

Chapter 16. LED 매트릭스

1. LED 매트릭스

LED 매트릭스는 LED를 행과 열의 매트릭스 형태로 배치하여 문자나 기호를 표시할 수 있도록 만들어진 출력장치의 일종이다. 특히 8×8 크기의 매트릭스는 알파벳이나 숫자 등 ASCII 코드에 정의된 문자를 표시할 수 있어 많이 사용된다. 시중에서 흔히 볼 수 있는 LED 매트릭스는 8×8 크기의 매트릭스로 16개의 핀을 가지고 있으며, 형변이 기록된 면의 가장 왼쪽 핀이 1번 핀이고 반시계 방향으로 핀 번호가 증가한다.

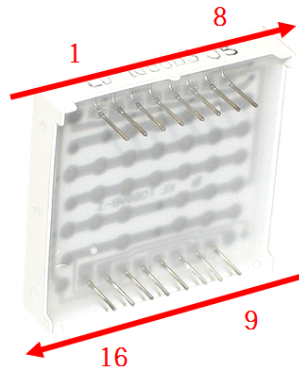
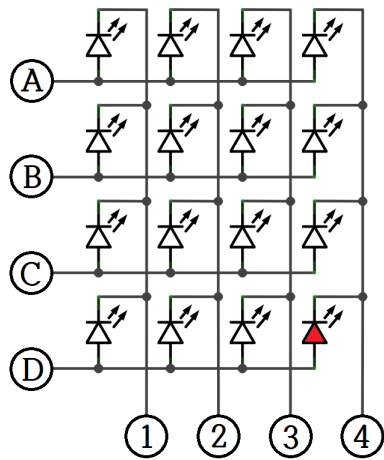
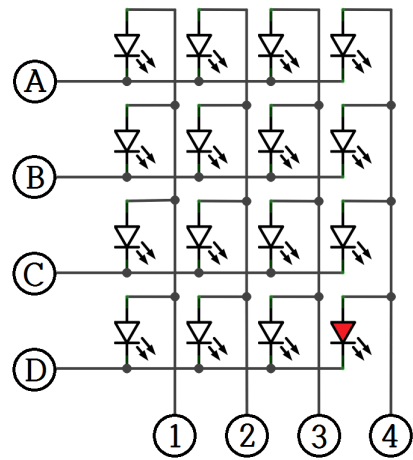


그림 1. LED 매트릭스의 핀 배치

8×8 LED 매트릭스에는 모두 64개의 LED가 포함되어 있다. 64개의 LED를 제어하기 위해서는 64개의 핀이 필요하지만 LED 매트릭스에는 16개의 핀뿐이다. LED 매트릭스 제어를 위해서는 4자리 7세그먼트를 제어에 사용하였던 잔상효과가 사용된다. 즉, 한 번에 하나의 행이나 열에 있는 8개 LED를 제어하고 빠른 속도로 행이나 열을 이동함으로써 64개 LED가 한꺼번에 켜지는 것처럼 보이게 할 수 있다. LED 매트릭스에서 한 번에 하나의 행에 포함되어 있는 8개 LED를 동시에 제어하는 방식을 행 단위 스캔 방식이라 하며, 한 번에 하나의 열에 포함되어 있는 8개 LED를 동시에 제어하는 방식을 열 단위 스캔 방식이라 한다. LED 매트릭스는 특정 위치의 LED를 켜는 방법에 따라 공통 행 양극 방식(또는 간단히 양극 방식)과 공통 행 음극 방식으로 나눌 수 있으며, 이 역시 7 세그먼트에서와 동일하다.



(a) 양극 방식



(b) 음극 방식

그림 2. 양극 방식과 음극 방식의 LED 매트릭스

그림 2에서 D행 4열 LED를 켜는 경우를 생각해 보자. 양극 방식의 경우 행에는 (+)를, 열에는 (-)를 가해주어야 해당 LED가 켜진다. 반면 음극 방식의 경우에는 양극 방식과 반대로 행에는 (-)를, 열에는 (+)를 가해주어야 한다.

