Bulkvalidator

cctalk Interface

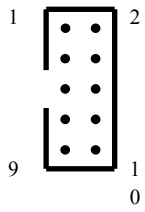
Autor: C.-P. Heins

Table of contents

1 General

1.1 Connector, 10 pin

PIN	*Parallel*
1	Data
2	Data Gnd
3	n.c.
4	n.c.
5	n.c.
6	n.c.
7	Vdd (12-24 DC)
8	GND
9	n.c.
10	n.c.



1.2 Serial Protocol

1.2.1 Voltage levels

Mark state (idle) +3.3V nominal Range 2.0V to 5.0V*
Space state (active) 0V nominal Range 0.0V to 1.0V

* Input is 5V tolerant, Output is 3.3V

1.2.2 Format

9600 baud, 1 start bit, 8 data bits, no parity bit, 1 stop bit
The rise and fall time at 9600 baud should be less than 10us

1.2.3 Bus address

The standard bus address is 2.
It can be factory programmed to any value from 2 to 250.

1.2.4 Message Structure

The Bulkvalidator uses the Standard structure with simple checksum. The Format with CRC is not supported.

For a payload of N data bytes…

[Destination Address]
[No. of Data Bytes]

[Source Address]
[Header]
[Data 1]
...
[Data N]
[Checksum]

Each communication sequence (a command or request for information) consists of 2 message packets structured in the above manner. The first will go from the master device to the slave device and then a reply will be sent from the slave device to the master device. The reply packet could be anything from a simple acknowledge message to a stream of data.

Note that the acknowledge message in cctalk conforms to the above structure in the same way all other messages do. Some protocols use a single byte acknowledge - this is not viewed as secure.

The structure does not alter according to the direction of the message packet. The serial protocol structure does not care who originates the message and who responds to it.

For a simple command or request with no data bytes…

[Destination Address]
[0]
[Source Address]
[Header]
[Checksum]

The acknowledge message is produced by setting the header to zero and having no data bytes…

[Destination Address]
[0]
[Source Address]
[0]
[Checksum]

For more detailed information refer to the official cctalk specification downloadable at www.cctalk.org

2 Commandset overview

Table 2.1: Implemented commands

Header	***Command***	***Group***
255	Factory setup and test	
254	Simple poll	1
249	Request polling priority	C
248	Request status	C
247	Request variable set	C
246	Request manufacturer id	1
245	Request equipment category id	1
244	Request product code	1
243	Request database version	C
242	Request serial number	2
241	Request software revision	2
240	Test solenoids	C
239	Operate motors	B
238	*Test output lines*	*C*
237	*Read input lines*	*C*
236	Read opto states	C
232	Perform self-check	C
231	Modify inhibit status	C
230	Request inhibit status	C
229	Read buffered credit or error codes	C
228	Modify master inhibit status	C
227	Request master inhibit status	C
192	Request build code	1
184	Request coin id	C
4	Request comms revision	2
1	Reset device	2

3 Commandset details

3.1 Header 255 - Factory set-up and test

Transmitted data : <variable>
Received data : ACK or <variable>

This command is reserved for functions needed during the manufacture of a product and is not intended for general use. Every manufacturer can define their own set of functions buried behind this header number.

Commands for testing components
['B']['V'][1][Test ID]
0x10 = Diskmotor off
0x11 = Diskmotor forward
0x12 = Diskmotor reverse
0x13 = change disk calibration value (temporarily)
0x15 = move disk to stop position

0x20 = Trashmotor off
0x21 = Trashmotor forward
0x22 = trashmotor reverse
0x23 = change trashmotor calibration value (temporarily)
0x24 = trashdoor small cycle (trashdoor is opened and closed)
0x25 = trash complete cycle (trashdoor is opened, the disk turns, trashdoor is closed)

3.2 Header 254 - Simple poll

Transmitted data : <none>
Received data : ACK

This command can be used to check that the slave device is powered-up and working. No data is returned other than the standard ACK message and no action is performed. It can be used at EMS (Early Morning Start-up) to check that the slave device is communicating. A timeout on this command indicates a faulty or missing device, or an incorrect bus address or baud rate.

3.3 Header 249 - Request polling priority

Transmitted data : <none>
Received data : [units] [value]

This is an indication by a device of the recommended polling interval for buffered credit information. Polling a device at an interval longer than this may result in lost credits.

[units]
0 = special case
1 = ms
2 = x 10 ms

The bulkvaldator will answer with [2][20] which means 200 ms.

3.4 Header 248 - Request status

Transmitted data : <none>
Received data : [status]

This command reports the status of the BV. Because the BV has non of the documented error states the answer will always be [0]

3.5 Header 246 - Request manufacturer id

Transmitted data : <none>
Received data : "NRI"

3.6 Header 245 - Request equipment category id

Transmitted data : <none>
Received data : "Coin Acceptor"

3.7 Header 244 - Request product code

Transmitted data : <none>
Received data : "BV"

The product code is returned. No restriction on format.

The complete product identification string can be determined by using ‘Request

product code' followed by 'Request build code'.

3.8 Header 243 - Request database version

Transmitted data : <none>
Received data : [calibration database no.]

