SV ARM mod. ccTalk COIN ACCEPTOR

Operator's manual

Rev. 2.02

Operator's Manual

CE

Alberici

Progettazione e produzione di sistemi di pagamento per l'Intrattenimento, i Lavaggi automatici, e il Vending
Design and manufacture of payment solutions for the Amusement, Gaming, Car-Wash, Laundries, and Vending

Via Cà Bianca, 421 - 40024
Castel San Pietro Terme (BO) - Italy

Tel.: +39.051.944300
Fax.: +39.051.944594

Web: www.alberici.net
E.mail: info@alberici.net

CONTENTS

STORICO REVISIONI			
Revisione n°	Data	Modifica	Note
Creazione v. 2.00	13.10.19	Creazione	Fw: u1.5 A1.0.9
Revisione v. 2.01	21.11.20	Gestione diagnostica e separatore iS Pulse	Fw: u1.5 A1.1.2
Revisione v. 2.02	14.01.22	Inserita gestione errori a Dip-Switch banco SW2	Fw: u1.5 A2.1.1

NOTICE

This manual has been drafted with the utmost care. Nevertheless, it is not possible to guarantee at all times the absolute correspondence of the descriptions contained therein with the actual characteristics of the product.
Alberici S.p.A. declines any and all responsibility towards the User with reference to damages, losses, or claims of third parties, resulting from the use of the product or caused by incorrect interpretations of this manual.
Alberici S.p.A. reserves the right to modify, without prior notice and in any way, any part of this manual and the technical specifications of this product, as part of the continuous pursuit of improvement of its products.

Dear Customer,

we would like to thank you and congratulate you on your choice of the SV modifiable ccTalk electronic coin selector. We are confident that you will appreciate the quality and performance.

Please read this manual carefully to get the most from this product.

1. Package content

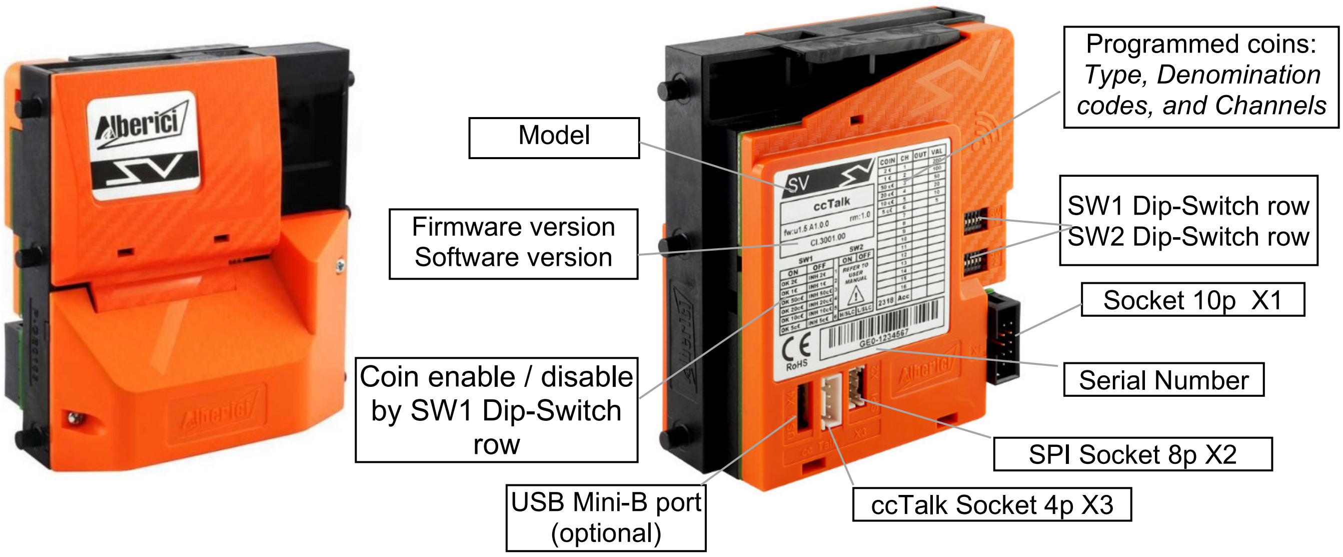
The package contains:

1. The SV modifiable ccTalk coin acceptor, type V;
2. The manual of the product (this manual), or the Data sheet

This product has been packed with the utmost care. If you receive it damaged or incomplete, please notify immediately your findings to the Carrier, and collect it only under conditional signature.

The packing material, wooden pallets included, must be disposed off according to your Country laws and regulations. The product itself, once dismissed, must be disposed off according to the same laws and regulations.

2. Product identification



3. Product description

The SV high-speed coin acceptor can process up to 10 coins per second. Its architecture has been designed specifically for the ccTalk protocol. The ccTalk signals sent to the machine CPU are channeled through the classic 4-pole TTL connector (ccTalk protocol: dip-switch 1 in SW1 = ON).
The SV is equipped with the latest generation 32-bit ARM microprocessor, with 128 KB flash memory, “automotive” model, whose performance is stable even in difficult environmental conditions. Its innovative identification technology, basing on wide spectre 2 magnetic sensors and 2 optical detector pairs, ensure excellent performance in identifying coins and discriminating counterfeits. Anti-fraud is limited to coin recognition, since the SV coinmech is designed for casino machines (and not for the unattended street market units), where the need is fast counting and negligeable security, since casino venue are continuously monitored by staff and videocameras.
The construction, with non-deformable HQ polymer resins, ensures strength and durability, helping to ensure constant reading conditions regardless of the environment in which the coin mechanism is working.
Using the Dip-Switches of bank SW1, coins to be accepted can be disabled or enabled. The acceptance tolerance can be changed dip-switch 6 of SW2 row.
Thanks to its SPI socket (X2), it can be interfaced with intelligent peripheral units, such as the iS coin sorters.

4. Technical Specifications

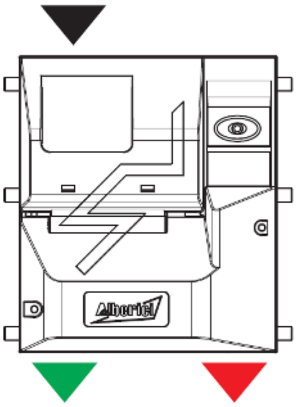
- > 32-bit top-performing ARM microcontroller with 36 KB Flash-memory, highly immune to magnetic interference and to environmental conditions.
- > Two magnetic sensors and one optic detector combine to guarantee the most accurate capacity of discrimination. Data-digitizer interface copes at best with bi-metal and magnetized coins.

The SV coin acceptor operates by modifiable ccTalk protocol.

It is available in the following mechanical configuration:
V = rejected coin sorted from bottom rear slot, accepted coin sorted from bottom front slot **I =** rejected coin sorted from bottom front slot, accepted coin sorted from bottom rear slot

Technical specs	
Mechanical features	
Format Size	3½" standard
Weight	88,5 x 102 x 51 mm
	161,60 g
Electrical features	
Supply voltage min.	12 V DC (min. 10 V DC)
Supply voltage max.	24 V DC (max. 26 V DC)
Power consumption:	
in acceptance	max. 400 mA(*30 ms*)/100 mA
in reading	≤100 mA
in Standby	≤50 mA
Output type	Open collector Darlington
Saturation output voltage	≤1 V
Output voltage max.	50 V (Active Low)
Output current max.	250 mA
Input trigger voltage min.	3 Vdc / 24 Vdc (Active High)
Input voltage max	50 V
Input impedance	≈55 kfi
Coin acceptance	
Coin channels number	16
Coin diametre min.	16 mm
Coin diametre max.	32 mm
Coin thickness	1 to 3,4 mm
Speed	3 coins/sec. (V) – 4 coins/sec. (S, K, I)
Communication mode	Modifiable ccTalk (USB-B port on request)
Timing data	
Power-up recovery time	≤ 200 ms
Wake-up recovery time	≤ 50 ms
Ambient conditions	
Working ambient temperature	0°C to 60°C
Storage temperature	- 30°C to 70°C
Humidity	up to 75% (*non condensing*) for standard up to 95% for tropicalized version

EMC performance
This product is compliant with EN55014-1 and EN55014-2 test specification

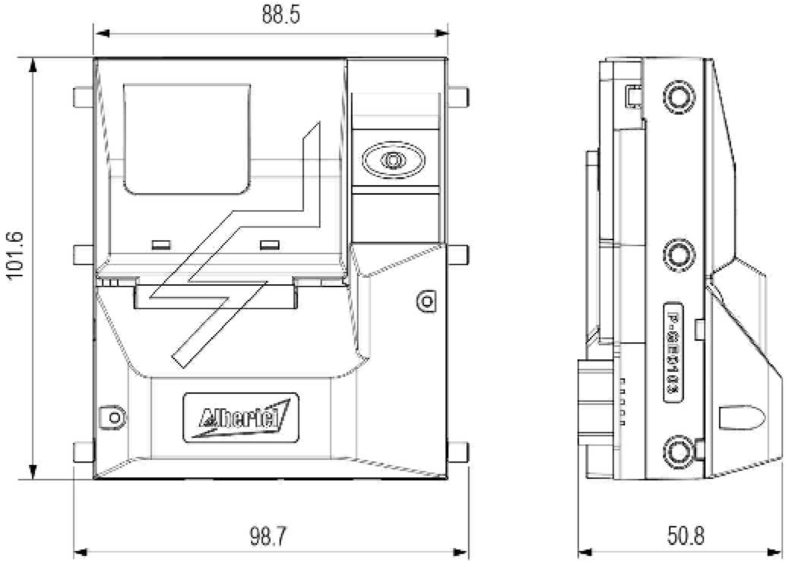


⬛	INGRESSO MONETA	COIN IN
🟩	MONETA VALIDA	VALID COIN
🟥	MONETA RIFIUTATA	REJECTED COIN

5. Installation

DIMENSIONS

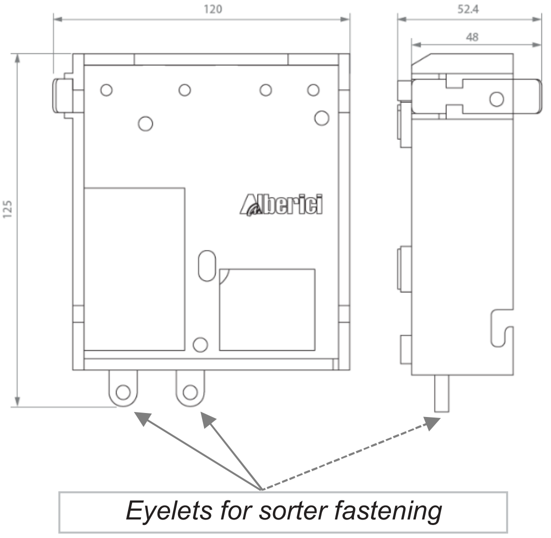
The SV is a 3,5" coin acceptor. It is interchangeable with the coin acceptors of the same type (compatible entries and exits of the coins) available on the market.



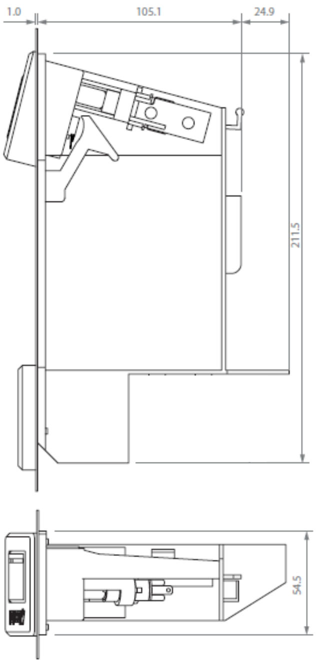
PAY ATTENTION! The coin selector must be installed 90°-95° with respect to level plane. Because of the built-in advanced security system, it is essential that the whole coin path gets not hindered. Alberici is not responsible for any malfunctioning due to lack of compliance with such recommendations.

MOUNTING of the "V"-TYPE COIN ACCEPTOR in C-CRADLE or in "SPANISH" CRADLE

The "V"-type (or "video") coin acceptor must be housed into the "C"-cradle (SG-7000) or into "Spanish" type cradles (SG-3001 / SG-4001). The preferred drop-in coin slot can be fitted onto the upper side of the C-cradle, or else the electronic rejecting device when the coin acceptor must be installed far from the entry slot.



