Problems with Interrupt putchar & getchar

3 posts / 0 new

Author Message

stefanieh

Level: New Member
Joined: Thu. Mar 17, 2011
Posts: 11 View posts

Posted by stefanieh: Thu. Mar 17, 2011 - 05:43 AM #1

Fivestar widget

Total votes: 0

Hi,
I use with my Atmega1280 USART0 & USART1. I also use the Codevision Compiler.

In former time I used the putchar() & getchar()-Functions for USART0&1 without interrupts and everything worked fine.

Now I putted the interrupt USART Code from the AVR-Codewizard in my components.c-File:

```
// Bit definitions from the USART control registers
#define RXB8 1
#define TXB8 0
#define UPE 2
#define OVR 3
#define FE 4
#define UDRE 5
#define RXC 7

#define FRAMING_ERROR (1<<FE)
#define PARITY_ERROR (1<<UPE)
#define DATA_OVERRUN (1<<OVR)
#define DATA_REGISTER_EMPTY (1<<UDRE)
#define RX_COMPLETE (1<<RXC)

// USART0 Receiver buffer
#define RX_BUFFER_SIZE0 8
char rx_buffer0[RX_BUFFER_SIZE0];

#if RX_BUFFER_SIZE0 <= 256
```

```
unsigned char rx_wr_index0,rx_rd_index0,rx_counter0;
#else
unsigned int rx_wr_index0,rx_rd_index0,rx_counter0;
#endif

// This flag is set on USART0 Receiver buffer overflow
bit rx_buffer_overflow0;

// USART0 Receiver interrupt service routine
interrupt [USART0_RXC] void usart0_rx_isr(void)
{
char status,data;
status=UCSR0A;
data=UDR0;
if ((status & (FRAMING_ERROR | PARITY_ERROR | DATA_OVERRUN))==0)
   {
   rx_buffer0[rx_wr_index0++]=data;
#if RX_BUFFER_SIZE0 == 256
   // special case for receiver buffer size=256
   if (++rx_counter0 == 0)
      {
#else
   if (rx_wr_index0 == RX_BUFFER_SIZE0) rx_wr_index0=0;
   if (++rx_counter0 == RX_BUFFER_SIZE0)
      {
      rx_counter0=0;
#endif
      rx_buffer_overflow0=1;
      };
   };
}

#ifndef _DEBUG_TERMINAL_IO_
// Get a character from the USART0 Receiver buffer
#define _ALTERNATE_GETCHAR_
#pragma used+
char getchar(void)
{
char data;
while (rx_counter0==0);
data=rx_buffer0[rx_rd_index0++];
#if RX_BUFFER_SIZE0 != 256
if (rx_rd_index0 == RX_BUFFER_SIZE0) rx_rd_index0=0;
#endif
#asm("cli")
--rx_counter0;
#asm("sei")
return data;
}
#pragma used-
#endif

// USART0 Transmitter buffer
#define TX_BUFFER_SIZE0 8
char tx_buffer0[TX_BUFFER_SIZE0];

#if TX_BUFFER_SIZE0 <= 256
unsigned char tx_wr_index0,tx_rd_index0,tx_counter0;
#else
```

```
unsigned int tx_wr_index0,tx_rd_index0,tx_counter0;
#endif

// USART0 Transmitter interrupt service routine
interrupt [USART0_TXC] void usart0_tx_isr(void)
{
if (tx_counter0)
   {
   --tx_counter0;
   UDR0=tx_buffer0[tx_rd_index0++];
#if TX_BUFFER_SIZE0 != 256
   if (tx_rd_index0 == TX_BUFFER_SIZE0) tx_rd_index0=0;
#endif
   };
}

#ifndef _DEBUG_TERMINAL_IO_
// Write a character to the USART0 Transmitter buffer
#define _ALTERNATE_PUTCHAR_
#pragma used+
void putchar(char c)
{
while (tx_counter0 == TX_BUFFER_SIZE0);
#asm("cli")
if (tx_counter0 || ((UCSR0A & DATA_REGISTER_EMPTY)==0))
   {
   tx_buffer0[tx_wr_index0++]=c;
#if TX_BUFFER_SIZE0 != 256
   if (tx_wr_index0 == TX_BUFFER_SIZE0) tx_wr_index0=0;
#endif
   ++tx_counter0;
   }
else
   UDR0=c;
#asm("sei")
}
#pragma used-
#endif

// USART1 Receiver buffer
#define RX_BUFFER_SIZE1  16
char rx_buffer1[RX_BUFFER_SIZE1];

#if RX_BUFFER_SIZE1 <= 256
unsigned char rx_wr_index1,rx_rd_index1,rx_counter1;
#else
unsigned int rx_wr_index1,rx_rd_index1,rx_counter1;
#endif

// This flag is set on USART1 Receiver buffer overflow
bit rx_buffer_overflow1;

// USART1 Receiver interrupt service routine
interrupt [USART1_RXC] void usart1_rx_isr(void)
{
char status,data;
status=UCSR1A;
data=UDR1;
if ((status & (FRAMING_ERROR | PARITY_ERROR | DATA_OVERRUN))==0)
```

```
   {
   rx_buffer1[rx_wr_index1++]=data;
#if RX_BUFFER_SIZE1 == 256
   // special case for receiver buffer size=256
   if (++rx_counter1 == 0)
      {
#else
   if (rx_wr_index1 == RX_BUFFER_SIZE1) rx_wr_index1=0;
   if (++rx_counter1 == RX_BUFFER_SIZE1)
      {
      rx_counter1=0;
#endif
      rx_buffer_overflow1=1;
      };
   };
}

// Get a character from the USART1 Receiver buffer
#pragma used+
char getchar1(void)
{
char data;
while (rx_counter1==0);
data=rx_buffer1[rx_rd_index1++];
#if RX_BUFFER_SIZE1 != 256
if (rx_rd_index1 == RX_BUFFER_SIZE1) rx_rd_index1=0;
#endif
#asm("cli")
--rx_counter1;
#asm("sei")
return data;
}
#pragma used-
// USART1 Transmitter buffer
#define TX_BUFFER_SIZE1 16
char tx_buffer1[TX_BUFFER_SIZE1];

#if TX_BUFFER_SIZE1 <= 256
unsigned char tx_wr_index1,tx_rd_index1,tx_counter1;
#else
unsigned int tx_wr_index1,tx_rd_index1,tx_counter1;
#endif

// USART1 Transmitter interrupt service routine
interrupt [USART1_TXC] void usart1_tx_isr(void)
{
if (tx_counter1)
   {
   --tx_counter1;
   UDR1=tx_buffer1[tx_rd_index1++];
#if TX_BUFFER_SIZE1 != 256
   if (tx_rd_index1 == TX_BUFFER_SIZE1) tx_rd_index1=0;
#endif
   };
}

// Write a character to the USART1 Transmitter buffer
#pragma used+
void putchar1(char c)
```

```
{
while (tx_counter1 == TX_BUFFER_SIZE1);
#asm("cli")
if (tx_counter1 || ((UCSR1A & DATA_REGISTER_EMPTY)==0))
   {
   tx_buffer1[tx_wr_index1++]=c;
#if TX_BUFFER_SIZE1 != 256
   if (tx_wr_index1 == TX_BUFFER_SIZE1) tx_wr_index1=0;
#endif
   ++tx_counter1;
   }
else
   UDR1=c;
#asm("sei")
}
#pragma used-

// Standard Input/Output functions
#include
```

