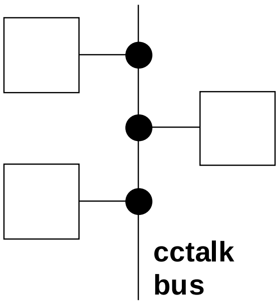


SR5R Serial Protocol
-
Issue 1.8

Revision History

Issue	Date	Comments
1.0	27-04-01	First Issue
1.1	08-05-01	Delay after programming of 50ms
		Remote Programming Files
	22-05-01	Minor text clarification
1.2	21-06-01	ACK returned for headers 219 & 218 (PIN no.)
1.3	22-08-01	Clarify reference to ‘virtual’ rotary switch
1.4	30-01-02	DE examples replaced with EU
		Latest cctalk generic specification is 4.2
1.5	31-01-02	‘Extra file’ clarification
1.6	07-02-02	Section on ‘run mode’ commands added
1.7	15-04-03	Added TSP number
		Changed Header and Footer
1.8	30-06-04	Changed footer

Contents

Introduction 3
Remote Programming Overview 3
Implementation on SR5R 4
Device Address 4
Electrical Connections 5
Inhibits and Overrides 5
Credit Polling 6
Sorter Operation 6
Serial Teach 7
Token Selection 7
Diagnostics 8
Command List 8
Run Mode Commands 14
Remote Programming in Detail 15
Remote Programming Files 15
Remote Programming Commands 17
Remote Programming Error Codes 20

Introduction

The SR5R coin acceptor is a serial-only validator.

The serial protocol is **cctalk**, now a standard for the money transaction industry.

Full details of the protocol can be found in the following document, available from the customer services department of Money Controls Ltd (MCL)…

'cctalk Serial Communication Protocol - Generic Specification - Issue 4.2'

The generic specification explains the history and design philosophy of the protocol, the message structure and a complete list of commands covering different types of money transaction peripherals.

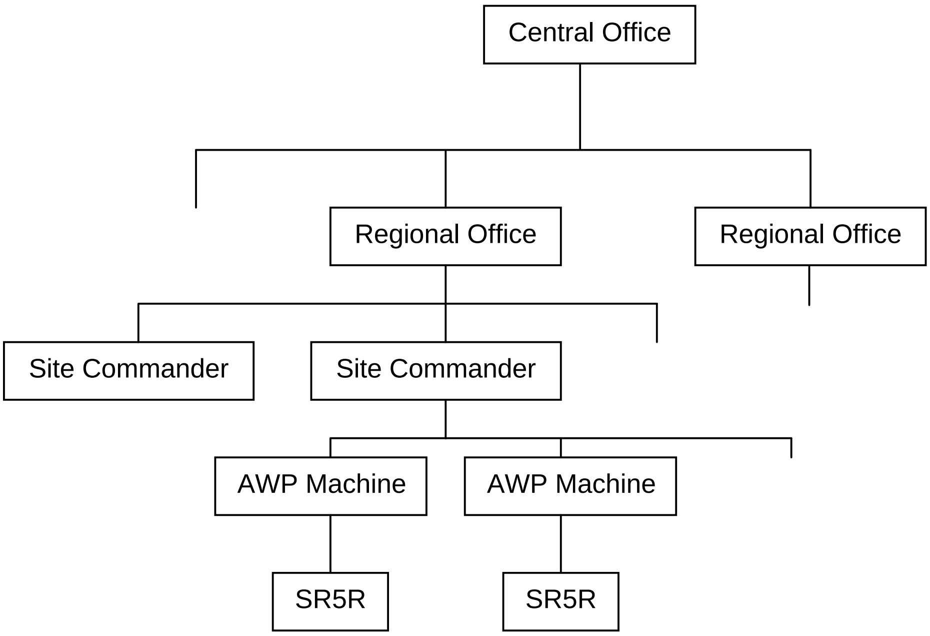
Listed below are the command headers available to SR5R and any product-specific features which you need to know about.

Remote Programming Overview

SR5R offers genuine remote coin programming on a serial-only interface. A single binary file is transferred into the coin acceptor using cctalk commands in order to reprogram that coin position. This may be an enhanced version of the existing coin specification or part of a new currency download. The binary file is the same for all SR5R's with the same build standard. No external calculations are required - all the 'intelligence' is in the SR5R itself.