This command retrieves a database number from 1 to 255 which may be used for remote coin programming.

A database number of 0 indicates remote coin programming is not possible.

3.9 Header 242 - Request serial number

Transmitted data : <none>
Received data : [serial 1] [serial 2] [serial 3] [serial 4]

serial 1 = LSB
decimal = [serial 1] + 256 * [serial 2] + 65536 * [serial 3] + ...

3.10 Header 241 - Request software revision

Transmitted data : <none>
Received data : "mm.ss"

mm = main revision
ss = sub revision

3.11 Header 240 - Test solenoids

Transmitted data : [bit mask]
Received data : ACK

The solenoids are pulsed for 200 ms. The bit mask indicates which solenoids to operate.

[bit mask]
The following bits have been defined for a coin acceptor :
Bit 0 - Accept gate solenoid. 0 = no action, 1 = activate for 200 ms.
Bit 1 – Hold Gate

The slave ACK is returned after the solenoid deactivates (200ms later)

3.12 Header 239 - Operate motors

Transmitted data : [Command] <var data>
Received data : ACK or <var data>

This command is to test motors & speed.
It´s for calibration purpose and not for everyday use.

[Command]
1 = trash cycle 1 (trashdoor is opened, the disk turns, trashdoor is closed)
2 = trash cycle 2 (for future use)
3 = trash cycle 3 (for future use)
10 = set speed (one data byte needed)
data = speed value in percent. 100 = 3 coins/s, 133 = 4 coins/s
NOTE: This changes the speed temporarily. After a reset this value is always 100%.
11 = get speed (one data byte returned)
data = speed value in percent.
Returns the actual speed value.
12 = get pocket time (returns one byte of data)
data = pocket time [in 4 ms steps]
If the disk turns the time from one pocket to the next is measured. This value can be used to determine the disk speed in coins/s. NOTE: only use this command if no coins are in the container, otherwise the returned value is not valid.

3.13 Header 236 - Read opto states

Transmitted data : <none>
Received data : [bit mask]

Checks the optical sensors at the coin exit.

[bit mask]
Bit 0: Coin Present Sensor (1 = active, there is at least one coin within the container)
Bit 1: Trashdoor (1 = open, 0 = closed)
Bit 2: Lower Sensor (Accept & Reject Path)
Bit 3: Upper Sensor (Accept path) (0 = opto clear, 1 = opto blocked)

3.14 Header 232 - Perform self-check

Format (a)
Transmitted data : <none>
Received data : [fault code]

0 = OK
1 = firmware checksum corrupted
2 = Fault on inductive coils (measurement system)
3 = fault on credit sensors
30 = Datablock checksum corrupted
253 = coin jam in measurement system
254 = Disk Blocked (Disk is blocked, device was not able to resolve blockage)
255 = unspecified alarm code

3.15 Header 231 - Modify inhibit status

Transmitted data : [inhibit mask 1] [inhibit mask 2]
Received data : ACK

This command sends an individual inhibit pattern to the bulk validator.
With a 2 byte inhibit mask, up to 16 coins can be inhibited or enabled.

[inhibit mask 1]
Bit 0 - coin 1
...
Bit 7 - coin 8

[inhibit mask 2]
Bit 0 - coin 9
...
Bit 7 - coin 16

0 = coin disabled (inhibited)
1 = coin enabled (not inhibited)

NOTE: This setting is temporary (RAM). After a device reset all coins are inhibited

3.16 Header 230 - Request inhibit status

Transmitted data : <none>

Received data : [inhibit mask 1] [inhibit mask 2]

This command requests an individual inhibit pattern from the bulk validator.

See ‘Modify inhibit status’ for more details.

3.17 Header 229 - Read buffered credit or error codes

Transmitted data : <none>
Received data : [event counter]
[result 1A] [result 1B]
[result 2A] [result 2B]
[result 3A] [result 3B]
[result 4A] [result 4B]
[result 5A] [result 5B]

This command returns a past history of event codes for a coin acceptor in a small data buffer. This allows a host device to poll a coin acceptor at a rate lower than that of coin insertion and still not miss any credits or other events.

The standard event buffer size is 10 bytes which at 2 bytes per event is enough to store the last 5 events.

A new event ripples data through the return data buffer and the oldest event is lost.

For example, consider a 5 event buffer :

result 5A ==> lost
result 5B ==> lost

result 4A ==> result 5A
result 4B ==> result 5B

result 3A ==> result 4A
result 3B ==> result 4B

result 2A ==> result 3A
result 2B ==> result 3B

result 1A ==> result 2A
result 1B ==> result 2B

new result A ==> result 1A

new result B ==> result 1B

An event counter is used to indicate any new events and this must be compared at each poll to the last known value.

event counter - last event counter	Stored in result…
0	-
1	1
2	1, 2
3	1, 2, 3
4	1, 2 ,3, 4
5	1, 2, 3, 4, 5
6+	1, 2, 3, 4, 5 & others are lost

[event counter]
0 (power-up or reset condition)
1 to 255 - event counter

[result A]
1 to 255 - credit
0 - error code

[result B] for credits
0 – Validation Event (valid & enabled coin was measured)
1 – Acception Event (coin was accepted and has exited the validator)