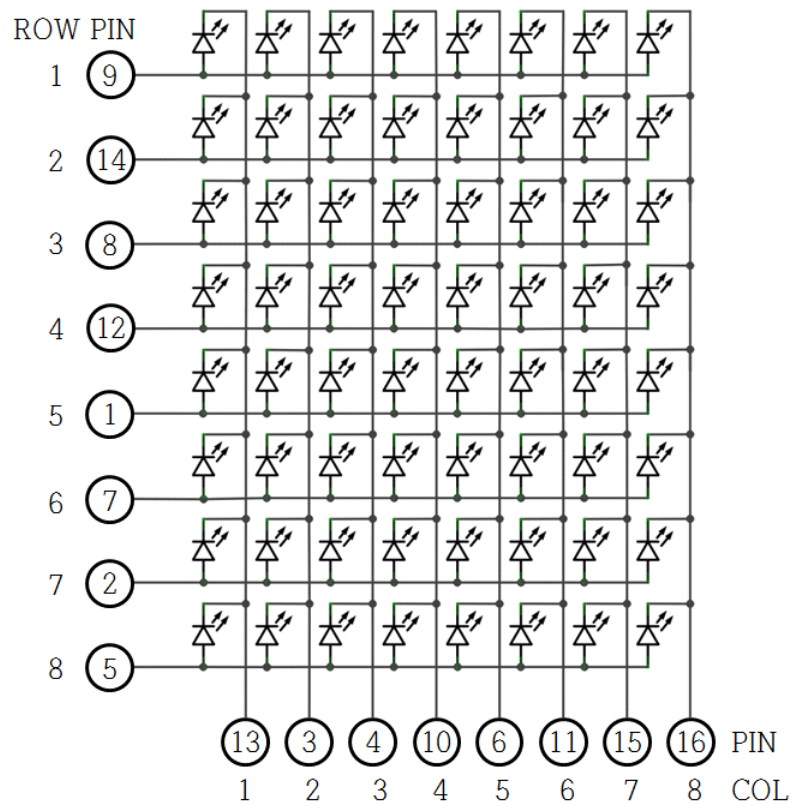


그림 3. LED 매트릭스 내부 회로도

양극 방식 8×8 LED 매트릭스의 핀 번호와 해당 행 및 열의 번호는 그림 3과 같다. 행이나 열의 번호와 핀의 번호 사이에는 아무런 연관성이 없으므로 연결 시에 주의하여야 한다.

열 단위 스캔의 예를 살펴보자. 열 단위 스캔에서는 하나의 열에 있는 8개 LED를 한꺼번에 제어한다. 그림 4는 열 단위 스캔을 통해 4번째 열에서 1, 3, 6, 8번 행의 LED만을 켜고 나머지는 끄는 예로, 행에는 10100101_2 을 열에는 11101111_2 을 가해주면 된다. 행 단위 스캔의 경우 하나의 행에 있는 8개 LED를 한꺼번에 제어한다는 점을 제외하면 열 단위 스캔과 동일하다.

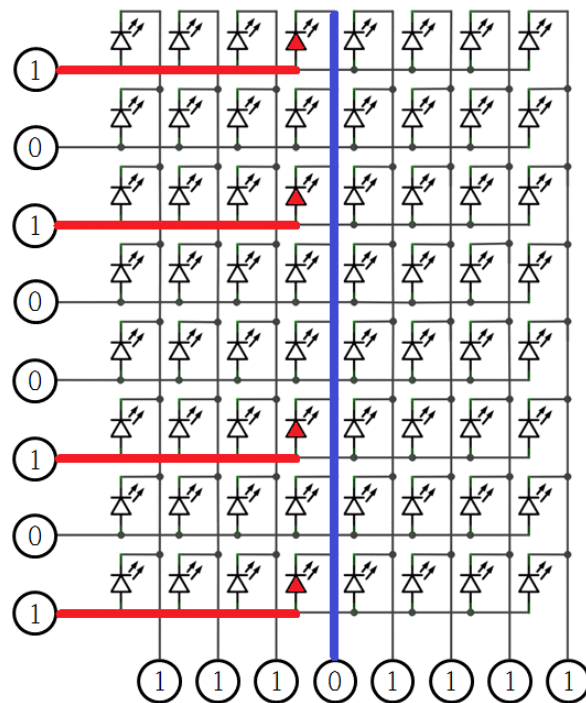


그림 4. 열 단위 스캔의 예

이 장에서는 양극 방식의 LED 매트릭스를 열 단위 스캔으로 제어하는 방법을 기준으로 한다.

2. 74595 - Serial In Parallel Out Shift Register

8×8 LED 매트릭스를 16개의 핀으로 제어가 가능하지만 16개의 핀 역시 적은 수가 아니다. 더 적은 수의 핀으로 LED 매트릭스를 제어하기 위해 사용하는 방법 중 하나가 74595 직렬 입력 병렬 출력 이동 레지스터를 사용하는 방법이다. 74595는 16핀의 칩으로 1개의 데이터 입력 핀과 2개의 제어 핀, 총 3개의 핀으로 8개의 디지털 출력을 사용할 수 있도록 해준다.