In order to support the additional functionality required for this type of remote coin programming, additional cctalk commands have been defined in the 'Application specific' section of the serial standard. These commands are specific to MCL products and are available for use for in applications which need them. The previously published cctalk public command headers operate as normal bearing in mind that parallel features have been removed. It is likely that most coin acceptor manufacturers will adopt proprietary programming algorithms.

An application which requires thousands of coin acceptors to be reprogrammed during the night could be achieved quite easily with SR5R. The updated coin specification files could be transmitted from a central office via any means (e.g. modem, internet, ADSL etc.) and a small software kernel in the target machine would re-program the coin acceptor using the standard cctalk interface. The bulk data transfer is unidirectional and would allow a tree-like distribution pattern.



Implementation on SR5R

The protocol conforms to cctalk b96.p0.v12.a5.d0.c5.m0.x8.e0.i0.r4

In other words…

- 9600 baud
- open-collector
- +12V nominal supply
- +5V data pull-up
- supply sink
- connector type 5
- slave device
- 8-bit addition checksum
- no encryption
- cctalk minor release 0
- cctalk major release 4

Device Address

All SR5R's leave the factory with **address 2**.

The address is stored in EEPROM and can be subsequently changed with serial commands. Unless you have an application requiring more than one coin acceptor on the serial bus, it is strongly recommended you leave the address alone. The default addresses for hoppers and bill validators have been made different and will not clash with the coin acceptor.

If the address has been changed to an unknown value then you will either have to search all through the entire address space (2 to 255) with the 'Simple poll' command until an ACK is returned or send the 'Address poll' command with the broadcast address.

Electrical Connections

Only 3 wires are required for the interface between a SR5R and the host machine.

```
(1) /DATA
(7) +12V
(8) 0V
```

The dual-mode SR5 used pin 9 as a serial mode selection pin (pulled to 0V). As SR5R operates in serial mode only, pin 9 is no longer used.

The bi-directional data line operates at 9600 baud, 1 start bit, 1 stop bit and no parity bits. There is no option to select a different baud rate.

The data pin on SR5R is pulled up to +5V via a **10K resistor** and should be driven with an open-collector transistor.

Pin 5 is a hardware /RESET line but no connection needs to be made. A software reset command is provided for this purpose.

Inhibits and Overrides

SR5R can accept **16 different coins** including a special token selected from a bank of 12 if in 'token mode'.

There are 16 inhibit bits used in the 'Modify inhibit status' command allowing any combination of coins to be accepted or rejected.

There is support for 8-way routing with a 7-way override.

There are 7 override bits used in the 'Modify sorter override status' command allowing complex routing patterns to be achieved.

Each coin can have **4 paths** programmed into EEPROM. If the first path has an active override, then the coin will be routed down the second path. This continues until all the assigned paths have overrides when the coin will be sent down the default path.

The default path can be changed with the 'Modify default sorter path' command.

SR5R has a command called 'Modify inhibit and override registers' which allows *current* and *next* coin values to be sent. This allows accurate coin-by-coin acceptance and routing, overcoming the inherent latency in serial operation.
At power-up or reset, all coins are inhibited and all sorter overrides are removed. Coins cannot be accepted until a non-zero inhibit mask is sent to the mech.

Credit polling

Coin credits are obtained by polling the coin acceptor at regular intervals using the ‘Read buffered credit or error codes’ command. Up to 5 credits or errors are stacked in the return buffer.

The return data is 11 bytes in length.

[event counter]
[result 1A] [result 1B]
[result 2A] [result 2B]
[result 3A] [result 3B]
[result 4A] [result 4B]
[result 5A] [result 5B]

A & B are formatted as [credit code] [sorter path] or [0] [error code] depending on the logged event. See Table 2 in the generic specification for a list of error codes.

Note that the event counter wraps from 255 to 1, not 0.

A suitable polling interval is anywhere between **100ms and 900ms**. If other cctalk peripherals are on the bus then these may have to be polled as well.

SR5R has a ***credit poll watchdog*** feature whereby if the mech is not polled for 1s or longer then the device auto-inhibits and does not accept any more coins. When polling resumes, the auto-inhibit is lifted. This feature prevents coin swallowing in the event the host machine serial link goes down.

Sorter Operation

SR5R can operate without a sorter, or in 4-way and 8-way routing configurations.

