YC31xx UART 应用说明

V1.0

Yichip Microelectronics

Revision History

Version	Date	Author	Description
V1.0	2020-2-25	Duanziyang	Initial version

Confidentiality Level:

confidential

目录

1. 文档说明

1.1 编写目的

为使用 UART 相关 Demo 及 UART 库函数 提供指南

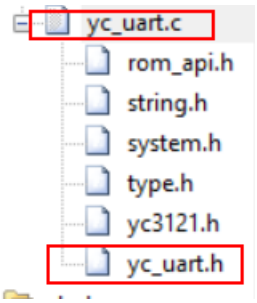
1.2 适用范围

31xx 系列芯片

1.3 文件说明

YC31xx 系列芯片 UARTx 只能使用 DMA 的方式，有两个缓冲区 buffer 做为 DMA 缓冲区，buffer 缓冲区的空间大小可根据实际应用修改

UART Demo 路径为 ModuleDemo\UART
UART 库文件为如下图 uart.c 与 uart.h,路径为 Librarier\sdk



UART Demo 中共有五个示例 Demo，依次做简要说明

1.3.1 UART_Interrupt

该 demo 主要示例 UART 接收字符串中断与发送字符串中断（发送完触发中断）

1.3.2 UART_RX_Recvbuf

该 demo 主要示例 UART 接受字符串中断

1.3.3 UART_RX_RecvData

该 demo 主要示例 UART 接受一个字节数据，并打印出来

1.3.4 UART_TX_Sendbuf

该 demo 示例 UART 发送字符串

1.3.5 UART_TX_SendData

该 demo 示例 UART 发送多个字节数据

2. 结构体说明

UART_InitTypeDef

元素名称	类型	说明	参数项
Mode	uint8_t	Uart 通信模式: 单工，双工全双工	Mode_Single_Line（单工） Mode_duplex （全双工）
RaudRate	uint32_t	波特率如： 9600，38400，115200	Max：300000，默认为 0
DataBits	uint8_t	数据位宽	Databits_8b（8bit，则无奇偶校验位） Databits_9b（9bit，第 9bit 为奇偶校验位）
StopBits	uint8_t	停止位	StopBits_1 （0） StopBits_2 （1）
Parity	uint8_t	奇偶校验位设置	Parity_None （无校验） Parity_Even （偶校验） Parity_Odd （奇校验）
FlowCtrl	uint8_t	指定是启用还是禁用硬件流控制模式。	FlowCtrl_None （禁用流控） FlowCtrl_Enable （启用流控）
RxBufLen	int	Uart DMA rx buff length	指定 Rx DMA buff 长度

3. 库函数说明

3.1 UART_StructInit

函数原型：void UART_StructInit(UART_InitTypeDef* UART_InitStruct);
说明： UART 预初始化函数，将 UART_InitStruct 结构体赋初值，见**注 1**

参数	方向	说明
UART_InitTypeDef* UART_InitStruct	OUT	UART InitStruct 指针指向的结构体将被初始化

表格 3-1-1 UART_StructInit 形参表

返回值	说明
None	None

表格 3-1-2 UART_StructInit 返回值

注 1：UART_InitStruct 初值如下图：

```
UART_InitStruct->BaudRate = 9600;
UART_InitStruct->DataBits = Databits_8b;
UART_InitStruct->FlowCtrl = FlowCtrl_None ;
UART_InitStruct->Mode = Mode_duplex;
UART_InitStruct->StopBits = StopBits_1;
UART_InitStruct->Parity = 0;
```

3.2 UART_Init

函数原型：void UART_Init(UART_TypeDef UARTx, UART_InitTypeDef* UART_InitStruct);
说明： 根据 UART_InitStruct 中的配置初始化 UARTx

参数	方向	说明
UART_TypeDef UARTx	IN	选择 USART 或 UART。该参数可设置为以下值之一：UART0，UART1.
UART_InitTypeDef* UART_InitStruct	IN	该指针指向内容包含所需要配置的 UART 中所有配置参数

表格 3-2-1 UART_Init 形参列表

返回值	说明
None	None

表格 3-2-2 UART_Init 返回值

3.3 UART_DeInit

函数原型：void UART_DeInit(UART_TypeDef UARTx);
说明： UART 去初始化函数，目的是恢复 UARTx 寄存器初始值

参数	方向	说明
UART_TypeDef UARTx	IN	选择 USART 或 UART。该参数可设置为以下值之一：UART0，UART1.