Eyelets for sorter fastening

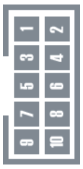


The sorter can be fastened at the bottom of the C-cradle, by tightening two screws to the two preset eyelets.

Connections

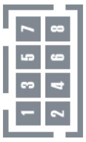
The selector connects to other peripherals and to the main board by the following sockets:

X1. Power supply and Pulse interface
It is used to connect the coin sorters in Pulse mode.
When the coin acceptor is connected to an “intelligent” sorter (e.g. iS1), the pins 7-10 are also used to monitor that the coins are sorted via the correct direction path.



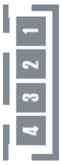
nr.	Description
1	Gnd
2	+12-24 Vdc
3	Out 5 / sorter coil B control line (dir. 2)
4	Out 6 / sorter coil A control line (dir. 3)
5	N.C.
6	+12-24 Vdc (sorter) / In 1 (inhibit)
7	Coin inlet to sensor (O.C. npn)
8	Coin exit from sensor (O.C. npn)
9	Optics check dir. 2 (O.C. npn)
10	Optic check dir. 3 (O.C. npn)

X2. SPI Display or intelligent peripherals.
Socket X2 (8 pins) is meant for display connection.
Displays with SPI or I²C bus protocols are supported (see DISPLAY AD01 SPI SERIAL PROTOCOL at page 20).



The acceptor must be preset by Alberici to host the display (AA-0410-0900). Display functions can then be modified by the ‘AlbericiCoinSelector’ programming software.

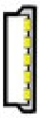
X3. CCTALK. The X3 4-p socket is for serial **ccTalk®** communication with the main board. The protocol is preset for “slave mode”operation,and is described in Chapter 6. This socket can be used to modify the programmed configuration of the selector, by PC and Alberici free programming software.



nr.	Description
1	Data
2	Gnd
3	N.C.
4	+12 Vdc

X4. USB Mini-B (option). The USB port allows to drive the acceptor directly the USB port of a PC *Mother Board. The USB port provides only the communication channel: the communication itself will be run by serial ccTalk® protocol, that must therefore be implemented in the Host software (see Chapter 6).*

To provide USB operation, the acceptor must be +12V powered through pins 2 and 4 in socket X3. *This socket will be enabled only after installation of the USB drivers on the Host (See ‘USB installation notice’ below).*



USB pre-activation must be requested at the time of order.

USB installation notice

- Install the drivers in the Host using the relevant executable file (i.e.: for Windows 7 64bit: “VCP_V1.5.0_Setup_W7 x64_64bits.exe”), available in the Alberici web site (FG coin acceptor page). Win 10 includes such drivers by default.
- *Only after* the drivers ("STMicroelectronics Virtual COM Port") are installed, connect the coin acceptor through the USB;
- Windows OS will automatically start the installation of the driver;
- At the end of the installation process, a dialogue window will confirm "STMicroelectronics Virtual COM Port (COMxx)”.
- It is now possible to connect +12V power to the pins 2 (GND) and 4 (+) of the X3 socket.

PAY ATTENTION! Using a standard ccTalk interface circuit, the master (i.e. host mother board) receives the echo back from the interface, and NOT from the slave (i.e. coin selector).
However, USB port is NOT a standard ccTalk interface circuit: no TX echo will be sent to the master!
Therefore, the master must be programmed so as not to expect to receive the echo, or communication will end.

DISABLE DENOMINATIONS BY THE SW1 ROW

EACH PROGRAMMED CHANNEL CONTAINS ONE DENOMINATION. TO DISABLE ACCEPTANCE OF ANY COINS, MOVE TO OFF POSITION THE DIP-SWITCH CORRESPONDING TO THE CHANNEL IN WHICH THE COIN IS PROGRAMMED (see grid below). THEN SWITCH OFF AND ON AGAIN.

SW1 Dip-Switch Row	DS1	DS2	DS3	DS4	DS5	DS6
ON	Enable CH1 (0,05€)	Enable CH2 (0,10€)	Enable CH3 (0,20€)	Enable CH4 (0,50€)	Enable CH5 (1,00€)	Enable CH6 (2,00€)
OFF	Disable CH1 (0,05€)	Disable CH2 (0,10€)	Disable CH3 (0,20€)	Disable CH4 (0,50€)	Disable CH5 (1,00€)	Disable CH6 (2,00€)

SETTINGS BY THE SW2 DIP-SWITCH ROW

SW2 Dip-Switch Row	DS2 OFF	DS2 ON	DS4 OFF	DS4 ON	NOTES
DS1 OFF	**Default: modifiable CCTALK**		Error events disabled (only events of accepted coins available)	Error events enabled (replying to ccTalk "Header 229 -Read buffered credit or error codes")	
DS1 ON					
DS3 OFF					
DS3 ON	Denominations top-down order in Channels (ch6=2€; ch5=1 €; … ch2=0,10€; ch1=0,05€)				
DS5 OFF	(U.C.S.)				Discrimination as per dip-switch 6
DS5 ON	(AVENTADOR)				Special discrimination of 1€ / 50c
DS6 OFF	STANDARD Discrimination (L / SLC)				
DS6 ON	HIGH Discrimination (H / SLC)				

HOW TO TEACH (NEW) COINS BY THE DIP-SWITCHES

1) Take note of the positions of the dip-switches.
2) Power off the coin acceptor.
3) Set all the dip-switches in SW1 row to OFF.
4) Move the dip-switch 2 of the SW2 row to ON.
5) Power up the coin acceptor: one single coil activation will confirm the start of the auto-programming mode.
6) In SW1 row, move the dip-switch corresponding to the channel in which the coins will be taught to ON .
7) Drop 10 such coins into the acceptor: double coil activation will confirm that the coins have been programmed.
8) Power off the coin acceptor and set the dip-switches according to the desired operation.

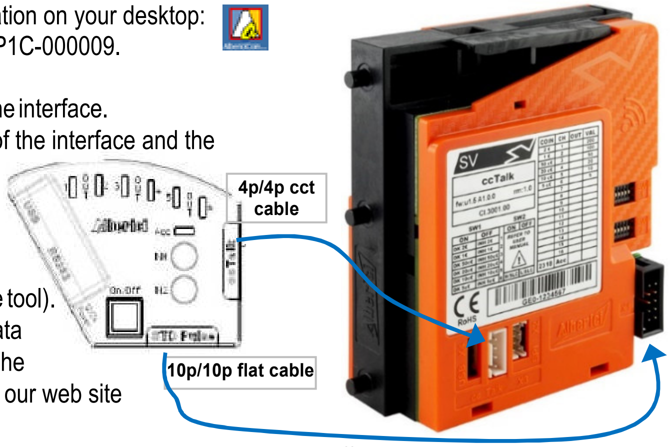
CANCEL THE LAST SELF-TEACHED CONFIGURATION

1) Take note of the positions of the dip-switches.
2) Power off the coin acceptor.
3) Set all the dip-switches in SW1 row to OFF.
4) Move the dip-switch 2 of the SW2 row to ON.
5) Set to ON both dip-switches 1 and 6 of the SW1 row.
6) Power up the coin acceptor: after few seconds, triple coil activation will confirm that the coins/channels that have been manually programmed have been cancelled, and that the previous configuration as been restored.
7) Power off the coin acceptor and set the dip-switches according to the desired operation.

PROGRAMMING BY PC SOFTWARE

THIS IS DONE THROUGH THE K-P1C-000009 PROGRAMMING KIT AND THE "Programming sw 66 v. 2 - Alberici Coin Selector" SOFTWARE.
THE SOFTWARE AND ITS MANUAL, CONTAINING INSTRUCTIONS FOR KIT CONNECTIONS AND PROGRAMMING PROCEDURES, ARE BOTH AVAILABLE FREE FOR DOWNLOAD ON OUR WEB SITE www.alberici.net.

1 Download the application "Programming sw 66 v. 2 - AlbericiCoinSelector.exe" and the drivers for the programming interface tool from our web site http://www.alberici.it, "Downloads" section at the bottom of the coin acceptor page).
2 Install the drivers into your PC, then install the application on your desktop:
3 Make use of the programming-testing-power tool K-P1C-000009.
4 Make sure that voltage supply is off.
5 Connect the USB port of the PC to the USB A port of the interface.
6 Connect the 4pin cable between the ccTalk output of the interface and the 4-pin socket (X3) of the acceptor.



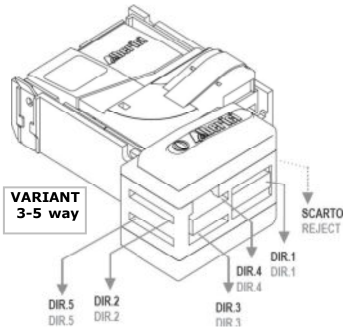
7 Connect the 10pin flat cable between the STD output of the interface and the 10pin socket (X1) of the acceptor:
8 Power the selector (green pushbutton on the interface tool).
9 Launch the application and modify the acceptor's data appearing on the screen according to your needs. The Instruction Manual is available on the same page of our web site from where you have downloaded the software.
10 Download the new configuration to the coin acceptor, switch power supply to selector off and then on again.
11 Test that operation corresponds to the new configuration programmed.
12 Switch power off and disconnect the selector from the PC USB port.

DRIVING THE NL3, NS3, VARIANT, iS2, iS3, and iS4 SORTERS

To attribute the proper sorting commands, please refer to the output paths shown below:

SORTING COMMANDS

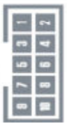
CCTALK SORTERS		PULSE SORTERS
Dir 1 = cctalk 01	activates	Dir 1 = no OUT
Dir 2 = cctalk 02	activates	Dir 2 = OUT 5
Dir 3 = cctalk 03	activates	Dir 3 = OUT 6
Dir 4 = cctalk 06	activates	Dir 4 = OUT 4 + OUT 5
Dir 5 = cctalk 07	activates	Dir 5 = OUT 4 + OUT 6



SORTERS SOCKETS

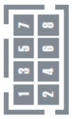
10 PIN PULSE (X1)

GND	1	2	+24 V
OUT5	3	4	OUT6
IN2/OUT7	5	6	IN1/INHIBIT
OUT1	7	8	OUT2
OUT3	9	10	OUT4



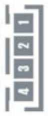
8 PIN SPI (X2)

DATA OUT	1	2	CLOCK
SELECT	3	4	DATA IN
GND LOGIC	5	6	N.C.
GND	7	8	+12/24 Vdc



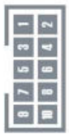
4 PIN CCTALK (X3)

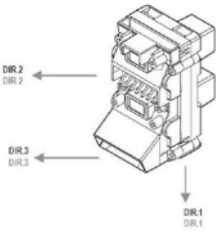
DATA	1
GND	2
NC	3
+24 V	4

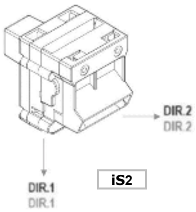


10 PIN CCTALK

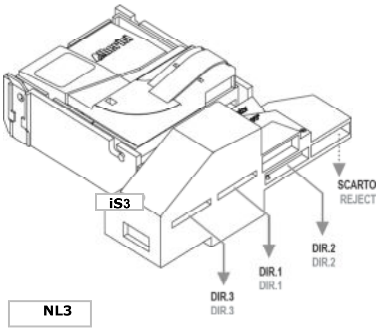
DATA	1	2	NC
NC	3	4	GND
NC	5	6	NC
+24 V	7	8	GND
NC	9	10	+24 V

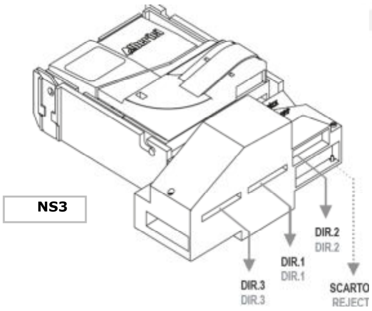




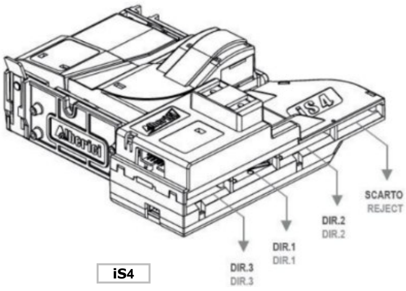


SPI mode is available only on sorters iS3 and iS4, and it is supported only by the AL66 FG and SV coin acceptors



