پورت سریال STM32 یا UART با توابع HAL - دریافت به روش BLOCKING

تاریخ انتشار۲۴ بهمن, ۱۴۰۲ توسط سید حسین سلطانی

سلام خدمت همه شما مایکروالکامی ها. در مطلب قبلی از سری مطالب STM32 به معرفی ارتباط پورت سریال (UART) در STM32 و ارسال به روش polling یا blocking mode پرداخته شد. در این مطلب به بررسی دریافت دیتا از پورت سریال با استفاده از توابع HAL در میکروکنترلر STM32 پرداخته خواهد شد. پس با من تا انتهای مطلب همراه باشید. همچنین شما میتونید سایر مطالب من رو از این لینک مطالعه و بررسی کنید.

مقدمه

ارتباط سریال یکی از پرکاربردترین پریفرال های سخت افزاری در میکروکنترلر ها و ماژول های الکترونیکی می‌باشد. در مطلب قبل به بررسی روش های blocking mode (روش polling) و non-blocking mode (روش های وقفه و DMA) پرداخته شد. در این مطلب به نحوه راه اندازی پورت سریال UART و دریافت دیتا از طریق پورت سریال در STM32 به روش polling خواهیم پرداخت.

ارسال دیتا از طریق پورت سریال (UART) در STM32

همانطور که در مطلب قبل بررسی شد برای ارسال دیتا با استفاده از پورت سریال (UART) از تابع HAL_UART_Transmit() استفاده می‌شود. با استفاده از این تابع باید علاوه بر مشخص کردن طول دیتا مورد نظر، مدت زمان مورد نیاز ارسال (timeout) را نیز مشخص می‌کردیم.

هرچه دیتا ارسالی طولانی تر باشد زمان مورد نیاز برای ارسال آن نیز بیشتر خواهد شد. در نتیجه باید timeout مشخص شده را متناسب با طول دیتا بکار برد. معمولا 100ms زمان معقولی برای اینکار است. حال اگر رشته (string) ارسالی طولانی تر باشد این زمان نیز بیشتر می‌شود.

در زمان ارسال دیتا از طریق پورت سریال UART با استفاده از این تابع، CPU میکروکنترلر (STM32) در این خط منتظر مانده تا ارسال کامل شود. پس از تکمیل ارسال، CPU از این تابع خارج شده و دستورات بعدی اجرا خواهند شد. از آنجا که CPU در این حالت درگیر شده تا ارسال کامل شود، به این روش blocking mode گویند.

دریافت دیتا از طریق پورت سریال (UART) در STM32

در روش دریافت یا خواندن دیتا از روی پورت سریال به روش polling نیز CPU با فراخوانی تابع دریافت، منتظر دریافت کامل دیتا شده و نهایتا دستورات بعدی را اجرا خواهد کرد. برای دریافت دیتا از پورت سریال توسط توابع HAL بایستی از دستور HAL_UART_Receive() استفاده کرد.

```
HAL_UART_Receive(UART_HandleTypeDef *huart, uint8_t *pData, uint16_t Size,
uint32_t Timeout);
```

ورودی های تابع فوق شامل موارد زیر می‌باشد.

- **ورودی اول:** اشاره گر (pointer) به ساختار تنظیمات UART میکروکنترلر
- **ورودی دوم:** اشاره گر (pointer) به آرایه محل ذخیره سازی دیتا مورد نظر
- **ورودی سوم:** یک عدد uint16_t بیانگر طول دیتای مورد نظر جهت دریافت
- **ورودی چهارم:** بازه زمانی برحسب میلی ثانیه و بیانگر مدت زمان مورد نیاز جهت دریافت دیتا

پیش از دریافت دیتا از پورت سریال (UART) در STM32 ابتدا باید یک آرایه (بافر) جهت ذخیره دیتا مشخص کنیم. طول آرایه باید متناسب با دیتا های دریافتی بوده و به اندازه ای باشد که برای دیتای دریافتی کم نباشد. معمولا طول آرایه را یکی بیشتر از حداکثر طول دیتا تنظیم می‌کنیم.

