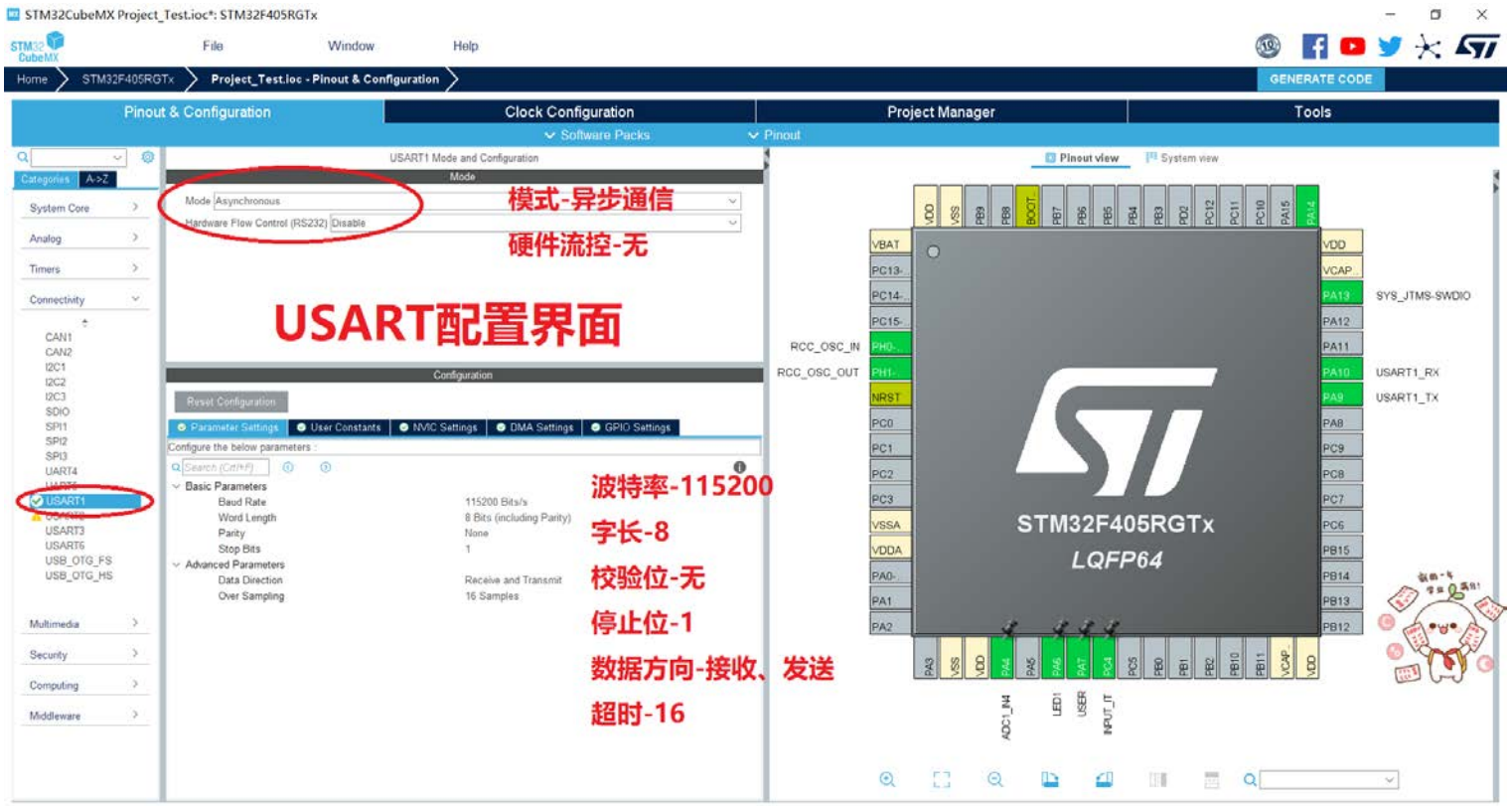
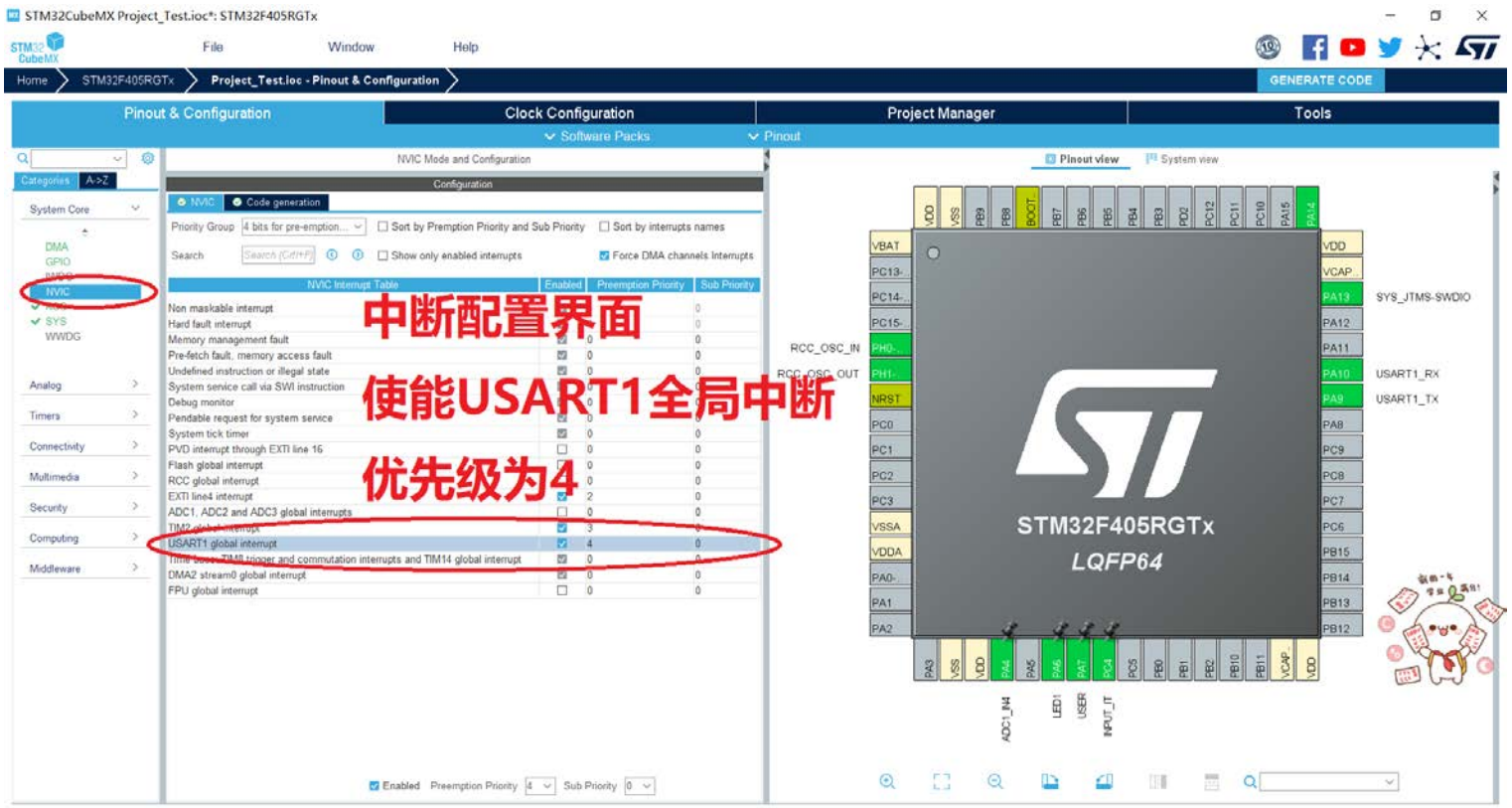
目录