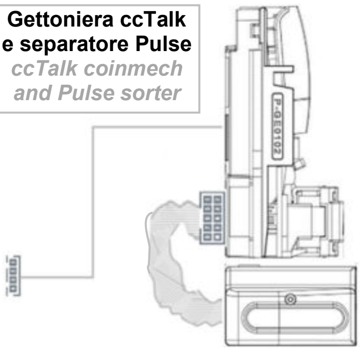


**SORTERS INCLUDE ONE BUILT-IN 10P FLAT CABLE.
ONE 4P CABLE IS ADDED TO CCTALK SORTERS.**



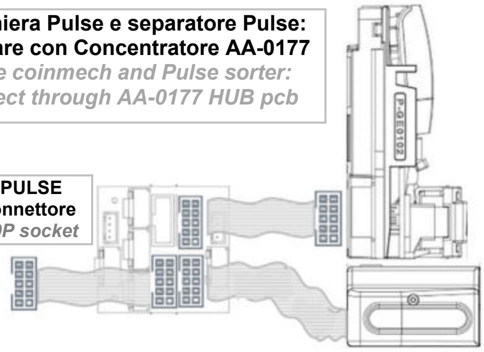
Gettoniera ccTalk e separatore Pulse
ccTalk coinmech and Pulse sorter

HOST CCTALK CPU: connettore 4P / *4P socket*

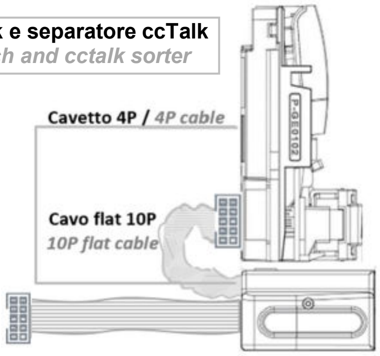


Gettoniera Pulse e separatore Pulse: collegare con Concentratore AA-0177
Pulse coinmech and Pulse sorter: connect through AA-0177 HUB pcb

HOST PULSE CPU: connettore 10P / *10P socket*



Gettoniera ccTalk e separatore ccTalk
ccTalk coinmech and cctalk sorter



HOST CCTALK CPU: connettore 10P / *10P socket*

6. CCTALK Communication Protocol

The new generation of coin selectors AL55 or AL66 use **cctalk®** communication protocol. This protocol was developed by company Emulator M (ex. Coin Controls) to enable connection of different peripheral devices[1] in small networks. Protocol is mostly used in gaming and casino machines but it can be implemented in any other types of machines that use same type of devices. It is a public protocol and free to use. Literature can be found at: www.cctalk.org.

The communication protocol of ALBERICI coin selectors AL55/66 comply to generic specification 4.4

The following ccTalk commands (Headers) are implemented for the serial operation of the coin acceptor:

254- Simple poll
253- Address poll (broadcast)
252- Address clash (no broadcast)
252- Address clash (broadcast)
249- Request polling priority
248- Request status
246- Request manufacturer id
245- Request equipment category id
244- Request product code
242- Request serial number
241- Request software revision
240- Test solenoids
237- Read input lines
236- Read opto states
232- Perform self test
231- Modify inhibit status
230- Request inhibit status
229- Read buffered credit or error codes
228- Modify master inhibit status

227- Request master inhibit status
226- Request insertion counter
225- Request acceptance counter
210- Modify sorter paths
209- Request sorter paths
197- Calculate ROM checksum
196- Request creation date
195- Request last modification date
194- Request reject counter
193- Request fraud counter
192- Request build code
188- Request default sorter path
184- Request coin id
170- Request base year
4- Request comms revision
3- Clear comms status variables
2- Request comms status variables
1- Reset device

6.1 Communication specifications

Serial communication was derivated from RS232 standard.
It is low data rate NRZ (***N**on-Return to **Z**ero*) asynchronous communication with:
Baud rate 9600, 1 start bit, 8 data bits, no parity, 1 stop bit.
RS232 handshaking signals (*RTS, CTS, DTR, DCD, DSR*) are not supported.
Message integrity is controlled by means of checksum calculation.

6.1.1 Baud rate

The baud rate of 9600 was chosen as compromise between cost and speed.
Timing tolerances is same as in RS232 protocol and it should be less than 4%.

6.1.2 Voltage level

To reduce the costs of connections the "Level shifted " version of RS232 is used.
The idle state on serial connector is 5V, and active state is 0V.

Mark state (*idle*)	+5V nominal	from 3.5V to 5V
Space state (active)	0V nominal	from 0.0V to 1.0V

Data I/O line is "open collector" type, so it is possible to use device in systems with different voltage.

6.1.3 Connection

The connection of Coin selector at network is achieved by means of 4 pole JST connector (*standard type 7*). The connector is used for power supply and communication as well.
For schematics and connector appearance see image1.

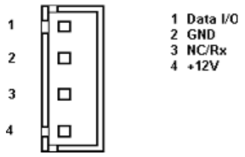


Image 6.1 communication connector

Recommended peripheral connector is:
JST B 4B-XH-A with crimping contacts SXH-001T-P0.6

[1] Coin selectors, Hoppers (*pay out devices*), Banknote readers etc.

6.2 Message structure

Each communication sequence consists of two message string.
Message string in case of simple checksum use is structured as follows:

[Destination address]
[Nr. of data bytes]
[Source address]
[Header]
[Data 1]
...
[Data n]
[Checksum]

There is an exception of message structure when device replies to instruction Address poll and Address clash[2]. The reply consists of only one byte representing address delayed for time proportional to address value. For CRC checksum case format is:

[Destination address] [Nr. of data bytes]
[CRC 16 LSB]
[Header]
[Data 1]
...
[Data n]
[CRC 16 MSB]

6.2.1 Address

Address range is from address 0 to address 255. Address 0 is special case or so called “broadcast” address and address 1 is default host address.
Table 6.1 shows the recommended address values of different devices.

Device category	Address	Add. addr.	Note
Coin Acceptor	**2**	**11 - 17**	**Coin validator, selector**
Payout	3	4 - 10	Hopper
Reel	30	31 - 34	
Bill validator	40	41 - 47	Banknote reader
Card Reader	50		-
Display	60		Alphanumeric LC display
Keypad	70		-
Dongle	80	85 - 89	Safety equipment
Meter	90		Replacement for el.mec. counters
Power	100		Power supply
Printer	110		Ticket printing
RNG	120		Random Number Generator

Table 6.1 Standard address for different types of devices

Address for ALBERICI coin selectors AL55/66 is factory set at value **2**.
User can change the default address using MDCES instructions: **Address change** or **Address random**.

6.2.2 Number of data byte

Number of data byte in each transfer could be from 0 to 252.
Value 0 means that there are no data bytes in the message, and total length of message string will be 5 bytes. Although theoretically it will be possible to send 255 bytes of data because of some limitations in small micro controllers the number is limited to 252[3].

[2] For details see cctalk44-2.pdf, Address poll

6.2.3 Command headers (*Instruction*)

Total amount of possible cctalk command header is 255 with possibility to add sub-headers using headers 100, 101, 102 and 103.

Header 0 stands for **ACK** (***ack**nowledge*) replay of device to host. **Header 5** stands for **NAK** (***N**o **ack**nowledge*) replay of device to host. **Header 6** is **BUSY** replay of device to host.

In all three cases no data bytes are transferred. Use of ACK and NAK headers are explained later on, for each specific message transfer.

Commands are divided in to several groups according to application specifics:

- Basic general commands
- Additional general commands
- Commands for Coin acceptors
- Commands for Bill validators
- Commands for Payout mechs
- MDCES commands.

6.2.4 Data

There is no limitation in use of data formats. Data could be BCD (***B**inary **C**oded **D**ecimal*)numbers, Hexa numbers or ASCII strings. Interpretation as well as format is specific to each header use, and will be explained in separate chapter.

6.2.5 Checksum

Message integrity during transfer is checked by use of simple zero checksum calculation.

Simple checksum is made by 8-bit addition (modulus 256) of all the bytes in the message. If message is received and the addition of all bytes are non-zero then an error has occurred[4].

For noisy environment or higher security application it is possible to use more complex, 16-bit CRC CCITT checksum based on a polynomial of:

$\mathbf{x^{16} + x^{12} + x^{5} + 1}$ and initial value of CRC register **0x0000**.

Coin selectors AL55/66 use simple checksum, but they can be set to operate by CRC-16 checksum on customer demand.

6.3 Timing specifications

The timing requirements of cctalk are not very critical but there are some important recommendations.

6.3.1 Time between two bytes

When receiving bytes within a message string, the communication software must wait up to **50 ms** for next byte if it is expected. If time out occurs, the software should reset all communication variables and get ready to receive next message. The interbyte delay during transition should be ideally **less than 2 ms** and **not greater than 10 ms**.

6.3.2 Time between command and reply

The time between command and reply is dependent on application specific for each command. Some commands return data immediately, and maximum time delay should be within **10 ms**.

Others commands that must activate some actions in device, may return reply after the action is finished[5].

6.3.3 Start-up time

After the power-up sequence coin selector should be ready to accept and reply to a cctalk message within time period of **less than 250 ms**.

During that period all internal check-up and system settings must be done, and coin acceptor should be able to recognize and accept coins.

[3] 252 bytes of data, source address, header and checksum (total of 255 bytes)

[4] See Error handling

[5] I.e., more than 100 milli sec for solenoid testing.

6.4 Error handling

If slave device receives the message with bad checksum or missing data no further action is taken and receive buffer will be cleared.

Host software should decide to re-transmit message immediately or after a fixed amount of time. In case when host receive message with error it has same options.

6.5 Command headers

Command header set, that host could use in communication with coin selectors AL55 and AL66 is given in the table 6.2. Command headers are divided in to 3 different groups:

- 6.5.1 Common command headers
- 6.5.2 Coin acceptor command headers
- 6.5.3 MDCES command headers
- 6.5.4 Alberici specific commands

The following ccTalk commands are implemented for the serial operation of the coin acceptor:

254-Simple poll
253-Address poll (broadcast)
252-Address clash (no broadcast)
252-Address clash (broadcast)
249- Request polling priority
248- Request status
246- Request manufacturer id
245- Request equipment category id
244- Request product code
242- Request serial number
241- Request software revision
240- Test solenoids
237- Read input lines
236- Read opto states
232- Perform self test
231- Modify inhibit status
230- Request inhibit status
229- Read buffered credit or error codes
228- Modify master inhibit status

227- Request master inhibit status
226- Request insertion counter
225- Request acceptance counter
210- Modify sorter paths
209- Request sorter paths
197- Calculate ROM checksum
196- Request creation date
195- Request last modification date
194- Request reject counter
193- Request fraud counter
192- Request build code
188- Request default sorter path
184- Request coin id
170- Request base year
4- Request comms revision
3- Clear comms status variables
2- Request comms status variables
1- Reset device

Table 6.2 cctalk instruction header list

6.5.1 Common command headers

Common commands are used in all type of devices to detect their presence on cctalk network or to describe them. Information like: manufacturer or product type id, serial number, different settings etc. are transmitted to host.

6.5.1.1 Command 254 [hexFE], Simple poll

The fastest way for host to detect all attached devices in cctalk network.

Addressed device-coin selector replies with ACK (*Acknowledge*).

If within predicted amount of time Coin selector does not reply, coin selector is probably not connected, powered or simple not working properly.

Message format is:

Host sends: **[Dir] [00] [01] [FE] [Chk]**

Coin s. replies: **[01] [00] [Dir] [00][Chk]**

As coin selector default address is 2, example of message string is:

Host sends: **[02] [00] [01] [FE] [FF]**

Coin s. replies: **[01] [00] [02] [00] [FD]** ACK message

....

