# Technical Bulletin

| | |
|---|---|
| Product: | XRP2 Panel Reader |
| Topic: | Serial Command Protocol (SCP) |
| Bulletin Number: | |
| Prepared by: | D Tottey |
| File: | G:\Projects\ID\11-167 Marconi\design\sw\SCP Commands\XRP2 Serial Command Protocol v1.1.docx |
| File Date: | 4 October 2016 |
| Version: | 1.1 |

## Overview

The SCP allows a PC application program to import/export data from/to the Tru-test XRP2 Panel Reader.

The SCP works either over Bluetooth (wireless) or over an RS232 serial cable.

Establishing a connection over RS232 will typically require a USB-Serial adapter.

Establishing a connection over Bluetooth will typically require a Bluetooth dongle unless the Laptop has Bluetooth built in. A separate Technical Bulletin covers these connections.

At the level of the Serial Command Protocol, the XRP2 is always a slave. (This is unrelated to the lower level of the Bluetooth device being a slave or master). The XRP2 is a slave because it always responds to commands - it never initiates them.

The XRP2 responds to SCP commands on the same serial port through which the command is received.

The default RS232 bit format is always asynchronous, 9600, N, 8, 1.

If the bit rate is changed (by the {SEBXm} command), the reader reverts to 9600 at the next Power ON.

## Manually testing SCP commands

To gain familiarity, it is recommended that you connect to the XRP2 with a terminal emulation program such as Window's Hyperterminal or Realterm, and type in commands you want to use. The protocol is made up entirely of printable ASCII characters to make this possible.

If using Hyperterminal, make sure you connect to the correct COM port, and set the baud rate to 9600 and turn on local echo so that you can see the commands that you type. Note: In Hyperterminal, you must disconnect and reconnect after changing its settings parameters. A trouble shooting section is at the end of this document.

The first command to type when manually testing is {ZA1}. This command turns on acknowledgements. You should see an acknowledgement come back. It is the carat '^' character.

Another command to try is {ZN}. It should return [XRP2] the name of the device.

As you can see, SCP commands always begins with an open curly brace character ‘{‘ and end with the close curly brace character ‘}’.

# Serial Command Protocol

The general form of an SCP command is {AAxx} where AA is a 2 to 4 letter command, and xx represents an arbitrary parameter string.

The SCP protocol is a command response protocol. The XRP2 only responds to commands. It’s important to wait for the response to a command before sending the next command.

The XRP2 responds to an SCP command in one of four ways:

1. Nothing
2. An acknowledgement which is always the single character the carat '^'.
3. A data response that begins with open square bracket '[' and ends with close square bracket ']'.
4. An error code that begins with open round bracket '(' and ends with close round bracket ')'.

Between the opening and closing curly braces there is, at a minimum, a 2 to 4 character command, all upper case.

After the 2..4-character command can be other information, which is command dependent.

By default the stick does not acknowledge commands or return error codes. Also it does not return carriage returns or linefeeds.

These can all be turned on with the respective commands:

- {ZA1} turn acknowledgements on.
- {ZC1} turn carriage return, linefeeds on.

The contents of [] responses is totally command dependent.

The XRP2 can take between a few milliseconds and 0.1 seconds to respond to commands. It is usually a good idea to use acknowledgements. However, it is important that the master device not wait indefinitely for an acknowledgement. If the character were to be lost, the master would then effectively hang.

You should timeout after 1 second and then (in general) try to do the command again or otherwise ascertain what the current connectivity status is and take appropriate action.

**Checksum and CRC**

An optional checksum or CRC may be included in an SCP command. If included, any square bracket response to that command will also include a checksum or CRC respectively. The carat and round bracket error responses do not come back with a checksum or CRC.

Although you won’t want to use checksums or CRCs when talking to the stick manually, one should be used for automated communications.

A checksum or CRC goes after all the other data in a command or a response and just before the closing } or ]. Acknowledgements and error codes do not use checksums or CRCs.

A checksum begins with the ~ character.

A CRC begins with the ` character (open quote) Note: Not the normal single quote, the one on the top left of your keyboard

A checksum uses two hex characters and a CRC is 4 hex characters. CRCs should be used in preference to checksums.

Both include the characters from and including the open curly brace up to but not including the ~ or `.

