RS232 Controller

1. General information

The RS232 Controller module is a controller that allows integration with devices equipped with the RS232 interface.

2. Example of usage in scripts

2.1. Sending a command to the device without waiting for a response

Sending the ASCII command "PLAY":

```
CLU->SerialControler->SetRepresentationType(1)  --  setting the data in ASCII
format, can be configured from the built-in feature position.
CLU->SerialControler->AddToTxBuffer("PLAY")  -- adding to the transmit buffer
(Tx).
CLU->SerialControler->SendTxBuffer(0)  -- sending the command without an end-
of-line character (0), after sending, the transmit buffer (Tx) is cleared.
```

Sending the HEX command for three hexadecimal values: 0x12, 0xAB, and 0x34:

```
CLU->SerialControler->SetRepresentationType(0)  --  setting the data in HEX
format, can be configured from the built-in feature position.
CLU->SerialControler->AddToTxBuffer(0x12) -- adding to the transmit buffer
(Tx).
CLU->SerialControler->AddToTxBuffer(0xAB) -- adding to the transmit buffer
(Tx).
CLU->SerialControler->AddToTxBuffer(0x34) -- adding to the transmit buffer
(Tx).
CLU->SerialControler->SendTxBuffer(0)  -- sending the command without an end-
of-line character (0), after sending, the transmit buffer (Tx) is cleared.
```

2.2. Detecting the received command

Script checking if the received response contains the expression "Status: Play":

> The script analyzing should be set under the `OnReceive` event. To trigger the event for EVERY received message on Rx, you should set `ResponseSize = 1`.

```
x = CLU->SerialControler->RxBuffer  -- Storing the buffer contents in variable
x.
if (string.match(x, "Status: Play")) then  -- The string.match function will
return "Status: Play" or nil. If it's nil, the condition will be treated as
false.
    print("Status: Play recognized")
end
CLU->SerialControler->ClearRxBuffer(0)  -- Clearing the entire (0-All) Rx
buffer after analysis.
```

2.3. Detecting the received command with value analysis

Script checking if the received response contains the expression "Track: 25" and extracting the value 25:

> The script analyzing should be set under the `OnReceive` event. To trigger the event for EVERY received message on Rx, you should set `ResponseSize = 1`.

```
x = CLU->SerialControler->RxBuffer  -- Storing the buffer contents in variable
x.
y = string.match(x, "Track: (%d+)")  -- The string.match function will return a
numeric value (%d+) or nil.
if y then  -- If y = nil, the condition will be treated as false.
    print("Track: " .. y)
end
CLU->SerialControler->ClearRxBuffer(0)  -- Clearing the entire (0-All) Rx
buffer after analysis.
```

The variable y contains an expression in the string. To convert the value to a numeric representation, you should use the `tonumber()` function.

Script checking if the received response contains the expression "Temperature: 25.5°C" and extracting the value 25.5:

```
x = CLU->SerialControler->RxBuffer  -- Storing the buffer contents in variable
x.
y = string.match(x, "Temperature: (%d+.%d+)°C")  -- The string.match function
will return a numeric value (%d+).(%d+) or nil.
if y then  -- If y = nil, the condition will be treated as false.
    print("Temperature: " .. y .. "°C")
end
CLU->SerialControler->ClearRxBuffer(0)  -- Clearing the entire (0-All) Rx
buffer after analysis.
```

3. Object description

FEATURES

Name	Description
`RepresentationType`	Data representation type
`BaudRate`	Transmission speed
`WordLenght`	Word length
`StopBits`	Number of stop bits
`Parity`	Bit parity control
`TxBuffer`	Transceiver buffer. Cleared automatically after calling SendTxBuffer
`RxBuffer`	Receive buffer
`ResponseSize`	Expected response size. This is the minimum number of bytes for which the OnReceive event will be triggered. If the message size is smaller than the specified number of bytes, the event will only trigger after accumulating the specified number of bytes in the buffer. If the value is set to 0, the OnReceive event will never be executed
`ResponseTimeout`	Response timeout

METHODS

Name	Description
`SetRepresentationType`	Sets the data representation type
`SetBaudRate`	Sets the transmission speed
`SetWordLenght`	Sets the word length
`SetStopBits`	Sets the number of stop bits
`SetParity`	Sets the parity
`AddToTxBuffer`	Adds data to the transmit buffer
`SetResponseSize`	Sets the length of the expected response
`SetResponseTimeout`	Sets the response timeout
`ClearRxBuffer`	Clears the receive buffer. Operates on a first-in, first-out basis - the first element added to the queue will also be the first one removed
`ClearTxBuffer`	Clears the transmit buffer. Operates on a last-in, first-out basis - the last element added to the queue will be the first one removed
`SendTxBuffer`	Sends the transmit buffer

EVENTS

Name	Description
`OnReceive`	Event occurs when the controller has received data
`OnTransmit`	Event occurs when the controller sends data
`OnTimeout`	Event occurs when the response time has been exceeded
`OnOverflow`	Event occurs when the receive buffer is overflowed
`OnTransmitError`	Event occurs when data is sent incorrectly