6.5.1.2 Command 246 [hexF6], Request manufacturer ID

Coin selector replies with ASCII string representing manufacturer name.
Message format is:
Host sends: **[Dir] [00] [01] [F6] [Chk]**
Coin s. replies: **[01] [Nr.b] [Dir] [00][a1] [a2]...........[an] [Chk]**
Nr. b is number of data bytes-characters sent by coin selector, and a1 to an are ASCII characters. For **Alberici** coin selector example of message string is:
Host sends: **[02] [00] [01] [F6] [07]**
Coin s. replies: **[01] [08] [02] [00] [41][6C][62][65][72][69][63][69] [DA]**

6.5.1.3 Command 245 [hexF5], Request equipment category ID

Reply to command header is standardized name for coin selectors, coin validators or coin mechs. Coin selector replies with ASCII string of characters representing standardized name for that type of device **Coin Acceptor**.
Message format is:
Host sends: **[Dir] [00] [01] [F5] [Chk]**
Coin s. replies: **[01] [0D] [Dir] [00] [43][6F][69][6E][20][41][63][63][65][70][74][6F][72] [Chk]**
Number of data byte is always 13, hex [0D].
Example of message string for coin selector(*address* 2) is:
Host sends: **[02] [00] [01] [F5] [08]**
Coin s. replies: **[01] [0D] [02] [00] [43][6F][69][6E][20][41][63][63][65][70][74][6F][72] [16]**

6.5.1.4 Command 244 [hexF4], Request product code

Coin selector replies with ASCII string of character, representing the factory type of coin selector.
For ALBERICI coin selectors of new generation possible response will be: **CS PCM 01**

In special version for Italian gambling machines response is always AL06V-c .
Host sends: **[Dir] [00] [01] [F4] [Chk]**
Coin s. replies: **[01] [07] [Dir][00] [a1][a2].........[a7] [Chk]**
Number of data bytes sent by coin selector is 6 or 7, hex [07].
Example of message string for coin selector(*address* 2) type **AL06V-c** is:
Host sends: **[02] [00] [01] [F4] [09]**
Coin s. replies: **[01] [07] [02] [00] [41][4C][30][36][56][2D][63] [1D]**

6.5.1.5 Command 242 [hexF2], Request serial number

Coin selector replies with three-byte serial number. Message format is:
Host sends: **[Dir] [00] [01] [F2] [Chk]**
Coin s. replies: **[01] [03] [Dir] [00] [Serial 1 - LSB] [Serial 2] [Serial 3 - MSB] [Chk]**
Serial 1 – first data byte sent is LSB of serial number.
Example of message string for coin selector(*address* 2) with serial number: **1234567** (hex [BC][61][4E]) is:
Host sends: **[02] [00] [01] [F2] [0B]**
Coin s. replies: **[01] [03] [02] [00] [4E][61][BC] [8F]**

6.5.1.6 Command 241 [hexF1], Request software revision

Coin selector return ASCII string of character representing software version and revision. Message format is:
Host sends: **[Dir] [00] [01] [F1] [Chk]**
Coin s. replies: **[01] [Nr.b] [Dir] [00][a1] [a2].......[an] [Chk]**
Number of data bytes in ASCII string is not limited and each producer has its own system of labelling.
Example of message string for coin selector(*address* 2) is:
Host sends: **[02] [00] [01] [F1] [0C]**
Coin s. replies: **[01] [09] [02] [00] [75][31][2E][30][20][70][31][2E][30][2E][30] [71]**

Coin selector replies is '**u1.0 p1.0.0**'.
ALBERICI coin selectors has program firmware label divided in two parts.
First label **u** is for protected FLASH memory program(*monitor program*) revision.
First digit is for major changes and second for minor changes. In this case it is **u1.0**.
Second label is revision of main program FLASH memory.
Main program software revision labelling use 3 digits. First most significant digit is for major software changes, second is for minor software changes and third for "bug" correction. In this case it is **u1.0.0**.

………………………………

6.5.1.7 Command 197 [hexC5], Calculate ROM checksum

Coin selector replies with four bytes of micro controller internal memory checksum. First two bytes are program ROM CRC and the second is data EEPROM CRC. Any changes in program or data will change the reply of coin selector. Message format is:
Host sends: **[Dir] [00] [01] [C5] [Chk]**
Coin s. replies: **[01] [4] [Dir] [00] [CRC1-H][CRC1-L] [CRC2-H] [CRC2-L] [Chk]**
Example of message string for coin selector(*address* 2) is:
Host sends: **[02] [00] [01] [C5] [38]**
Coin s. replies: **[01] [04] [02] [00] [D9][2A][7E][79] [96]**

…………….

6.5.1.8 Command 192 [hexC0], Request build code

Coin selector replies with ASCII string of character representing its hardware version and revision. Suppose revision of printed circuit board for coin selectors AL55/66 as:**AL66 V1.0**.
Message format is:
Host sends: **[Dir] [00] [01] [C0] [Chk]**
Coin s. replies: **[01] [Nr.b] [Dir] [00] [a1] [a2]……. [an] [Chk]**
Example of message string for coin selector(*address* 2) is:
Host sends: **[02] [00] [01] [C0] [3D]**
Coin s. replies: **[01] [09] [02] [00] [41][4C][2D][30][35][20][56][35][30] [FA]**

………………………………

6.5.1.10 Command 4 [hex04], Request comms revision

Coin selector replies with three-byte data information about level of cctalk protocol implementation, major and minor revision. Message format is:
Host sends: **[Dir] [00] [01] [04] [Chk]**
Coin s. replies: **[01] [03] [Dir] [00] [Level] [Mag.rev.] [min. rev.] [Chk]**
Example of message string for coin selector(*address* 2) with level of implementation **1**, cctalk protocol issue **4.4** is: Host sends: **[02] [00] [01] [04] [F9]**
Coin s. replies: **[01] [03] [02] [00] [01][04][04] [F1]**

6.5.1.11 Command 3 [hex03], Clear comms status variables

After acceptance of command header 3, coin selector clears all three bytes of communication errors counters and replies with ACK message. Message format is:
Host sends: **[Dir] [00] [01] [03] [Chk]**
Coin s. replies: **[01] [00] [Dir] [00] [Chk]** ACK message
Example of message string for coin selector(*address* 2) is:
Host sends: **[02] [00] [01] [03] [FA]**
Coin s. replies: **[01] [00] [02] [00] [FD]** ACK message

6.5.1.12 Command 2 [hex02], Request comms status variables

Coin selector replies with three-byte data representing communication errors.
First byte is received time out counter, second byte is number of ignored receive bytes[6] and third byte is number of checksum errors. Message format is:
Host sends: **[Dir] [00] [01] [02] [Chk]**
Coin s. replies: **[01] [03] [Dir] [RxErr1] [RxErr2] [RxErr3] [Chk]**
Example of message string for coin selector(*address* 2) with no errors is:
Host sends: **[02] [00] [01] [02] [FB]**
Coin s. replies: **[01] [03] [02] [00] [00] [00] [00] [FA]**

[6] Number of receive buffer overflow bytes.

6.5.1.13 Command 1 [hex01], Reset device

After acceptance of Reset command, coin selector execute software reset and clear all variables in RAM or set them at default value, including different counters and credit buffer. ACK message is sent before reset of coin selector. Host software must set again:

- inhibit state
- sorter path
- master inhibit (*if necessary*)

Message format is:
Host sends: **[Dir] [00] [01] [01] [Chk]**
Coin s. replies: **[01] [00] [Dir] [00] [Chk]** ACK message
Example of message string for coin selector(*address* 2) is:
Host sends: **[02] [00] [01] [01] [FC]**
Coin s. replies: **[01] [00] [02] [00] [FD]** ACK message
Host software must wait at least **100 ms**, to continue communication with coin selector after reset instruction!

6.5.2 Coin acceptor specific command headers

Coin selectors use some specific commands, mostly for control of coin input, acceptance and direction[7].
Some commands are shared with other device like banknote reader or payout device.

6.5.2.1 Command 249 [hexF9], Request polling priority

Basic principle of detecting credit input or eventual errors from coin selector is sequential polling[8]. Coin selectors due to differences in mechanical and electrical construction has different acceptance speed. All events are registered in memory buffer with limited size[9] . To avoid credit loss, host must read coin selector credit buffer within limited time period. Coin selector has internal mechanism to block the coin acceptance and registration of all events if polling time elapse.
For ALBERICI coin selector acceptance speed is from 3 to 4 coins per second[10].
Considering that it is possible to register 5 events in the buffer, the adequate polling time will be about 1 sec. Because of necessity to register even “close” and non-accepted coins polling time must be even shorter.
For ALBERICI coin selectors AL55/66 using cctalk interface, poll time is set to 500 ms.
Coin selectors that use standard 10 pole interface are not necessary to poll.
In that case polling time unit is set to 0(*no polling*)!
Minimum time for polling must not be shorter than overall message time[11].
Coin selector replies to command with two bytes of data. First byte is poll time unit and second is polling time value[12].
Message format is:
Host sends: **[Dir] [00] [01] [F9] [Chk]**
Coin s. replies: **[01] [01] [Dir] [Time] [Chk]**
Example of message string for coin selector(*address* 2) is:
Host sends: **[02] [00] [01] [F9] [04]**
Coin s. replies: **[01] [02] [02] [00] [02] [32] [C7]**
First byte **02** is unit **x10ms** , and second byte is time value **hex32 = 50**.
Polling time is calculated as: **T = 10 x 50 = 500 ms**

6.5.2.2 Command 248 [hexF8], Request Status

ALBERICI coin selectors has no additional COS[13] and return mechanism.
Response to that command is always hex**[00]**, coin selector Ok.
Example of message string for coin selector(*address* 2) is:

Host sends: **[02] [00] [01] [F8] [05]**
Coin s. replies: **[01] [01] [02] [00] [00] [FC]**

…………………………….

[7] Sorter control commands
[8] Reading memory buffer from coin selector
[9] Five stage double byte memory buffer
[10] Dependant on mechanical type of coin selector (K, S type is faster) and coin
[11] For coin selector with replying time 2 ms and byte gap 1 ms it is 38 ms
[12] For details see, cctalk44-2.pdf
[13] **C**oin **O**n **S**tring

6.5.2.4 Command 240 [hexF0], Test solenoids

Host sends one byte mask to determinate which solenoid must be tested.
Coin selector accept gate solenoid or sorter solenoid will be switched on for period of 100 ms and after that, ACK message will be transmitted. Message format is:
Host sends: **[Dir] [01] [01] [F0] [Mask.] [Chk]**
Coin s. replies: **[01] [00] [Dir][00] [Chk]** ACK
Example of message string for coin selector(*address 2*) acceptance gate test is:
Host sends: **[02] [01] [01] [F0] [01] [0B]**
Coin s. replies: **[01] [00] [02][00] [FD]** Single click -> 100 ms, ACK
Bit position for output that is used to drive sorter coil are:
bit 0 = accept gate coil
bit 1 = sorter coil "A"(out 6/pin 4)
bit 2 = sorter coil "B"(out 5/pin 3)
bit 3 = sorter coil "C"(out 4/pin 10)
If output selected with bit in mask is not programmed for sorter activation it will not be activated but coin selector will still response with ACK.

....