This table shows how the different modes relate to each other.

cctalk Route Code	cctalk Override Bit Position	8-way Route Number	4-way Route Letter	8-way Manifold Active ?
1	0	1	D	Yes
2	1	2	C	Yes
3	2	3	B	Yes
4	3	4	A	Yes
5	4	5	a	No
6	5	6	b	No
7	6	7	c	No
8	-	8	d	No

The route numbers and letters refer to physical exit positions. Looking down from the front of the product (accept gate side), the sorter exits appear as…

C B

D A

(4-way)

or

8 7 6 5

1 2 3 4

(8-way)

The default sorter route (cashbox path) for a 4-way sorter is usually ‘A’ and for a 8-way sorter is ‘8’. Note that cctalk on SR5R uses the 8-way sorter notation even if a 4-way only sorter is fitted. This is important for driver software because the default route ‘A’ is usually programmed as the number ‘5’. As can be seen from the above table, routes 4 and 5 are both ‘A’ when a 4-way sorter is fitted.

Serial Teach

Coins and tokens can be taught using the ‘Teach mode control’ and ‘Request teach status’ commands. This method involves the insertion of a small sample of coins (typically 8) but the process can be controlled entirely over serial.

If a token is taught then after the teach process send a ‘Reset device’ command to force the new token to be used (alternatively, cycle the power).

Token Selection

SR5R is a 16 coin validator with the capability to reprogram any of the 16 coins remotely. However, to retain compatibility with SR5 (and C435, C335 etc.) the validator can also accept any 1 of 12 pre-programmed tokens and make use of the Dual Coin Entry (DCE) chute for determining whether a coin or a token was inserted.

Token 12 or position ‘C’ on the former rotary switch can be taught using the serial ‘teach mode control’ command. The other token slots cannot be taught or reprogrammed remotely.

When SR5R powers-up, token 12 is selected by default. If another token is required, the host software can select it with the ‘virtual’ rotary switch. The ‘Handheld function’ command which was thoughtfully added to the cctalk specification to allow parallel functionality to be transferred across to serial as seamlessly as possible.

Entire Transmitted data : [2] [2] [1] [177] [1] [TOKEN_NUMBER] [CHK]
Entire Received data : ACK

The TOKEN_NUMBER should be 1 to 12. The value of 0 disables token acceptance.

Diagnostics

Since SR5R is a serial-only coin acceptor, the parallel diagnostic connector available on SR5 has been removed. However, it is still possible to execute a full diagnostic test of the validator by sending the cctalk 'Perform self-check' command. In fact, it is recommended that as part of the power-up initialisation of the host machine and peripherals that the SR5R is always checked in this manner. This will then notify the host software of any possible reason for reduced performance prior to the coin inhibits being lifted.

Command List

The following table shows all the cctalk commands available for customer use on SR5R. More details can be found in the generic specification.

Any [data bytes] are shown in decimal.

DCE = Dual Coin Entry rundown. This is a 'Y' chute with optos in the coin and token entry slots.

MDCES = Multi-Drop Command Extension Set. These commands are only used when peripheral addresses are unknown.

Header numbers are shown in descending order.

***** SR5R has 71 serial commands *****

Header :Sub-header	Function	Returned Data and Comments
254	Simple poll	ACK returned. All cctalk peripherals should reply to a simple poll - use this command for testing the comms link.
253	Address poll	MDCES support
252	Address clash	MDCES support
251	Address change	MDCES support
250	Address random	MDCES support
249	Request polling priority	[1] [200] = 200 ms
248	Request status	[0] = 'OK' [1] = 'Flight deck open'
247	Request variable set	Nothing of interest
246	Request manufacturer id	'Money Controls'
245	Request equipment category id	'Coin Acceptor'
244	Request product code	'SR5R'
243	Request database version	[0] if no calibration support, otherwise new coin sets can be remotely programmed.
242	Request serial number	Supported. Range 0 to 16,777,215. This is an internal electronic serial number and may be different to the batch serial number on the label. The serial number is unique on SR5R and may not be modified by conventional means.
241	Request software revision	'SR5R-V1.00' or a subsequent revision.
240	Test solenoids	Bit 0 = Accept gate Bit 1 = Lower sorter flap Bit 2 = Upper sorter flap Bit 3 = 8-way Manifold flap (if used) Pulsed for 500ms
236	Read opto states	Bit 0 = DCE Coin opto (1 = blocked / not fitted) Bit 1 = DCE Token opto Bit 2 = Sorter optos Sorter optos cannot be tested individually - they are combined in series.
232	Perform self-check	Supported. Refer to table 3 in the generic specification.
231	Modify inhibit status	[inhibit 1] [inhibit 2] Support for 16 coins. 0 = inhibited, 1 = enabled Inhibits are stored in RAM and are lost at power-down or reset. The power-up state is 'all coins inhibited'.
230	Request inhibit status	Supported.

