GETT. ELETTRONICA AL66 FG ARM ccTalk Italy AES

Manuale del Prodotto

Rev. 2.01

Manuale Operatore

CE

Alberici

Progettazione e produzione di sistemi di pagamento e accessori per macchine Gaming, Vending e Car-Wash

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

SOMMARIO

STORICO REVISIONI			
Revisione n°	**Data**	**Modifica**	**Note**
Creazione v. 2.00	11.06.19	Creazione versione con Microprocessore ARM 3	Fw: u3.1A +
Versione v. 2.01	18.09.20	Inserito separatore iS4	Fw: u3.1A +

NOTA

Ogni possibile cura è stata posta nella redazione del presente manuale. Ciò nonostante, non è possibile garantire in ogni momento la corrispondenza assoluta delle descrizioni, in esso contenute, con le caratteristiche del prodotto.
La Alberici S.p.A. declina ogni e qualsivoglia responsabilità verso l'utilizzatore con riferimento a danni, perdite, o reclami di terze parti, conseguenti all'uso del prodotto o causate da errate interpretazioni del presente manuale.
La Alberici S.p.A. si riserva il diritto di modificare, senza preavviso e in qualunque modo, qualsiasi parte del presente manuale e delle specifiche tecniche dell'apparecchio, nell'ambito del perseguimento continuo del miglioramento dei propri prodotti.

Gentile Cliente,

desideriamo ringraziarla e congratularci con Lei per la scelta del selettore elettronico di di monete AL66 FG ARM ccTalk Italy. Siamo certi che ne apprezzerà la qualità e le prestazioni. Questa gettoniera comunica con la scheda host utilizzando il protocollo seriale ccTalk definito dalle Normative AAMS (Legge n. 289), e rispetta i requisiti di non modificabilità da queste prescritti.

Legga attentamente il presente manuale per ottenere il massimo rendimento da questo prodotto.

1. Contenuto della confezione

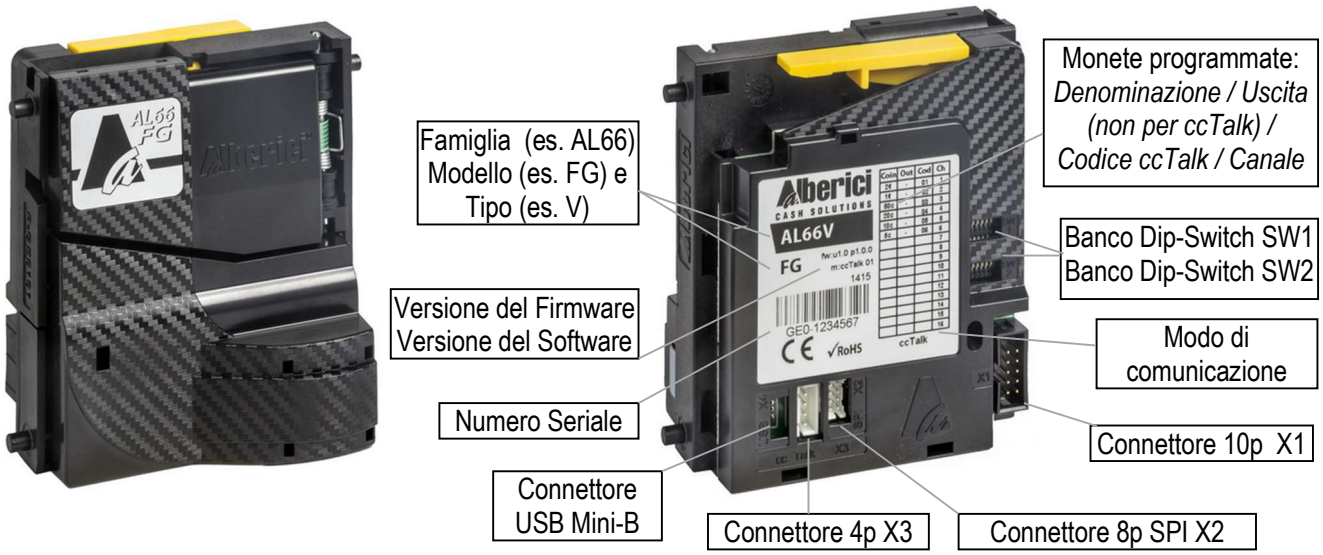
La confezione contiene:

1. il selettore elettronico di monete AL66 FG cctalk Italy, modello V, o modello S, o modello I
2. il manuale del prodotto (questo manuale) o la Scheda Dati

Il prodotto è stato imballato con la massima cura. Prima di firmare il documento di ritiro, vi preghiamo di aprire l'imballo e di verificare se il prodotto è stato danneggiato, o se presenta anomalie. In tal caso segnalatelo subito allo spedizioniere, facendogli trascrivere il reclamo sul documento di ricezione che vi richiede di firmare.

Tutto il materiale d'imballo deve essere riciclato o smaltito in conformità alle leggi vigenti.

