Serial Protocol for the STn00 and SRn00 devices

Issue 2
Date 5/6/00

1 Introduction

This document describes the serial protocol for re-programming the STn00 and SRn00 range of transmitter and receiver modules.

There is an important difference between the VHF and the UHF products which concerns the Intermediate Frequency (IF) used.

To differentiate between the ST and SR each has a unique product code. For the SRn00 this is 0F6 hex and for STn00 it is 0F5 hex.

2 Data Format

Two baud rates are used for differing functions. Throughout this document, which one is being used is given. The two baud rates are:

Baud rate 19,200, 8 data bits, 1 start bit, 1 stop bit, no parity
Baud rate 1,200, 8 data bits, 1 start bit, 1 stop bit, no parity

The internal PIC is running an RC clock which can vary considerable with component tolerance and temperature. To overcome this, a serial protocol is used which looks a the input waveform from the PC as a pulse, the duration of which determines whether the input data is a binary 1 or 0. If the baud rate is 19,200, the data byte to be transmitted to the radio is as follows:

0 (decimal)	=	binary 0
255 (decimal)	=	binary 1

During readback the data can vary and so must be interpreted as follows:

Value $<$ 241 (dec) = binary 0
Value $\geq$ 241 (dec) = binary 1

For the one data byte, 8 bytes must be sent representing each bit.

If the baud rate is 1,200, then the data sent is a normal 8 bit byte.

Note: throughout the document the value of the data is given assuming an RS232 interface. If TTL or similar logic levels are used then the data must be inverted.

3 Definitions

The following give details of the terms used thoughout this document.

3.1 Fosc

This is the value of the Reference Oscillator in the radio. This is **NOT** programmable and the value is not stored in EEPROM. Currently there are two types used by the ST/SRn00 units:

12.8 and 14.85 MHz.

3.2 Fc

Fc is the Comparison Frequency used by the synthesizer in the radio. This is a programmable parameter, but care should be taken if it is changed, because the synthesiser hardware is optimized for certain values of Fc. The Fc in use with the ST/SRn00 family vary for the different applications, but is one of these discrete values:

5, 6.25, 10, 12.5 ,20 or 25 KHz
(refer to Wood & Douglas for the actual values allowed).

Fc is not directly stored as a number in the radio's EEPROM. It is rather indirectly stored as the "Rvalue" (see below).

3.3 Rvalue

This number is the result of the division:

Rvalue = Fosc / Fc

Since the ST/SRn00 family can have one of two Fosc: 12.8 or 14.85 MHz, the possible Rvalues for the radio are:

If Fosc= 12.8MHz:

RR1(5)	=	2560 (dec), A00 (hex)
RR1(6.25)	=	2048 (dec), 800 (hex)
RR1(10)	=	1280 (dec), 500 (hex)
RR1(12.5)	=	1024 (dec), 400 (hex)
RR1(20)	=	640 (dec), 280 (hex)
RR1(25)	=	512 (dec), 200 (hex)

If Fosc= 14.85MHz:

RR2(5)	=	2970 (dec), B9A (hex)
RR2(6.25)	=	2376 (dec), 948 (hex)
RR2(10)	=	1485 (dec), 5CD (hex)
RR2(12.5)	=	1188 (dec), 4A4 (hex)
RR2(20)	=	NOT possible (result is not integer no.)
RR2(25)	=	594 (dec), 252 (hex)

From above it is clear that we have 11 possible values for the combination of 2 Fosc and 6 Fc's. When the Rvalue is read from the unit the current Fc and Fosc. can be determined from these values.

Note: Rvalue is stored as 2 bytes in EEPROM locations 61, 62

3.4 IF

IF is the Intermediate Frequency of the receiver. Its value for the STn00 family is always zero, but for the SRn00 family is as follows:

SRn00 with frequency range < 200 MHz:, "High Side" (+45 MHz)
SRn00 with frequency range > 200 MHz, "Low Side" (- 45 MHz).
STn00, any frequency IF = 0

Note: The IF value is not stored in the unit's EEPROM. The type of ST or SRn00 must be known before programming can be carried out.

4 Channel Information and Storage

For both the ST and SRn00 family, 16 random and 112 sequential channels can be stored in the non-volatile EEPROM memory area of the internal PIC processor. Channels 0 - 15 are stored individually in the EEPROM and channels 16 - 127 are calculated from the start frequency and the table step increment.

In principle frequency is stored in a two byte format plus the high byte offset. The data stored is not the frequency itself, but the frequency divided by the Fc. In other words if the desired frequency for the random channel N is Freq(N), then the value stored is F(N) where F(N) is calculated as:

F(N) = [Freq(N) + IF] / Fc

The result is stored as:

EEPROM(60)	= int [F(N)/65536] *256	high byte offset
EEPROM(N)	= int [F(N)/256] - EEPROM(60)	msb
EEPROM(N + 16)	= F(N)- [EEPROM(60) + EEPROM(N)] *256	lsb

Or if a read-back is performed, the frequency on channel N is calculated as:

Freq(N) = {{{[EEPROM(60)+ EEPROM(N)]*256 }+ EEPROM(N+16)} - IDF }*Fc Hz

where IDF = IF/Fc

Similarly, for the Sequential table start, if desired frequency is FreqS, then:

FS = [FreqS + IF] / Fc

and the result is stored as:

EEPROM(56)	=	int[FS/65536] *256	high byte offset
EEPROM(57)	=	int[FS/256] - Eeprom(56)	msb
EEPROM(58)	=	FS- [EEPROM(56) + Eeprom(57)] *256	lsb