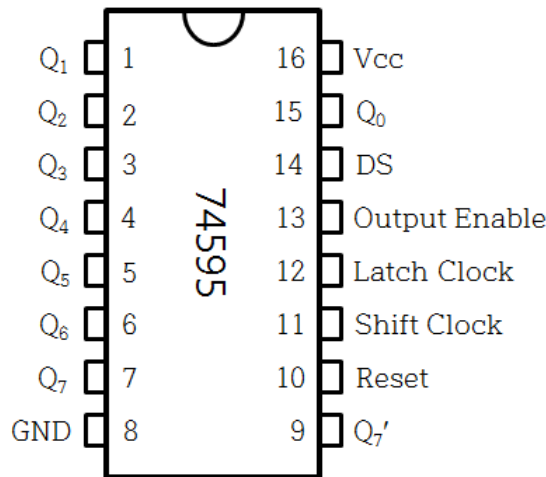


그림 5. 74595 핀 배치도

74595 칩은 직렬로 8비트의 데이터를 입력 받아 이를 병렬로 출력하는 (Serial In Parallel Out) 역할을 한다. 그림 5에서 14번 핀 DS가 직렬 입력에 해당하며, Q_0 에서 Q_7 는 데이터를 병렬로 출력하는 핀이다. 74595는 11번과 12번 핀에 두 개의 CLOCK을 필요로 한다. 11번 핀의 Shift Clock은 74595 칩으로 전달되는 데이터를 동기화하여 1비트의 데이터가 전송되었을 때 74595 내부에서 데이터를 1비트를 이동시키기 위해 사용된다. 반면 12번 Latch Clock은 74595 칩에서 병렬로 데이터가 출력되는 시점을 결정하기 위해 사용되는 클록이다. Shift Clock이 8번 발생하면 8비트의 데이터가 전송 완료되어 8비트 래치에 저장되고, 이 때 Latch Clock을 발생시키면 현재 래치에 저장된 데이터가 Q_0 에서 Q_7 핀을 통해 출력된다. Output Enable은 LOW 입력이 가해질 때 출력단의 래치(latch)를 활성화 시켜 출력이 가능하도록 하는 역할을 하므로 Output Enable을 GND(LOW 상태)에 연결하고 출력 시점은 Latch Clock으로 조절함으로써 병렬 데이터 출력이 가능하다. 10번 Reset 핀은 비동기 초기화 핀으로 VCC에 연결시키면 아무런 동작도 수행하지 않는다. 9번 Q_7' 는 쉬프트 레지스터의 값을 직렬로 출력하는 핀으로 74595를 여러 개 연결시킬 수 있도록 해준다.

8×8 LED 매트릭스를 제어하기 위해서는 16개의 출력 핀이 필요하므로 2개의 74595 칩을 사용한다. 첫 번째 칩의 Q_7' 핀을 두 번째 칩의 DS 핀으로 연결하고 1바이트 데이터를 두 번 전송하면 첫 번째 바이트는 두 번째 74595 칩으로, 두 번째 바이트는 첫 번째 74595 칩에 저장된다. 이 때 Latch Clock과 Shift Clock은 공통으로 사용하면 된다. LED 매트릭스를 2개의 74595 칩을 사용하여 아두이노 메가2560과 연결해 보자. 2개의 74595 칩 제어를 위해서는 데이터, 클록, 래치 클록을 위해 3개의 핀만을 연결하면 된다.