2. Identificazione del contenuto



3. Descrizione del prodotto

La gettoniera AL66 FG è stata progettata con una specifica architettura dedicata al protocollo ccTalk. La comunicazione ccTalk con la scheda di controllo della macchina è canalizzata tramite il classico connettore 4poli TTL, oppure, a richiesta, mediante la porta USB Mini-B presente a bordo. La versione ccTalk può essere facilmente trasformata in versione criptata AES D.H. 256. La AL66 FG è dotata di microprocessore ARM 32 bit di ultima generazione, con 128 KB di memoria flash, modello "automotive", le cui prestazioni sono stabili anche in condizioni ambientali difficili.
Grazie ai suoi 6 sensori, 2 coppie di sensori ottici e a innovative tecniche di rilevamento, genera 13 parametri di riconoscimento (di cui 12 dinamici, ossia auto-adeguantisi alla situazione in cui avviene la misurazione) che assicurano prestazioni eccellenti nell'identificazione delle monete e di discriminazione dei falsi. La vigilanza antifrode è garantita da 4 coppie ottiche direzionali e dispositivi meccanici (trancia filo e strappa filo) che proteggono contro "ripescaggio" e "pendolino" e altri tipi di manomissioni E' dotata di dispositivi e controlli antifrode, meccanici ed elettronici, contro le più diffuse e recenti tecniche di truffa.
La tipologia costruttiva, con resine polimeriche HQ indeformabili, assicura robustezza e durabilità, contribuendio a garantire condizioni costanti di lettura indipendentemente dall'ambiente in cui la gettoniera viene utilizzata.
Mediante i Dip-Switch del banco SW1 si disabilitano e abilitano le monete da accettare. Si può modificare la tolleranza dell'accettazione tramite il dip-switch 6 del Banco SW2.
Grazie all'interfacciamento SPI (connettore X2), supporta display e periferiche intelligenti (es. separatori iS1 e iS4) .

4. Dati Tecnici

La gettoniera AL66 FG ccTalk Italy rispetta i requisiti di non modificabilità prescritti dalle Normative AAMS (Legge n. 289)

E' disponibile nelle seguenti versioni meccaniche di montaggio::
V = modello con uscita posteriore in basso della moneta rifiutata, uscita anteriore in basso della moneta accettata
I = modello con uscita anteriore in basso della moneta rifiutata, uscita posteriore in basso della moneta rifiutata
S = modello con introduzione moneta frontale e uscita frontale della moneta scartata

Dati Tecnici	
Caratteristiche meccaniche	
Formato	3½" standard
Dimensioni	88 x 102 x 52 mm = 3½"
Peso	212 g
Caratteristiche elettriche	
Tensione di alimentazione	12 - 24V DC (min. 10 - max. 26 VDC)
Assorbimento:	
In accettazione	350 mA(*30 ms*)/100 mA
In misurazione	Š30 mA
In attesa (stand by)	Š25 mA
Tipo uscita	Open Collector Darlington
Tensione uscita di saturazione	Š1 V
Tensione uscita max.	50 V (Attivo Basso)
Corrente uscita max.	250 mA
Tensione attivazione ingresso	3-24 Vdc (Attivo Alto)
Tensione ingresso max	50 V
Impedenza d'ingresso	=55 kfi
Accettazione monete	
Velocità di accettazione	3 mon/sec. (V) - 4 mon/sec. (K,S)
Numero canali moneta	16
Diametro min. moneta	16 mm
Diametro max. moneta	32 mm
Spessore moneta	1 - 3,4 mm
Communication modes	
Pulse	switching by Dip-Switch or by
ccTalk modificabile	programming software
Dati risposta	
Tempo di attivazione all'accensione	Š200 ms
Tempo di attivazione al risveglio	Š50 ms
Tolleranza impulso e time-put	± 2%
Condizioni ambientali	
Temperatura ambiente operativo	0°C to 60°C
Temperatura di conservazione	-30°C to 70°C
Umidità	fino a 75% *non condensata*
	fino a 95% (vers. tropicalizzata)
EMC performance	
This product is compliant with EN55014-1 and EN55014-2 test specification	

V

S

I

Tecnologie di riconoscimento e anti-frode	
N° parametri di riconoscimento	13
N° sensori riconoscimento	10 (3 coppie induttive e 5 ottiche)
N° sensori anti-frode	6 (3 coppie ottiche)
N° dispositivi meccanici anti-frode	1 trancia-filo
	2 strappa-filo