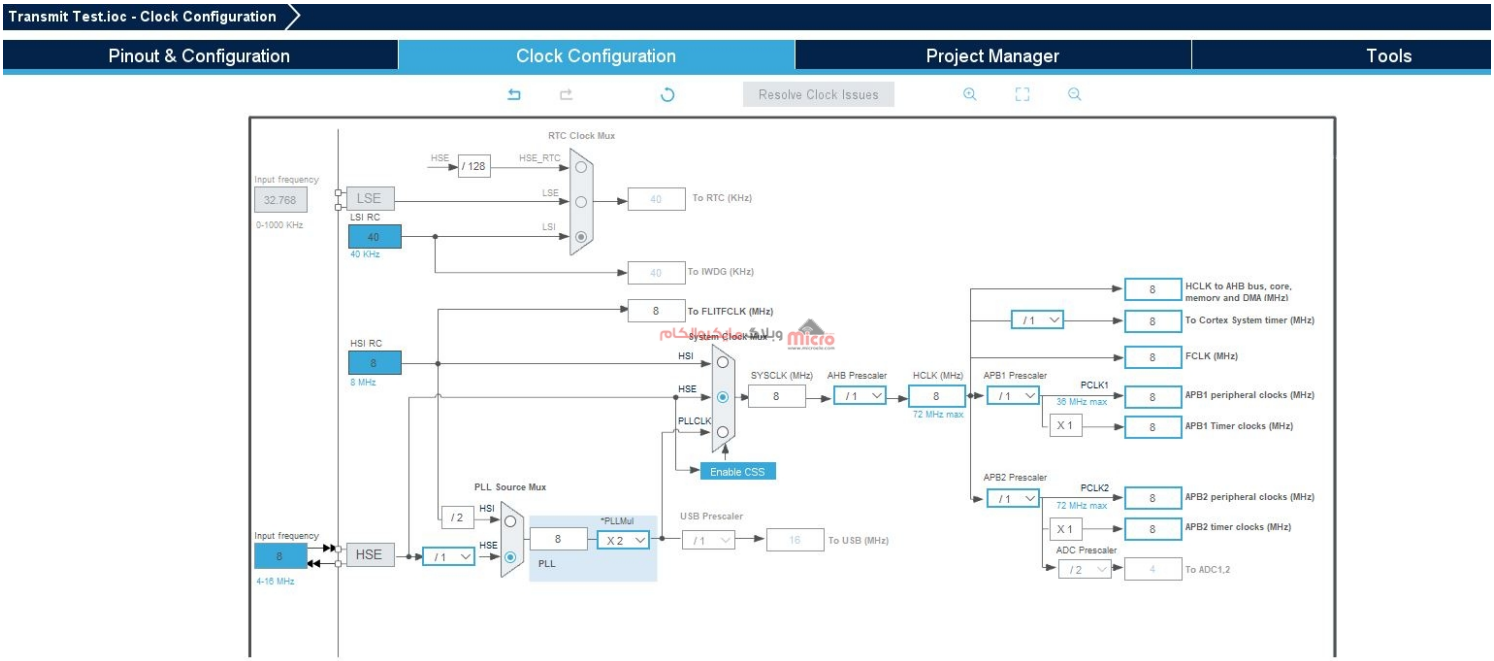
به عنوان مثال برای دریافت یک رشته یا دیتا به اندازه 10 بایت طول دیتا، طول آرایه جهت ذخیره دیتا را 11 انتخاب می‌کنیم. اگر دیتا دریافتی بیشتر از طول آرایه باشد مابقی دیتا را از دست داده و فقط تعداد مشخص شده را قادر به ذخیره آن هستیم.

ایجاد پروژه

در این مطلب برای ارسال دیتا از توابع HAL استفاده می‌کنیم. در ابتدا یک پروژه جدید ایجاد کرده و میکروکنترلر مورد نظر (در این مطلب STM32F103C8T6) را انتخاب و تنظیمات آن را مطابق مراحل زیر انجام می‌دهیم.