表格 3-3-1 UART_DeInit 形参表

返回值	说明
None	None

表格 3-3-2 UART_DeInit 返回值

3.4 UART_AutoFlowCtrlCmd

函数原型：void UART_AutoFlowCtrlCmd(UART_TypeDef UARTx, FunctionalState NewState);
说明： 启用或禁用 UARTx 自动流控制

参数	方向	说明

UART_TypeDef UARTx	IN	选择 USART 或 UART。该参数可设置为以下值之一：UART0，UART1.
FunctionalState NewState	IN	ENABLE or DISABLE auto flow control

表格 3-4-1 UART_AutoFlowCtrlCmd 形参表

返回值	说明
None	None

表格 3-4-2 UART_AutoFlowCtrlCmd 返回值

3.5 UART_IsRXFIFOFull

函数原型：Boolean UART_IsRXFIFOFull(UART_TypeDef UARTx);
说明： 判断 UARTx 接收空间是否满

参数	方向	说明
UART_TypeDef UARTx	IN	选择 USART 或 UART。该参数可设置为以下值之一：UART0，UART1.

表格 3-5-1 UART_IsRXFIFOFull 形参表

返回值	说明
TRUE	接收空间已满
FALSE	接收空间未满

表格 3-5-2 UART_IsRXFIFOFull 返回值

3.6 UART_IsRXFIFONotEmpty

函数原型：Boolean UART_IsRXFIFONotEmpty(UART_TypeDef UARTx);
说明： 判断接收空间是否为空

参数	方向	说明
UART_TypeDef UARTx	IN	选择 USART 或 UART。该参数可设置为以下值之一：UART0，UART1.

表格 3-6-1 UART_IsRXFIFONotEmpty 形参表

返回值	说明
TRUE	接收空间不为空
FALSE	接收空间为空

表格 3-6-2 UART_IsRXFIFONotEmpty 返回值

3.7 UART_IsUARTBusy

函数原型：Boolean UART_IsUARTBusy(UART_TypeDef UARTx);
说明： 判断 UARTx 是否为忙

参数	方向	说明
UART_TypeDef UARTx	IN	选择 USART 或 UART。该参数可设置为以下值之一: UART0，UART1.

表格 3-7-1 UART_IsUARTBusy 形参表

返回值	说明
TRUE	UART is busy
FALSE	UART is not busy

表格 3-7-2 UART_IsUARTBusy 返回值

3.8 UART_ReceiveData

函数原型: uint8_t UART_ReceiveData(UART_TypeDef UARTx);
说明: 通过 UARTx 接收单个数据（1 字节）

参数	方向	说明
UART_TypeDef UARTx	IN	选择 USART 或 UART。该参数可设置为以下值之一: UART0，UART1.

表格 3-8-1 UART_ReceiveData 形参表

返回值	说明
Data	A byte of data received

表格 3-8-1 UART_ReceiveData 返回值

3.9 UART_RecvBuf

函数原型: int UART_RecvBuf(UART_TypeDef UARTx, uint8_t* buf, int len);
说明: 通过 UARTx DMA buf 接收数据

参数	方向	说明
UART_TypeDef UARTx	IN	选择 USART 或 UART。该参数可设置为以下值之一: UART0，UART1.
uint8_t* buf	OUT	接收数据存放空间的首地址
int len	OUT	接收数据长度

表格 3-9-1 UART_RecvBuf 形参表

返回值	说明
length	Returns the number of bytes accepted

表格 3-9-2 UART_RecvBuf 返回值

3.10UART_ReceiveDataLen

函数原型: uint16_t UART_ReceiveDataLen(UART_TypeDef UARTx);
说明: 获取 UARTx 接收缓冲区中待接收数据长度

参数	方向	说明
UART_TypeDef UARTx	IN	选择 USART 或 UART。该参数可设置为以下值之一: UART0，UART1.