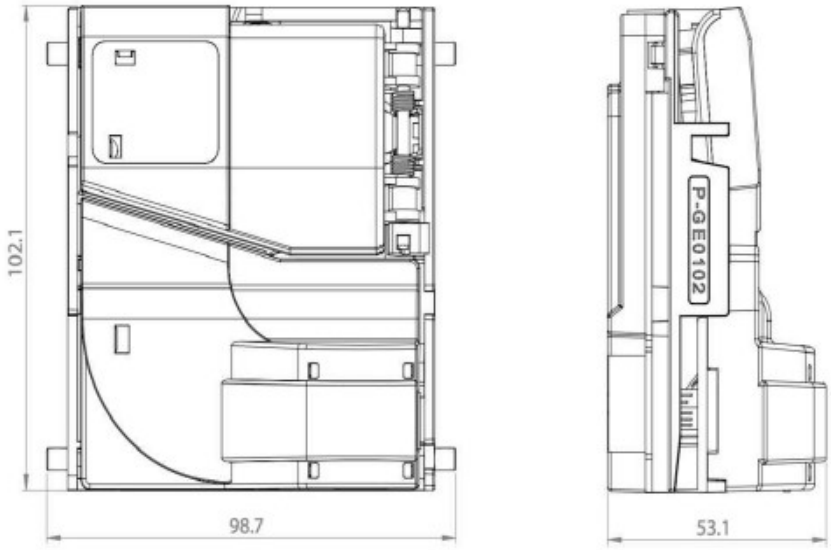
INGRESSO MONETA	COIN IN
MONETA VALIDA	VALID COIN
MONETA RIFIUTATA	REJECTED COIN

> Microcontrollore ARM a 32 bit con memoria Flash da 36 KB, immune alle interferenze magnetiche e alle condizioni ambientali.
> Sei sensori magnetici e un rilevatore ottico si combinano per garantire la più precisa capacità di discriminazione.
> Tagliafili (e strappafili nella versione V) proteggono meccanicamente dal ripescaggi, affiancando il .sistema anti-frode Coin-Guard, basato sulla sinergia di 3 sensori ottici opportunamente posizionati.

5. Installazione

DIMENSIONI

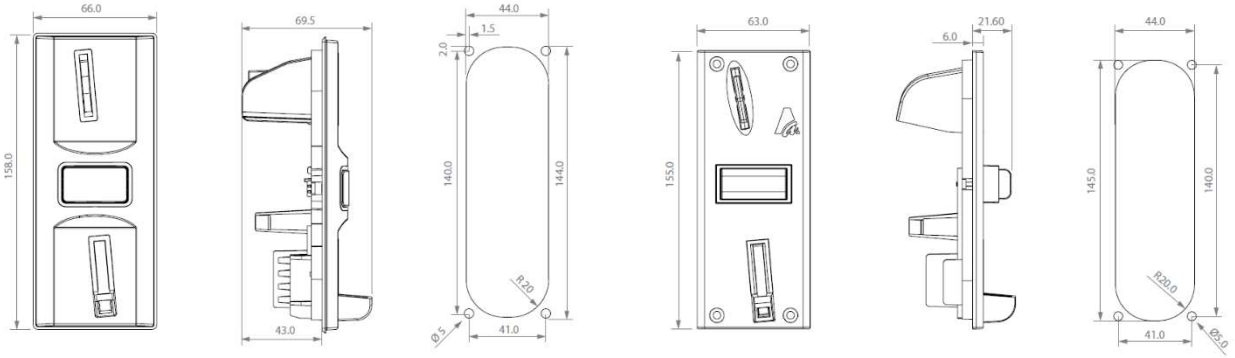
La gettoniera AL66 FG è una gettoniera da 3,5". E' perfettamente intercambiabile con le gettoniere presenti sul mercato e aventi le stesse tipologie di entrata e di uscita delle monete.



ATTENZIONE! La gettoniera deve essere montata tassativamente **da 90 a 95 gradi** rispetto al piano orizzontale.
Inoltre, in ragione dei sofisticati sistemi antifrode utilizzati su questo prodotto, è indispensabile che non venga ostacolato il percorso della moneta fino alla sua totale fuoriuscita dalla gettoniera e/o dal separatore.
L'azienda declina qualsiasi responsabilità per malfunzionamenti causati dall'inosservanza di queste raccomandazioni.

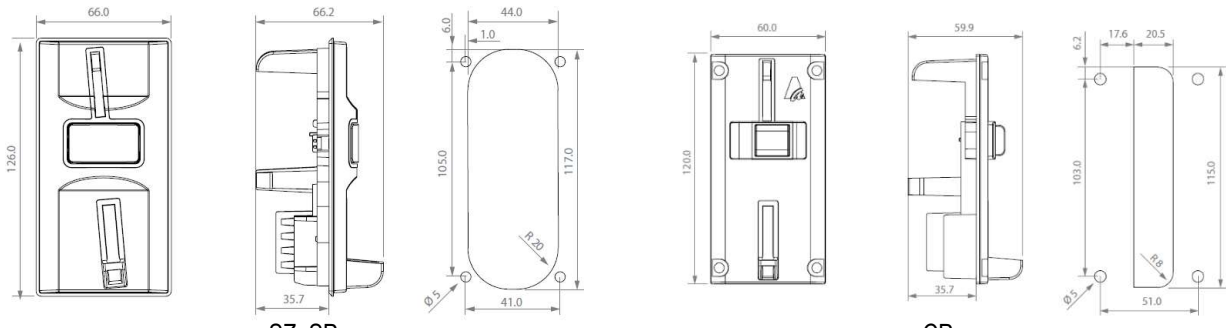
MONTAGGIO GETTONIERA TIPO "S" CON INTRODUZIONI FRONTALI

Sono disponibili diversi modelli di Introduzioni frontali, alla cui staffa la gettoniera si aggancia saldamente. Le introduzioni KZ, KP, KA, SZ, e SP sono progettate per installazione esterna al pannello. Le introduzioni KB, KI, KF, CP ed NP possono essere applicate sia esternamente che internamente. L'introduzione PP è progettata apposta per installazione interna con profilo esterno a filo.