تنظیمات کلاک

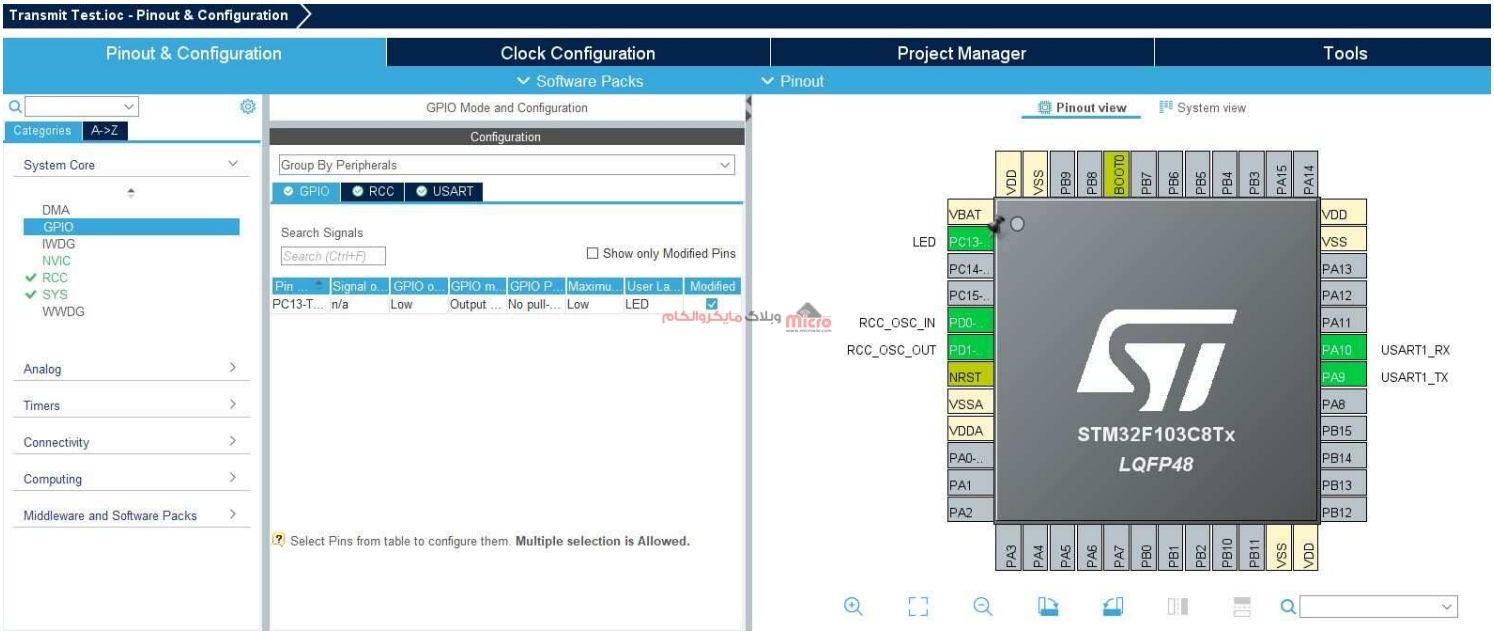
بعد از باز شدن پنجره Mx از بخش System Core وارد RCC شده و منبع کلاک (HSE) را کریستال خارجی انتخاب و نهایتا در بخش Clock Configuration فرکانس کاری میکرو را مطابق با نیاز انتخاب نمایید.



