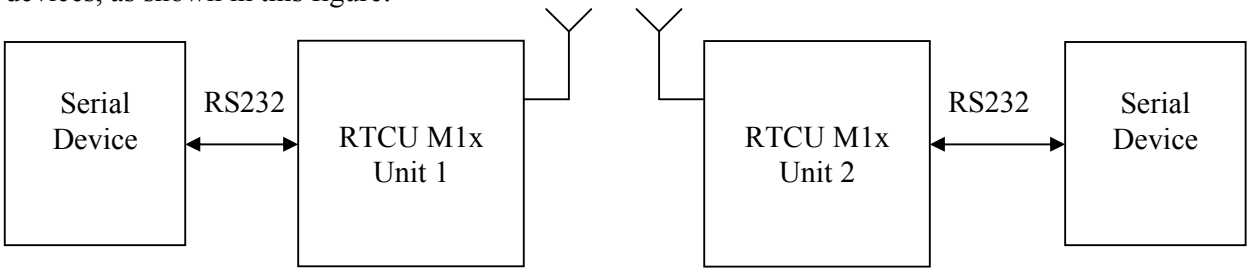
Logic IO
www.logicio.com

Application Note
Serial communication forwarder
Ver. 1.0

This application note describes how to configure two RTCU M1x units to forward serial communication between two serial devices, using the GPRS Gateway.
The hardware is simply a standard RS232 null-modem cable between the RTCU units and the serial devices, as shown in this figure:



Connect the null-modem to serial port 2 on the RTCU units.

The software for the basic functionality described here is bundled with this document.
Once the program is uploaded to the RTCU units all that is needed is to set the NodeId of the receiving RTCU unit. This is done by sending a SMS to the RTCU units.

The simplicity of the software comes with a price however, as there is no control over configuration of destination NodeId, which makes it easy to hijack the data. Also when receiving data there is no control of where it comes from, and so it is possible to insert foreign data into the system.
Another point is that only the NodeId of another RTCU is guaranteed to work correctly.

Logic IO ApS.
Holmboes Allé 14,1
8700 Horsens
Denmark

Ph: (+45) 7625 0210
Fax: (+45) 7625 0211
Email: info@logicio.com
www.logicio.com / www.rtcu.dk

APPENDIX 1: Source code listing

```
//-------------------------------------------------------------------------------
// Ser2Ser.vpl, created 2004-12-07 09:29
//
//-------------------------------------------------------------------------------
INCLUDE rtcu.inc
INCLUDE tcpip.inc

VAR
   rxpdu : ARRAY[0..139] of SINT;
   txpdu : ARRAY[0..139] of SINT;
   buf   : ARRAY[0..139] of SINT;
END_VAR;

FUNCTION strFormatHex : STRING;
VAR_INPUT
   v        : SINT;
END_VAR;
VAR
   temp     : SINT;
   nible    : SINT;
   i        : INT;
END_VAR;

   // Setup
   nible := 16#F0;
   // Iterate
   FOR i := 0 TO 1 DO
      // Move nible to start
      temp := v AND nible;
      temp := shr8( in := temp, n := SINT((1-i)*4) );
      // Add character to string
      CASE SINT(temp) OF
         16#01 : strFormatHex := strConcat( str1 := strFormatHex, str2 := "1" );
         16#02 : strFormatHex := strConcat( str1 := strFormatHex, str2 := "2" );
         16#03 : strFormatHex := strConcat( str1 := strFormatHex, str2 := "3" );
         16#04 : strFormatHex := strConcat( str1 := strFormatHex, str2 := "4" );
         16#05 : strFormatHex := strConcat( str1 := strFormatHex, str2 := "5" );
         16#06 : strFormatHex := strConcat( str1 := strFormatHex, str2 := "6" );
         16#07 : strFormatHex := strConcat( str1 := strFormatHex, str2 := "7" );
         16#08 : strFormatHex := strConcat( str1 := strFormatHex, str2 := "8" );
         16#09 : strFormatHex := strConcat( str1 := strFormatHex, str2 := "9" );
         16#0A : strFormatHex := strConcat( str1 := strFormatHex, str2 := "A" );
         16#0B : strFormatHex := strConcat( str1 := strFormatHex, str2 := "B" );
         16#0C : strFormatHex := strConcat( str1 := strFormatHex, str2 := "C" );
         16#0D : strFormatHex := strConcat( str1 := strFormatHex, str2 := "D" );
         16#0E : strFormatHex := strConcat( str1 := strFormatHex, str2 := "E" );
         16#0F : strFormatHex := strConcat( str1 := strFormatHex, str2 := "F" );
      ELSE
         strFormatHex := strConcat( str1 := strFormatHex, str2 := "0" );
      END_CASE;
      // Move reference
      nible := shr8( in := nible, n := 4 );
   END_FOR;

END_FUNCTION;

FUNCTION PrintIncoming
VAR_INPUT
   len      : INT;
END_VAR;
VAR
   linelen : INT;
   start   : INT;
   i       : INT;
   str     : STRING;
END_VAR;
   // Init
   start := 0;
```


