIC microprocessors and the 6800".

[2] Electronic product associates, inc., micro 68, users mannual.