229	Read buffered credit or error codes	Supported. 5 event buffer. This is the only command which can be used to obtain coin credit information.
228	Modify master inhibit status	Supported 0 = inhibited, 1 = enabled The master inhibit flag is stored in EEPROM. The master inhibit flag is not normally used and all SR5R’s which leave the factory are ‘enabled’. If a coin is put down a master inhibited mech then the ‘Inhibited coin’ error is returned.
227	Request master inhibit status	Supported
226	Request insertion counter	Supported The counter is stored in RAM and is cleared at power-down or reset.
225	Request accept counter	Supported The counter is stored in RAM and is cleared at power-down or reset.
222	Modify sorter override status	Bit 0 = override route 1 (0 = override) Bit 1 = override route 2 Bit 2 = override route 3 Bit 3 = override route 4 Bit 4 = override route 5 Bit 5 = override route 6 Bit 6 = override route 7 Bit 7 = { not used) Overrides are stored in RAM and are lost at power-down or reset. The power-up state is ‘no overrides’.
221	Request sorter override status	Supported
219	Enter new PIN number	ACK returned but PIN number mechanism not used.
218	Enter PIN number	ACK returned but PIN number mechanism not used.
216	Request data storage availability	[2] [2] [8] [2] [8] EEPROM storage is available for customer use… 2 blocks of 8 bytes for reading 2 blocks of 8 bytes for writing A number of security features could be implemented with this 16 byte memory such as logging machine ident numbers during installation.
215	Read data block	Supported
214	Write data block	Supported
213	Request option flags	Bit 0 = credit code format (0 = position)

212	Request coin position	Supported Returns position of coins with specified cctalk credit codes.
210	Modify sorter paths	[coin position] [path 1] [path 2] [path 3] [path 4] coin position : coin 1 to 16 path : route 1 to 8 route 1 ⇔ D (4-way…) route 2 ⇔ C route 3 ⇔ B route 4 ⇔ A route 5 ⇔ a (8-way…) route 6 ⇔ b route 7 ⇔ c route 8 ⇔ d Sorter paths are stored in EEPROM.
209	Request sorter paths	4 paths returned
202	Teach mode control	[coin position] Specify 1 to 16 for a coin or 17 for token position 'C'. The 'virtual' rotary switch must be on 'C' for the teach token to be accepted. If the teach mechanism has been disabled for security reasons then the 'teach error' code is returned.
201	Request teach status	Supported
199	Configuration to EEPROM	Stores the current inhibit & override settings into EEPROM so that they are then used at the next power-up. Use with caution as this modifies the factory default settings !
197	Calculate ROM checksum	4 byte ROM checksum returned
196	Request creation date	Supported
195	Request last modification date	Supported
194	Request reject counter	Supported The counter is stored in RAM and is cleared at power-down or reset.
193	Request fraud counter	Supported The counter is stored in RAM and is cleared at power-down or reset. Note : Fraud coins can only be counted if they are factory pre-programmed into the coin acceptor and marked as 'bad'
192	Request build code	8 character ASCII string representing major build options. May contain spaces.

189	Modify default sorter path	[default path] Specified as 1 to 8. This is the path the coin is routed to after all the override paths have been used. The default path is stored in EEPROM and is used at the next power-up. The default path for 4-way routing is A. The default path for 8-way routing is 8.
188	Request default sorter path	Supported
185	Modify coin id	Supported 6 x ASCII characters e.g. GB100A
184	Request coin id	Supported 6 x ASCII characters e.g. GB100A
182	Download calibration info	Supported No customer parameters. 168 bytes in length.
181	Modify security setting	Supported Security settings are stored in RAM and are cleared at power-down or reset. 0 = factory default setting Each coin can have a different security setting.
180	Request security setting	Supported
179	Modify bank select	[bank no.] 0 = default (both banks enabled) 1 = bank 1 only enabled (coins 1 to 8) 2 = bank 2 only enabled (coins 9 to 16) The action of this command is the same as using the 'Modify inhibit status' command.
177	Handheld function	Low level engineering functions… mode = 0 : function = 0 return token in use mode = 0 : function = 1, [token] select token to be used (virtual rotary switch) mode = 0 : function = 5 fix EEPROM checksum
176	Request alarm counter	[alarm count] The alarm counter is stored in RAM and is cleared at power-down or reset. Additionally, it is cleared **after each request** and so should be used cumulatively. Alarm conditions… a) blocked sorter exit optos b) blocked credit sensor c) coin going backwards detection
173	Request thermistor reading	Supported
170	Request base year	'2001'

