```c
/*
 * uart1.c
 *
 * Created: 30.1.2017 13:56:41
 *  Author: atom2
 */

#include <avr/io.h>
#include <stdio.h>
#include <string.h>
#include "common_defs.h"
#include "uart1.h"
#include "def_init.h"
#include <avr/interrupt.h>
#include "trinamic.h"


#define MACRO_BAUDRATE(BAUDRATE) (UART1_UBRRL = (((F_CPU) / (BAUDRATE * 16UL)) - 1 + (F_CPU % (BAUDRATE * 16UL) > (BAUDRATE * 8UL) ? 1 : 0)))


struct tmp
{
    uint8_t tmpData;
    uint8_t tmpStatus;
    uint8_t tmpTimer;
};

struct tmp tmpUart;


volatile uint8_t uart1_buf_rx[BUFFER_CHAR_PACKET];
volatile uint8_t uart1_buf_tx[BUFFER_CHAR_PACKET];

volatile uint8_t uart1_sum=0;
volatile uint8_t uart1_i=0;
volatile uint8_t uart1_ret=0, uart1_check_sum=0;

volatile uint8_t uart1_tx_flag=FALSE;
volatile uint8_t uart1_tx_iptr=0;
volatile uint8_t uart1_tx_ptr=0;

volatile uint8_t uart1_rx_flag=FALSE;
volatile uint8_t uart1_rx_iptr=0;
volatile uint8_t uart1_rx_ptr=0;


Trinamicpac TR_Buf_In;


uint8_t uart1_init(uint32_t MYUBRR1)
{
    uint16_t UBRR1_COUNT = 0;
    // Výpočet rychlosti
    UBRR1_COUNT = (((F_CPU) / (MYUBRR1 * 16UL)) - 1 + (F_CPU % (MYUBRR1 * 16UL) > (MYUBRR1 * 8UL) ? 1 : 0));
```

```
55      //UBRR1_COUNT = MYUBRR
56      // Nastavení UBRR0 pro rychlost
57      UART1_UBRRL = (unsigned char) (UBRR1_COUNT);
58      UART1_UBRRH = (unsigned char) (UBRR1_COUNT >> 8);
59      // Povolení RX a TX pinů
60      UART1_UCSRB |= BV(UART1_TXEN) | BV(UART1_RXEN);
61      // 8bit, 1stop, no parity
62      UART1_UCSRC |= BV(UART1_UCSZ10) | BV(UART1_UCSZ11);
63      return 0;
64  }
65
66  uint8_t uart1_interrupt_rx(uint8_t enable)
67  {
68      if (enable)
69          UART1_UCSRB |= BV(UART1_RXIE);
70      else
71          UART1_UCSRB &= BV(UART1_RXIE);
72
73      return 0;
74  }
75
76  uint8_t uart1_interrupt_tx(uint8_t enable)
77  {
78      if (enable)
79          UART1_UCSRB |= BV(UART1_TXIE);
80      else
81          UART1_UCSRB &= BV(UART1_TXIE);
82
83      return 0;
84  }
85
86  uint8_t uart1_ptr_ask()
87  {
88      return uart1_rx_ptr;
89  }
90
91
92  void uart1_receive_char(uint8_t data)
93  {
94      uart1_buf_rx[uart1_rx_ptr++] = data;
95      uart1_rx_iptr++;
96  }
97
98
99  ISR(UART1_RX_vect)
100 {
101     tmpUart.tmpData = UART1_UDR;
102     tmpUart.tmpStatus = UART1_UCSRA;
103     tmpUart.tmpTimer =  DEFAULT_TIMEOUT;
104     uart1_receive_char(tmpUart.tmpData);
105 }
106
107 ISR(UART1_TX_vect)
108 {
109     if (uart1_tx_flag)
110     {
```

```
111         //Odeslání 9 bytů dat
112         if (uart1_tx_iptr > 8)
113         {
114             // Vypnutí odesílání a povolení příjmu
115             uart1_tx_flag = FALSE;
116             RS485_EN_EXT_receive;
117             uart1_tx_iptr=0;
118         }
119         else
120         {
121             UART1_UDR = TR_Buf_In.b[uart1_tx_iptr];
122             uart1_tx_iptr++;
123         }
124     }
125 }
126 
127 
128 
129 uint8_t check_uart1(uint8_t data)
130 {
131     // Vnitřní čítač 9 příchozích Bytů
132     if (uart1_rx_iptr > 8)
133     {
134         uart1_rx_flag = TRUE;
135     }
136     // Vypnutí přerušení před kontrolou dat
137     cli();
138     if (uart1_rx_flag)
139     {
140         uart1_sum=0;
141         uart1_i=0;
142         for (uart1_i=9; uart1_i>1; uart1_i--)
143         {
144             uart1_sum += uart1_buf_rx[uart1_rx_ptr-uart1_i];
145         }
146         uart1_check_sum = uart1_buf_rx[uart1_rx_ptr-1];
147         if (uart1_sum == uart1_check_sum)
148         {
149             uart1_ret = 1;
150             uart1_rx_iptr=0;
151         }
152         else
153         {
154             uart1_rx_iptr=0;
155             uart1_rx_ptr=0;
156             uart1_ret=2;
157         }
158         // Vynulování crc
159         uart1_check_sum=0;
160     }
161     else
162     {
163         uart1_ret = 0;
164     }
165     sei();
166     uart1_rx_flag=FALSE;
```

```
167     return uart1_ret;
168
169 }
170
171
172 void uart1_transmit_char(uint8_t data)
173 {
174     while ( !( UART1_UCSRA & (1 << UART1_UDRE)) );
175     UART1_UDR = data;
176 }
```