تنظیم کلاک در *STM32* برای ارتباط سریال

تنظیمات GPIO

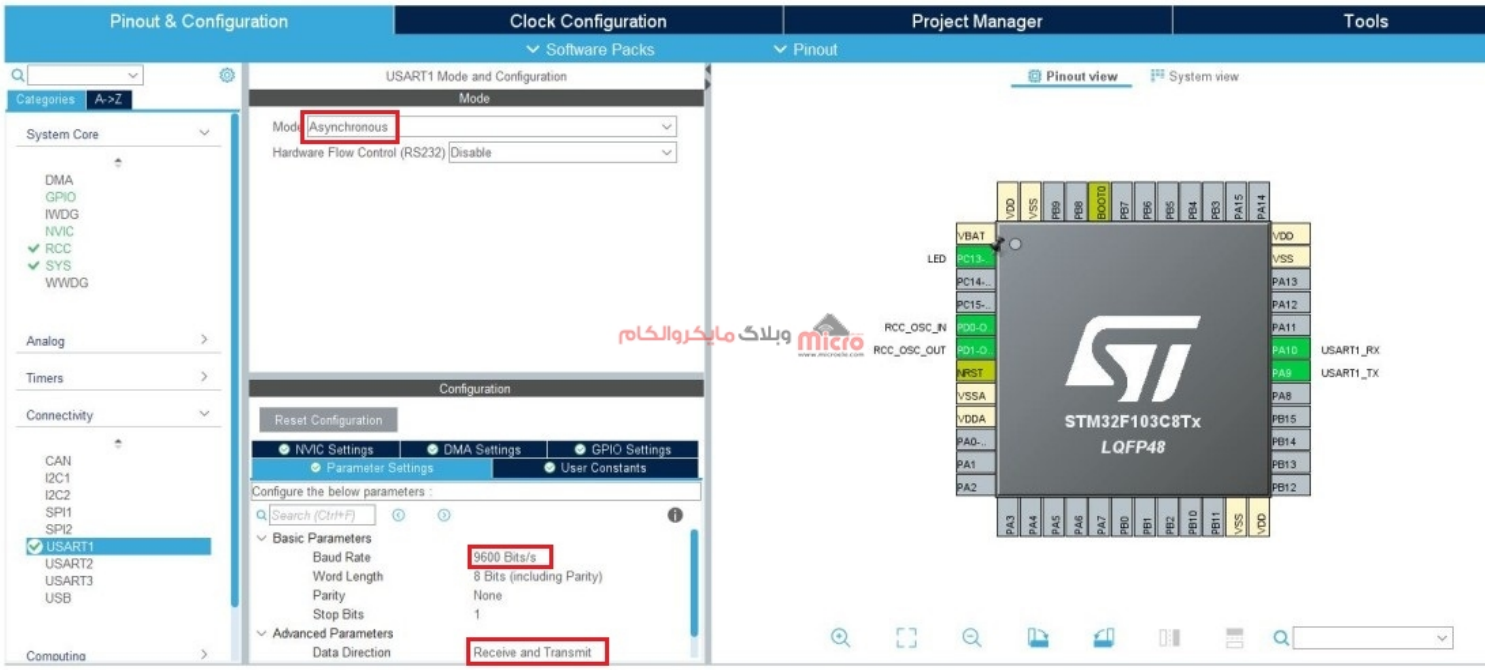
در Pinout & Configuration وارد System Core و GPIO شده و یک پایه را به دلخواه برای اتصال یک LED و چشمک زدن آن انتخاب می‌کنیم. در تصویر زیر PC13 انتخاب شده و یک label با نام LED به آن اختصاص داده شده است.



تنظیمات *GPIO* در *STM32*

تنظیمات اولیه پورت سریال

سپس به بخش Pinout & Configuration بازگشته و از بخش Connectivity گزینه UART1 را انتخاب نمایید. در ابتدا Mode را به حالت Asynchronous تغییر داده و در بخش Parameters Settings باودریت مورد نظر و تنظیمات دلخواه (مطابق با نیاز پروژه) را انجام می‌دهیم. چون نیاز به دریافت و ارسال دیتا داریم حالت را برروی Receive and Transmit تنظیم شده است.



تنظیمات اولیه راه اندازی پورت سریال *UART* در *STM32*

برنامه دریافت دیتا از پورت سریال UART در STM32

در برنامه زیر کد های مربوط به دریافت دیتا از پورت سریال STM32 نوشته شده که توسط دستور HAL_UART_Receive() دیتا را دریافت و آن را در آرایه data ذخیره می‌کنیم. برای بررسی دیتا که آیا برابر با رشته مد نظر است یا خیر از تابع strcmp() استفاده شده است. درصورتی که پیام دریافتی از سریال UART برابر "toggle" باشد، LED تغییر وضعیت داده و پیامی برروی پورت سریال ارسال می‌شود. شرط زیر اگر toggle یا toggle بهمراه enter ارسال شود اجرا خواهد شد.