表格 3-10-1 UART_ReceiveDataLen 形参表

返回值	说明
ReceiveDataLen	接收缓冲区中待接收数据长度

表格 3-10-2 UART_ReceiveDataLen 返回值

3.11 UART_SendBuf

函数原型：void UART_SendBuf(UART_TypeDef UARTx, uint8_t* buf, int len);
说明： 通过 UARTx DMA 发送数据，数据发送后返回，**数据发送过程中阻塞**

参数	方向	说明
UART_TypeDef UARTx	IN	选择 USART 或 UART。该参数可设置为以下值之一: UART0，UART1.
uint8_t* buf	IN	待发送发送数据所在的首地址
Int len	IN	待发送数据的长度

表格 3-11-1 UART_SendBuf 形参表

返回值	说明
None	None

表格 3-11-2 UART_SendBuf 返回值

3.12 UART_DMASendBuf

函数原型：void UART_DMASendBuf(UART_TypeDef UARTx,uint8_t* buf,int len);
说明： 发送数据给 UARTx DMA，DMA 特性，**数据发送过程中并不阻塞（与 3.11 区别）**

参数	方向	说明
UART_TypeDef UARTx	IN	选择 USART 或 UART。该参数可设置为以下值之一：UART0，UART1.
uint8_t* buf	IN	待发送发送数据所在的首地址，**注意该指针指向的变量必须在数据发送完之后才可被销毁（最好全局变量）**
int len	IN	待发送数据的长度

表格 3-12-1 UART_DMASendBuf 形参表

返回值	说明
None	None

表格 3-12-2 UART_DMASendBuf 返回值

3.13UART_SendData

函数原型：void　UART_SendData(UART_TypeDef　UARTx, uint8_t　Data);
说明：　　通过 UARTx 发送一个字节数据

参数	方向	说明
UART_TypeDef UARTx	IN	选择 USART 或 UART。该参数可设置为以下值之一：UART0，UART1.
uint8_t　Data	IN	待发送数据（一个字节）

表格 3-13-1　UART_SendData 形参表

返回值	说明
None	None

表格 3-13-2　UART_SendData 返回值

3.14UART_ITConfig

函数原型：void UART_ITConfig(UART_TypeDef UARTx, uint32_t UART_IT,
FunctionalState NewState);

说明：Config Interrupt trigger mode

参数	方向	说明
UART_TypeDef UARTx	IN	选择 USART 或 UART。该参数可设置为以下值之一：UART0，UART1.
uint32_t UART_IT	IN	UART_IT:中断触发方式，可设置以下参数 **UART_IT_TX:** 发送完数据后触发中断. **UART_IT_RX:** 接收数据时触发中断.
FunctionalState NewState	IN	ENABLE or DISABLE UART 中断

表格 3-14-1 UART_ITConfig 形参表

返回值	说明
None	None

表格 3-14-2 UART_ITConfig 返回值

3.15UART_ClearIT

函数原型：void UART_ClearIT(UART_TypeDef UARTx);

说明： Clear IT

参数	方向	说明
UART_TypeDef UARTx	IN	选择 USART 或 UART。该参数可设置为以下值之一：UART0，UART1.