And the following lines in my components.h-File:

```
#ifndef _DEBUG_TERMINAL_IO_
// Get a character from the USART0 Receiver buffer
#define _ALTERNATE_GETCHAR_
#pragma used+
char getchar(void);
#pragma used-
#endif

#ifndef _DEBUG_TERMINAL_IO_
// Write a character to the USART0 Transmitter buffer
#define _ALTERNATE_PUTCHAR_
#pragma used+
void putchar(char c);
#pragma used-
#endif

// Get a character from the USART1 Receiver buffer
#pragma used+
char getchar1(void);
#pragma used-
// USART1 Transmitter buffer

// Write a character to the USART1 Transmitter buffer
#pragma used+
void putchar1(char c);
#pragma used-
```

I can use the putchar1() & putchar() Function on my component.c - File and also in my main-File 8 chars before I am Calling antoher function from my project everything stops.
I didn't get the 8 char's before that function and nothing goes on.

Can anybody tell my, if my implemention of the USART0 & 1 Interruptsfunction was right and how I can find my error?!

I am searching for a long time now and can't find the reason for.

Do I have to change anything else in my project, when I am using the interrupt-functions?
I have different files where I am using putchar(), putchar1(), getchar1() and getchar1() mixed.

Thank you for every tipp!

Tags:
AVR Microcontrollers, **megaAVR and tinyAVR**, **ATmega128**, **ATmega1280**

Top

MartinM57

Level: Resident
Joined: Thu. Dec 15, 2005
Posts: 842 **View posts**
Location: Bristol, UK

Posted by **MartinM57**: Thu. Mar 17, 2011 - 09:18 AM

#2

Total votes: 0

Which version of Codevision?

...and I'm not sure that you should be posting Infotech's wizard code that's only available in the commercial version. But I'm not sure about that..

Top

david.prentice

Level: 10k+ Postman
Joined: Sat. Feb 12, 2005
Posts: 27316 **View posts**
Location: Wormshill, England

Posted by **david.prentice**: Thu. Mar 17, 2011 - 09:30 AM

#3

Total votes: 0

No, you do not need to do anything special.

If you ask the CodeWizard to generate IRQ driven stdio, you simply use getchar(), putchar(), printf() in the regular way.

I dislike using getchar() without a 'kbhit()' test. i.e. I always look first to see if there is a character available. The kbhit() is different for IRQ or polled stdin.

However, if you ever disable IRQs in some other part of your program, you will lose the stdio IRQs too. So you should never disable for two whole character times. e.g. < 2.08 ms @ 9600 baud 8-N-1.

Personally, I would put my main logic in a separate file. Then let the CodeWizard generate the initialisation.

You can then experiment with polled or IRQ stdio, or with different buffer sizes.

I suspect that you are using CLI() somewhere else in your program.

David.

Top

Log In or **Register** to post comments

Jump To

--megaAVR and tinyAVR ▼