169	Request address mode	[132] Address is stored in EEPROM and may be changed serially (non-volatile).
162	Modify inhibit and override registers	Supported
96:255	Begin packet upload	Start remote programming
96:254	Upload packet data	Send data packets
96:253	End packet upload & program	Finish remote programming
96:252	Modify coin signature	Re-program coin signature after reading it.
96:251	Request coin signature	Read coin signature for a later ‘refreshing’ operation.
96:250	Verify coin signature	Check the coin signature hasn’t changed.
96:249	Remove coin signature	Delete a coin signature. This stops a coin being accepted and is more permanent than modifying the inhibits.
96:248	Request extended coin id	The extended coin id contains 2 extra characters which reflect window security. 8 x ASCII characters e.g. EU100A-2
96:247	Calculate ROM secure signature	The ROM checksum is calculated with a 32-bit seed and uses a XOR / rotate algorithm for improved security.
4	Request comms revision	[2] [4] [0] cctalk level = 2 (remote download capability) specification = 4.0
3	Clear comms status variables	Supported
2	Request comms status variables	Supported The rx buffer can store complete messages up to 176 bytes in length.
1	Reset device	Supported This command performs a ‘software reset’.

Run Mode Commands

Only a subset of the cctalk command set on this product is designed for use in ‘run mode’, i.e. when the coin acceptor is in a machine and accepting coins. The other commands are designed for off-line use or during the machine / peripheral initialisation process. Using these commands at other times could seriously reduce true coin accept rate or generate unwanted errors as priority is taken away from coin detection and validation.

These commands can be used in run mode.

Header 229, Read buffered credit or error codes

Header 231, Modify inhibit status
Header 230, Request inhibit status
Header 222, Modify sorter override status
Header 221, Request sorter override status

Header 226, Request insertion counter
Header 225, Request accept counter
Header 194, Request reject counter
Header 193, Request fraud counter

Header 254, Simple poll
Header 248, Request status
Header 184, Request coin id
Header 179, Modify bank select
Header 176, Request alarm counter
Header 162, Modify inhibit and override registers
Header 181, Modify security setting
Header 180, Request security setting

Note that requesting the serial number with header 242 should only be done during initialisation - it is not a run mode command. If a coin acceptor is replaced in a machine while it is operating them there will be multiple communication errors during credit polling. When the comms resumes working, a serial number check could then be done for security.

Remote Programming in Detail

A coin packet for remote programming consists of a binary data file in DOS / Microsoft Windows format with the extension ‘.bin’. It will be encrypted and protected by checksum. There will be one file per coin, and each file size will range typically from 1,000 bytes to 3,500 bytes depending on the proximity of close frauds etc. The neural network algorithm used is extremely accurate and requires variable amounts of data.

MCL will distribute new coin files by whatever means is appropriate at the time. This may be by CD, email or via the MCL web site.

Sending coin data to the SR5R validator will be a simple matter of transferring the download file using the new cctalk serial commands described below. As the maximum cctalk data size is 252 bytes, the file will need to be split into ‘chunks’ as it is sent. An ACK return message will confirm each data chunk has been received successfully and the next chunk can be sent.

A mechanism will be provided whereby the host machine can ‘refresh’ the coin windows at early morning start-up. This is entirely optional as the coin windows are stored permanently in EEPROM but it may be thought there are security advantages in negating any interference to the unit while it is ‘off-line’. Serial commands will be provided for reading coin signatures out of the validator. These can be stored locally and used to reprogram the validator as part of a ‘once per day’ security procedure. Coin signatures are specific to a particular validator and will not work if transferred to another one. They are also encrypted.