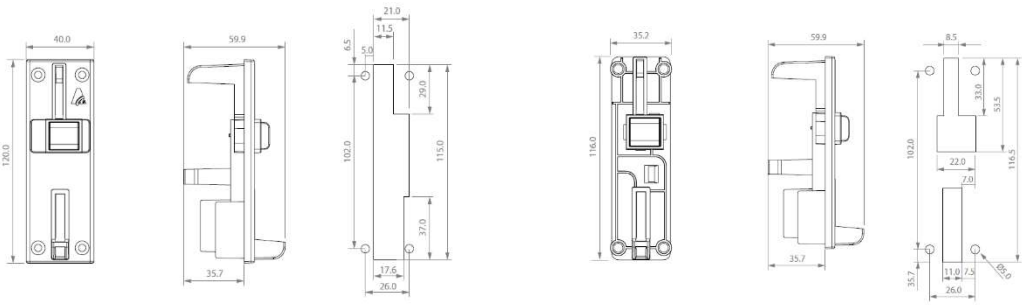
KZ, KP, KA

KB, KI, KF



SZ, SP

CP

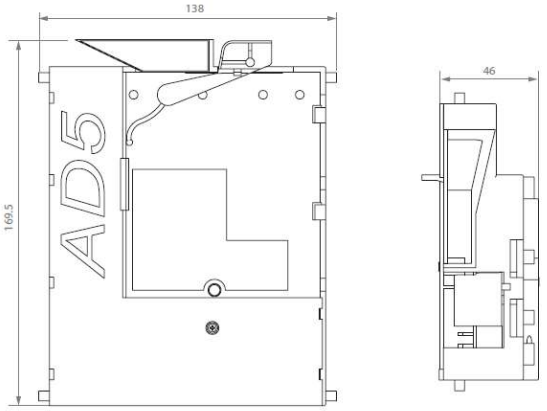


NP PP

L'eventuale separatore si monta sulla gettoniera frontale S mediante il supporto a staffa dedicato (AA-0703).

MONTAGGIO GETTONIERA TIPO "S" in ADATTATORE 5"

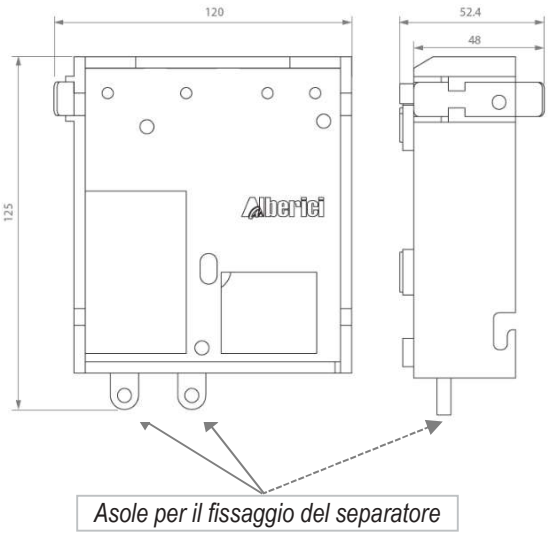
Incorporata nell'Adattatore 5" (SG-5000), e con l'aggiunta di leva di scarto superiore e relativa molla di torsione, la gettoniera da 3,5" può sostituire gettoniere da 5" aventi lo stesso protocollo di comunicazione.



L'adattatore 5" è disponibile anche con separatore integrato e/o con Kit di scarto elettronico per installazione remota.

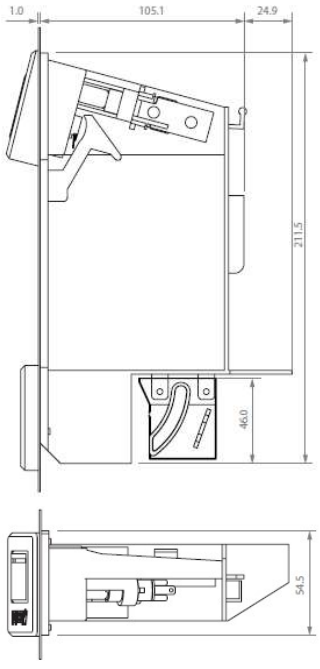
MONTAGGIO GETTONIERA TIPO V in SUPPORTO "C" o in SUPPORTO "SPAGNOLO"

La gettoniera verticale "V" (o "video") si installa nel Supporto "C" (SG-7000) o all'interno di Supporti di tipo "Spagnolo" (SG-3001 / SG-4001). Il supporto "C" consente anche di fissare alla sua sommità l'Introduzione verticale preferita, o il kit di scarto elettronico quando si preferisce situare la gettoniera lontano dall'ingresso monete per proteggerla da vandalismi.



Asole per il fissaggio del separatore

L'eventuale separatore si monta sul supporto "C" avvitandolo direttamente sulle due asole predisposte nel suo lato inferiore.



