UART communication on STM32 Microcontrollers using HAL

You can use the STM32CubeMX tool to create the necessary config. files to enable the SPI-Drivers.
The HAL library provides the necessary functions to communicate to with the SPI protocol.
I'm going to show you how to output SPI with the HAL library using a 74HC595 8-Bit shift register.

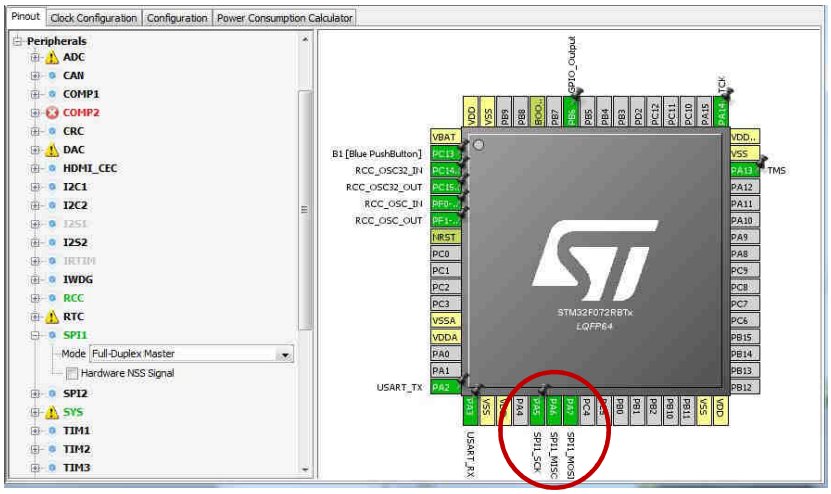
This tutorial uses the following equipment:

- NUCLEO-F072RB Board
- Keil uVision 5 with the necessary packages for Nucleo boards installed
- STLink USB Driver
- STM32CubeMX
- 74HC595 shift register chip
- (Oscilloscope to analyse the signal)

STM32CubeMX

Generating the config. files from STM32CubeMX.

1. Open STM32CubeMX and open a new project.
2. Select the Nucleo-F072RB from the Borards tab
3. Enable FreeRTOS
4. Set the RCC (HSE & LSE) to Crystal/Ceramic Resonator
5. Enable the USART2 port in Asynchronous mode
6. **Enable the SPI1 module and remap the MISO / MOSI / SCK pins to PA5 / PA6 / PA7**
7. Set the baud rate **Prescaler** under Configuration > SPI1 > Parameter Settings to 16
8. At the same menu, set the Data Size to 8 Bits
9. Go to Project > Generate code
10. Enter a project name and select MDK-ARM V5
11. Generate the code and open the project in Keil uVision



The code generator should output the following code:

```
173  /* SPI1 init function */
174  void MX_SPI1_Init(void)
175  {
176
177    hspi1.Instance = SPI1;
178    hspi1.Init.Mode = SPI_MODE_MASTER;
179    hspi1.Init.Direction = SPI_DIRECTION_2LINES;
180    hspi1.Init.DataSize = SPI_DATASIZE_8BIT;
181    hspi1.Init.CLKPolarity = SPI_POLARITY_LOW;
182    hspi1.Init.CLKPhase = SPI_PHASE_1EDGE;
183    hspi1.Init.NSS = SPI_NSS_SOFT;
184    hspi1.Init.BaudRatePrescaler = SPI_BAUDRATEPRESCALER_16;
185    hspi1.Init.FirstBit = SPI_FIRSTBIT_MSB;
186    hspi1.Init.TIMode = SPI_TIMODE_DISABLED;
187    hspi1.Init.CRCCalculation = SPI_CRCCALCULATION_DISABLED;
188    hspi1.Init.CRCPolynomial = 7;
189    hspi1.Init.CRCLength = SPI_CRC_LENGTH_DATASIZE;
190    hspi1.Init.NSSPMode = SPI_NSS_PULSE_ENABLED;
191    HAL_SPI_Init(&hspi1);
192
193  }
194
```

You should check that the DataSize is set to 8-Bit, the rest should be fine as is.
Also you can change the bit order under FirstBit (if not set within the cube Configuration menu).

How to send data over SPI

Sending bytes of data using the function provided within the HAL library.

```
1  uint8_t outputBuffer = 0xFF;
2  HAL_SPI_Transmit(&hspi1,&outputBuffer, 1, 1);
```

The first argument is a pointer to the SPI instance.
The second argument is a pointer to the data to be sent (uint8_t).
The third argument is the size of the data in bytes.
Last argument is the timeout, a value of 1 should do the job just fine.

Sending multiple bytes at once

```
1  uint8_t outputBuffer[4] = {0xFF, 0x0F, 0xF0, 0x00};
2
3  HAL_SPI_Transmit(&hspi1, outputBuffer, 4, 1);
4
```

To send multiple bytes of data, you can create an array as a buffer and give it to the HAL function.
Also change the **size** argument to the number of bytes you want to read.

Example: Blink all leds connected to the 74hc595

```
1  void StartDefaultTask(void const * argument)
2  {
3
4    /* USER CODE BEGIN 5 */
5    uint8_t leds[4] = {0xFF, 0x0F, 0xF0, 0x00};
6    /* Infinite loop */
7    for(;;)
8    {
9      HAL_SPI_Transmit(&hspi1, leds, 1, 1);
10     GPIOB -> ODR &= ~GPIO_PIN_6;
11     GPIOB -> ODR |= GPIO_PIN_6;
12     osDelay(500);
13     HAL_SPI_Transmit(&hspi1, &leds[3], 1, 1);
14     GPIOB -> ODR &= ~GPIO_PIN_6;
15     GPIOB -> ODR |= GPIO_PIN_6;
16     osDelay(500);
17   }
18   /* USER CODE END 5 */
19 }
20
```

Note that PB6 is connected to the latch pin (pin 12, ST_CP) of the 74hc595.

How to receive Data over SPI

```
1  uint8_t outputBuffer[4];
2
3  HAL_SPI_Receive (&hspi1, outputBuffer, 4, 1);
4
```

The listing above reads 4 bytes of data to the outputBuffer.

Kommentiert [BS1]: ToDo: Test this function e.g. with 74hc165

Document Created by Simon Burkhardt

This tutorial is very basic and might not show the best way to use the STM32 environment.
It still might help you get into the whole HAL philosophy of STM if you are coming from another platform. This document is free of copyright under the Creative Commons Zero attribution.

Simon Burkhardt
electronical engineering
simonmartin.ch

History:

V1.0	tested the code, created this document (HAL_SPI_Receive not yet tested)
V1.0.1	added contact info and cc0 notice