7 segment with 74HC595 IC

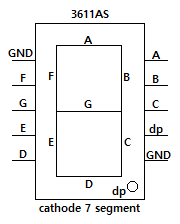
김성희

# 설명

7 segment와 74HC595 칩을 이용하여 0~9, 0**.** ~ 9**.** 에 대한 카운트를 반복하는 프로그램을 짠다.

# 회로도

## 7 segment - 3611AS

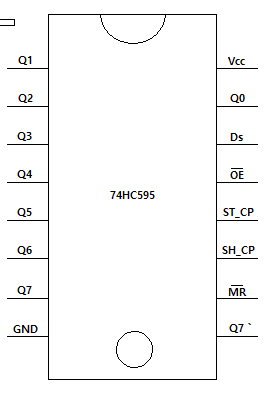


A ~ G, dp pin == cathode

## 74HC595

MR : Master Reset (input)

OE : Output Enable (input)

ST\_CP : Latch Clock (input)

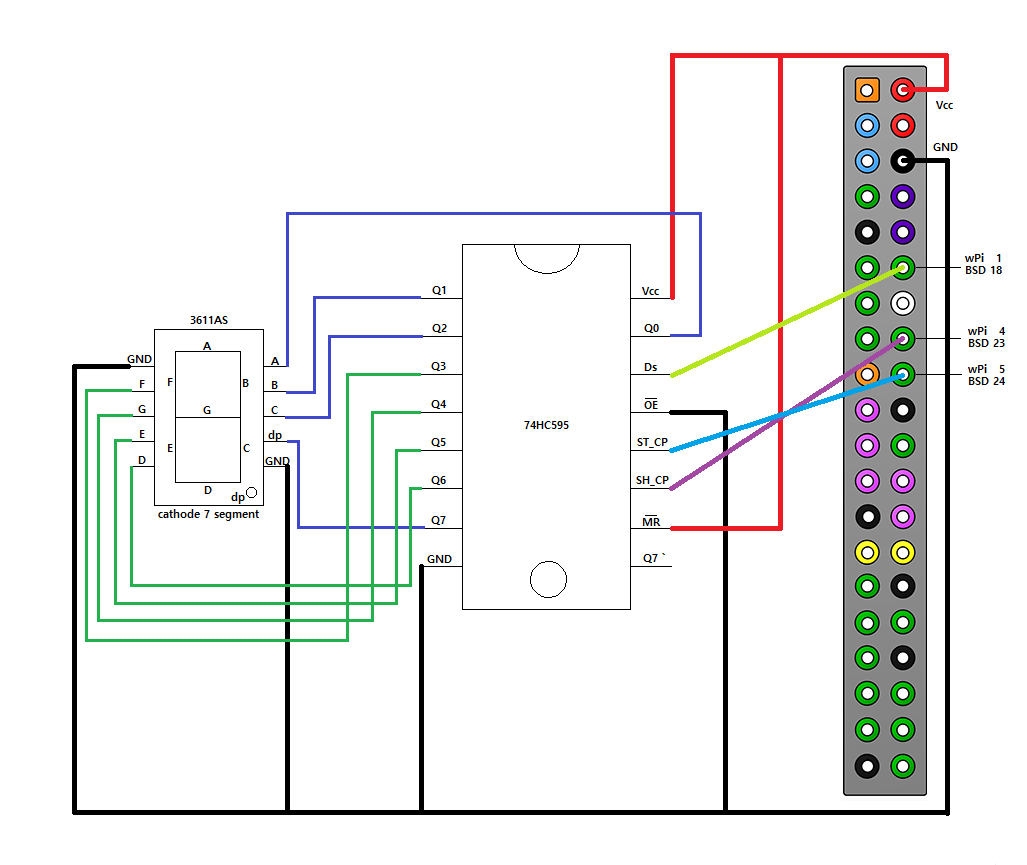
SH\_CP : Shfit Clock (input)

Ds : Data sequences (input)

