ATxmega 演示代码之 USART

——基于 ATxmega64A3-EK 和 JTAGICE mkII-CN

文档编号	MAN0007A_CH				
文档版本	Rev. A				
文档摘要	详细讲述了 XMEGA64A3 的串口使用方法				
关键词	AVR、XMEGA、ATxmega64A3-EK 开发板、USB JTAGICE mkII-CN、USART、串口通信				
创建日期	2010-05-17	创建人员	Kiler	审核人员	Hotislandn
文档类型	公开发布/仿真器配套文件				
版权信息	Mcuzone 原创文档，转载请注明出处				

更新历史

版本	时间	更新	作者
Rev. A	2010-05-17	初始创建	Kiler

微控电子 乐微电子
杭州市登云路 639 号 2B143
销售 TEL：+86-571-89908193 13957118045
支持 TEL：13957118045 18913989166
FAX：+86-571-88908193
www.mcuzone.com www.atarm.com

1.概述

ATxmega 是 ATMEL 推出的一款全新 MCU，与之前的 AVR 相比速度更快，ADC 和 DAC 性能更佳，功耗更低，而且有多达 8 个串口，因此在工控领域有较大用途。 本系列文档以 ATxmega64A3-EK 开发板和本站的 USB AVR JTAGICE mkII-CN 仿真器为平台演示 ATxmega 的一些片上外设的操作。

本文演示代码为 64A3 的 USART 操作。
XMEGA64A3 共有 7 个串口，全双工操作，同步和异步模式。

2. 寄存器描述

DATA - USART I/O Data Register

Bit	7	6	5	4	3	2	1	0
+0x00	RXB[[7:0]							
	TXB[[7:0]							
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

数据寄存器，用来缓存发送或者接收的数据。

STATUS - USART Status Register

Bit	7	6	5	4	3	2	1	0	
+0x01	RXCIF	TXCIF	DREIF	FERR	BUFOVF	PERR	-	RXB8	STATUS
Read/Write	R	R/W	R	R	R	R	R	R/W	
Initial Value	0	0	1	0	0	0	0	0	

状态寄存器，第 7 位是接收中断标志位，用于判断是否接收到数据；第 6 位是发送中断标志位；第 5 位是数据寄存器空标志位，发送数据前必须先确定数据寄存器是空的。详情请参阅数据手册第 21 章。

CTRLA – USART Control Register A

Bit	7	6	5	4	3	2	1	0	
+0x03	-	-	RXCINTLVL[1:0]		TXCINTLVL[1:0]		DREINTLVL[1:0]		CTRLA
Read/Write	R	R	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

控制寄存器 A，第 5 位和第 4 位可以设置接收中断级，并且使能接收中断；第 3 位和第 2 位则是设置发送中断级，并且使能发送中断的；第 1 位和第 0 位是设置数据寄存器空中断级的。

CTRLB - USART Control Register B

Bit	7	6	5	4	3	2	1	0	
+0x04	-	-	-	RXEN	TXEN	CLK2X	MPCM	TXB8	CTRLB
Read/Write	R	R	R	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

控制寄存器 B，第 4 位使能发送，第 3 位使能接收，详情请参阅数据手册第 21 章。

BAUDCTRLA - USART Baud Rate Register

Bit	7	6	5	4	3	2	1	0	
+0x06	BSEL[7:0]								BAUDCTRLA
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

BAUDCTRLB - USART Baud Rate Register

Bit	7	6	5	4	3	2	1	0	
+0x07	BSCALE[3:0]				BSEL[11:8]				BAUDCTRLB
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

波特率寄存器，用于设置波特率。

通过配置以上寄存器可以实现串口通信的基本功能。
还有一个寄存器未用到，即 CTRLC，详情请参阅数据手册第 21 章。

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Page
+0x00	DATA	DATA[7:0]								249
+0x01	STATUS	RXCIF	TXCIF	DREIF	FERR	BUFOVF	PERR	-	RXB8	249
+0x02	Reserved	-	-	-	-	-	-	-	-	
+0x03	CTRLA	-	-	RXCINTLVL[1:0]		TXCINTLVL[1:0]		DREINTLVL[1:0]		251
+0x04	CTRLB	-	-	-	RXEN	TXEN	CLK2X	MPCM	TXB8	251
+0x05	CTRLC	CMODE[1:0]		PMODE[1:0]		SBMODE	CHSIZE[2:0]			253
+0x06	BAUDCTRLA	BSEL[7:0]								255
+0x07	BAUDCTRLB	BSCALE[3:0]				BSEL[11:8]				254

