

Eight LEDs and a Shift Register

Overview



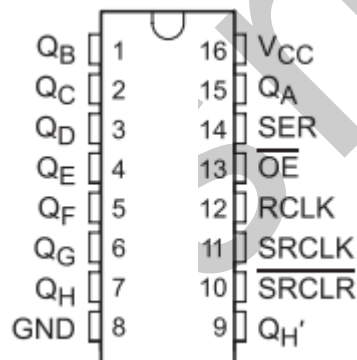
In this lesson, you will learn how to use eight large red LEDs with an Arduino without needing to give up 8 output pins!

Specification

Please view 74HC595-datasheet.pdf

Path: \Public_materials\Datasheet\74HC595-datasheet.pdf

Pin definition







GND	8	10	—	Ground Pin
\overline{OE}	13	17	I	Output Enable
Q_A	15	19	O	Q_A Output
Q_B	1	2	O	Q_B Output
Q_C	2	3	O	Q_C Output
Q_D	3	4	O	Q_D Output
Q_E	4	5	O	Q_E Output
Q_F	5	7	O	Q_F Output
Q_G	6	8	O	Q_G Output
Q_H	7	9	O	Q_H Output
$Q_{H'}$	9	12	O	$Q_{H'}$ Output
RCLK	12	14	I	RCLK Input
SER	14	18	I	SER Input
SRCLK	11	14	I	SRCLK Input
SRCLR	10	13	I	SRCLR Input
NC	—	1	—	No Connection
		16		
		11		
		16		
V_{CC}	—	20	—	Power Pin

