UART

Discussion III (Version 2.0)

UMBC - CE

September 16, 2014

Version 1.0 - Initial Document
Version 2.0 - Fixed typos and some illogical assumptions !

Objectives

- Introduce UART (Universal Asynchronous Receiver Transmitter)

Objectives

- Introduce UART (Universal Asynchronous Receiver Transmitter)
- Interface PC with AVR Butterfly via UART

Objectives

- Introduce UART (Universal Asynchronous Receiver Transmitter)
- Interface PC with AVR Butterfly via UART
- Implement UART communications using AVR Assembly

UART

- The simplest form of communication with a computer **COM** or serial port uses only 3 wires, **GND**, **RX** and **TX** (named with respect to the master)

UART

- The simplest form of communication with a computer **COM** or serial port uses only 3 wires, **GND**, **RX** and **TX** (named with respect to the master)
- Slave Device **(TX)** $\longrightarrow$ Master Device **(RX)**

UART

- The simplest form of communication with a computer **COM** or serial port uses only 3 wires, **GND**, **RX** and **TX** (named with respect to the master)
- Slave Device **(TX)** $\longrightarrow$ Master Device **(RX)**
- Master Device **(TX)** $\longrightarrow$ Slave Device **(RX)**

UART

- The simplest form of communication with a computer **COM** or serial port uses only 3 wires, **GND**, **RX** and **TX** (named with respect to the master)
- Slave Device **(TX)** $\longrightarrow$ Master Device **(RX)**
- Master Device **(TX)** $\longrightarrow$ Slave Device **(RX)**
- The data is transmitted without a clock and is instead transmitted at a rate predetermined or pre-negotiated rate known on both sides

UART

- The simplest form of communication with a computer **COM** or serial port uses only 3 wires, **GND**, **RX** and **TX** (named with respect to the master)
- Slave Device **(TX)** $\longrightarrow$ Master Device **(RX)**
- Master Device **(TX)** $\longrightarrow$ Slave Device **(RX)**
- The data is transmitted without a clock and is instead transmitted at a rate predetermined or pre-negotiated rate known on both sides
- The baud rate defines the length of each bit as $\frac{1}{baudrate}$

The UART Frame

- The data to be sent is **framed** by a start bit and a stop bit

The UART Frame

- The data to be sent is **framed** by a start bit and a stop bit

- The start bit is typically a high signal to start the data frame, and the stop bit is typically a low signal, often 1, 1.5 or 2 times as long as the other bits

The UART Frame

- The data to be sent is **framed** by a start bit and a stop bit

- The start bit is typically a high signal to start the data frame, and the stop bit is typically a low signal, often 1, 1.5 or 2 times as long as the other bits
- Optionally, a parity bit may be transmitted after the data

Transmitting data

- Typically, a FIFO buffer of 1 or more bytes is used for transmission and reception

Transmitting data

- Typically, a FIFO buffer of 1 or more bytes is used for transmission and reception
- The status of the buffers is probed internally by **empty** and possibly **full** status bits

Transmitting data

- Typically, a FIFO buffer of 1 or more bytes is used for transmission and reception
- The status of the buffers is probed internally by **empty** and possibly **full** status bits
- If the read buffer is only 1 byte, **NOT EMPTY** is the same as **DATA READY**

Transmitting data

- Typically, a FIFO buffer of 1 or more bytes is used for transmission and reception
- The status of the buffers is probed internally by **empty** and possibly **full** status bits
- If the read buffer is only 1 byte, **NOT EMPTY** is the same as **DATA READY**
- If the write buffer is only 1 byte, **NOT EMPTY** is the same as **NOT READY/FULL**

Handling Overflow

- What happens if the Master sends data to the slave receive buffer faster than it can be unloaded ?

Handling Overflow

- What happens if the Master sends data to the slave receive buffer faster than it can be unloaded ?
- To prevent overflow, additional hardware lines can be used to communicate status and perform handshaking

Handling Overflow

- What happens if the Master sends data to the slave receive buffer faster than it can be unloaded ?
- To prevent overflow, additional hardware lines can be used to communicate status and perform handshaking
- A software handshaking can also be implemented

Handling Overflow

- What happens if the Master sends data to the slave receive buffer faster than it can be unloaded ?
- To prevent overflow, additional hardware lines can be used to communicate status and perform handshaking
- A software handshaking can also be implemented
- The stop/start bits, parity, hardware/software hand shaking, and baud rate must be configured on both ends

AVR Butterfly UART Registers

- The following set of registers are used to communicate over UART

AVR Butterfly UART Registers

- The following set of registers are used to communicate over UART
  - UCSRA

AVR Butterfly UART Registers

- The following set of registers are used to communicate over UART
  - UCSRA
  - UCSRB