使用 STM32CubeMX 配置 USART





初始化代码

在 usart.c 源文件中，有如下配置代码

```
1.  void MX_USART1_UART_Init(void)
2.  {
3.
4.    /* USER CODE BEGIN USART1_Init 0 */
5.
6.    /* USER CODE END USART1_Init 0 */
7.
8.    /* USER CODE BEGIN USART1_Init 1 */
9.
10.   /* USER CODE END USART1_Init 1 */
11.   huart1.Instance = USART1;
12.   huart1.Init.BaudRate = 115200;
13.   huart1.Init.WordLength = UART_WORDLENGTH_8B;
14.   huart1.Init.StopBits = UART_STOPBITS_1;
15.   huart1.Init.Parity = UART_PARITY_NONE;
16.   huart1.Init.Mode = UART_MODE_TX_RX;
17.   huart1.Init.HwFlowCtl = UART_HWCONTROL_NONE;
18.   huart1.Init.OverSampling = UART_OVERSAMPLING_16;
19.   if (HAL_UART_Init(&huart1) != HAL_OK)
20.   {
21.     Error_Handler();
22.   }
23.   /* USER CODE BEGIN USART1_Init 2 */
24.
25.   /* USER CODE END USART1_Init 2 */
26.
27. }
```

常用 USART 相关操作函数 stm32f4xx_hal_uart.c

1.1 开始串口中断接收 HAL_UART_Receive_IT

```
1.  HAL_StatusTypeDef HAL_UART_Receive_IT(UART_HandleTypeDef *huart, uint8_t *pD
    ata, uint16_t Size)
2.  {
3.  /* Check that a Rx process is not already ongoing */
4.  if (huart->RxState == HAL_UART_STATE_READY)
5.  {
```

```
    if ((pData == NULL) || (Size == 0U))
    {
      return HAL_ERROR;
    }

    /* Process Locked */
    __HAL_LOCK(huart);

    /* Set Reception type to Standard reception */
    huart->ReceptionType = HAL_UART_RECEPTION_STANDARD;

    return(UART_Start_Receive_IT(huart, pData, Size));
  }
  else
  {
    return HAL_BUSY;
  }
}
```