3. 演示代码

示例代码 1：
此代码实现将接收到的数据发送回去的功能，波特率为 9600。

```
int main(void)
{
      unsigned char RXData;
      PORTC_DIR |= 0x08;
      Usart_Init();
      Usart_PutString("Mcuzone--");
    while(1)
    {
       while(!(USARTC0_STATUS & 0x80));   //判断接收是否完成
       RXData = USARTC0_DATA;
       Usart_PutChar(RXData);
      }
}

void Usart_Init(void)
{
      USARTC0_CTRLA = 0x14;                  //设置中断级，并使能中断
      USARTC0_CTRLB |= 0x18;                 //使能
      USARTC0_BAUDCTRLA = 0x0c;              //设置波特率为 9600
}

void Usart_PutChar(unsigned char cTXData)
{
      while(!(USARTC0_STATUS & 0x20));   //只有数据寄存器为空时才能发送数据
      USARTC0_DATA = cTXData;
}

void Usart_PutString(unsigned char *pcString)
{
      while (*pcString)
      {
            Usart_PutChar(*pcString++);
      }
      Usart_PutChar(0x0D);
      Usart_PutChar(0x0A);   //结尾发送回车换行
}
```

示例代码 2：

此代码实现 485 与 485 之间的通信。

```
void Usart_Init(void)
{
      USARTE0_CTRLA = 0x14;                 //设置中断级，并使能中断
      USARTE0_CTRLB |= 0x18;                //使能
      USARTE0_BAUDCTRLA = 0x0c;             //设置波特率
      USARTF0_CTRLA = 0x14;                 //设置中断级，并使能中断
      USARTF0_CTRLB |= 0x18;                //使能
      USARTF0_BAUDCTRLA = 0x0c;             //设置波特率

}

int main(void)
{
      unsigned char i,j;
      PORTE_DIR = 0x0b;
      PORTF_DIR = 0xc8;
      Usart_Init();
    while(1)
    {
       PORTF_OUT &= 0x7f;           //U5 收
       PORTF_OUT |= 0x40;           //U4 发

       while(!(USARTE0_STATUS & 0x20));   //U4 发
       USARTE0_DATA = 0x01;

       while(!(USARTF0_STATUS & 0x80));    //U5 收
       i = USARTF0_DATA;
       PORTE.OUT |= i;
//-------------------------------------
       PORTF_OUT |= 0x80;           //U5 发
       PORTF_OUT &= 0xbf;           //U4 收

      while(!(USARTF0_STATUS & 0x20));   //U5 发
       USARTF0_DATA = 0x02;

       while(!(USARTE0_STATUS & 0x80));    //U4 收;
       j = USARTE0_DATA;
       PORTE.OUT |= j;
     }
}
```

示例代码 3：

此代码实现 485 与电脑之间的通信（PC 端使用的是本站的 FT232 USB 转串口模块，加焊了 SP3485 芯片）。

```
void Usart_Init(void)
{
      USARTE0_CTRLA = 0x14;                  //设置中断级，并使能中断
      USARTE0_CTRLB |= 0x18;                 //使能
      USARTE0_BAUDCTRLA = 0x0c;               //设置波特率
}

void delay(void)
{
      unsigned char i,j;
      for(i = 0;i < 10;i ++)
        for(j = 0;j < 70;j ++);
}

int main(void)
{
      unsigned char j;
      PORTE_DIR = 0x0b;
      PORTF_DIR = 0xc8;
      Usart_Init();
      delay();
     while(1)
     {
        PORTF_OUT &= 0xbf;            //U4 收

        while(!(USARTE0_STATUS & 0x80));     //U4 收;
        j = USARTE0_DATA;

        delay();                                  //切换状态需要等待时间
        PORTF_OUT |= 0x40;            //U4 发
        while(!(USARTE0_STATUS & 0x20));    //U4 发
        USARTE0_DATA = j;
        delay();
      }
}
```