AVR Butterfly UART Registers

- The following set of registers are used to communicate over UART
  - UCSRA
  - UCSRB
  - UCSRC

AVR Butterfly UART Registers

- The following set of registers are used to communicate over UART
  - UCSRA
  - UCSRB
  - UCSRC
  - UBRRH & UBRRL

AVR Butterfly UART Registers

- The following set of registers are used to communicate over UART
  - UCSRA
    - Flags for various errors that might occur during data transmission, e.g. parity error, frame error etc.
  - UCSRB
  - UCSRC
  - UBRRH & UBRRL

AVR Butterfly UART Registers

- The following set of registers are used to communicate over UART
  - UCSRA
    - Flags for various errors that might occur during data transmission, e.g. parity error, frame error etc.
  - UCSRB
    - Contains lot of enable bits. eg. different interrupt enable bits and the receiving and transmitting enable bits
  - UCSRC
  - UBRRH & UBRRL

AVR Butterfly UART Registers

- The following set of registers are used to communicate over UART
  - UCSRA
    - Flags for various errors that might occur during data transmission, e.g. parity error, frame error etc.
  - UCSRB
    - Contains lot of enable bits. eg. different interrupt enable bits and the receiving and transmitting enable bits
  - UCSRC
    - Set the parity mode, stop bits etc.
  - UBRRH & UBRRL

AVR Butterfly UART Registers

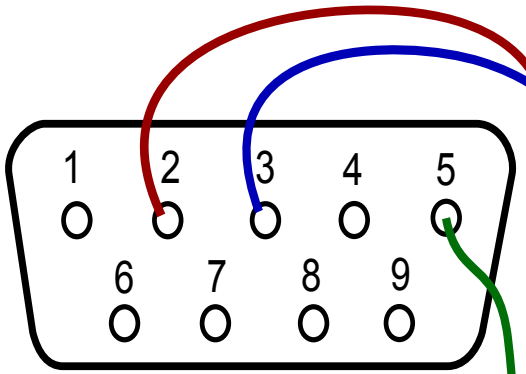
- The following set of registers are used to communicate over UART
  - UCSRA
    - Flags for various errors that might occur during data transmission, e.g. parity error, frame error etc.
  - UCSRB
    - Contains lot of enable bits. eg. different interrupt enable bits and the receiving and transmitting enable bits
  - UCSRC
    - Set the parity mode, stop bits etc.
  - UBRRH & UBRRL
    - The higher byte (UBRRH) and lower byte (UBRRL) is stored for generating the required baud rate

AVR Butterfly UART Registers

- The following set of registers are used to communicate over UART
  - UCSRA
    - Flags for various errors that might occur during data transmission, e.g. parity error, frame error etc.
  - UCSRB
    - Contains lot of enable bits. eg. different interrupt enable bits and the receiving and transmitting enable bits
  - UCSRC
    - Set the parity mode, stop bits etc.
  - UBRRH & UBRRL
    - The higher byte (UBRRH) and lower byte (UBRRL) is stored for generating the required baud rate
- For Our AVR, the required **registers** have slightly different names and are in the EXTENDED I/O, meaning, we must access them as memory and not as registers