Remote Programming Files

This is an example of a set of programming files for the Euro currency.

```
EU200A-0 BIN         1,503  22/01/02   8:30 EU200A-0.bin
EU100A-0 BIN         1,656  22/01/02   8:30 EU100A-0.bin
EU050A-0 BIN         1,797  22/01/02   8:30 EU050A-0.bin
EU020A-0 BIN         1,656  22/01/02   8:30 EU020A-0.bin
EU010A-0 BIN         2,103  22/01/02   8:30 EU010A-0.bin
EU005A-0 BIN         1,962  22/01/02   8:30 EU005A-0.bin
EU002A-0 BIN         2,691  22/01/02   8:30 EU002A-0.bin
EU001A-0 BIN         2,691  22/01/02   8:30 EU001A-0.bin
TK309A-0 BIN         2,703  22/01/02   8:30 TK309A-0.bin
TK381A-0 BIN         2,997  22/01/02   8:30 TK381A-0.bin
TK418A-0 BIN         2,832  22/01/02   8:30 TK418A-0.bin
XTRAEU~1 BIN            63  22/01/02   8:30 XtraEU_SR5R.bin
```

The 2nd column of this DOS directory listing shows the file size in bytes and is typically similar for other currencies.

This set consists of 8 Euro coins, 2 tokens (= TK) and 1 extra file. The extra file contains information on top of the coin data which is required for that currency. Not all currencies require extra information to be programmed. When reprogramming the SR5R, it is best to select all the coins from within a currency directory as these coins have been qualified and released together. Mixing coins from different currency directories could possibly result in overlaps between coin windows and a chance of cross-crediting.

The **extra file** has a filename beginning with the characters ‘Xtra’. It is programmed into the validator in exactly the same way as a coin using ‘Begin packet upload’, ‘Upload packet data’ and ‘End packet upload & program’ commands. **It only needs to be programmed once**, not for every coin. Where in the coin programming sequence it is sent is not important. The ‘End packet upload & program’ command requires a [coin position] parameter. For extra files this is not used so a zero byte can be sent for the coin position.

Some currencies are released as dual currency sets. In this case the extra file could be called XtraEUGB_SR5R.bin for combined EU and GB coins.

Examples

a) EU100A-0.bin

country code = EU (Euro)
coin value = 100 (€1 or 100 cents)
mint issue = A
window type = 0 (default)

The binary file is encrypted but contains a plain text stub which can be viewed in a text editor under Microsoft Windows or typed on a DOS screen.

```
C SR5     STD     00008
```

The format is...

```
<file type, 1 char> <space> <product name, 8 chars> <variant name, 8
chars> <specification version, 5 chars>
```

A file type of ‘C’ means a coin file. The product name and variant name must match the coin acceptor being programming. SR5R uses SR5 coin specifications and is available in ‘STD’ and ‘FRT’ variants for standard (top entry) and front entry builds.

The specification version is increased for subsequent releases to the coin specification file based on new information about close frauds etc. It is preferable to always use the latest version available from MCL.

b) XtraEU_SR5R.bin

coin specification = EU (Euro)
product = SR5R

The binary file is encrypted but contains a plain text stub which can be viewed in a text editor or typed on a DOS screen.

```
M SR5     STD     00008
```

The format is...

```
<file type, 1 char> <space> <product name, 8 chars> <variant name, 8
chars> <specification version, 5 chars>
```

A file type of ‘M’ means a macro file. This is a file containing a list of commands on top of those needed to reprogram a coin. The remaining fields are described above.

Remote Programming Commands

The modes below are sub-headers below the cctalk command header 96. The sub-header is always the first data byte.

As an example, this is the entire packet structure for the ‘Begin packet upload’ command below, assuming a host address of 1 and a mech address of 2.

Entire Transmitted data : [2] [1] [1] [96] [255] [157]
Entire Received data : [1] [0] [2] [0] [253] - ACK

Mode 255 : Begin packet upload

Transmitted data : [255]
Received data : ACK

This command clears any internal pointers and prepares for a new coin file to be sent to the validator.

Mode 254 : Upload packet data

Transmitted data : [254] [data 1] [data2] [data 3]… [data N]
Received data : ACK

N = 1 to 251

The amount of data sent in each packet can vary. The validator will keep track and update the address pointer accordingly. If the data is received with a checksum or length error then no ACK is returned and the host has the option of re-sending that packet or starting again with the ‘Begin packet upload’ command.
Sending 251 bytes in each packet will result in the fastest possible programming time but 250 bytes or 128 bytes may be chosen as being more convenient.