6.5.2.6 Command 237 [hexED], Read input lines

Coin selector replies with two data byte representing state of DIP-switches and state of inputs In1(*pin 6*) and In2(*pin 5*)[14].
ALBERICI coin selectors has one or two banks of DIP-switches for various data or operating modes setting. First data byte is state of first DIP-switch(*bank 1*) and In1, while second represent second DIP-switch(*bank 2*) and In2. LSb is first switch in bank and MSb is state of input. Switch closed state is represented with logic "1", and input active state is logic "1". Message format is:
Host sends: **[Dir] [00] [01] [ED] [Chk]**
Coin s. replies: **[01] [02] [Dir] [Mask1] [Mask2] [Chk]**
Example of message string for coin selector(*address 2*), with all switches "off" and inputs not active is:
Host sends: **[02] [00] [01] [ED] [10]**
Coin s. replies: **[01] [02] [02] [00] [00] [00] [FB]**
Example of message string for coin selector(*address 2*), with all switches "on" and input 1(*inhibit acceptance*) active is:
Host sends: **[02] [00] [01] [ED] [10]**
Coin s. replies: **[01] [02] [02] [00] [BF] [00] [3C]**

6.5.2.7 Command 236 [hexEC], Read opto status

Coin selector replies with one data byte representing the state of opto pairs.
ALBERICI coin selectors has up to 3 pairs of optical sensor[15] for detection of coin position, speed and direction and 2 pairs of opto sensors for diameter measurement.
Bit position for opto pairs is:

- bit 0 Diam. measure opto 1
- bit 1 Diam. measure opto 2
- bit 2 Control opto 1
- bit 3 Control opto 2
- bit 4 Control opto 3
- bit 5 Not used
- bit 6 Not used
- bit 7 Not used

Control opto sensor 2 is called "credit" opto sensor exist in all version of coin selectors and it is placed after the acceptance gate. Other pairs are optional and some coin selectors has 2 and some 3 control optical pairs. Number of control pairs make part of coin selector type label. For example, coin selector type AL66V2 has 2 control opto sensor pairs. The unused bits or non-existing optical sensors are always read as 0.
Interruption of light barrier of opto sensor correspond to bit value 1.
Message format is:
Host sends: **[Dir] [00] [01] [EC] [Chk]**
Coin s. replies: **[01] [01] [Dir] [Mask.] [Chk]**
Example of message string for coin selector(*address 2*) with opto sensors cleared is:

Host sends: **[02] [00] [01] [EC] [11]**
Coin s. replies: **[01] [01] [02] [00] [00] [FC]**

[14] If In2 is programmed as input

[15] In some case group could contain more than one opto pairs

6.5.2.9 Command 232 [hexE8], Perform self-test *(no longer available starting from Fw v. u1.8 A4.1.9)*

Coin selector replies to command with one or two bytes of data according to table 6.3. First byte is fault code and second is optional data, usually representing fault sensor number(*from 1 to 3*).

Code	Fault	Optional data	Comment
0	OK No fault detected	-	-
2	Fault on inductive sensor	Sensor number	-
3	Fault on credit sensor	-	Control opto sensor 2
6	Fault on diameter sensor	-	-
18	Fault on reject sensor	-	Control opto sensor 3
33	Power supply out of limits	-	-
34	Temperature out of limit	-	Optional
255	Unspecified fault code	-	-

Table 6.3 Fault codes for AL55/66 coin selectors

Inductive sensor numbers are:

01 Upper inductive sensor	02 First lower inductive sensor	03 Second lower inductive sensor

Message format is:
Host sends: **[Dir] [00] [01] [E8] [Chk]**
Coin s. replies: **[01] [01/02] [Dir] [Fault c.][Data opt.] [Chk]**
Example of message string for coin selector(*address* 2) with no fault detected is:
Host sends: **[02] [00] [01] [E8] [15]**
Coin s. replies: **[01] [01] [02] [00][00] [FC]** No fault detected
Example of message string for coin selector(*address* 2) with first lower sensor fault detected is:
Host sends: **[02] [00] [01] [E8] [15]**
Coin s. replies: **[01] [02] [02] [00][02][02] [F7]** Fault on first lower sensor detected

6.5.2.10 Command 231 [hexE7], Modify inhibit status

With this command host is able to inhibit the acceptance of some or all coins.
Acceptance or inhibition is set with a two-byte mask sent by host.
Bits from 0 to 15 determinate coin positions from 1 to 16[16].
Number of coin channels in new ALBERICI coin selectors AL55/66 is same as number of position(*16*). Message format is:
Host sends: **[Dir] [02] [01] [E7] [LSB Mask.] [MSB Mask.] [Chk]**
Coin s. replies: **[01] [00] [Dir][00] [Chk]** ACK
Example of message string to enable all position for coin selector(*address 2*) is:
Host sends: **[02] [02] [01] [E7] [FF] [FF] [16]**
Coin s. replies: **[01] [00] [02][00] [FD]** ACK
After that all programmed coins will be enabled. Command has no effect on coin position that are not programmed.
Initially coin channels could be programmed with acceptance enabled or disabled.

For coin selectors that are using only cctalk interface, all coins' positions must be initially inhibited!

6.5.2.11 Command 230 [hexE6], Request inhibit status

Coin selector replies with two-byte data that correspond to inhibit state mask for all 16 positions of coin. If bit value is 1 acceptance of coin in that position is enabled. If bit value is 0 coin is inhibited. Message format is:
Host sends: **[Dir] [02] [00] [E6] [Chk]**
Coin s. replies: **[01] [02] [Dir] [00] [LSB Mask.] [MSB Mask.] [Chk]**
Example of message string for coin selector(*address* 2) **AL06V-c**[17] after power-up or reset is:
Host sends: **[02] [00] [01] [E6] [17]**
Coin s. replies: **[01] [02] [02] [00] [00] [00] [FB]**

Example of message string for coin selector(*address* 2) with programmed positions from 1 to 6, after receiving command to enable acceptance of all 16 position is:
Host sends: **[02] [00] [01] [E6] [17]**
Coin s. replies: **[01] [02] [02] [00] [3F] [00] [BC]**
First byte represents the mask for coin positions 1 to 8 and second for 9 to 16.
Coin channels(*positions*) that are not programmed are always represented as zero bit!

[16] Positions are sent by coin selector during reading credit buffer or error codes (*header 229*)

[17] Coin selector for Italian gambling machines

6.5.2.12 Command 229 [hexE5], Read buffered credit or error codes

This is the most important command used by host to detect import of coins in to a machine and to report eventual errors. As previously mentioned coin selectors store all events in volatile memory called credit buffer. Buffer has 5 level and use two bytes for each event. In first byte coin position or coin value[18] is stored. The second byte point to a sorter path or indicate error code.

If during coin acceptance any error occurs, stored value of coin position is 0, hex [00].

Error codes supported in ALBERICI coin selectors AL55/66 are shown in table 6.4.

Code d.	Code h.	Error	Coin rejected
0	00	Null event	No
1	01	Reject coin (not recognized)	Yes
2	02	Inhibited coin (master inhibit)	Yes
3	03	Multiple window (fraud or similar coin)	Yes
5	05	Validation (measuring) time out	Yes
6	06	Credit sensor (recognition to opto 2) time out	Possible
8	8	Second close coin	Yes/both
16	10	Credit sequence error (Yo-yo)	No
18	12	Coin too fast (opto 2 minimum time not elapsed)	No
19	13	Coin too slow (opto 2 time out)	No
128	80	Inhibited coin (position 1)	Yes
…	…	Inhibited coin (position n)	Yes
143	8F	Inhibited coin (position 16)	Yes
255	FF	Unspecified alarm code	-

Table 6.4 Acceptance error codes

Coin selectors also use one eight-bit counter[19] that is incremented each time a new coin is detected. At the same time data in coin credit buffer are shifted two positions to the right. When counter reaches the value of 255 it toggles to a value 1 and continue to increment on each event. Event counter is set to value “0” after each power-up or acceptance of reset command. The first two byte (*LSB*) in coin credit buffer always contain the data of last event. Host software must read event counter and coin credit buffer data in period short enough to prevent the loss of coin data[20]. Message format is:

Host sends: **[Dir] [00] [00] [E5] [Chk]**

Coin s. replies: **[01][0B] [Dir] [00] [Ev.cnt.][coin code 1][dir/err] [coin code 2][dir/err] . . .**
. . . [coin code 5][dir/err] [Chk]

Examples of message string for coin selector(*address 2*) after coin insertions:

Host sends: **[02] [00] [00] [E5] [18]** Polling minimum each 500 ms

Coin s. replies: **[01] [0B] [02] [00] [00][00][00][00][00][00][00][00][00][00][00] [F2]**
The reply after power-up or reset:

Coin s. replies: **[01] [0B] [02] [00] [01][01][02][00][00][00][00][00][00][00][00] [EE]**
First event, coin position 1, sorter path 2 accepted

Coin s. replies: **[01] [0B] [02] [00] [02][02][01][01][02][00][00][00][00][00][00] [EA]**
Second event, coin position 2, sorter path 1 accepted

Coin s. replies: **[01] [0B] [02] [00] [03][00][02][02][01][01][02][00][00][00][00] [E7]**
Third event, coin rejected due to master inhibit active

Coin s. replies: **[01] [0B] [02] [00] [04][00][83][00][02][02][01][01][02][00][00] [63]**
Forth event, coin position 4 inhibited and rejected

From example we can notice shifting of data in the coin credit and error buffer and increment of event counter.

6.5.2.13 Command 228 [hexE4], Modify master inhibit status

This command is used to inhibit acceptance of all coins and has same effect as command modify inhibit status with sent with two bytes of zeros. Host sends only one byte of data. If first bit (*LSb*) is set to “0” coin selector is inhibited. Bits 1 to 7 has no influence to coin selector. Message format is:

Host sends: **[Dir] [01] [01] [E4] [Mask.] [Chk]**

Coin s. replies: **[01] [00] [Dir][00] [Chk]**

ACK **Initially coin selectors are programmed with acceptance enabled. Change is stored in RAM location.**

[18] If coin selector use CVF (*Coin Value Format*)

[19] Event counter

[20] See command 249 Request polling priority

On customer demand it is possible to set inhibition as default .

Example of message string to inhibit the acceptance for coin selector(*address 2*) is:
Host sends: **[02] [01] [01] [E4] [00] [18]**
Coin s. replies: **[01] [00] [02][00] [FD]** ACK
After that coin selector acceptance will be inhibited till reset or next instruction that will change master inhibit status.

6.5.2.14 Command 227 [hexE3], Request master inhibit status

Coin selector replies with one byte data information of main inhibit status.
Only first (*LSb*) bit is used. If bit 0 is “1” acceptance is enabled, and if bit 0 is “0” coin selector is inhibited and acceptance is disabled.
Other bits have no meaning and always read as “0”. Message format is:
Host sends: **[Dir] [00] [00] [E3] [Chk]**
Coin s. replies: **[01] [01] [Dir] [00] [Mask.] [Chk]**
Example of message string for coin selector(*address 2*) after power-up is:
Host sends: **[02] [00] [01] [E3] [1A]**
Coin s. replies: **[01] [01] [02] [00][01] [FB]** Acceptance enabled (*default*)
Example of message string for coin selector(*address 2*) after activation of master inhibit[21] is:
Host sends: **[02] [00] [01] [E3] [1A]**
Coin s. replies: **[01] [01] [02] [00][00] [FC]** Coin selector inhibited

6.5.2.15 Command 226 [hexE2], Request insertion counter

Coin selector replies with three bytes of insertion counter data.
First byte is LS byte of three-byte counter in RAM. Insertion counter is set to zero after power up or reset command. It is incremented each time a new coin is inserted in to coin acceptor. Message format is:
Host sends: **[Dir] [00] [00] [E2] [Chk]**
Coin s. replies: **[01] [03] [Dir] [00] [Cunt1-LSB] [Cunt2] [Cunt3-MSB] [Chk]**
Example of message string for coin selector(*address 2*) after power-up is:
Host sends: **[02] [00] [01] [E2] [1B]**
Coin s. replies: **[01] [03] [02] [00] [00] [00] [00] [FA]**

6.5.2.16 Command 225 [hexE1], Request accept counter

Coin selector replies with three bytes of acceptance counter data.
First byte is LS byte of three-byte counter in RAM. Acceptance counter is set to zero after power up or reset command. It is incremented each time a new coin pass acceptance sensor[22]. Message format is:
Host sends: **[Dir] [00] [00] [E1] [Chk]**
Coin s. replies: **[01] [03] [Dir] [00] [Cunt1-LSB] [Cunt2] [Cunt3-MSB] [Chk]**
Example of message string for coin selector(*address 2*) after power-up is:
Host sends: **[02] [00] [01] [E1] [1C]**
Coin s. replies: **[01] [03] [02] [00] [00] [00] [00] [FA]**

……………………………

6.5.2.22 Command 210 [hexD2], Modify sorter path

With this command host is able to change direction of coins in sorter if sorter is supported. Host sends two bytes of data to select the coin position and sorter path (*direction of exit*). Firs byte of data (*LSB*) represent coin position and second byte of data point to sorter path. ALBERICI coin selectors has support for most existing sorters that has direct drive of coils from coin selector with open collector transistor. Most common are 3- or 4-way sorter with two coils[23] , but 5-way sorters[24] with 3 coils are in use as well. Message format is:
Host sends: **[Dir] [02] [01] [D2] [Coin pos.] [Sort.Path] [Chk]**
Coin s. replies: **[01] [00] [Dir] [00] [Chk]** ACK if sorter path is possible to set
Coin s. replies: **[01] [00] [Dir] [05] [Chk]** NAK if coin selector does not support setting

Initially all coin position has sorter paths set to direction 1 hex[01]. If sorter is not supported, sorter path is set initially to 0 hex[00]!