AVR Butterfly Interfacing

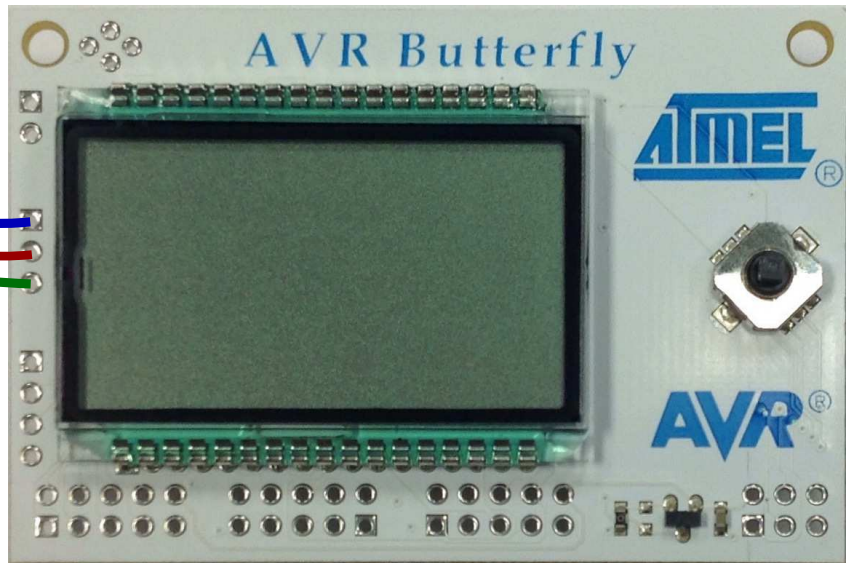
RS232 Communications Port



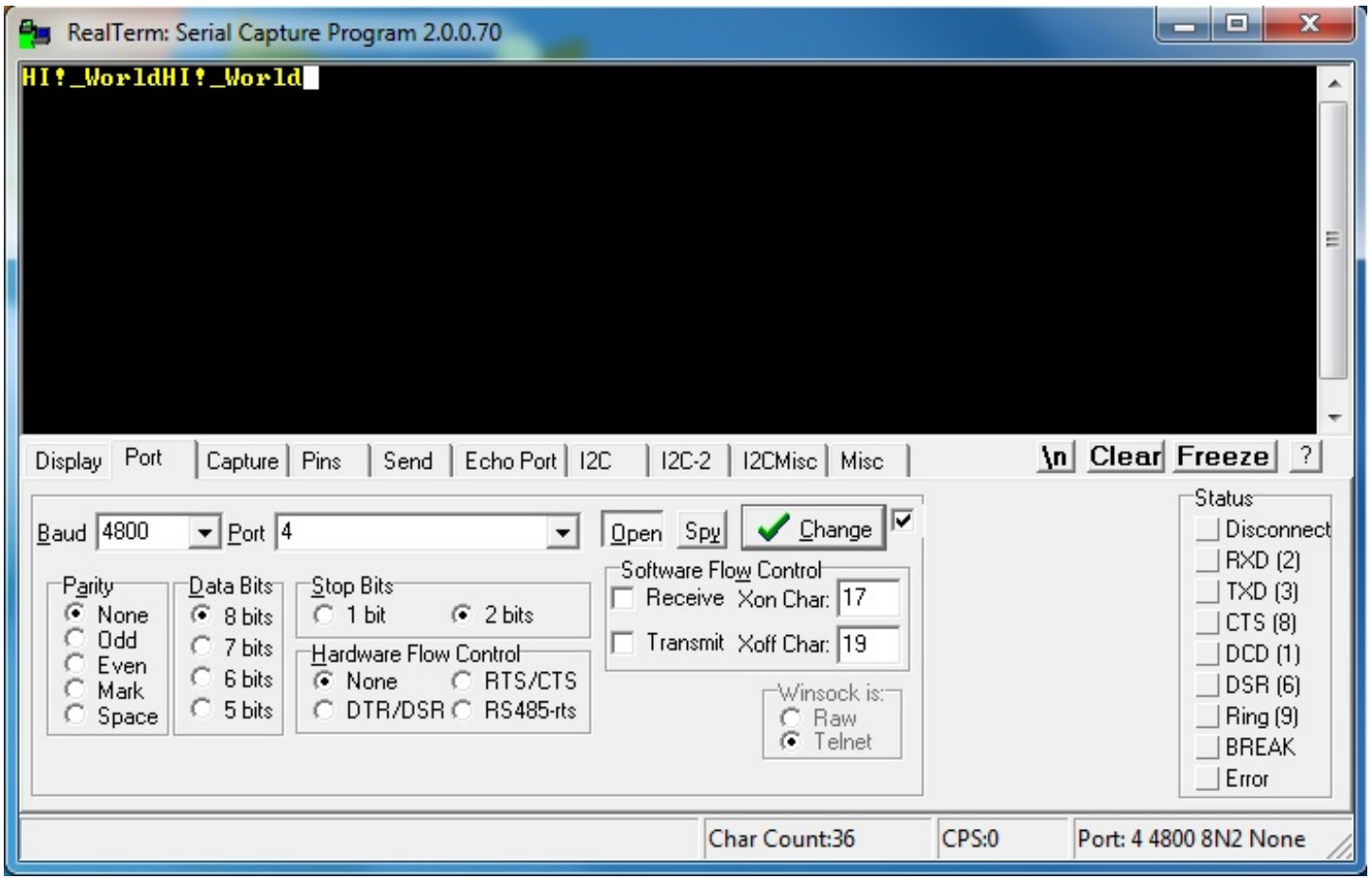
Connector Pin-out

Pin #	RS-232
1	Data Carrier Detect (DCD)
2	RXD
3	TXD
4	DataTerminal Ready (DTR)
5	Ground (GND)
6	Data Set Ready (DSR)
7	Request To Send (RTS)
8	Clear To Send (CTS)
9	Ring Indicator (RI)

AVR Butterfly	COM
Pin 1 (RXD)	Pin 3
Pin 2 (TXD)	Pin 2
Pin 3 (GND)	Pin 5



RealTerm Settings



Interfacing Code

Download interfacing code from your instructor's website (uart.asm)