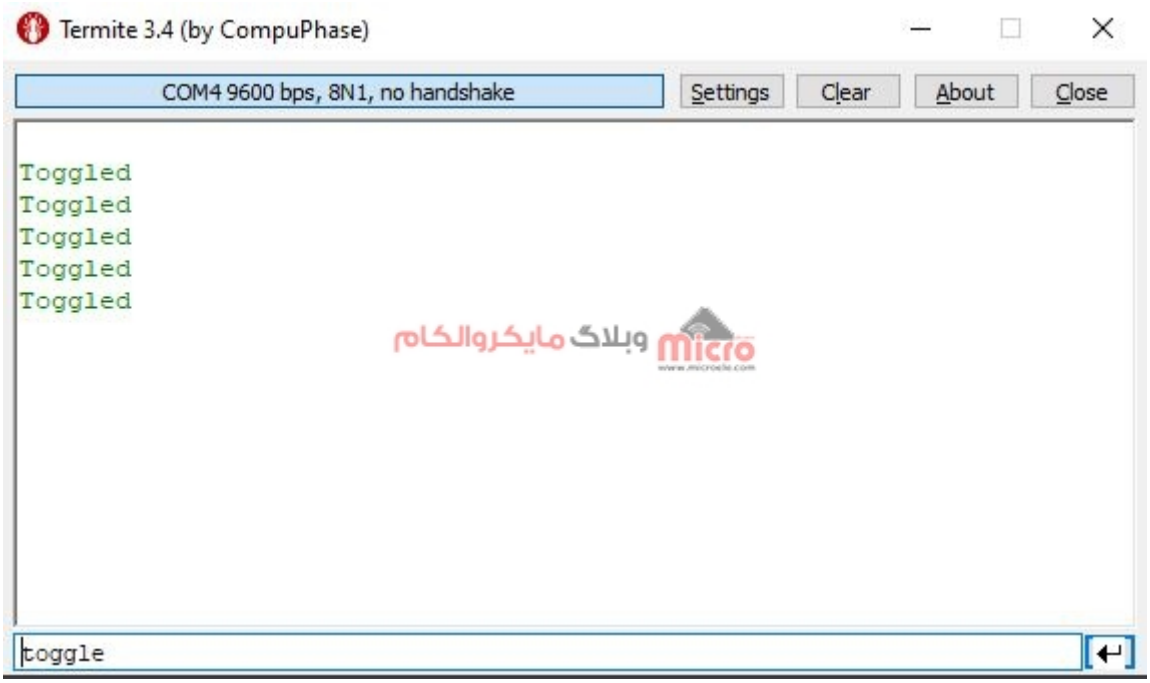
```
#include "main.h"
#include <string.h>

UART_HandleTypeDef huart1;
```

```
void SystemClock_Config(void);
static void MX_GPIO_Init(void);
static void MX_USART1_UART_Init(void);

uint8_t data[11];
unsigned char echo_flag = 0;

int main(void)
{
  HAL_Init();

  SystemClock_Config();

  MX_GPIO_Init();
  MX_USART1_UART_Init();

  while (1)
  {
    HAL_UART_Receive(&huart1, data, 10, 500);

    if ((strcmp((char *)data, "toggle") == 0) || (strcmp((char *)data,
"toggle\r") == 0))
    {
        HAL_UART_Transmit(&huart1, (uint8_t *) "Toggled\r", 8, 50);
        HAL_GPIO_TogglePin(LED_GPIO_Port, LED_Pin);
        memset(data, 0, 11);
    }

  }
}

void SystemClock_Config(void)
```

```
{
  RCC_OscInitTypeDef RCC_OscInitStruct = {0};
  RCC_ClkInitTypeDef RCC_ClkInitStruct = {0};

  RCC_OscInitStruct.OscillatorType = RCC_OSCILLATORTYPE_HSE;
  RCC_OscInitStruct.HSEState = RCC_HSE_ON;
  RCC_OscInitStruct.PLL.PLLState = RCC_PLL_NONE;
  if (HAL_RCC_OscConfig(&RCC_OscInitStruct) != HAL_OK)
  {
    Error_Handler();
  }

  RCC_ClkInitStruct.ClockType = RCC_CLOCKTYPE_HCLK|RCC_CLOCKTYPE_SYSCLK
                              |RCC_CLOCKTYPE_PCLK1|RCC_CLOCKTYPE_PCLK2;
  RCC_ClkInitStruct.SYSCLKSource = RCC_SYSCLKSOURCE_HSE;
  RCC_ClkInitStruct.AHBCLKDivider = RCC_SYSCLK_DIV1;
  RCC_ClkInitStruct.APB1CLKDivider = RCC_HCLK_DIV1;
  RCC_ClkInitStruct.APB2CLKDivider = RCC_HCLK_DIV1;

  if (HAL_RCC_ClockConfig(&RCC_ClkInitStruct, FLASH_LATENCY_0) != HAL_OK)
  {
    Error_Handler();
  }
}

static void MX_USART1_UART_Init(void)
{
  huart1.Instance = USART1;
  huart1.Init.BaudRate = 9600;
  huart1.Init.WordLength = UART_WORDLENGTH_8B;
  huart1.Init.StopBits = UART_STOPBITS_1;
  huart1.Init.Parity = UART_PARITY_NONE;
```

```