Examples:

{ZA1~47}

{ZA1`73C9}

Note: The 47 and 73C9 is the correct Checksum or CRC for this command

The checksum is the 8 bit sum of the characters, discarding any carry and then converting the result to two hex characters. The above example is calculated as follows.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | '{' | + | 'Z' | + | 'A' | + | '1' |
| = | 7B | + | 5A | + | 41 | + | 31 (all hex) |
| = | 147 (hex) | | | | | | |

The carry is discarded and the two checksum characters become 4 and 7.

The CRC algorithm used is the popular CRC-16 based on x^16+x^15+x^2+1

The CRC calculation function source code can be obtained from Tru-Test.

# SCP commands

Following are tables of SCP commands.

# General commands

| Command | Parameters details | Explanation |
|---|---|---|
| {SEBXm} | m = 1200/2400/ 4800 / 9600 /19200 / 38400 / 57600 / 115200 | Set RS232 port bit rate (bps) to m. Acknowledgement is sent at original bit rate. |
| {VA} | [s] | Get software version. |
| {VU} | | Initiate ISP bootloader |
| {ZA}<br>{ZAn} | [0/1]<br>n = 0/1 | Get, set acknowledge off/on |
| {ZC}<br>{ZCn} | [0/1]<br>n = 0/1 | Get, set Carriage return & Line feed on responses off/on |
| {ZN} | [XRP2] | Request device name |

# Device Import/Export commands

To understand the importing/exporting commands, it is necessary to understand the tables that exist inside the XRP2 Panel Reader.

**Scanned Records Table**
Up to 20,480 records containing 4 fields: Session, EID, Date, Time.

The command {DLi} downloads one record. It has an index that begins at 0 for the oldest record.

The table contains all the records of all sessions in session order.

Sessions are punctuated by session markers, which are records that have a zero EID. The presence of these records needs to be accounted for when importing sessions.

The {DSi} command allows finding the locations of all the session markers. In this way the beginning and ending point for the records of each session can be determined. As well as the beginning and ending indexes, it establishes how many sessions are present, how many records are in each, and each session's date and time. When a single session is imported, only the records between the beginning and ending indexes for that session need be imported.

The {DN} command can be used to find out the total number of scanned records in memory.

The {CL} command is used to clear all scanned records in all sessions.

This table can only be imported.

The following SCP commands are specific to importing/exporting data for the tables discussed above:

| | | |
|---|---|---|
| {CL} | Clears All Records | This command will typically be sent after all records have been successfully downloaded. |
| {DN} | [n]<br>0..20479. | Returns the number of scanned EID records currently in memory. |
| {DSi} | [j,time stamp]<br>i: starting index<br>j: Session Marker Index<br>time stamp: Time Stamp in YYYY-MM-DD HH:MM format<br>When there are no more session markers to find, returns empty square brackets e.g. []. | Downloads the next session marker from the specified starting index i.<br>The complete Session Marker table can be downloaded by starting with {DS0} and continuing with i = j+1 until [] is received.<br>Eg<br>{DS0}[14,2010-11-08 14:22]<br>{DS15}[26,2010-11-08 16:19]<br>{DS27}[134,2010-11-08 20:13]<br>{DS135}[]<br><br>Note: The response time can be up to 3 seconds because internally the XRP2 must search linearly for the session markers. |

| | | |
|---|---|---|
| {DLi} | [session,eid,,Date,Time,]<br>i: Index( 0..20479)<br>session: Session Number<br>eid: EID is format as specified by {FS}.<br>Date: Date in YYYY-MM-SS format<br>Time: Time in HH:MM:SS format<br>When the index, i, equals or exceeds the number of records in memory, it returns empty square brackets e.g. []. | Download one record from the XRP2 as specified by Index i.<br>This command downloads EID scan records one at a time. |
| {DLi,n} | [session,eid,,Date,Time,;<br>session,eid,,Date,Time,;……<br>session,eid,,Date,Time,]<br>n: Number of records (maximum 5) | Downloads multiple records (max 5) starting at index i. |