Mode 253 : End packet upload & program

Transmitted data : [253] [coin position]
Received data : ACK or [error code]

This command signals the end of the packet upload process, starts the packet validation algorithm and if successful programs a coin into the specified coin position. Either an ACK message is returned which means the coin was programmed successfully or an error message is returned. See the error code table for a list of all possible error conditions.

[coin position]
1 to 16

[error code]
See error code section.

After sending this command, the host machine should wait at least **50ms** before sending another cctalk command. After programming a coin, the validator performs a software reset and serial communication activity during this time will be ignored.

Mode 252 : Modify coin signature

Transmitted data : [252] [coin position] [data 1] [data 2] [data 3] … [data N]
Received data : ACK or [error code]

[error code]
See error code section.

The coin signature is the essence of the coin acceptance criteria for each programmed coin. Coin signatures are 44 bytes on SR5R. They are unique to each manufactured coin acceptor. Reading a coin signature from mech A and sending it to mech B will always fail.

Mode 251 : Request coin signature

Transmitted data : [251] [coin position]
Received data : [data 1] [data 2] [data 3]… [data N]

If the coin position is not valid then there is no reply.

The requested coin signature can be stored and used later with the 'Modify coin signature' and 'Verify coin signature' commands.

Mode 250 : Verify coin signature

Transmitted data : [250] [coin position] [data 1] [data 2] [data 3] … [data N]
Received data : ACK or [error code]

[error code]
See above.

This command reports back to the host whether a modification has been made to the coin signature. The existing data is not altered though.

Mode 249 : Remove coin signature

Transmitted data : [249] [coin position]
Received data : ACK or [error code]

The coin signature at the specified position is erased and that coin will no longer be accepted. A temporary reject of a coin can be performed using the 'Modify inhibit status' command as per SR5.

Mode 248 : Request extended coin id

Transmitted data : [248] [coin position]
Received data : [data 1] [data 2] [data 3]… [data 8]

A string identifier is returned showing which coin is programmed into the specified coin position. If the coin position is not valid then there is no reply.

The string is an extra 2 characters longer than the one returned by the existing 'Request coin id' command.

e.g. ZA100B-0.bin

country code = ZA (South Africa)
coin value = 100 (1 Rand)
mint issue = B
window type = 0 (default)

The same coin may be released with a number of different window types to tackle different fraud situations. Some window types may result in a slight loss in true coin accept rate in order to eliminate very close frauds.

Mode 247 : Calculate ROM secure signature

Transmitted data : [247] [seed 1] [seed 2] [seed 3] [seed 4]
Received data : [signature 1] [signature 2] [signature 3] [signature 4]

The signature is calculated from the seed value and the entire contents of the microcontroller ROM using a XOR / rotate algorithm.

This is a more secure version of the 'Calculate ROM checksum' command which is still available for backwards compatibility. The signature can be verified against the seed and is difficult to fraud.

Remote Programming Error Codes

Error Code	Description	What Went Wrong ?
255	Packet too short	Less data than expected was sent to SR5R during the packet upload process
254	Packet too long	More data than necessary was sent to SR5R during the packet upload process
253	Too much data	SR5R ran out of memory for storing the coin packet ! Should never happen.
252	File verification error type 1	Data corruption was detected in the coin packet.
251	File verification error type 2	Data corruption was detected in the coin packet.
250	File verification error type 3	Data corruption was detected in the coin packet or a coin signature was transferred to another mech.
249	File verification error type 4	Data corruption was detected in the coin packet.
240	Unsupported packet header	The packet upload file is not recognised.
239	Unsupported file format	The packet upload file is not supported.
238	Non-matching product name	The wrong upload file is being used. Contact MCL for the correct version for your application.
237	Non-matching variant name	The wrong upload file is being used. Contact MCL for the correct version for your application.
236	Non-matching database version	The wrong upload file is being used. Contact MCL for the correct version for your application.
235	Mech does not support calibration	Calibration was factory-disabled in this coin acceptor. No coins can be programmed remotely.
234	Mech does not allow calibration	Calibration has been disabled in this coin position. It may be possible to program other coin positions.
232	Key mis-match	The wrong upload file is being used. Contact MCL for the correct version for your application.
231	Sensor mis-match	The wrong upload file is being used. Contact MCL for the correct version for your application.
101	Failed coin signature verification	The coin signature has been changed.
100	Illegal coin position	The coin position requested is outside the legal range for SR5R. Use a value from 1 to 16 only.