**Lesson 24**

**HAL. SPI. LED Static indication**

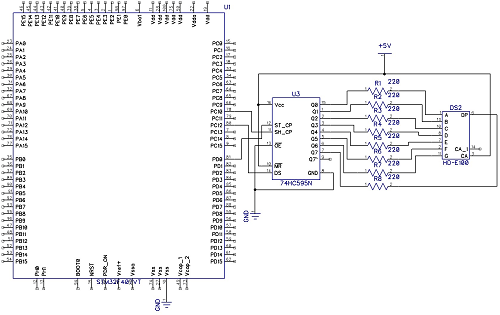
In the [**last lesson,**](http://narodstream.ru/stm-urok-23-hal-spi-sdvigovyj-registr-74hc595/) we connected via the **SPI** bus the **74HC595** chip , which is a shift register.

Today, with the help of this chip, we will try to adjust the seven-segment LED indicator with a common anode.

Here is the pinout of this indicator

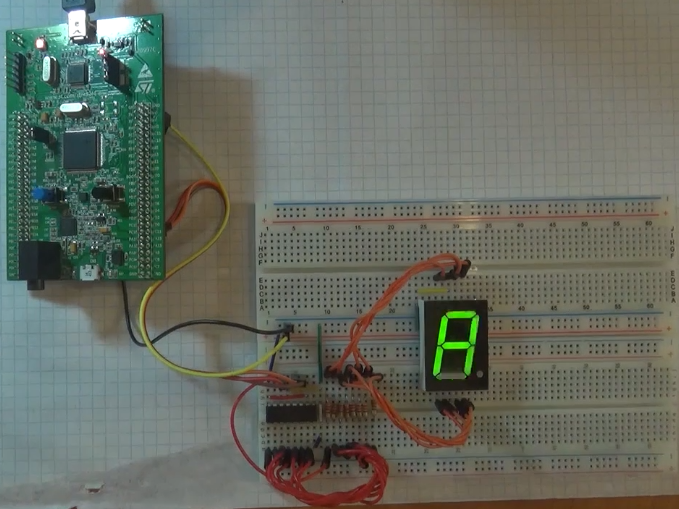


And here it is the connection scheme (click on the image to enlarge the image)

[](http://narodstream.ru/wp-content/uploads/2017/01/Image01.jpg)

Similarly, everything is connected, just like the LEDs, Only the segments are already connected not to the common wire, but to the power wire, and to the controller they are already reversed by cathodes.

But here we have everything connected in practice



Also note that we already with the same shift register have already connected the indicator to the controller AVR in the [**appropriate lesson**](http://narodstream.ru/avr-urok-26-spi-podklyuchaem-led-indikator/) . So those interested can look.

We create the project from the previous project SPI595, we call it SPI\_595\_LED.

From the project LED\_STAT we copy the files led.c and led.h to the corresponding directory.

Start the Cube. Generally nothing is done there, we generate and open the project in Keil.

We move on to the project. We connect the file led.c and configure the programmer.

We will collect the project.

Connect led.h to main.h

#include "stm32f4xx\_hal.h"

**#include "led.h"**

In the main () function, we correct the lines as follows, since everything is upside down, and in order to extinguish the LEDs, we need to reverse all the outputs of the unit

        cs\_set ();

~~aTxBuffer [0] = 0x01;~~

~~HAL\_SPI\_Transmit (& hspi3, (uint8\_t \*) aTxBuffer, 1, 5000);~~

~~cs\_strob ();~~

~~HAL\_Delay (1000);~~

        aTxBuffer [0] = 0xFF;

        HAL\_SPI\_Transmit (& hspi3, (uint8\_t \*) aTxBuffer, 1, 5000);

        cs\_strob ();

        HAL\_Delay ( **200**);

In the infinite loop, we correct the cycle as follows

  / \* USER CODE BEGIN WHILE \* /

  while (1)

  {

                for (i = 0; i <= **9**; i ++)

                {

**segchar (i);**

                        HAL\_Delay ( **1000**);

                }

  / \* USER CODE END WHILE \* /

The main patches will affect the led library. Therefore, go to the file led.h

Here we clean it completely

#include "main.h"

~~#define SA GPIO\_PIN\_7~~

~~#define SB GPIO\_PIN\_8~~

~~#define SC GPIO\_PIN\_9~~

~~#define SD GPIO\_PIN\_10~~

~~#define SE GPIO\_PIN\_11~~

~~#define SF GPIO\_PIN\_12~~

~~#define SG GPIO\_PIN\_13~~

~~#define SH GPIO\_PIN\_14~~

#define SA\_SET HAL\_GPIO\_WritePin (GPIOE, SA, GPIO\_PIN\_RESET);

From the next 16 lines, we delete first the semicolon, we'll put it in the calls.