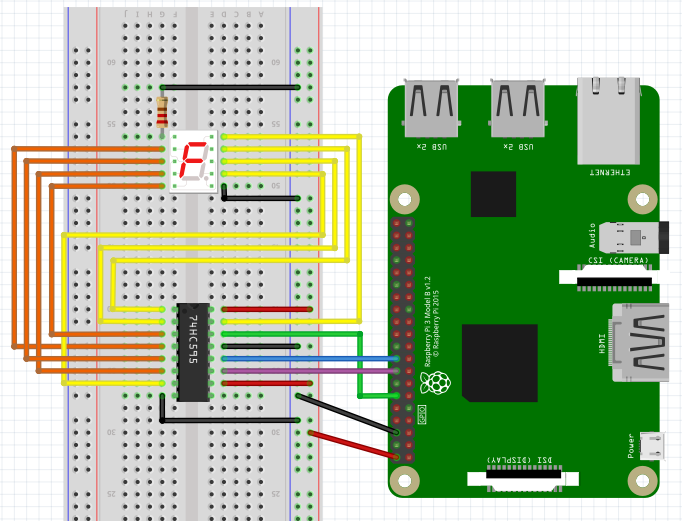
Q0 ~ Q7 : output pin

## 전체 연결

### 논리



### 물리



# 동작 원리

* 74HC595는 데이터를 읽을 때 사용하는 shift register와 데이터를 저장하는 Latch register가 존재한다.
* 74HC595의 Q0~7 핀을 7segment의 A~G,dp와 순서대로 연결했다.
* 따라서 1을 켜기 위해서 b01100000 데이터를 Ds pin으로 보내야 하며, 뒤의 데이터부터 보내야 Q0~7에 01100000가 순서대로 들어간다.

1. Write Data to Shift register from Ds pin

SH\_CP로 clock 신호를 보내서 데이터를 쓴다. (rising edge clock에 쓴다.)

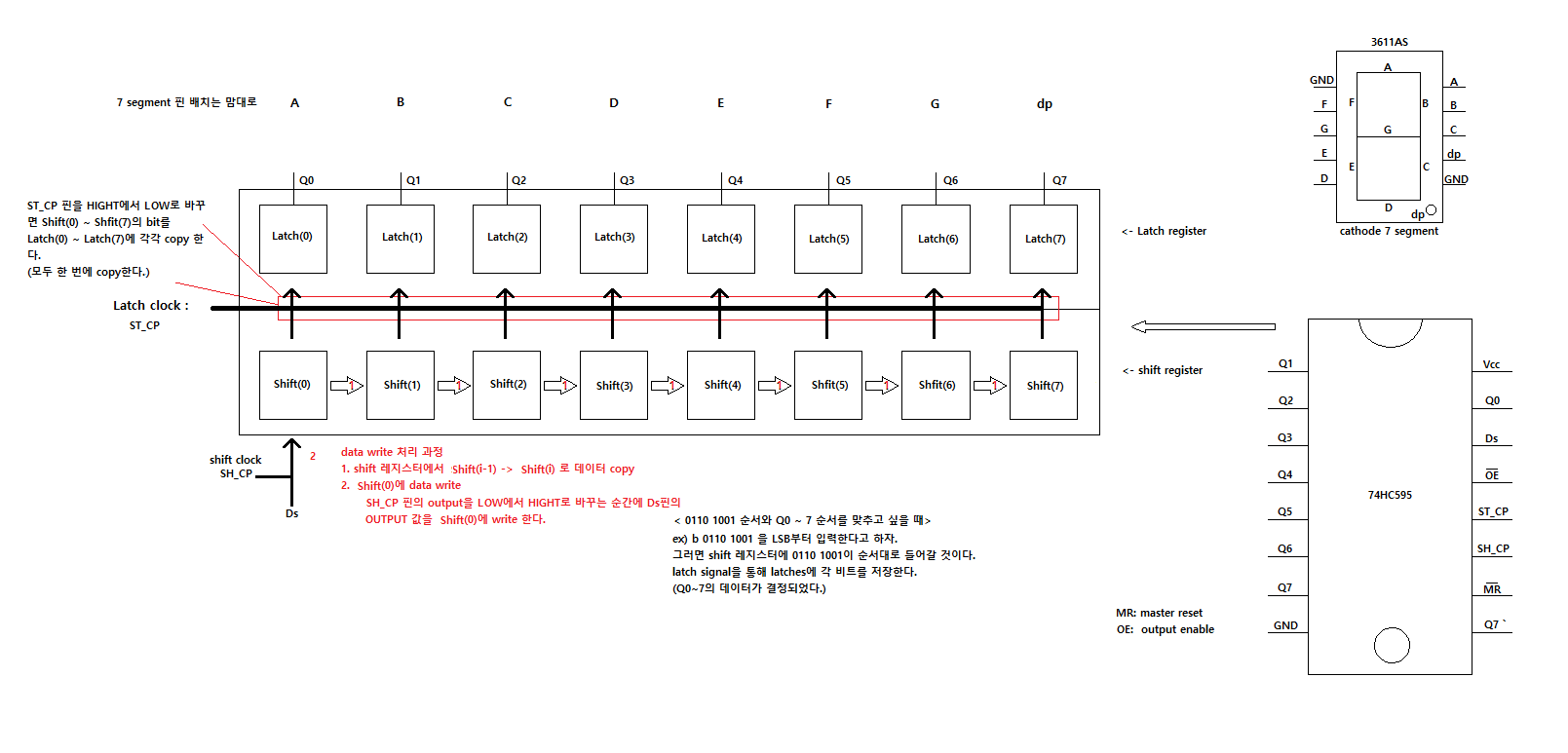
데이터를 shift register에 쓰기 전에 먼저 shift register의 각 latch의 값을 오른쪽으로 밀고 가장 왼쪽에 있는 latch에 쓴다. (오른쪽, 왼쪽은 편의상 설정.)