Connessioni

Il selettore si collega alle periferiche e alla scheda della macchina mediante i connettori seguenti:

X1. Alimentazione e interfaccia standard. Il connettore X1 è una presa IDC 10 poli, il cui schema compare nella tabella a fianco.
Non è usato per il collegamento ccTalk della gettoniera.
Presenta:
- 2 piedini di alimentazione (*pin 1 comune/terra & pin 2 positivo*),
1 input *(pin 6, normalmente dedicato all'inibizione dell'accettazione)*,
e 7 output "open collector" (*pin 3,4,5,7,8,9 & 10*), uno dei quali (pin 5) normalmente dedicato al totalizzatore multi-pulse (output "open collector"), oppure come secondo input (es.. per richiesta di credito).
OUTPUT : STATO ATTIVO = BASSO
INPUT: STATO ATTIVO = ALTO

nr.	Descrizione
1	Gnd
2	8-26 Vdc
3	Out 5 / bobina separatore B
4	Out 6 / bobina separatore A
5	Out 7 (totalizzatore) / In 2
6	In 1 (inibizione)
7	Out 1
8	Out 2
9	Out 3
10	Out 4 / bobina separatore C

X3. CCTALK. Il connettore X3 a 4 poli è *usato per la comunicazione seriale* ***cctalk®*** *con la scheda macchina*. Il protocollo è predisposto per funzionare in modalità "slave". I comandi da implementare nell''host sono descritti in dettaglio con esempi nel capitolo 6 del manuale.
Nei selettori Pulse standard e ccTalk modificabile, questo connettore viene usato per programmarli via PC mediante il software dedicato Alberici.

nr.	Descrizione
1	Dati
2	Gnd
3	NC
4	12 Vdc

DISABILITAZIONE MONETE MEDIANTE BANCO DIP-SWITCH SW1

AD OGNI CANALE PROGRAMMATO CORRISPONDE UNA E SOLO UNA DENOMINAZIONE. AD OGNI CANALE CORRISPONDE UN DIP-SWITCH DEL BANCO SW1.
PER INIBIRE LE MONETE, SPOSTARE SU OFF IL DIP-SWITCH CORRISPONDENTE (CONSULTARE LA TABELLA QUI SOTTO), POI SPEGNERE E RIACCENDERE.

BANCO SW1	DS1	DS2	DS3	DS4	DS5	DS6
ON	**Abilita CH1 (es. 2€)**	**Abilita CH2 (es. 1€)**	**Abilita CH3 (es. 0,50€)**	**Abilita CH4 (es. 0,20€)**	**Abilita CH5 (es. 0,10€)**	**Abilita CH6 (es. 0,05€)**
OFF	**Disabilita CH1 (es. 2€)**	**Disabilita CH2 (es. 1€)**	**Disabilita CH3 (es. 0,50€)**	**Disabilita CH4 (es. 0,20€)**	**Disabilita CH5 (es. 0,10€)**	**Disabilita CH6 (es. 0,05€)**

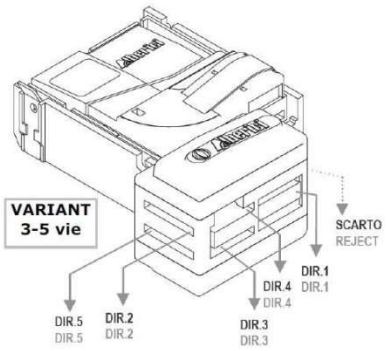
SETTAGGI MEDIANTE BANCO DIP-SWITCH SW2

BANCO SW2	DS2 OFF	DS2 ON	DS4 OFF	DS4 ON	
DS1 OFF	PULSE (*)	MDB (*)			
DS1 ON	CCTALK	SAS (*)			
DS3 OFF					Segnale del totalizzatore da OUTPUT 7
DS3 ON					Segnale del totalizzatore da OUTPUT 3
DS5 OFF	Moltiplica / dividi il valore del credito (*)		x 1	x 4	
DS5 ON			x 2	: 10	
DS6 OFF					Selettività STANDARD (L / SLC)
DS6 ON					Selettività ALTA (H / SLC)

(*) Non disponibile in questa versione

PILOTAGGIO DEI SEPARATORI

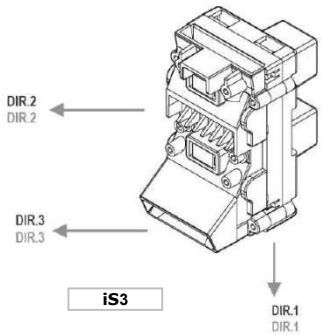
Per inviare i corretti comandi ai separatori, cfr. le indicazioni sulle direzioni di separazione qui sotto:



COMMANDI di SEPARAZIONE

	SEPARATORI CCTALK	SEPARATORI PULSE
Dir 1 = cctalk 01	attiva	Dir 1 = nessun OUT
Dir 2 = cctalk 02	attiva	Dir 2 = OUT 5
Dir 3 = cctalk 03	attiva	Dir 3 = OUT 6
Dir 4 = cctalk 06	attiva	Dir 4 = OUT 4 + OUT 5
Dir 5 = cctalk 07	attiva	Dir 5 = OUT 4 + OUT 6