In the file led.c we add the following variables (the first two can be copied from the file main.c)

#include "led.h"

**extern uint8\_t aTxBuffer [1];**

**extern SPI\_HandleTypeDef hspi3;**

**uint8\_t portseg = 0;**

We need a variable portseg to store the state of the output port of our chip, which we do not track, but we will just remember.

Also copy from the main.c file here these lines and insert them into led.c

uint8\_t portseg = 0;

**#define cs\_set () HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_0, GPIO\_PIN\_RESET)**

**#define cs\_reset () HAL\_GPIO\_WritePin (GPIOD, GPIO\_PIN\_0, GPIO\_PIN\_SET)**

**#define cs\_strob () cs\_reset (); cs\_set ()**

void segchar (uint8\_t seg)

Let's return to led.h.

First we fix the first line, part of the code can be taken from main.c

#define SA\_SET **aTxBuffer [0] = portseg & = ~ 0x01; HAL\_SPI\_Transmit (& hspi3, (uint8\_t \*) aTxBuffer, 1,5000); cs\_strob ()**

#define SA\_RESET HAL\_GPIO\_WritePin (GPIOE, SA, GPIO\_PIN\_SET);

Now we make the second line, copying the code from the first, slightly converting it

**#define SA\_RESET aTxBuffer [0] = portseg | = 0x01; HAL\_SPI\_Transmit (& hspi3, (uint8\_t \*) aTxBuffer, 1,5000); cs\_strob ()**

Now the code from these lines will be transferred to the remaining 14 lines, changing only the bit in portseg

#define SB\_SET aTxBuffer [0] = portseg & = ~ **0x02**; HAL\_SPI\_Transmit (& hspi3, (uint8\_t \*) aTxBuffer, 1,5000); cs\_strob ()

#define SB\_RESET aTxBuffer [0] = portseg | = **0x02**; HAL\_SPI\_Transmit (& hspi3, (uint8\_t \*) aTxBuffer, 1,5000); cs\_strob ()

#define SC\_SET aTxBuffer [0] = portseg & = ~ **0x04**; HAL\_SPI\_Transmit (& hspi3, (uint8\_t \*) aTxBuffer, 1,5000); cs\_strob ()

#define SC\_RESET aTxBuffer [0] = portseg | = **0x04**; HAL\_SPI\_Transmit (& hspi3, (uint8\_t \*) aTxBuffer, 1,5000); cs\_strob ()

#define SD\_SET aTxBuffer [0] = portseg & = ~ **0x08**; HAL\_SPI\_Transmit (& hspi3, (uint8\_t \*) aTxBuffer, 1,5000); cs\_strob ()

#define SD\_RESET aTxBuffer [0] = portseg | = **0x08**; HAL\_SPI\_Transmit (& hspi3, (uint8\_t \*) aTxBuffer, 1,5000); cs\_strob ()

#define SE\_SET aTxBuffer [0] = portseg & = ~ **0x10**; HAL\_SPI\_Transmit (& hspi3, (uint8\_t \*) aTxBuffer, 1,5000); cs\_strob ()

#define SE\_RESET aTxBuffer [0] = portseg | = **0x10**; HAL\_SPI\_Transmit (& hspi3, (uint8\_t \*) aTxBuffer, 1,5000); cs\_strob ()

#define SF\_SET aTxBuffer [0] = portseg & = ~ **0x20**; HAL\_SPI\_Transmit (& hspi3, (uint8\_t \*) aTxBuffer, 1,5000); cs\_strob ()

#define SF\_RESET aTxBuffer [0] = portseg | = **0x20**; HAL\_SPI\_Transmit (& hspi3, (uint8\_t \*) aTxBuffer, 1,5000); cs\_strob ()

#define SG\_SET aTxBuffer [0] = portseg & = ~ **0x40**; HAL\_SPI\_Transmit (& hspi3, (uint8\_t \*) aTxBuffer, 1,5000); cs\_strob ()

#define SG\_RESET aTxBuffer [0] = portseg | = **0x40**; HAL\_SPI\_Transmit (& hspi3, (uint8\_t \*) aTxBuffer, 1,5000); cs\_strob ()

#define SH\_SET aTxBuffer [0] = portseg & = ~ **0x80**; HAL\_SPI\_Transmit (& hspi3, (uint8\_t \*) aTxBuffer, 1,5000); cs\_strob ()

#define SH\_RESET aTxBuffer [0] = portseg | = **0x80**; HAL\_SPI\_Transmit (& hspi3, (uint8\_t \*) aTxBuffer, 1,5000); cs\_strob ()

We collect, sew, look



In the [**next lesson**](http://narodstream.ru/stm-urok-25-hal-spi-led-dinamicheskaya-indikaciya/) we will try to work with dynamic indication by connecting two shift registers at once.