[21] Using command 210, Modify master inhibit status

[22] Credit sensor

[23] Maximum current consumption for each coil is 500 mA

[24] 5-way VARIANT sorter from ALBERICI

If host sends command to modify sorter path that is not existent or for coin not programmed, the coin selector will reply with message NAK. Ex. of message string for coin selector(*address* 2) redirection of coin **pos. 1** into **path 2** is:

Host sends: **[02] [02] [01] [D2] [01] [02] [26]**

Coin s. replies: **[01] [00] [02][00] [FD]** ACK

After acceptance of command, accepted coins with position 1 will exit in direction 2 of the sorter.
The path or direction 1 is usually one without activation of any coil.
Different coil activation schematics is possible to program by setting the sorter type.

6.5.2.23 Command 209 [hexD1], Request sorter path

Host send one byte of coin position and coin selector replies with one byte of sorter path. Message format is:

Host sends: **[Dir] [01] [00] [D1] [Coin pos.] [Chk]**

Coin s. replies: **[01] [01] [Dir] [00] [Sort.Path] [Chk]**

Example of message string for coin selector(*address* 2) for initial sorter path 1 of coin position 1:

Host sends: **[02] [01] [01] [D1] [01] [2A]**

Coin s. replies: **[01] [01] [02] [00] [01][FB]**

Example of message string for coin selector (*address* 2) for sorter path 2 of coin position 1:

Host sends: **[02] [01] [01] [D1] [01] [2A]**

Coin s. replies: **[01] [01] [02] [00] [02][FA]**

If host request sorter path for non-programmed coins or non-existent position[25], the coin selector will reply with message NAK !

………………………………

6.5.2.26 Command 196 [hexC4], Request creation date

Coin selector replies with two byte of data that represent codified date of production. Date of production is codified in so called *RTBY (**R**elative **T**o **B**ase **Y**ear)*[26] format. Message format is:

Host sends: **[Dir] [00] [01] [C4] [Chk]**

Coin s. replies: **[01] [02] [Dir] [00] [LSB] [MSB] [Chk]**

Example of message string for coin selector (*address* 2) with date of production *05.07.2003* is:

Host sends: **[02] [00] [01] [C4] [39]**

Coin s. replies: **[01] [02] [02] [00] [E5] [06] [10]**

ALBERICI coin selectors has date of production written in monitor part of MCU FLASH memory which is not possible to change without factory FLASH reprogramming.

6.5.2.27 Command 195 [hexC3], Request last modification date

Coin selector replies with two byte of data that represent codified date of last modification of software[27].
Date of modification is codified also in RTBY format.
Message format is:

Host sends: **[Dir] [00] [01] [C3] [Chk]**

Coin s. replies: **[01] [02] [Dir] [00] [LSB] [MSB] [Chk]**

Example of message string for coin selector (*address* 2) with date of modification *23.07.2003* is: Host sends:

[02] [00] [01] [C3] [3A]

Coin s. replies: **[01] [02] [02] [00] [F7] [06] [FE]**

NOTICE: after each up-grade of coin selector program FLASH memory date will correspond to software modification date, not to the actual date of up-grade!

6.5.2.28 Command 194 [hexC2], Request reject counter

Coin selector replies with three bytes of reject counter data.
First byte is LS byte of three-byte counter in RAM. Reject counter is set to zero after power up or reset command. It is incremented each time a coin is inserted but not recognized. Message format is:

Host sends: **[Dir] [00] [00] [C2] [Chk]**

Coin s. replies: **[01] [03] [Dir] [00] [Cunt1-LSB] [Cunt2] [Cunt3-MSB] [Chk]**

Example of message string for coin selector(*address* 2) after power-up is:

Host sends: **[02] [00] [01] [C2] [3B]**

Coin s. replies: **[01] [03] [02] [00] [00] [00] [00] [FA]**

[25] Position bigger than 16

[26] For details see cctalk protocol, document cctalk44-2.pdf

[27] Up-grade of FLASH program memory

6.5.2.29 Command 193 [hexC1], Request fraud counter

Coin selector replies with three bytes of fraud coins counter data.
First byte is LS byte of three-byte counter in RAM. Fraud counter is set to zero after power up or reset command. It is incremented each time a coin acceptor recognize coin that is programmed as "fraud" coin[28]. Message format is:
Host sends: **[Dir] [00] [00] [C1] [Chk]**
Coin s. replies: **[01] [03] [Dir] [00] [Cunt1-LSB] [Cunt2] [Cunt3-MSB] [Chk]**
Example of message string for coin selector(*address 2*) after power-up is:
Host sends: **[02] [00] [01] [C1] [3C]**
Coin s. replies: **[01] [03] [02] [00] [00] [00] [00] [FA]**
....

6.5.2.30 Command 188 [hexBC], Request default sorter path

For ALBERICI coin selectors AL55/66 the default sorter path is always hex**[01]**. Example of message string for coin selector(*address 2*) is:
Host sends: **[02] [00] [01] [BC] [41]**
Coin s. replies: **[01] [01] [02] [00] [01] [FB]**

6.5.2.32 Command 184 [hexB8], Request coin ID

Host use this command at initialization process to build table for each coin position value. If coin selector uses CVF it is obsolete command.
Host send one byte data of coin position and coin selector replies with 6-byte ASCII string of characters that describes the requested coin position.
Message format is:
Host sends: **[Dir] [01] [01] [B8] [Coin pos] [Chk]**
Coin s. replies: **[01] [06] [Dir] [00] [a1][a2][a3][a4][a5][a6] [Chk]**
Example of message string for coin selector(*address 2*) and coin position 1(*2 Euro*) is:
Host sends: **[02] [01] [01] [B8] [01] [43]**
Coin s. replies: **[01] [06] [02] [00][45][55][32][30][30][41] [8A]** Coin 'EU200A'
For none-programmed position the ASCII string is: '..... '.
Example of message string for coin selector(*address 2*) and coin position 12 that is not programmed is:
Host sends: **[02] [01] [01] [B8] [0C] [38]**
Coin s. replies: **[01] [06] [02] [00] [2E][2E][2E][2E][2E][2E] [E3]** Coin not programmed

6.5.2.35 Command 170 [hexAA], Request base year

Coin selector replies with four-byte ASCII string of character representing the base year for calculation of exact date of production. Message format is:
Host sends: **[Dir] [00] [01] [AA] [Chk]**
Coin s. replies: **[01] [04] [Dir] [00] [a1][a2][a3][a4] [Chk]**
For ALBERICI coin selectors base year is **2000**.
Example of message string for coin selector(*address 2*) is:
Host sends: **[02] [00] [01] [AA] [53]**
Coin s. replies: **[01] [04] [02] [00] [32][30][30][30] [37]**

6.5.3 MDCES command headers

MDCES stands for **M**ulti-**D**rop **C**ommand **E**xtension **S**et, or so-called Multi-drop buss commands. Multi-drop bus commands give additional functionality to systems that require change of address for devices in cctalk network. Some of commands has different message format than usual cctalk message.
Commands are:

- Address poll
- Address clash
- Address change
- Address random

Because host always use address 1 and address 0 is for broadcast message all commands that change the address should not accept these settings.

All changes are stored in non-volatile memory, EEPROM !

[28] Coins with close recognition parameters sometime called "killer coin or channel"

6.5.3.1 Command 253 [hexFD], Address poll

This is a broadcast message used by host to determinate all address of device attached on cctalk network. Coin selector replies with only one byte (*non-standard message format*), after a delay that is proportional to address value multiplied with 4 milliseconds. Message format is:

Host sends: **[00] [00] [01] [FD] [Chk]** Broadcast
message Coin s. replies: **Dly -> [Address]**

Example of message string for coin selector(*address 2*) is:

Host sends: **[00] [00] [01] [FD] [02]**
Coin s. replies: **Dly=8 ms-> [02]** Address is 2

Example of message string for coin selector (*address 250*) is:

Host sends: **[00] [00] [01] [FD] [02]**
Coin s. replies: **Dly=1 s-> [FA]** Address is 250

6.5.3.2 Command 252 [hexFC], Address clash

Command Address clash has same reply from coin selector but host issue this command with specific device address. Coin selector replies with only one byte (*non-standard message format*), after a random value of time delay to prevent collision if two devices share same address. Message format is:

Host sends: **[Dir] [00] [01] [FC] [Chk]**
Coin s. replies: **Random Dly -> [Address]**

Example of message string for coin selector(*address 2*) **AL06V-c** is:

Host sends: **[02] [00] [01] [FC] [01]**
Coin s. replies: **Random Dly-> [02]** Address is 2

6.5.3.3 Command 251 [hexFB], Address change (*not supported before Fw v. u1.8 A4.1.9 !*)

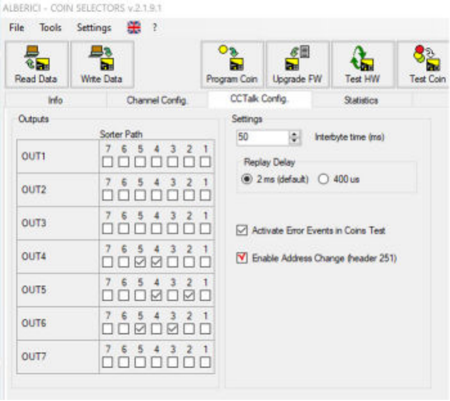
To *enable* Header 251, use programming sw ≥ v.2.1.9.1 and tick 'Enable Address Change' box (in red colour in figure beside). Command Address change is issued to a specified device only. Coin selector replies with ACK message.

Message format is:

Host sends: **[Dir] [01] [01] [FB] [Address] [Chk]**
Coin s. replies: **[01] [00] [02] [00] [FD]** ACK

Example of message string for coin selector with default *address 2* to be changed to *address 16*:

Host sends: **[02] [01] [01] [FB] [10] [ED]**
Coin s. replies: **[01] [00] [02] [00] [FD]** ACK
Address is now 16.



Sorter Path

	7	6	5	4	3	2	1
OUT1	□	□	□	□	□	□	□
OUT2	□	□	□	□	□	□	□
OUT3	□	□	□	□	□	□	□
OUT4	□	□	☑	☑	□	□	□
OUT5	□	□	□	☑	□	☑	□
OUT6	□	□	☑	□	☑	□	□
OUT7	□	□	□	□	□	□	□

50 Interbyte time (ms)

Replay Delay
● 2 ms (default) ○ 400 us

☑ Activate Error Events in Coins Test
☑ Enable Address Change (header 251)

Valid address values are: 2 (=default), or 11÷17. Else, the coin acceptor replies NAK.
If the command is NOT enabled (default), the coin acceptor ignores the command (does not answer!)

6.5.4 ALBERICI specific commands

First command header for specific factory setting and testing is **255 Factory set-up and test**. This command has several modes of use and some of them are factory secret and are not explained in this document.

Beside this next chapter describe two modified cctalk commands (Commands 214 Write data block and 215 Read data block) for encrypted exchange of data between host and coin selector.

6.5.4.1 Command 255, Factory set-up and test

This instruction header is used only for factory testing and programming!

With instruction command header one data byte must be sent for definition of command mode. Modes are:

- Mode 0(*hex 00*) No description – reserved for factory use only
- Mode 1(*hex 01*) No description – reserved for factory use only
- Mode 2(*hex 02*) Read coin selector memory data
- Mode 3(*hex 03*) Write coin selector memory data
- Mode 4(*hex 04*) Analog circuit test
- Mode 5(*hex 05*) Coin parameter test
- Mode 6(*hex 06*) Up-grade FLASH(*program memory*)
- Mode 7(*hex 07*) Re-initialization

First two modes are reserved for factory use - not described in this document!

To use read and write instruction user must have basic knowledge about coin selectors memory data organization (Details are available on customer request, see document AL55/66-MemDataOrg-v1.pd)!