PRESE DISPONIBILI SUI SEPARATORI



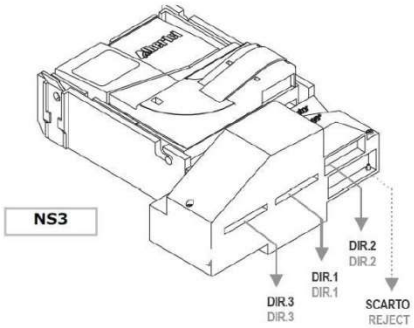
iS3

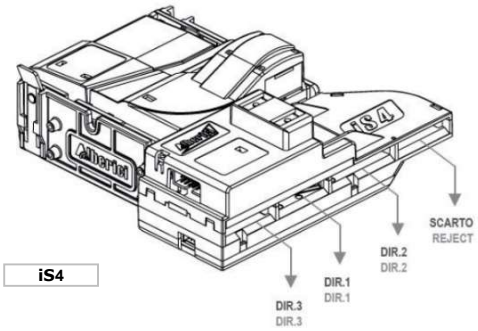
10 PIN PULSE (X1)

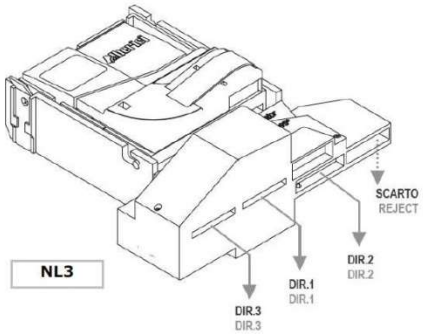
GND	1	2	+24 V
OUT5	3	4	OUT6
IN2/OUT7	5	6	IN1/INHIBIT
OUT1	7	8	OUT2
OUT3	9	10	OUT4

4 PIN CCTALK (X3)

DATA	1
GND	2
NC	3
+24 V	4







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

N.B.: la modalità SPI è disponibile unicamente sui separatori della serie iS: iS1 e iS4, ed è pilotabile unicamente dalla gettoniera AL66 FG

6. Protocollo di comunicazione ccTalk

ccTalk® communication protocol is the Money Controls serial communication protocol for low speed control networks. It was designed to allow the interconnection of various cash handling devices (*Hoppers, Bill validators, Coin selectors etc*.), mostly in AWP and gaming Industry, but allso in other devices that use those components.
cctalk® is an open standard. All documentation is available at web site: www.cctalk.org.

6. Serial cctalk communication (English language)
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 (Coin selectors, Hoppers(*pay out device*), Banknote readers etc.) in small network.
Protocol is mostly used in gaming and casino machines but it can be implemented in any other tipes of machines that use same type of devices. It is public protocol and free to use. Pls. find documentation on: www.cctalk.org.
Communication protocol of ALBERICI coin selectors AL55I66 complies with generic specification 4.4

6.1 Communication specifications
Serial communication was derivated from RS232 standard.
It is low data rate NRZ (***N****on* ***R****eturn to* ***Z****ero*) asyncronous 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 suported.
Message integrity is controled by means of checksum calculation.

6.1.1 Baud rate
The baud rate of 9600 was chosen as compromise betwen 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 voltages.

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

nr.	Descrizione
1	Dati
2	Gnd
3	NC
4	12 Vdc

Image 6.1 communication connector

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

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 responds to instruction Address poll and Address clash5. The responds 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 caled
"brodcast" address and address 1 is default host address.
Table 6.1 shows the recommended address values of different devices.

Device category	Address	Additional addr.	Note
Coin Acceptor	2	11 - 17	Coin validator, coin selector, coinmech...
Payout	3	4 - 10	Hopper
Bill validator	40	41 - 47	Banknote reader
Card Reader	50		
Display	60		Alphanumeric LC display
Keypad	70		
Dongle	80	85	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 Hopper is factory set as 3; the user can change the default address by using the MDCES commands Address change or Address random or by setting Hopper dip-switches. For details see cctalk42-2.pdf, Address poll.

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 packet 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 (252 bytes of data, source address, header and checksum: total of 255 bytes).

6.2.3 Command headers (*Instructions*)

Total amount of cctalk command header is 255 with possibility to add sub-headers using headers 100, 101, 102, 103.
Header 0 stands for **ACK** (***ack****nowledge*) replay of device to host.
Header 5 stands for **NAK** (***N****o* ***ac****knowledge*) 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

ALBERICI Coin selectors AL55I66 use total of 557 instructions-headers.

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. Intrepretation as well as format is specific to each header use, and will be explained in separate chapter.

6.4.5 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 (See Error handling).
For noisy environment or higher security application it is possible to use more complex, 16 bit CRC CCITT checksum based on a polynomial of:
x16 + x12 + x5 + 1 and initial value of CRC register **0x0000**.
Hopper are using simple checksum, but they could be set to operate with CRC-16 checksum on customer demand.

6.5 Timing specification

The timing requirements of cctalk are not very critical but there are some recommendations (i.e. more than 100 mili sec for solenoid testing).

6.5.1 Time between two bytes

When receiving bytes within a message packet, 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 inter-byte delay during transmission should be ideally **less than 2 ms** and **not greater than 10 ms**.

6.5.2 Time between command and replay

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**. Other commands that must activate some actions in device may return reply after the action is finished

6.5.3 Start-up time

After the power-up sequence Hopper should be ready to accept and answer 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 Hopper should be able works fine.

6.4 Error handling

If slave device receive 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.