1. Copy Data to Latch register from Shift register

ST\_CP로 clock 신호를 보내서 데이터를 복사한다. (falling edge clock에 쓴다.)

Shift register의 8개의 latch의 data(bit)를 Latch register의 8개의 latch에 한 번에 복사한다.

1. Latch register에 저장된 비트 데이터에 따라서 7 segment의 led가 켜진다.

[동작 원리 그림]

# 이슈와 해결

“사물인터넷을 위한 리눅스 프로그래밍 with 라즈베리파이” 책에서 IC 7447을 이용해서 7 segment를 켜는 예제가 있다. 그러나 현재 가지고 있는 IC 칩이 74HC595여서 데이터시트와 매뉴얼을 보고 대강의 원리를 찾아서 코드를 직접 짜게 되었다. 가장 헷갈렸던 부분이 비트열을 write할 때, 74HC595의 output pin Q0~7과 어떻게 매칭되는지였다. 상식적으로 비트열의 MSB와 Q0, LSB와 Q7 pin이 대응될 것이라 예상하여 실험을 해봤고 결과적으로 잘 들어 맞았다.

두 번째 이슈가 있었는데, 저항을 연결하지 않아서 LED가 망가졌다. 7 segment input pin 8개에 각각 연결하고 새로운 7 segment display를 껴서 해결했다. 이 후에 7 segment의 GDN하나에만 저항을 연결하고 시도해봤더니 정상적으로 잘 작동했다. 회로를 다룰 때 항상 쇼트에 대한 손상을 고려해야 겠다는 생각이 들었다.

# 소스 코드

|  |
| --- |
| #include <stdio.h>  #include <stdlib.h>  #include <wiringPi.h>  #define PIN\_LATCH 5  #define PIN\_SHIFT 4  #define PIN\_DATA 1  #define PIN\_HIGH(X) digitalWrite(X, HIGH)  #define PIN\_LOW(X) digitalWrite(X, LOW)  #define PIN\_SET(X,Y) digitalWrite(X,Y)  #define LATCH\_HIGH() PIN\_SET(PIN\_LATCH, HIGH)  #define LATCH\_LOW() PIN\_SET(PIN\_LATCH, LOW)  #define LATCH\_SIGNAL() {LATCH\_HIGH();LATCH\_LOW();}  #define SHIFT\_HIGH() PIN\_SET(PIN\_SHIFT, HIGH)  #define SHIFT\_LOW() PIN\_SET(PIN\_SHIFT, LOW)  #define SHIFT\_SIGNAL() {SHIFT\_LOW();SHIFT\_HIGH();}  #define DATA\_SET(X) PIN\_SET(PIN\_DATA, X)  void pin\_init(void);  void write\_data(unsigned int data\_out, int dp);  // cathode 방식  const int seg7[10] = {  // 0x?? => 0b A B C D E F G dp ( Q0~7을 A, B, C, ~ , dp 에 연결 )  // 1을 켤라면, BC를 켜야 하므로 b0110 0000 == 0x60  // 2를 켤라면, ABDEG를 켜야 하므로 b1101 1010 == 0xDA  0xFC, 0x60, 0xDA, 0xF2, 0x66, 0xB6, 0xBE, 0xE4, 0xFE, 0xE6  };  void write\_data(unsigned int data\_out, int dp)  {  int i;  unsigned hold;  data\_out |= dp; // dp : either 0 or 1  for (i = 0; i < 8; i++) {  DATA\_SET( (data\_out >> i) & 0x01 );  SHIFT\_SIGNAL();  }  LATCH\_SIGNAL();  }  //#define TEST  int main(void)  {  pin\_init();  #ifdef TEST  int pos = 1;  while (1) {  write\_data(pos);  printf("%d\n", pos);  pos = (pos >= 128) ? 1 : (pos << 1);  delay(1000);  }  #endif  #ifndef TEST  int counter = 0;  int dp = 0; // dot : either 0 or 1  while (1) {  write\_data(seg7[counter], dp);  printf("%d\n", counter);  if (counter >= 9) {  counter = 0;  dp ^= 1; //toggle  } else {  counter++;  }  delay(1000);  }  #endif  return 0;  }  void pin\_init(void)  {  wiringPiSetup();  pinMode(PIN\_LATCH, OUTPUT);  pinMode(PIN\_SHIFT, OUTPUT);  pinMode(PIN\_DATA, OUTPUT);  SHIFT\_LOW();  LATCH\_HIGH();  DATA\_SET(LOW);  } |

# wiringPi 라이브러리

## 설치

wget <https://github.com/WiringPi/WiringPi.git>  
cd WiringPi  
git pull origin  
./build

## 컴파일

gcc wiringfnd.c -o fnd -lwiringPi

## wPi and BSD pin

$ gpio readall