## Other commands

| | | |
|---|---|---|
| {ZF} | | Set to factory defaults ( setups are all set to factory defaults, data files are not cleared) |
| {ZZ} | | Soft reset (processor restarts, peripherals are reinitialized, setups and data files are unchanged) |
| {VU} | | Initiate firmware upload (Hex format binary follows. Format and sub protocol to be defined by manufacturer) |
| {VS} | [n] | Get serial number. Every unit is factory programmed with a different serial number. |
| {BP}<br>{BPx} | [x]<br>0: disabled<br>1: enabled | Beeper on/off |
| {BR} | [id] | Gets the stick's Bluetooth MAC address in the form as described in the {BS} command. |
| {BT}<br>{BTx} | [x]<br>0 : Disabled<br>1 : Manual<br>2: Auto | Bluetooth mode. (Same as in Menus)<br>Disabled – always off<br>Manual – Accepts incoming connections (Slave). User can search and connect as from Menus as a Master.<br>Auto – On power up, automatically connects to last connected device. If the device is not available, it will try the next device on the 'Paired Devices' List. If 'Paired Device' list is empty, the XRP2 will attempt to search for and connect to the following devices:<br>XR3000BT, ID3000BT, Eziweigh7, XR5000, Gallagher W810 as well as Series3000 dongles and Eziweigh dongles . |
| {BTC} | | Clear the 'Paired' device list. |
| {BTP} | [$Name_1$,$MAC_1$;$Name_2$,$MAC_2$;……$Name_N$,$MAC_N$] | Returns the Paired Devices List |

| | | |
|---|---|---|
| {BTPm,n} | m= MAC address (12 chars/ 8 bytes)<br>n= BT friendly name (max 10 chars) | Adds a device to the top of the Bluetooth Paired Devices List<br>Eg {BTP000000000000,GGLscale} |
| {BV} | [X.XXV] if not charging<br>[X.XXV (CH)] if charging<br>X.XX: Voltage | Returns the Battery Voltage and charging status |
| {CD}<br>{CDx} | [x]<br>x = 0: disabled<br>x : 1: enabled | Check for duplicates.<br>Controls whether or not the reader checks for duplicates. Checks for duplicate tags in the current session up to a maximum of 500 tags. |
| {FO} | [status]<br>status: 1 = enabled<br>status: 0 = disabled | Returns the status of the RS232 Force ON setting. |
| {FOx} | x = 1: enable<br>x = 0: disable | Enables/Disables RS232 Force ON feature. |
| {FS}<br>{FSx} | [x]<br>x = 0 (with space)<br>x = 1 (no space)<br>x = 2 (hex)<br>x = 3 (ISO) | Output format (RS232/BT/Download)<br>==> 982 123456789012<br>==> 982123456789012<br>==> 8000F580011D47F6<br>==> 0x02100000009821234567890 12 |
| {IA}<br>{IAx} | [x]<br>x = 1: enable<br>x = 0: disable | When enabled, ignores the Animal Bit (i.e. assumes the Animal bit is set) |
| {NS} | New Session | Starts a new sessions (inserts a session marker record and resets counter)<br>(Same as in Menus) |
| {OM}<br>{OMx} | [x]<br>x=0 Single Output mode<br>x=1:Continuous mode | Sets the Output Mode to ether Single or Continuous output mode |
| {PS}<br>{PSx} | [x]<br>x=0: Phase is 0 degrees<br>x=1: Phase is 180 degrees<br>x = 2 Toggle 0←→180 | Gets/Sets the Phase Switching.<br><br>NOTE: This command is only used with 2 synchronised readers with closely coupled antennas (Only change the setting on one reader) |

| | | |
|---|---|---|
| {RM}<br>{RMx} | [x]<br>x=0 Dual Output Mode(FDX and HDX)<br>x=1:FDX Only<br>x=2:HDX Only | Sets the read mode to read either FDX or HDX or both types of tags |
| {RR} | [n rpm]<br>N: Number of reads / minute | Returns the EID read rate. |
| {SD}<br>{SDdate} | [date]<br>format of date is ISO8601 standard:<br>YYYY-MM-DD<br>date example :<br>2008-12-31 | Get or Set the Date |
| {ST}<br>{STtime} | [time]<br>format of time is ISO8601 standard:<br>HH:MM:SS (24hour)<br>time example:<br>23:59:59 | Get or Set the Time |
| {TLx}<br><br>{TL} | x=1 :High<br>x=0: Low<br>[x] | Set the transmitter power level (High or Low)<br>Get the transmitter power level |
| {TV} | [TV,t f Hz]<br>t: tuning value (0..31)<br>f: Free running carrier frequency | Returns the current tuning value |
| {UC} | [BT]<br>Or<br>[RS232] | Returns either [BT] or [RS232] depending on which UART is connected. |
| {VL}<br>{VLn} | [n] | Return the current language setting.<br>Set the current language.<br>{VL0} – English<br>{VL1} – Spanish<br>{VL2} – Portuguese<br>{VL3} – French<br>{VL4} – German<br>{VL5} – Italian<br>{VL6} – Danish<br>{VL7} – Swedish |