254	FE	Simple poll	228	E4	Modify master inhibit status
253	FD	Address poll (broadcast)	227	E3	Request master inhibit status
252	FC	Address clash (no broadcast)	226	E2	Request insertion counter
252	FC	Address clash (broadcast)	225	E1	Request acceptance counter
251	FB	Address change (not supported in coin acceptors)	210	D2	Modify sorter paths
249	F9	Request polling priority	209	D1	Request sorter paths
248	F8	Request status	197	C5	Calculate ROM checksum
246	F6	Request manufacturer id	196	C4	Request creation date
245	F5	Request equipment category id	195	C3	Request last modification date
244	F4	Request product code	194	C2	Request reject counter
242	F2	Request serial number	193	C1	Request fraud counter
241	F1	Request software revision	192	C0	Request build code
240	F0	Test solenoids	188	BC	Request default sorter path
237	ED	Read input lines	184	B8	Request coin id
236	EC	Read opto states	170	AA	Request base year
232	E8	Perform self test	4	04	Request comms revision
231	E7	Modify inhibit status	3	03	Clear comms status variables
230	E6	Request inhibit status	2	02	Request comms status variables
229	E5	Read buffered credit or error codes	1	01	Reset device

Table 6.2 cctalk instruction header list

Command headers are divided in to 3 different groups:
- Common command headers
- Coin acceptor command headers
- MDCES command headers

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 responds with ACK (*Acknowledge*).
If within predicted amount of time Coin selector does not responds coin selector is probably not connected, powered or simple not working properly.
Message format is:
Host sends: **[Dir] [00] [01] [FE] [Chk]**
Coin s. responds: **[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. responds: **[01] [00] [02] [00] [FD]** ACK message

6.5.1.2 Command 246 [hexF6], Request manufacturer ID

Coin selector responds with ASCII string representing manufacturer name.
Message format is:
Host sends: **[Dir] [00] [01] [F6] [Chk]**
Coin s. responds: **[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. responds: **[01] [08] [02] [00] [41][6C][62][65][72][69][63][69] [DA]**

6.5.1.3 Command 245 [hexF5], Request equipment category ID

Responds to command header is standardized name for coin selectors, coin validators or coin mechs. Coin selector responds 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. responds: **[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. responds: **[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 responds with ASCII string of character, representing the factory type of coin selector. For ALBERICI coin selectors of new generation possible response will be:
- **AL55V1, AL55K1, AL55I1**
- **AL66V2, AL66K3, AL66I3**

In special version for italian gambling machines response is allways **AL06V-c** .

Host sends: **[Dir] [00] [01] [F4] [Chk]**
Coin s. responds: **[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. responds: **[01] [07] [02] [00] [41][4C][30][36][56][2D][63] [1D]**

6.5.1.5 Command 242 [hexF2], Request serial number
Coin selector responds with three byte serial number. Message format is:
Host sends: **[Dir] [00] [01] [F2] [Chk]**
Coin s. responds: **[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. responds: **[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. responds: **[01] [Nr.b] [Dir] [00] [a1] [a2]. ... [an] [Chk]**
Number of data bytes in ASCII string is not limited and each producer has it's own system of labelling. Example of message string for coin selector(*address 2*) is:
Host sends: **[02] [00] [01] [F1] [0C]**
Coin s. responds: **[01] [09] [02] [00] [75][31][2E][30][20][70][31][2E][30][2E][30] [71]**
Coin selector response 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 responds 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 responds of coin selector.
Message format is:
Host sends: **[Dir] [00] [01] [C5] [Chk]**
Coin s. responds: **[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. responds: **[01] [04] [02] [00] [D9][2A][7E][79] [96]**

6.5.1.8 Command 192 [hexC0], Request build code
Coin selector responds with ASCII string of character representing it's hardware version and revision. Last revision of printed circuit board for coin selectors AL55/66 is: **AL66 V1.0**.
Message format is:
Host sends: **[Dir] [00] [01] [C0] [Chk]**
Coin s. responds: **[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. responds: **[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 responds 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. responds: **[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. responds: **[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 responds with ACK message. Message format is:
Host sends: **[Dir] [00] [01] [03] [Chk]**
Coin s. responds: **[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. responds: **[01] [00] [02] [00] [FD]** ACK message
10 Details of description see in public document cctalk44-2.pdf
11 Address change, Address random

6.5.1.12 Command 2 [hex02], Request comms status variables

Coin selector responds with three byte data representing communication errors.
First byte is receive time out counter, second byte is number of ignored receive bytes (Number of receive buffer overflow bytes), and third byte is number of checksum errors. Message format is:
Host sends: **[Dir] [00] [01] [02] [Chk]**
Coin s. responds: **[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. responds: **[01] [03] [02] [00] [00] [00] [00] [FA]**

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. responds: **[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. responds: **[01] [00] [02] [00] [FD]** ACK message
Host software must wait at least **100 ms**, to continue comunication 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 (Sorter control commands). 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 polling1 (Reading memory buffer from coin selector). Coin selectors due to differences in mechanical and electrical construction has different acceptance speed. All events are registered in memory buffer with limited size1 (Five stage double byte memory buffer). 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 (Dependant on mechanical type of coin selector - S type is faster than V type - and coin.
Considering that it is possible to register 5 event 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. It is 38 ms for coin selector with response time 2ms and byte gap 1 ms.