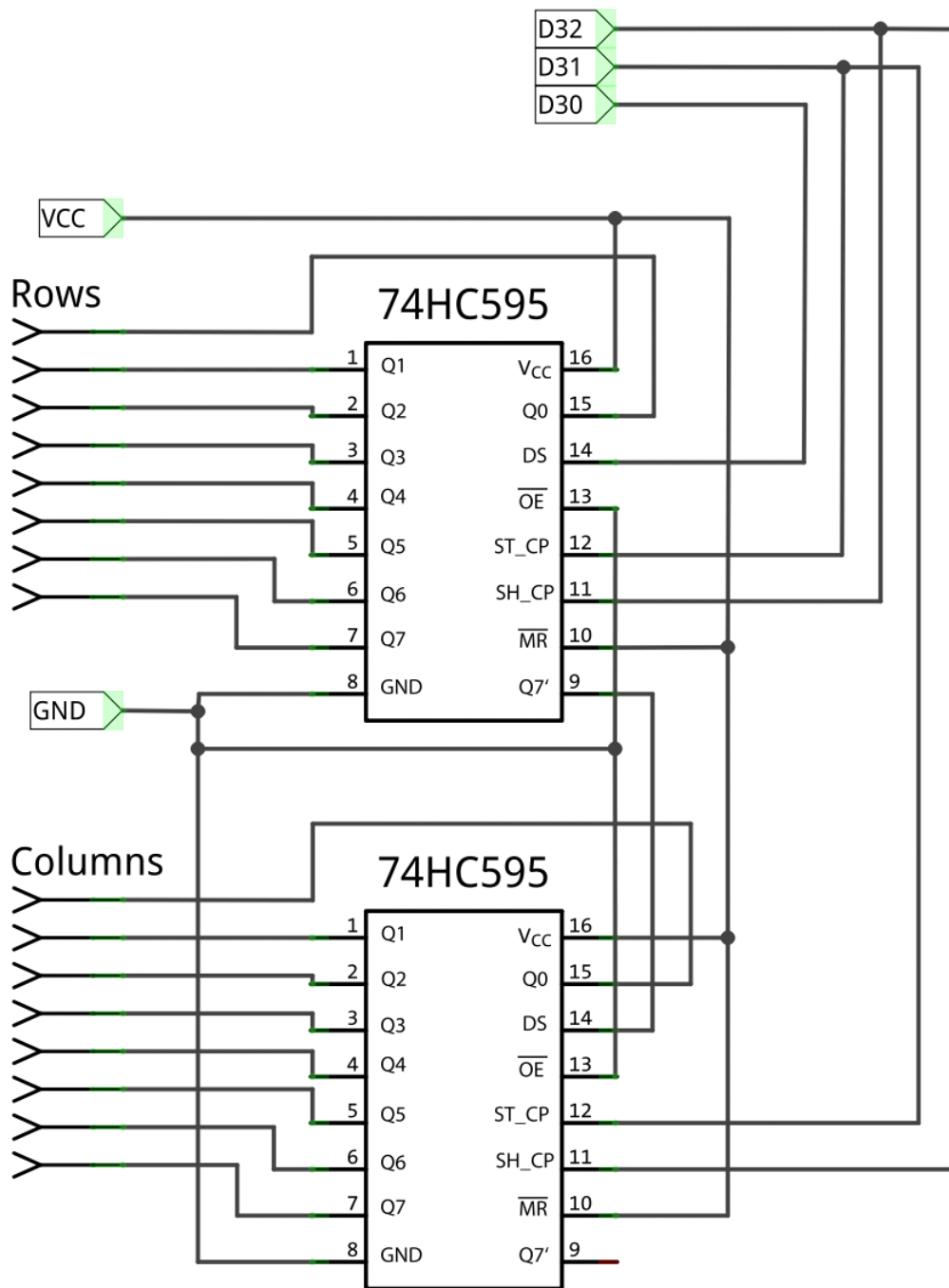


그림 6. 74595 연결 회로도

학습 보드에는 LED 매트릭스와 매트릭스 제어를 위한 2개의 74595 칩이 포함되어 있다.