The sequential table step is stored in location 57 as a multiple of Fc

The sequential table is limited by the value stored in location 51 (0-111), but it can not exceed 111. If the value is > 111 then limit should be imposed as follows:

```
MaxCh = EEPROM(51)      Sequential ch. Limit
If MaxCh > 111 then
MaxCh = 111             Limit MaxCh Number to 111 !
endif
```

If read-back is performed, the Start of the Sequential table is calculated as:

SeqFreqStart = {{{[EEPROM(56)+ EEPROM(57)]*256 }+ EEPROM(58)}- IDF }*Fc

where IDF = IF/Fc this is Intermediate Frequency as a multiple of Fc

And a sequential table step is:

SeqFreqStep = EEPROM(59)*Fc [Hz]

5 EEPROM locations

The PIC processor used in the STn00 and SRn00 units has an area of non-volatile memory EEPROM area. The contents of this data is given in the following table:

Location	Description
0 - 15	RX/TX div msbs
16 - 31	RX/TX div lsbs
32 - 47	not used
48 - 49	reserved
50	not used
51	is sequential table limit channel (0-111)
52 - 53	reserved
54	SETUP flags byte (default is FF). At the moment only relevant flag is one occupying bit no 1 and it is Serial or Parallel mode of operation. In Serial Mode Channel selected is the one defined in loc. 63 of EEPROM and is written using PC and serial interface. In parallel mode the switch in the unit (D0 to D2 lines) defines channel selected by the unit. In parallel mode only ch. 0 to 7 are accessible.
55	reserved
56	high byte offset for sequential table
57	sequential table start hi byte
58	sequential table start lo byte
59	sequential table step increment
60	high byte offset
61	Rvalue hi byte
62	Rvalue lo byte
63	serial channel selected

Note: RESERVED locations are planned for future use
NOT USED locations are not used with no plans for future use.

6 Command Types

There are 3 types of packet to be sent to the radio from the PC:

Short packet
Long packet
Direct packet

These are described in the following sections.

6.1 Short packet

This is used to send out single instruction to the radio to carry out functions like setting serial or parallel mode of operation, setting the serial channel number etc.

For the short packet the 8 data bytes to be transmitted are:

X	product ID. For the SRn00 = 0F6 hex and for STn00 = 0F5 hex
A	EEPROM address or DATA1
B	EEPROM data byte or DATA2
C	not used (0) or DATA3
D	not used (0) or DATA4
COM	command byte
chkH	check sum hi byte } A + B + C + D + COM
chkL	check sum lo byte }

Note: This data is sent out at 19200 baud as detailed in section 2. Each data byte within the packet should be separated by 10 to 15 msec. delay and the delay between 2 successive packets should be > 100 msec.

The Command (COM) byte is defined as follows:

6.1.1 COM ≤ 63 dec

This denotes a long packet as described in section 6.2 for re-programming the EEPROM area.

6.1.2 COM = 64 dec

This is the direct packet mode as described in section 6.3 and is used to change the channel of the unit.

6.1.3 COM = 66 dec

This invokes the EEPROM read-back mode. After this command and a packet, the radio will respond with a reading back of the whole content of the EEPROM (64 bytes), one after another, without any address bytes etc.

The data returned from the radio is at 19,200 as defined in section 2 of this document.

6.1.4 COM = 67 dec

This invokes the single byte programming of the EEPROM where:

A	EEPROM address
B	EEPROM data

This is used to change the mode of operation (Serial / Parallel) or for the setting of the serial channel number.

As an example, to set the parallel mode of operation only, the following packet must be sent:

X	chr(0F6) hex for SRn00 or chr(0F5) hex for STn00	
A	EEPROM address byte chr(54 dec)	
B	EEPROM data byte chr(255) for parallel	
C	not used (0)	
D	not used (0)	
COM	chr(67) dec	
chkH	chr(1)	}
chkL	chr(120 dec)	} for parallel mode (total check sum 376 dec.)

For setting the serial channel, two packets must be sent. First the short packet must be sent with B data = 253, followed by the Direct packet as shown in section 6.3.

6.1.5 COM = 70 dec

This resets the radio and should be sent at the end of the Long packet after the EEPROM contents have been re-programmed.

6.2 **Long Packet**

The Long packet mode is intended to re-write the entire EEPROM area of the unit (64 locations). The format is similar to the short packet with a few differences as shown below:

X	Product ID. For the SRn00 = 0F6 hex and for STn00 = 0F5 hex
A	EEPROM DATA1 (from addr COM + 0)
B	EEEPROM DATA2 (from addr. COM +1)
C	EEPROM DATA3 (from addr. COM +2)
D	EEPROM DATA4 (from addr. COM +3)
COM	command byte = EEPROM address of the first data byte in a block
chkH	check sum hi byte }A + B + C + D + COM
chkL	check sum lo byte }

16 packets like above are expected once started, covering all 64 EEPROM locations.

Note: This data is sent out at 19200 baud as detailed in section 2. There should be > 15 msec delay between bytes and > 100 msec. in between packets.

A short packet (with COM = 70) should be sent at the end of the Long packet to reset the device.

6.3 **Direct Packet**

Currently there is only one “Direct packet” in use and it is intended for the channel number change.. This 3 byte packet is sent at 1,200 baud as detailed in section 2. The form of the data packet is as follows:

chr(64 dec)
chr(Channel Number)
chr(149 dec)