6.5.4.2 Command 255 mode 2, Read coin sel. memory

This instruction sends back to host block of memory data that was requested.
Data memory of coin selector is divided into a 6 group:

- Coin channel data address hex 0600 do 06FF(*256 byte-a*)
- Input/Output data address hex 0700 do 073F(*64 byte-a*)
- Factory common data address hex 0740 do 079F(*96 byte-a*)
- User general data address hex 07A0 do 07E9(*74 byte-a*)
- Statistic setting address hex 07EA do 07FD(*20 byte-a*)
- Statistic counters address hex 0800 do 083F(*64 byte-a*)

Factory key or user PIN protection are sometime set to disable the access to some blocks of memory.
Usually, all memory is accessible for red instruction.
On any attempt to read memory location that is protected without clearing key or PIN protection mechanism, coin selector will reply with **NAK** message.
If number of blocks to read exceed block range, coin selector will also reply with **NAK** message. Maximum memory block read of 137 data bytes are limited by coin selector transmit buffer that has 142 bytes (Destination address+Bytes nr.+Source address+Header+59 data+Checksum=142). If block size extends this number coin selector will reply with NAK message to. Message string format is:
Host sends: **[Dir] [04] [01] [FF] [02][Start add-hi][Start add-lo][data nr.] [Chk]**
Coin s. replies: **[01] [n] [Dir] [00] [data 1] [data 2] . . . [data n] [Chk]**
Start add-hi is start address hi byte, while start add-lo is address low byte. Data nr. Represents the number of data to be read (block size).
Data 1 to data n are requested coin selector memory data.
The example of message string for reading of first coin channel data is:
Host sends: **[02] [04] [01] [FF][02] [06][00] [10] [E2]**
Coin s. replies: **[01][10][02][00][89][87][B4][77][A7][9F][08][08][0C][16][0A][04][08][01][09][C8][52]**

6.5.4.3 Command 255 mode 3, Write coin sel. memory

This command sends block of data to be written in the coin selector RAM memory.
As for the previous command(*read*) access to some memory blocks could be protected by factory key or user PIM mechanisam. Factory common data are usually protected with factory key and statistic counters and setting with user PIN. Exeption is made for statistic counter write. Any attempt to write in to statistic counter will erase those couners, thus protecting user from manipulation with statistics!
Number of bytes sent in command string are representing number of statistic to be erased.
Message string format is:
Host sends: **[Dir] [n+3] [01] [FF] [03][Start add-hi][Start add-lo][data 1]. . . [data n] [Chk]**
Coin s. respond: **[01][00] [Dir] [00] [Chk]** ACK

The example of message string for programming first output of coin selector **AL55/66** (*address 2*) as pulse parallel output for coin(*channel*) position 1 is:
Host sends: **[02] [0B] [01] [FF][03] [07][00] [01][00][00][00][14][00] [D2]**
Coin s. respond: **[01][00][02][00] [FD]** ACK

The example of message string for erase of all 20 statistic counters (20 x 3 byte of memory!) for coin selector **AL55/66** (*address 2*) is:
Host sends: **[02] [04] [01] [FF][02] [08][00] [14] [DB]**
Coin s. respond: **[01][00][02][00] [FD]** ACK

User PIN must be sent before erasing (*write*) statistic counters, otherwise selector will reply by **NAK** message.

Data write will change data in coin selector RAM, and changes will be lost if power supply turns off!
Use write to FLASH command to save changes (special case write command)**!**

To save all coin selector RAM data in to FLASH, send write command with no data:
Host sends: **[02] [01] [01] [FF][03] [FA]**
Coin s. respond: **[01][00][02][00] [FD]** dly 20 -100 ms **->**ACK
Coin selector will reply with ACK if write to FLASH was successful.

6.5.4.4 Command 255 mode 4, Analog circuit test

This command returns to host string of data that can be used to test analog measuring circuit state.
Details are described in internal documents and not available to users.

6.5.4.5 Command 255 mode 5, Coin parameter test

This command returns to host string of data from the last coin measurement.
This data are coin measured parameters and could be used to create and analyze coin data base.

Message string format is:
Host sends: **[Dir] [01] [01] [FF][05] [Chk]**
Coin s. replies: **[01] [6] [Dir] [00] [data 1] [data 2] . . . [data 6] [Chk]**
Data string description:

- data 1 Measured coin parameter 1, AM1
- data 2 Measured coin parameter 2, PH1
- data 3 Measured coin parameter 3, AM2
- data 4 Measured coin parameter 4, PH2
- data 5 Measured coin parameter 5, PI3
- data 6 Measured coin parameter 6, DIM

The example of message string after insertion of one Euro coin for coin selector **AL66** (*address 2*) is:
Host sends: **[02] [01] [01] [FF][05] [F8]**
Coin s. s: **[01] [6] [02] [00] [8A] [80][8C] [5C] [AB] [8A] [D0]**
Coin measured parameters(*data*) are available for read till next coin insertion. This instruction is used by the Alberici coin programming software.

6.5.4.6 Command 255 mode 6, Up-grade FLASH

This command is used to up-grade coin selector FLASH program memory.
User can up-grade coin selector in cases when he needs some new function or improvement, which were not available at the moment when coin selector was purchased. This instruction is used by our coin selector programming software and generally is not useful to most users. Additional information is available on request.
Up-grade files are encrypted in factory and decryption is done internally by monitor program of coin selector. Coin selector will accept only correct type of up-grade file!

6.5.4.7 Command 255 mode 7, Factory reset

This command will set all coin selector data to initial factory programming values!
Warning: This could lead to unwanted coin selector function if different coin, input / output or user setting data where programmed!
The example of message string reset the factory settings is:
Host sends: **[02] [01] [01] [FF][07] [F6]**
Coin s. replies: **[01][00][02][00] [FD]** ACK
After that, coin selector must be switched off/on!

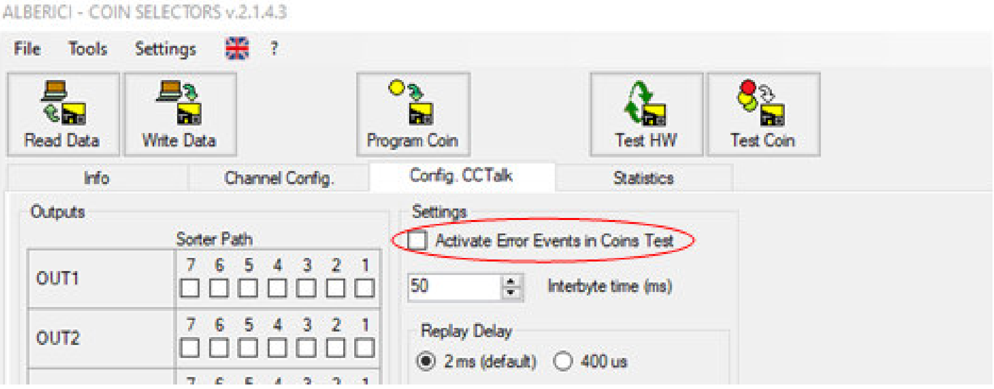
6.6 Error Codes

PLEASE NOTE: default status does not include Error Codes. The coin acceptor can be preset for error code transmission IF REQUESTED WHEN PLACING THE ORDER FOR THE COIN ACCEPTOR(S).

ERROR CODE LIST

Code d.	Code h.	Error	Coin rejected
0	00	Null event	No
1	01	Reject coin (not recognized)	Yes
2	02	Inhibited coin (master inhibit)	Yes
3	03	Multiple window (fraud or similar coin)	Yes
5	05	Validation (measuring) time out	Yes
6	06	Credit sensor (recognition to opto 2) time out	Possible
8	8	Second close coin	Yes/both
16	10	Credit sequence error (Yo-yo)	No
18	12	Coin too fast (opto 2 minimum time not elapsed)	No
19	13	Coin too slow (opto 2 time out)	No
128	80	Inhibited coin (position 1)	Yes
…	…	Inhibited coin (position n)	Yes
143	8F	Inhibited coin (position 16)	Yes
255	FF	Unspecified alarm code	-

Else, make use of the Alberici Coin Selector programming software to enable transmission. Tick the checkbox in the menu “Config. CCTalk","Settings"/"Activate Error Events in Coins Test", then press “Write data” key to transfer the instruction into the coin acceptor, i.e.:



OUT1 7 6 5 4 3 2 1 □ □ □ □ □ □ □

OUT2 7 6 5 4 3 2 1 □ □ □ □ □ □ □

Settings

□ Activate Error Events in Coins Test

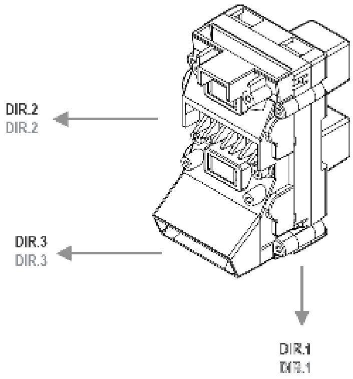
50 Interbyte time (ms)

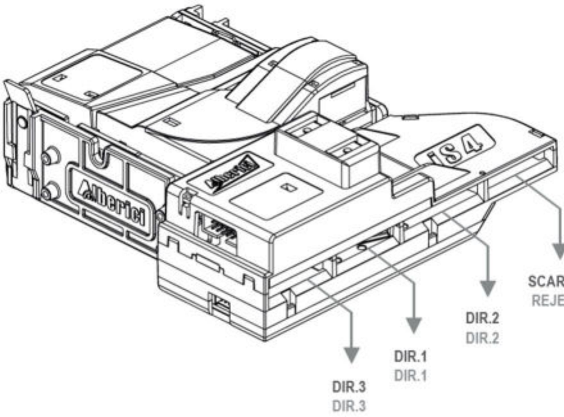
Replay Delay

● 2 ms (default) ○ 400 us

7. Driving SPI or Pulse peripheral devices

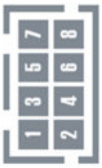
To attribute the appropriate sorting commands to the iS3 and iS4 SORTERS, please refer to the output paths shown below, and either to the directions given in the section "**SPI DISPLAY AD01 SERIAL PROTOCOL**", or to the usual Pulse commands.





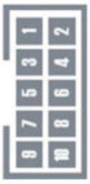
8 PIN SPI (X2)

DATA OUT	1	2	CLOCK
SELECT	3	4	DATA IN
GND LOGIC	5	6	+5 Vdc
GND	7	8	+24 Vdc



10 PIN PULSE (X1)

GND	1	2	+24 Vdc
OUT5	3	4	OUT6
IN2/OUT7	5	6	IN1/INHIBIT
OUT1	7	8	OUT2
OUT3	9	10	OUT4



Please notice: SPI mode does not allow the same processing speed as the Pulse mode

SPI Display AD01: serial protocol

The Alberici display AD01 consists of standard 16x2 characters LCD module with parallel interface, background light, equipped with a small controller pcb for SPI communication with the host. It has been designed to be used as a display accessory for SV coin selectors, but it can be used with any other host capable to interface through SPI.

7.1 Communication specifications

Slave mode operation
8 bit rate
Maximum Clk freq. = 2 MHz
Clk phase = 0; Transfer on the first clock edge
Clk polarity = 0; Transfer on clock transition low to high

7.1.1 Clock frequency

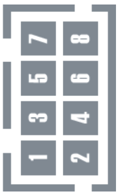
Clock frequency is not critical but if it is to low thandisplay will blink whennewdata or refresh of the olddata is done. Maximum frequency is limited with connection cable capacity and should be max 2 MHz for short connection(less *than 300 mm*).Recommended frequency is above 100 KHz till 1 MHz.

7.1.2 Power supply

The background light of the display must be powered by 8-26 Vdc . The logic circuit must be powered by 5 Vcd. See connector in section 7.1.3.

7.1.3 Connection (SPI Interface)

DATA OUT	1	2	CLOCK
SELECT	3	4	DATA IN
GND LOGIC	5	6	+5 Vdc
GND	7	8	+12/24 Vdc



The connector is pin-to-pin compatible with the one on the AL66 FG coin selectors

7.2 Message structure

The display can work only as slave and does not reply to any of the command messages sent by the host.
There are two different sets of commands:

- Display coin selector data
- Display message

First set is used to show specific coin selector data and second is used to show any display message.

7.2.1 Display coin selector data

The coin selector will send to the display controller the following command string:

[0xC0]
[0x08]
[Display mode]
[Display multiplier and decimal position]
[Display light int.]
[Accumulated credit]
[Accumulated value]
[Time in minutes]
[Time in sec.]
[Display warnings]

The first byte **0xC0** in message string means the type of message. The Hex code **0xCO** displays specific data of the coin selector.