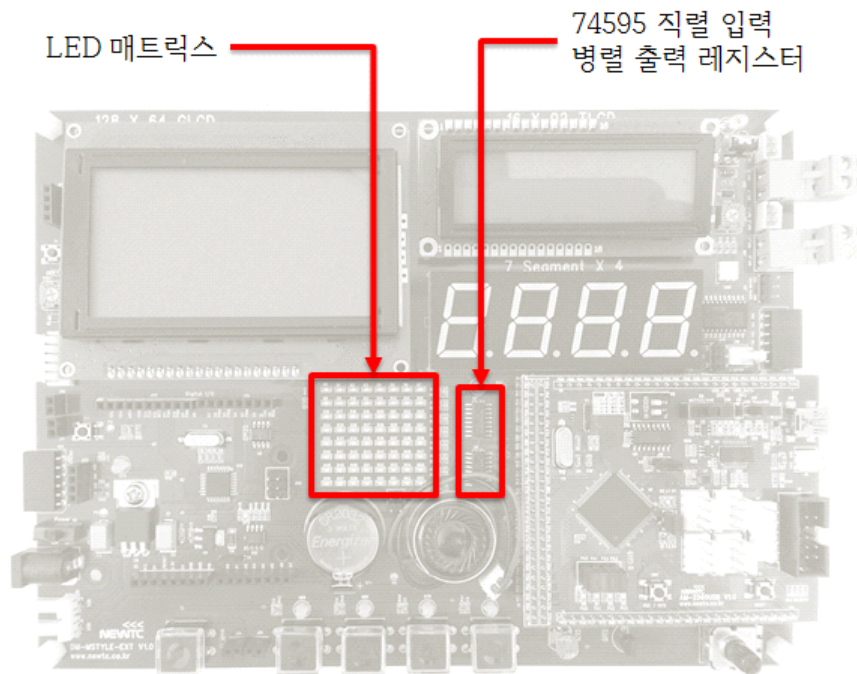


그림 7. 학습보드에 포함된 LED 매트릭스와 74595 칩

3. LED 매트릭스 제어

LED 매트릭스의 연결 방법은 복잡하지만 사용 방법은 간단하다. LED 매트릭스에 스마일 문자를 표시해 보자. 스마일 문자의 정의는 그림 8과 같다.

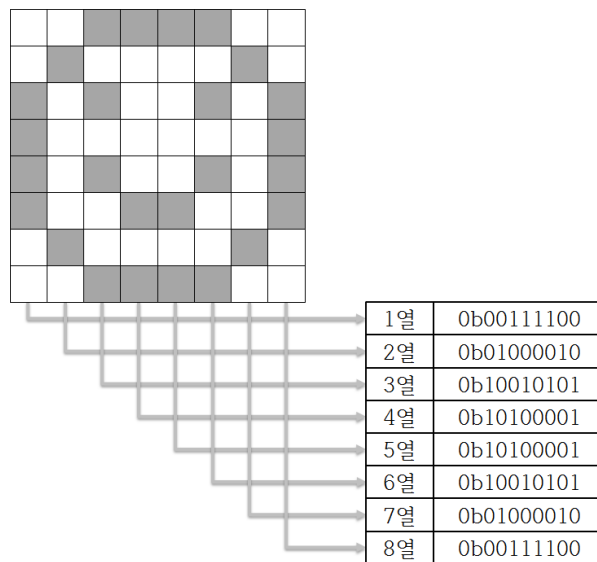


그림 8. 스마일 문자

74595 칩으로 데이터를 전송하기 위해서는 데이터와 데이터 이동을 위한 클럭을 함께 전송해야 하며 이를 위해 shiftOut 함수를 사용한다.

```
void shiftOut(uint8_t dataPin, uint8_t clockPin, uint8_t bitOrder, uint8_t value)
```

- 매개변수

dataPin : 비트 출력이 일어날 핀

clockPin : 한 비트 출력 후 데이터 출력을 알려주기 위해 펄스가 출력되는 핀

bitOrder : 비트 출력 순서, MSBFIRST 또는 LSBFIRST

value : 출력될 데이터 (byte)

- 반환값 : 없음

스케치 1은 LED 매트릭스에 스마일 문자를 표시하는 스케치의 예이다. 열 데이터는 하나의 0만을 가지고 나머지는 모두 1의 값을 가지지만 직관적으로 이해하기 쉽도록 하나의 1만을 가지고 나머지는 모두 0의 값을 가지도록 정의하고 출력 시 반전하여 사용하였다.

스케치 1. 스마일 문자 출력

```
int latchPin = 31;
```

```
int clockPin = 32;
```

```
int dataPin = 30;
```

```
// 행 데이터로 선택된 LED만 1의 값을 가짐
```

```
byte row_data[] = {
```

```
    0b00111100, 0b01000010,
```

```
    0b10010101, 0b10100001,
```

```
    0b10100001, 0b10010101,
```

```
    0b01000010, 0b00111100 };
```

```
// 열 데이터로 선택된 열만 1로 표시되어 있음. 실제로는 반전하여 출력
```

```
byte col_data[8] = {
```

```
    0x01, 0x02, 0x04, 0x08,
```

```
    0x10, 0x20, 0x40, 0x80 };
```

```
void setup()
{
  pinMode(latchPin, OUTPUT);          // 각 제어 핀을 출력으로 설정
  pinMode(dataPin, OUTPUT);
  pinMode(clockPin, OUTPUT);
}

void loop()
{
  for(int i = 0; i < 8; i++){
    digitalWrite(latchPin, LOW);      // 래치 출력 금지

    // 열 데이터 → 행 데이터 순으로 출력
    shiftOut(dataPin, clockPin, MSBFIRST, ~col_data[i]); // 반전 출력
    shiftOut(dataPin, clockPin, MSBFIRST, row_data[i]);

    digitalWrite(latchPin, HIGH);     // 래치 출력 허용
  }
}
```