表格 3-15-1 UART_ClearIT 形参表

返回值	说明
None	None

表格 3-15-2 UART_ClearIT 返回值

3.16UART_SetITTimeout

函数原型：void UART_SetITTimeout(UART_TypeDef UARTx,uint16_t timeout);

说明： 设置 UART 接收数据字符间的超时时间，超过该时间触发中断

参数	方向	说明
UART_TypeDef UARTx	IN	选择 USART 或 UART。该参数可设置为以下值之一：UART0，UART1.
uint16_t timeout	IN	接收超时中断时间值（value*X），X 随主频变动。 例：主频 24M，X 为 24

表格 3-16-1 UART_SetITTimeout 形参表

返回值	说明
None	None

表格 3-16-2 UART_SetITTimeout 返回值

3.17UART_SetRxITNum

函数原型：void UART_SetRxITNum(UART_TypeDef UARTx, uint8_t Bcnt);
说明： 设置 UART 接收时一次中断最多接收字节数

参数	方向	说明
UART_TypeDef UARTx	IN	选择 USART 或 UART。该参数可设置为以下值之一：UART0，UART1.
uint8_t Bcnt	IN	0 为不触发，该参数控制中断一次最多接收字节数

表格 3-17-1 UART_SetRxITNum 形参表

返回值	说明
None	None

表格 3-17-2 UART_SetRxITNum 返回值

3.18UART_GetITIdentity

函数原型：uint8_t UART_GetITIdentity(UART_TypeDef UARTx);
说明： 获取 UARTx 的中断号

参数	方向	说明
UART_TypeDef UARTx	IN	选择 USART 或 UART。该参数可设置为以下值之一：UART0，UART1.

表格 3-18-1 UART_ClearIT 形参表

返回值	说明
IT Identity	中断号

表格 3-18-2 UART_ClearIT 返回值

4. Demo 函数说明

4.1 UART_Configuration

```
void UART_Configuration(void)
{
    UART_InitTypeDef UART_InitStruct;
    /* Configure serial ports 0 RX and TX for IO. */
    GPIO_Config(GPIOA, GPIO_Pin_1, UART0_TXD);    函数见 GPIO 应用说明
    GPIO_Config(GPIOA, GPIO_Pin_0, UART0_RXD);
    /* Configure serial ports 1 RX and TX for IO. */
    GPIO_Config(GPIOC, GPIO_Pin_7, UART1_TXD);
    GPIO_Config(GPIOC, GPIO_Pin_8, UART1_RXD);

    /* USARTx configured as follow:
  - BaudRate = 115200 baud
  - Word Length = 8 Bits
  - Stop Bit = 1 Stop Bit
  - Parity = No Parity
  - Hardware flow control disabled (RTS and CTS signals)
  - Receive and transmit enabled*/

    配置结构体 UART_InitStruct，也可调用 UART_StructInit 使用默认参数
    UART_InitStruct.BaudRate = uartBaud;   //Configure serial port baud rate, the baud rate defaults to 128000.
    UART_InitStruct.DataBits = Databits_8b;
    UART_InitStruct.StopBits = StopBits_1;
    UART_InitStruct.Parity = Parity_None;
    UART_InitStruct.FlowCtrl = FlowCtrl_None;
    UART_InitStruct.Mode = Mode_duplex;

    UART_Init(UART0, &UART_InitStruct);
    UART_Init(UART1, &UART_InitStruct);
}
```

4.2 UART1_IRQHandler

UART 中断服务函数

```
void UART1_IRQHandler(void)
{
    uint8_t rbuf[10], r;
    uint8_t tbuf[] = "UART1 TX INTERRUPT test successful!\n";

    if (UART_GetITIdentity(UART1) == UART_IT_RX)       判断是否接收中断
    {
        if (UART_IsRXFIFONotEmpty(UART1))              判断接收空间是否为空
        {
            r = UART_RecvBuf(UART1, rbuf, 9);
            rbuf[r] = '\0';
        }
        MyPrintf("UART1 RX INTERRUPT test successful, this buf is %s!\n", rbuf);
        UART_ClearIT(UART1);                           清中断
    }
    else if (UART_GetITIdentity(UART1) == UART_IT_TX)
    {
        UART_SendBuf(UART1, tbuf, sizeof(tbuf)-1);
        UART_ClearIT(UART1);
    }
}
```