The second byte is always **0x08** and represents the 8 bytes of data in the string.
Display mode is the code for showing further and different coin selector data. It is not possible to present all the data of the coin selector at the same time. Possible combinations and codes are shown in table below:

Code	Function
0x00	stop display
0x01	display Credit
0x02	display Value
0x03	display Credit and Value
0x04	display Time
0x05	display Credit and Time
0x06	display Value and Time
0x08	display Warning messages
0x09	display Credit and Warnings
0x0A	display Value and Warnings
0x0C	display Time and Warnings

Table 2 AD01 - Coin selector display modes

A special case of message is when this byte is **0x00** and there are no other of data to consider. It is used to turn off the background light of the display, stop the operation of the display device, and set to power saving mode (*stop mode*) the MCU communication MCU. Details concerning such data are not described in this document.

The **Display multiplier and decimal position** code must be used when the inserted value must be displayed. Coin selectors use relative coin value for their programming - but this is not at all a practical information to be displayed for Users. Because of that, this value must usually be multiplied and decimals must be set.
For instance: if the 50 cent coin has been programmed with relative value = 1, in order to display its value as an understandable 0,50 (€), the relevant value 1 must be multiplied by 50. For this reason, 2 decimal positions must be made available. Tables 3a and 3b show the codes corresponding to the various multipliers and decimal positions.

Code hex	Dec possition
X**0**	No decimal
X**1**	One dec. place
X**2**	Two dec. place
X**3**	Three dec. places

Table 3b Decimal places codes

Code hex	Multiplyer
0X	1
1X	1
2X	2
3X	4
4X	5
5X	10
6X	20
7X	25
8X	50
9X	100
AX	200
BX	250
CX	500
DX	1000
EX	1000
FX	1000

Table 3a Value multiplier codes

The byte of the **Display light intensity** allows to set the duty cycle of the driver of the light brightness (*Pulse Width Modulation*).
When the display is connected to the coin selector, the intensity of its background light will then be stored into the coin selector NV memory (*FLASH*), and it will remain stable even if power supply is changed.

> *Warning: when the host sends the brightness value to the display, be aware that high power supply can damage the light driver or the background LED-s if a high brightness value has been set.*

The **Accumulated credit** byte represents the amount of the accumulated credit for the coin selectors that are programmed for releasing the credit signal only after the “Request” command has been activated.

The **Accumulated value** can be the coin value inserted in the coin selector, or the credit value remaining after the credit impulse has been activated, or the remaining time after the timer count-down has been activated. The accumulated value will be displayed according to the multiplier and decimal position code!

The **Time** is displayed in format: 'mm : ss'. The coin selector will refresh the display every second as long as the timer output is active.

The **Display warnings** can show the status of the coin selector or other warnings. Warnings and relevant codes are shown in table 4.

Code hex	Warnings
00	Ready
01	Hardware error
06	Coin error
10	Rest value
20	Credits
40	Time out

Table 4 Warning codes

Any other code sent will produce no result on the display, so this one will be blank.

7.3 Display message

The Host can send commands to show any text on a 16x2 digits display. The following command strings must be used:
[0xC1]
[number of bytes] [Character address] [Data 1]
[Data 2]
……
[Data n]

First byte **0xC1** string means the type of message. The Hex code 0xC1 is for general message.
The second byte means the amount of **data bytes (n)** in the message, including the address (n+1).
The third byte corresponds to the character address: it starts by 0x00 for the beginning of the first digit on first line (top), and finishes by 0x1F for the last digit on second line (bottom). T
he second line begins by address 0x10.
The bytes that follow share the same structure of the character codes of the Display table (Table 4).

8. Disposal of the product

WARNING! DISPOSE OF ACCORDING TO THE GOVERNING LAW IN YOUR COUNTRY!
This equipment may not be treated as household waste. Instead, it must be handed over to the applicable collection point for the recycling of electric and electronic equipment. By ensuring that this product is disposed of correctly, you will help to prevent potential negative consequences for the environment and human health, which could otherwise be caused by inappropriate waste handling of this product.
For more detailed information about recycling of this product, please contact the Dealer where you purchased this product.

9. Warranty

The manufacturer will fix malfunctions arising from production faults in this machine or parts of it within 12 months from the date of sale.

All communications referring to guarantee repairs or replacements must be accompanied by the product serial number and the copy of the sale invoice.
To obtain your guarantee repair, please send the part to the Dealer where you purchased the machine, together with the following documents:
- copy of the sale invoice
- delivery note stating “returned for guarantee repair”
- detailed report of the problem found and the circumstances in which it occurs.

Before sending the product, please get in touch with your Dealer or with Alberici S.p.a. (+39 051 944300); very malfunctions can be fixed via a simple phone call, saving you costs and time.

Alberici S.p.a. will verify that warranty is applicable, i.e. that problem is not caused by:
- transport damages
- damages from incorrect installation or wrong configuration
- installation in premises or areas not complying with the prescribed safety requirements
- intentional or unwilled tampering
- wrong or careless use or maintenance
- non-compliance with precautions prescribed (see Chapter 4. Caution)
- natural disasters, vandalisms, intentional or unintentional damage

Guarantee is considered automatically expired if outer and inner labels are missing.

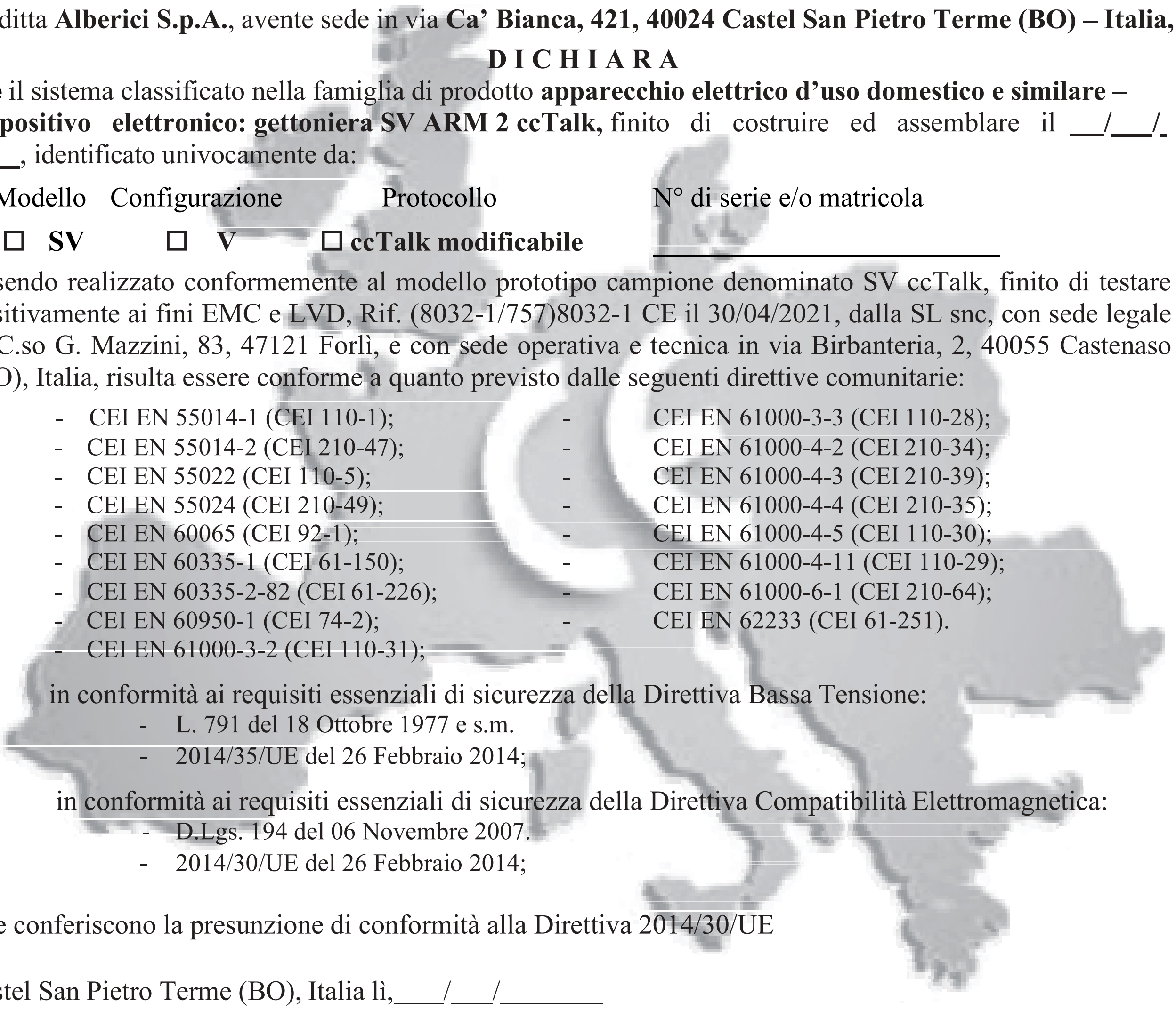
Transport costs of repaired products are at the Customer’s charge.

10. Customer Service

Alberici S.p.a. will be pleased to offer all the necessary information on use, ordinary maintenance and technical service. Please call (+39) 051 944300 and specify if your request concerns information on use or technical support.

DICHIARAZIONE DI CONFORMITÀ CE

DIRETTIVA 2014/35/UE – DIRETTIVA 2014/30/UE



La ditta **Alberici S.p.A.**, avente sede in via **Ca' Bianca, 421, 40024 Castel San Pietro Terme (BO) – Italia,**

DICHIARA

Che il sistema classificato nella famiglia di prodotto **apparecchio elettrico d'uso domestico e similare – dispositivo elettronico: gettoniera SV ARM 2 ccTalk,** finito di costruire ed assemblare il ___/___/____, identificato univocamente da:

Modello	Configurazione	Protocollo	N° di serie e/o matricola
□ **SV**	□ **V**	□ **ccTalk modificabile**	____________________

Essendo realizzato conformemente al modello prototipo campione denominato SV ccTalk, finito di testare positivamente ai fini EMC e LVD, Rif. (8032-1/757)8032-1 CE il 30/04/2021, dalla SL snc, con sede legale in C.so G. Mazzini, 83, 47121 Forlì, e con sede operativa e tecnica in via Birbanteria, 2, 40055 Castenaso (BO), Italia, risulta essere conforme a quanto previsto dalle seguenti direttive comunitarie:

a)
- CEI EN 55014-1 (CEI 110-1);
- CEI EN 55014-2 (CEI 210-47);
- CEI EN 55022 (CEI 110-5);
- CEI EN 55024 (CEI 210-49);
- CEI EN 60065 (CEI 92-1);
- CEI EN 60335-1 (CEI 61-150);
- CEI EN 60335-2-82 (CEI 61-226);
- CEI EN 60950-1 (CEI 74-2);
- CEI EN 61000-3-2 (CEI 110-31);
- CEI EN 61000-3-3 (CEI 110-28);
- CEI EN 61000-4-2 (CEI 210-34);
- CEI EN 61000-4-3 (CEI 210-39);
- CEI EN 61000-4-4 (CEI 210-35);
- CEI EN 61000-4-5 (CEI 110-30);
- CEI EN 61000-4-11 (CEI 110-29);
- CEI EN 61000-6-1 (CEI 210-64);
- CEI EN 62233 (CEI 61-251).

b) in conformità ai requisiti essenziali di sicurezza della Direttiva Bassa Tensione:
- L. 791 del 18 Ottobre 1977 e s.m.
- 2014/35/UE del 26 Febbraio 2014;

c) in conformità ai requisiti essenziali di sicurezza della Direttiva Compatibilità Elettromagnetica:
- D.Lgs. 194 del 06 Novembre 2007.
- 2014/30/UE del 26 Febbraio 2014;

Che conferiscono la presunzione di conformità alla Direttiva 2014/30/UE

Castel San Pietro Terme (BO), Italia lì,____/___/________

Fabrizio Alberici

Il Presidente

NOTICE

Alberici S.p.A. reserves the right to make changes to the equipment described and to its technical specifications at any time and without notice, in pursuit of continual improvement of this product.