```
/************************************************
* "uart.c":                                     *
* Implementation file for the Mega32 UART driver. *
*  STK500 setup:                                *
*  Header "RS232 spare" connected to RXD/TXD:   *
*  RXD = PORTD, bit0                            *
*  TXD = PORTD, bit1                            *
*                                               *
* David Buhauer, 22/05 2013                     *
************************************************/
#include <avr/io.h>
#include <stdlib.h>
#include "UART.h"

// Constants
#define XTAL 3686400UL

/**************************************************************************
USART initilization.
    Asyncronous mode.
    RX and TX enabled.
    No interrupts enabled.
    Number of Stop Bits = 1.
    No Parity.
    Baud rate = Parameter.
    Data bits = Parameter.
Parameters:
    BaudRate: Wanted Baud Rate.
    Databits: Wanted number of Data Bits.
**************************************************************************/
void InitUART(unsigned long BaudRate, unsigned char DataBit)
{
unsigned int TempUBRR;

  if ((BaudRate >= 110) && (BaudRate <= 115200) && (DataBit >=5) && (DataBit <= 8))
  {
    // "Normal" clock, no multiprocesser mode (= default)
    UCSRA = 0b00100000;
    // No interrupts enabled
    // Receiver enabled
    // Transmitter enabled
    // No 9 bit operation
    UCSRB = 0b00011000;
    // Asynchronous operation, 1 stop bit, no parity
    // Bit7 always has to be 1
    // Bit 2 and bit 1 controlles the number of databits
    UCSRC = 0b10000000 | (DataBit-5)<<1;
    // Set Baud Rate according to the parameter BaudRate:
    // Select Baud Rate (first store "UBRRH--UBRRL" in local 16-bit variable,
    //                   then write the two 8-bit registers seperately):
    TempUBRR = XTAL/(16*BaudRate) - 1;
    // Write upper part of UBRR
    UBRRH = TempUBRR >> 8;
    // Write lower part of UBRR
    UBRRL = TempUBRR;
  }
}


/**************************************************************************
  Returns 0 (FALSE), if the UART has NOT received a new character.
  Returns value <> 0 (TRUE), if the UART HAS received a new character.
**************************************************************************/
unsigned char CharReady()
{
   return UCSRA & (1<<7);
}

/**************************************************************************
Awaits new character received.
Then this character is returned.
**************************************************************************/
char ReadChar()
```

```
{
  // Wait for new character received
  while ( (UCSRA & (1<<7)) == 0 )
  {}
  // Then return it
  return UDR;
}

/*****************************************************************************
Awaits transmitter-register ready.
Then it send the character.
Parameter :
    Tegn : Character for sending.
*****************************************************************************/
void SendChar(char Tegn)
{
  // Wait for transmitter register empty (ready for new character)
  while ( (UCSRA & (1<<5)) == 0 )
  {}
  // Then send the character
  UDR = Tegn;
}

/*****************************************************************************
Sends 0-terminated string.
Parameter:
   Streng: Pointer to the string.
*****************************************************************************/
void SendString(char* Streng)
{
  // Repeat untill zero-termination
  while (*Streng != 0)
  {
    // Send the character pointed to by "Streng"
    SendChar(*Streng);
    // Advance the pointer one step
    Streng++;
  }
}
```