Logic IO ApS.
Holmboes Allé 14,1
8700 Horsens
Denmark

Ph: (+45) 7625 0210
Fax: (+45) 7625 0211
Email: info@logicio.com
www.logicio.com / www.rtcu.dk

```
   // Main teration
   WHILE start < len DO
      // Line start
      str   := strConcat(str1:=strFormatHex(v:=SINT(start)),str2:=" => ");
      IF (len - start) > 16 THEN linelen := 16; ELSE linelen := len - start; END_IF;
      // Iterate line
      FOR i := 0 TO linelen-1 DO
         str := strConcat(str1:=str,str2:=strConcat(str1:=strFormatHex(v:=rxpdu[i+start]),str2:=" "));
      END_FOR;
      // Zero pad
      IF linelen < 16 THEN FOR i := linelen TO 15 DO str := strConcat(str1:=str,str2:="00 "); END_FOR;
END_IF;
      // Insert text
      str   := strConcat(str1:=str,str2:=strFromMemory(src:=ADDR(rxpdu[start]),len:=linelen));
      DebugMsg(message:=str);
      // Next line
      start := start + 16;
   END_WHILE;
END_FUNCTION;

FUNCTION PrintOut
VAR_INPUT
   len     : INT;
END_VAR;
VAR
   linelen : INT;
   start   : INT;
   i       : INT;
   str     : STRING;
END_VAR;
   // Init
   start := 0;

   // Main teration
   WHILE start < len DO
      // Line start
      str   := strConcat(str1:=strFormatHex(v:=SINT(start)),str2:=" => ");
      IF (len - start) > 16 THEN linelen := 16; ELSE linelen := len - start; END_IF;
      // Iterate line
      FOR i := 0 TO linelen-1 DO
         str := strConcat(str1:=str,str2:=strConcat(str1:=strFormatHex(v:=buf[i+start]),str2:=" "));
      END_FOR;
      // Zero pad
      IF linelen < 16 THEN FOR i := linelen TO 15 DO str := strConcat(str1:=str,str2:="00 "); END_FOR;
END_IF;
      // Insert text
      str   := strConcat(str1:=str,str2:=strFromMemory(src:=ADDR(buf[start]),len:=linelen));
      DebugMsg(message:=str);
      // Next line
      start := start + 16;
   END_WHILE;
END_FUNCTION;

PROGRAM Ser2Ser;
VAR
   pdu    : gsmIncomingPDU;
   sms    : gsmIncomingSMS;
   rx     : serFrameReceiver;
   port   : SINT:=1;
   dst    : STRING;
   rc     : INT;
   linsec : DINT;
END_VAR;
   DebugMsg(message:="Program started");

   // Read dst. NodeID
   rc := LoadData(index:=1,data:=ADDR(rxpdu),datasize:=SIZEOF(rxpdu));
   IF rc = 16 AND rxpdu[0] = -34 AND rxpdu[1] = -83 AND rxpdu[2] = -38 AND rxpdu[3] = -19 AND rxpdu[4] = -
86 AND rxpdu[5] = 10 THEN
      // Header: 16#DEADDAED  = -34 -83 -38 -19
```


Logic IO ApS.
Holmboes Allé 14,1
8700 Horsens
Denmark

Ph: (+45) 7625 0210
Fax: (+45) 7625 0211
Email: info@logicio.com
www.logicio.com / www.rtcu.dk

```
      // Type:   16#AA (Text) = -86
      // Length: 10 Bytes
      dst := strFromMemory(src:=ADDR(rxpdu[6]),len:=rxpdu[5]);
      DebugMsg(message:=strConcat(str1:="Destination: ",str2:=dst));
   END_IF;

   // Init GPRS
   gsmPower(power:=ON);
   gprsOpen();

   // Init serial port
   serOpen(port:=port,baud:=9600,bit:=8,parity:=0);
   serSetHandshake(port:=port,RtsCts:=TRUE);
   rx(port:=port,enable:=true,frame:=ADDR(buf),maxSize:=SIZEOF(buf),sof:=0,eof:=0);

   // Init PDU
   pdu.message := ADDR(rxpdu);

   // Serial Timeout
   linsec      := clockNow() + 5;

   DebugMsg(message:="Initialized");

BEGIN
   // Test for incoming SMS
   IF sms.status > 0 THEN
      IF strLen(str:=sms.message) = 9 THEN
         dst := strConcat(str1:="@",str2:=sms.message);
         DebugMsg(message:=strConcat(str1:="New destination: ",str2:=dst));
         // Save NodeId
         txpdu[0] := 16#DE;   // Header
         txpdu[1] := 16#AD;
         txpdu[2] := 16#DA;
         txpdu[3] := 16#ED;
         txpdu[4] := 16#AA;   // String data
         txpdu[5] := 10;      // 10 Bytes long
         strToMemory(dst:=ADDR(txpdu[6]),str:=dst,len:=10);
         SaveData(index:=1,data:=ADDR(txpdu),datasize:=16);
      END_IF;
   END_IF;

   // Test for incoming PDU
   IF pdu.status = 1 OR pdu.status = 2 THEN
      serSendData(port:=port,data:=ADDR(rxpdu),size:=pdu.length);
      DebugFmt(message:="Serial data received, length=\1",v1:=pdu.length);
//      PrintIncoming(len:=pdu.length);
   END_IF;

   // Serial timeout
   IF clockNow() >= linsec THEN
      serForceDataReady(port:=port);
      // Reset timeout
      linsec   := clockNow() + 5;
   END_IF;

   // Test for incomming serial data
   rx();
   IF rx.ready THEN
      // Send data
      rc       := gsmSendPDU(phonenumber:=dst,message:=ADDR(buf),length:=rx.size);
      DebugFmt(message:="Serial data send, rc=\1 length=\2",v1:=rc,v2:=rx.size);
      DebugMsg(message:=strConcat(str1:="Routed to ",str2:=dst));
//      PrintOut(len:=rx.size);
      // Reset timeout
      linsec   := clockNow() + 5;
      // Flag as completed
      serFrameReceiveDone(port:=port);
   END_IF;
END;

END_PROGRAM;
```


Logic IO ApS.
Holmboes Allé 14,1
8700 Horsens
Denmark

Ph: (+45) 7625 0210
Fax: (+45) 7625 0211
Email: info@logicio.com
www.logicio.com / www.rtcu.dk