[result B] for error codes
0 = NULL event, no error / previous error solved
1 = unknown coin rejected
2 = inhibited coin rejected
5 = validation timeout
6 = credit sensor timeout
14 = credit sensor blocked
17 = coin going backwards
29 = unexpected acceptance (accept gate forced active/blocked)
30 = unexpected rejectance (accept gate forced inactive/blocked)
255 = unspecified alarm code

The event counter is incremented every time a new credit or error is added to the buffer. When the event counter is at 255 the next event causes the counter to change to 1. The only way for the counter to be 0 is at power-up or reset. This provides a convenient signalling mechanism to the host machine that a major fault has occurred.

3.17.1 Static errors

For static errors like 'credit sensor blocked' there will be a NULL-event after the error is solved.

3.17.2 Coin acception

For each valid and accepted coin the Bulkvalidator will send two events:

1) Validation Event: coin was validated successful and is about to be accepted
2) Acception Event: coin was accepted and has left the validator

The Acception event is for crediting purpose.
The Validation event is only informational and can be discarded. It can be used to have earlier information on the coin that will be accepted to control a subsequent sorter.

Because a coin can fall off the disk after it is measured and signalled there can be a validation event without a subsequent acception event. Therefore only acception events should be used for crediting.

3.17.3 Automatic inhibition

The device will stop accepting coins for the following reasons:

1. The device was not polled (Header 229) for more than 500ms
2. The event buffer is full (contains more than 5 events)

If the BV is enabled it will automatically disable itself because it considers the machine controller to be down.

3.18 Header 228 - Modify master inhibit status

Transmitted data : [XXXXXXX | master inhibit status]
Received data : ACK

[master inhibit status]
Bit 0 only is used.
0 – bulk validator is disabled (disk stops movement)
1 – bulk validator is enabled (disk starts turning, coins will be processed)

NOTE:
After a reset the bulkvalidator is disabled.
In an optional configuration the BV will not stop immediately if it is disabled. It will stop acception immediately but continues turning the disk until the container is empty. At least this will take 3 seconds.

3.19 Header 227 - Request master inhibit status

Transmitted data : <none>
Received data : [XXXXXXX | master inhibit status]

[master inhibit status]
Bit 0 only is used.
0 – bulk validator is disabled (disk stops movement)
1 – bulk validator is enabled (disk starts turning, coins will be processed)

3.20 Header 192 - Request build code

Transmitted data : <none>
Received data : ASCII

The product build code is returned. No restriction on format.

>>TBD<<

3.21 Header 184 - Request coin id

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

Each coin position, for example 1 to 16, is interrogated for an ASCII identifier. This consists of 6 characters representing the coin name.

The identifier is made up as follows…

[C] [C] [V] [V] [V] [I]

CC = Standard 2 letter country code e.g. GB for the U.K. (Great Britain)
VVV = Coin value in terms of the base unit appropriate to that country
I = Mint Issue. Starts at A and progresses B, C, D, E…

The country code for the ‘Euro’, the Common European currency, is ‘EU’.

If the country code is ‘TK’ then a token occupies this position rather than a coin. In this case the VVV field represents a token number in ASCII rather than a value which could change from one jurisdiction to another.

It is possible to have more than one mint issue in circulation at any particular time -

for instance during a transition period from ‘old’ coins to ‘new’ coins. Serial coin acceptors can be programmed with both types and the ‘old’ coins inhibited by the host machine when they officially go out of circulation.

Examples:
EU100A = 1,00 Euro
CA005B = 0,05 Canadian Dollar

3.22 Header 4 - Request comms revision

Transmitted data : <none>
Received data : [1] [4] [4]

3.23 Header 1 - Reset device

Transmitted data : <none>
Received data : ACK

This command forces a soft reset in the slave device. It is up to the slave device what action is taken on receipt of this command and whether any internal house-keeping is done. The action may range from a jump to the reset vector to a forced physical reset of the processor and periph eral devices. This command is worth trying before a hard reset (or power-down where there is no reset pin) is performed.

The slave device should return an ACK immediately prior to resetting and allow enough time for the message to be sent back in full.

The host device should wait an appropriate amount of time after issuing a ‘Reset device’ command before sending the next command.

4 Bulkvalidator Operation

To operate the Bulkvalidator the minimum steps are:

1. appropriate power connection 12..24V (+/- 10%), >= 500mA
2. Enable one or more coins (Header 231)
3. poll the device (Header 229) and process events/errors
4. repeatedly request coin present sensor (CPS) (Header 236)
5. if CPS is inactive proceed with step 3
6. Enable device (Header 228)
7. repeatedly poll the device and process events (Header 229) and where required request CPS state (Header 236)
8. if price is reached or if coin present sensor is inactive and there was no coin validated for at least 3 seconds disable device and proceed with step 3
9. proceed with step 7

For the loops within the 9 steps (polling) we recommend a period of 200 ms.

All other commands can be used to request more information (i.e. Programmed coins). They are not neccessary to operate the Bulkvalidator.