  huart1.Init.Mode = UART_MODE_TX_RX;
  huart1.Init.HwFlowCtl = UART_HWCONTROL_NONE;
  huart1.Init.OverSampling = UART_OVERSAMPLING_16;
  if (HAL_UART_Init(&huart1) != HAL_OK)
  {
    Error_Handler();
  }
}

static void MX_GPIO_Init(void)
{
  GPIO_InitTypeDef GPIO_InitStruct = {0};

  __HAL_RCC_GPIOC_CLK_ENABLE();
  __HAL_RCC_GPIOD_CLK_ENABLE();
  __HAL_RCC_GPIOA_CLK_ENABLE();

  HAL_GPIO_WritePin(LED_GPIO_Port, LED_Pin, GPIO_PIN_RESET);

  GPIO_InitStruct.Pin = LED_Pin;
  GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
  HAL_GPIO_Init(LED_GPIO_Port, &GPIO_InitStruct);
}

void Error_Handler(void)
{
  __disable_irq();
  while (1)
  {
  }
```

```
}

#ifdef  USE_FULL_ASSERT

void assert_failed(uint8_t *file, uint32_t line)
{
  /* USER CODE BEGIN 6 */
  /* User can add his own implementation to report the file name and line
number,
     ex: printf("Wrong parameters value: file %s on line %d\r\n", file, line)
*/
  /* USER CODE END 6 */
}
#endif
```



راه اندازی پورت سریال و دریافت دیتا در *STM32*

برای مطالعه قسمت بعدی (روش وقفه) از این لینک اقدام نمایید.

نتیجه گیری

در این مطلب به نحوه راه اندازی پورت سریال STM32 و دریافت دیتا بیان شد. همانطور که پیش تر نیز ذکر گردیده بود روش blocking mode باعث درگیری CPU میکروکنترلر شده و در کاربرد های خاص استفاده نمی‌شود. اما جایگزین آن می‌توان از روش های وقفه یا DMA به این منظور استفاده کرد.

امیدوارم از این مطلب کمال بهره را برده باشید. در صورت داشتن هرگونه نظر یا سوال درباره این مطلب یا تجربه مشابه اون رو در انتهای همین صفحه در قسمت دیدگاه ها قرار بدید. در کوتاه ترین زمان ممکن به اون ها پاسخ خواهم داد. اگر این مطلب براتون فید بود، اون رو به اشتراک بگذارید تا سایر دوستان هم بتوانند استفاده کنند. همینطور میتونید این مطلب را توی اینستاگرام با هشتگ #microelecom به اشتراک بگذارید و پیج مایکروالکام (@microelecom) رو هم منشن کنید.