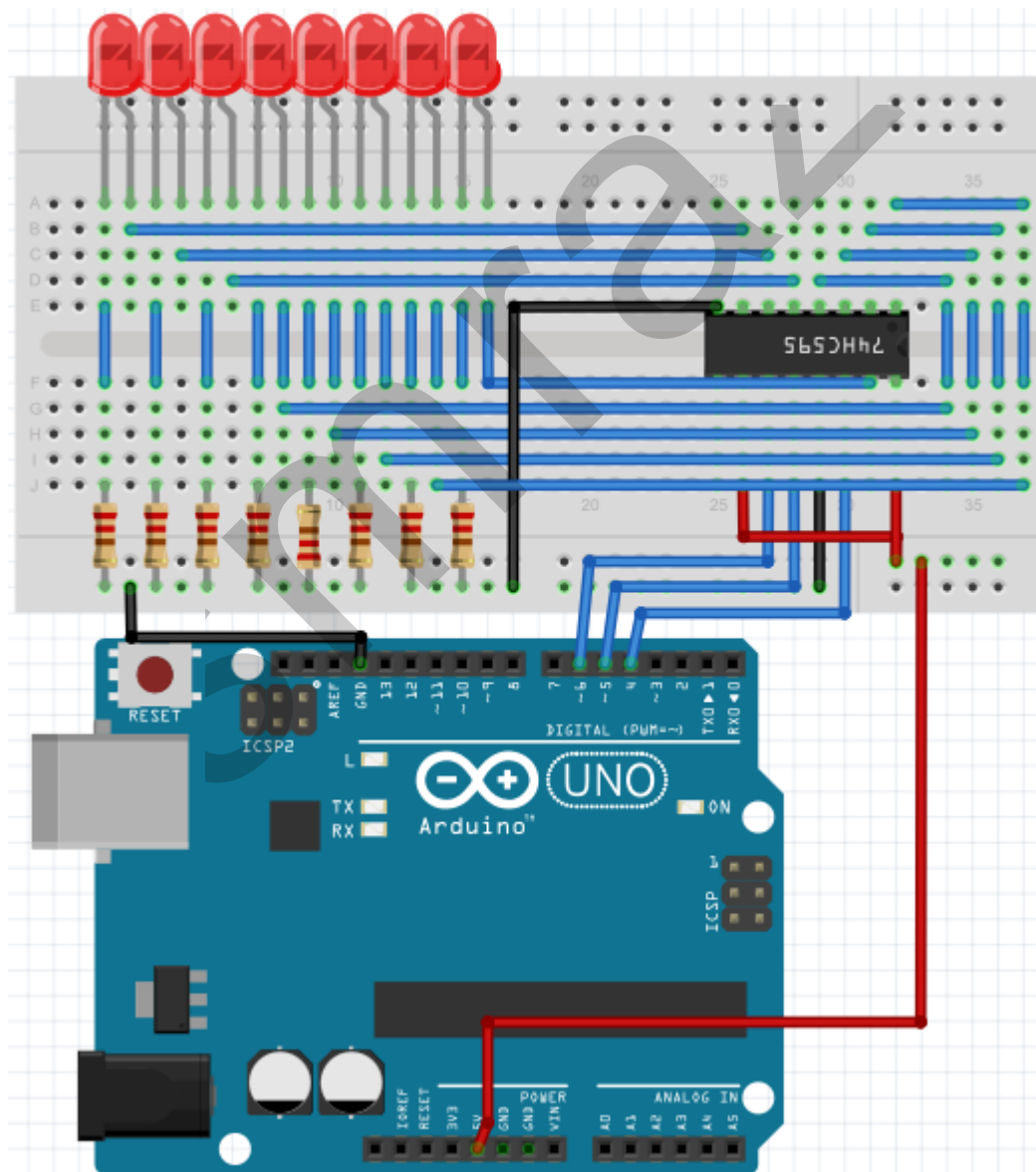
Hardware required

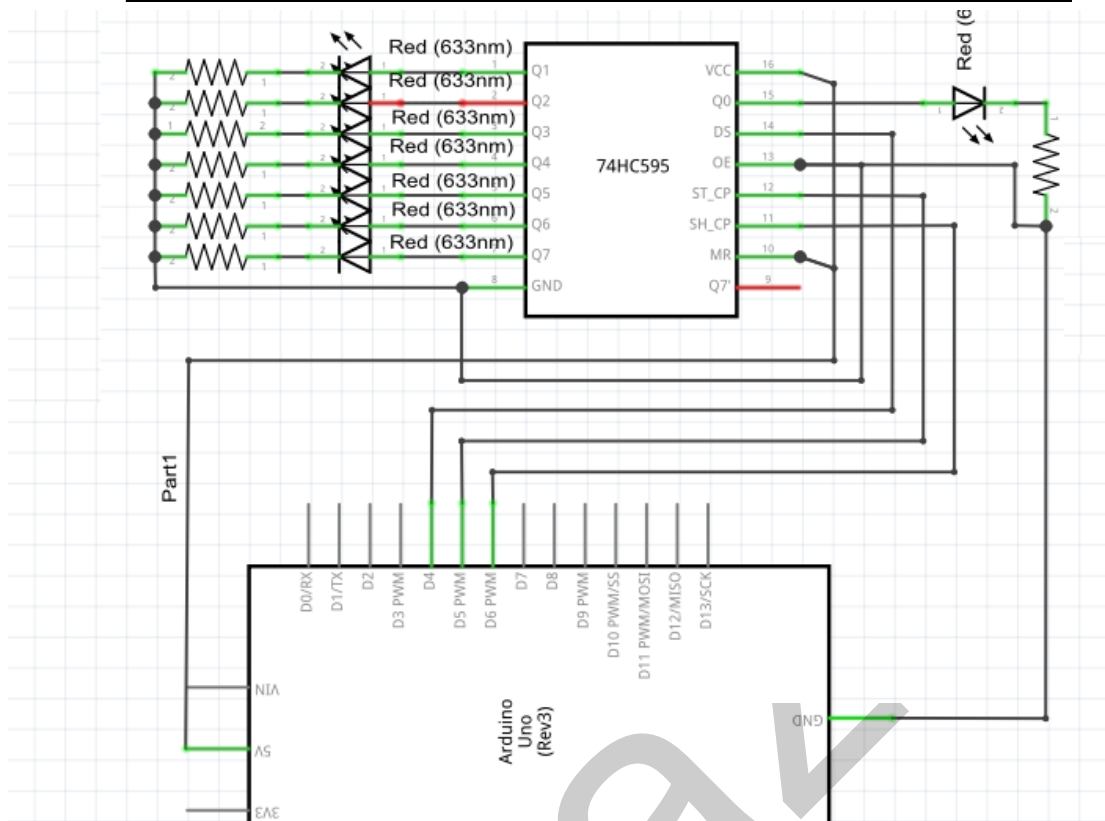
Material diagram	Material name	Number
	74HC595	1
	LED	8
	10KΩ resistor	8

V1.0

	USB Cable	1
	UNO R3	1
	Breadboard	1
	Jumper wires	Several

Connection diagram

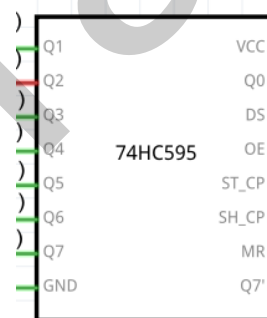




Connection:

Q1 -> LED1
Q2 -> LED2
Q3 -> LED3
Q4 -> LED4
Q5 -> LED5
Q6 -> LED6
Q7 -> LED7
GND -> GND

VCC -> VCC
Q0 -> LED8
DS -> D4
OE -> GND
ST_CP -> D5
SH_CP -> D6
MR -> VCC
Q7' -> null



Note :

Pay attention to the direction of 74HC595.

Sample code

Note: sample code under the **Sample code** folder

```
int latchPin = 5;
int clockPin = 6;
int dataPin = 4;
```

```
byte leds = 0;
```

```
void setup()
{
```

V1.0

```

pinMode(latchPin, OUTPUT);
pinMode(dataPin, OUTPUT);
pinMode(clockPin, OUTPUT);
}

void loop()
{
  leds = 0;
  updateShiftRegister();
  delay(500);
  for (int i = 0; i < 8; i++)
  {
    bitSet(leds, i);
    updateShiftRegister();
    delay(500);
  }
}

void updateShiftRegister()
{
  digitalWrite(latchPin, LOW);
  shiftOut(dataPin, clockPin, LSBFIRST, leds);
  digitalWrite(latchPin, HIGH);
}
/*
The function 'updateShiftRegister', first of all sets the latchPin to low, then calls the Arduino
function 'shiftOut' before putting the 'latchPin' high again. This takes four parameters, the
first two are the pins to use for Data and Clock respectively.
The third parameter specifies which end of the data you want to start at. We are going to
start with the right most bit, which is referred to as the 'Least Significant Bit' (LSB).
The last parameter is the actual data to be shifted into the shift register, which in this case is
'leds'.
If you wanted to turn one of the LEDs off rather than on, you would call a similar Arduino
function (bitClear) on the 'leds' variable. This will set that bit of 'leds' to be 0 and you would
then just need to follow it with a call to 'updateShiftRegister' to update the actual LEDs.
*/

```

Language reference

[byte](#)

Application effect

3 LED ports can be used to control the eight IO.

You will see all the LEDs turn on or turn off regularly.