入口参数：USART 句柄、数据存放地址指针、接收数据的大小(Byte)
返回值：HAL_ERROR(失败) or Data or HAL_BUSY(繁忙)
使用示例：

```
uint8_t Buffer;
HAL_UART_Receive_IT(&huart1,&Buffer,1);
```

1.2 开启串口 DMA 接收 HAL_UART_Receive_DMA

```
HAL_StatusTypeDef HAL_UART_Receive_DMA(UART_HandleTypeDef *huart, uint8_t *pData, uint16_t Size)
{
/* Check that a Rx process is not already ongoing */
if (huart->RxState == HAL_UART_STATE_READY)
{
  if ((pData == NULL) || (Size == 0U))
  {
    return HAL_ERROR;
  }

  /* Process Locked */
  __HAL_LOCK(huart);

  /* Set Reception type to Standard reception */
  huart->ReceptionType = HAL_UART_RECEPTION_STANDARD;
```

```
16.
17.   return(UART_Start_Receive_DMA(huart, pData, Size));
18. }
19. else
20. {
21.   return HAL_BUSY;
22. }
23. }
```

入口参数：USART 句柄、数据存放地址指针、接收数据的大小(Byte)
返回值：HAL_ERROR(失败) or Data or HAL_BUSY(繁忙)
使用示例：

```
1.  uint8_t Buffer[10];
2.  HAL_UART_Receive_DMA(&huart1,Buffer,10);
```

1.3 串口接收中断回调函数 HAL_UART_RxCpltCallback

```
1.  __weak void HAL_UART_RxCpltCallback(UART_HandleTypeDef *huart)
2.  {
3.  /* Prevent unused argument(s) compilation warning */
4.  UNUSED(huart);
5.  /* NOTE: This function should not be modified, when the callback is needed,
6.           the HAL_UART_RxCpltCallback could be implemented in the user file
7.   */
8.  }
```

在 HAL 库中，我们对__weak 函数重写，将函数重写至 usart.c 原文件中
使用示例：

```
1.  void HAL_UART_RxCpltCallback(UART_HandleTypeDef *huart)
2.  {
3.    /* Prevent unused argument(s) compilation warning */
4.    UNUSED(huart);
5.    /* NOTE: This function should not be modified, when the callback is needed
    ,
6.             the HAL_UART_RxCpltCallback could be implemented in the user file

7.     */
8.    if(huart->Instance == USART1)
9.    {
10.     //用户代码
11.     HAL_UART_Receive_IT(&huart1,&Buffer,1);
12.   }
```

```
13. }
```

>判断是否是串口 1 发生了接收中断
>用户代码处理
>重新开始接收 1 个字节的串口数据

中断机制讲解

HAL 库的中断机制大同小异，这里以 USART 串口中断为例，讲解 HAL 库中的中断

中断流程

在 stm32f4xx_it.c 源文件中，存放着单片机所有的硬件中断服务函数
以下为串口中断的流程
>硬件调用函数 USART1_IRQHandler

```
1.  void USART1_IRQHandler(void)
2.  {
3.  /* USER CODE BEGIN USART1_IRQn 0 */
4.
5.  /* USER CODE END USART1_IRQn 0 */
6.  HAL_UART_IRQHandler(&huart1);
7.  /* USER CODE BEGIN USART1_IRQn 1 */
8.
9.  /* USER CODE END USART1_IRQn 1 */
10. }
```

>在中断服务函数中，调用了中断处理函数 HAL_UART_IRQHandler

```
1.  HAL_UART_IRQHandler(&huart1);
```

>在中断处理函数中，进行了一系列串口错误的判断(parity,noise,frame,over-run…)
>如果某些错误发生，则停止串口接收
>如果没有错误发生，调用中断接收回调函数 HAL_UART_RxCpltCallback，也就是之前用户重写的函数，在中断接收回调函数中，用户可进行数据的处理，以及开启下一次接收等操作。

判断是何种错误发生

在函数 HAL_UART_IRQHandler 中，有进行类似以下的判断

```
1.  if (((isrflags & USART_SR_PE) != RESET) && ((cr1its & USART_CR1_PEIE) != RES
    ET))
2.  {
3.  huart->ErrorCode |= HAL_UART_ERROR_PE;
```

```
4.  }
```

>每当有错误发生，错误代码 |= 错误位
>用户可根据 huart->ErrorCode 判断是何种错误

6-APR-2021
厦大嘉庚 TCR 嵌入式