Coin selector responds to command with two bytes of data. First byte is poll time unit and second is polling time value (For details see, cctalk44-2.pdf). Message format is:
Host sends: **[Dir] [00] [01] [F9] [Chk]**
Coin s. responds: **[01] [01] [Dir] [Time] [Chk]**
Example of message string for coin selector(*address 2*) is:
Host sends: **[02] [00] [01] [F9] [04]**
Coin s. responds: **[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 (**C**oin **O**n **S**tring) 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. responds: **[01] [01] [02] [00] [00] [FC]**

6.5.2.3 Command 243 [hexF3], Request data base version
The responds to that command is version of coin data base. Version of data base is important for coin selectors with remote programming support. For all ALBERICI coin selectors type AL55/66 current data base version is 00.
Message format is:
Host sends: **[Dir] [00] [01] [F3] [Chk]**
Coin s. responds: **[01] [01] [Dir] [Ver.] [Chk]**
Example of message string for coin selector(*address 2*) is:
Host sends: **[02] [00] [01] [F3] [0A]**
Coin s. responds: **[01] [01] [02] [00] [00] [FC]**

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 100ms and after that, ACK message will be transmitted. Message format is:
Host sends: **[Dir] [01] [01] [F0] [Mask.] [Chk]**
Coin s. responds: **[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. responds: **[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.

6.5.2.5 Command 238 [hexEE], Test output lines
Host sends one byte mask to determinate which output line must be tested.
Coin selector output line that corresponds to bit set in the mask will be pulsed for 100 ms and after that, ACK message will be transmitted. Message format is:
Host sends: **[Dir] [01] [01] [EE] [Mask.] [Chk]**
Coin s. responds: **[01] [00] [Dir] [00] [Chk]** ACK
Example of message string for coin selector(*address 2*) first output(*pin7*) is:
Host sends: **[02] [01] [01] [EE] [01] [0D]**
Coin s. responds: **[01] [00] [02] [00] [FD]** Single pulse out 1 -> 100 ms, ACK

Bit positions for output test are:
- bit 0 Output 1(*pin 7*)
- bit 1 Output 2(*pin 8*)
- bit 2 Output 3(*pin 9*)
- bit 3 Output 4(*pin 10*)
- bit 4 Output 5(*pin 3*)
- bit 5 Output 6(*pin 4*)
- bit 6 Output 7(*pin 5*)
- bit 7 Not used

Unused output (*not programmed*) will not be turned on, but message ACK will be returned.

6.5.2.6 Command 237 [hexED], Read input lines

Coin selector responds with two data byte representing the state of DIP-switches and state of inputs In1(*pin 6*) and In2(*pin 5*), if In2 is programmed as input. 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. responds: **[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. responds: **[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. responds: **[01] [02] [02] [00] [BF] [00] [3C]**

6.5.2.7 Command 236 [hexEC], Read opto states

Coin selector responds with one data byte representing the state of opto pairs.
ALBERICI coin selectors has up to 3 pairs of optical sensor (notice: in some case group could contain more than one opto pairs) for detection of coin position, speed and direction and 2 pairs of opto sensors for diameter measurement.
Bit position for opto pairs are:
- 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 AL66V**2** 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 corresponds to bit value 1.
Message format is:
Host sends: **[Dir] [00] [01] [EC] [Chk]**
Coin s. responds: **[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. responds: **[01] [01] [02] [00] [00] [FC]**

6.5.2.8 Command 233 [hexE9], Latch output lines

This instruction is similar to instruction 238. Host sends one byte mask to determinate which output line must be activated(*latch*). ACK message will be transmitted immediate.
Coin selector output line that corresponds to bit set in the mask will be latched and active till reset or new instruction with bit cleared is sent. Message format is:

Host sends: **[Dir] [01] [01] [E9] [Mask.] [Chk]**
Coin s. responds: **[01] [00] [Dir] [00] [Chk]** ACK
Example of message string for coin selector(*address 2*) first output(*pin7*) is:
Host sends: **[02] [01] [01] [E9] [01] [12]**
Coin s. responds: **[01] [00] [02] [00] [FD]** Latch out 1 -> ACK

6.5.2.9 Command 232 [hexE8], Perform self-test
IMPORTANT WARNING: send only after at least 2 seconds have elapsed from power-on or from reset command.
Coin selector responds 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. responds: **[01] [01I02] [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. responds: **[01] [01] [02] [00] [00] [FC]** No fault detected
Ex. of message string for coin selector (*addr. 2*) with first lower sensor fault detected is:
Host sends: **[02] [00] [01] [E8] [15]**
Coin s. responds: **[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 do 15 determinate coin positions from 1 to 16. Positions are sent by coin selector during reading credit buffer or error codes (*header 229*).
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. responds: **[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. responds: **[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 position must be initially inhibited!

6.5.2.11 Command 230 [hexE6], Request inhibit status
Coin selector responds with two byte data that corresponds 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. responds: **[01] [02] [Dir] [00] [LSB Mask.] [MSB Mask.] [Chk]**
Example of message string for coin selector(*address 2*) **AL06V-c** (Coin selector for Italian gambling machines) after power-up or reset is:

Host sends: **[02] [00] [01] [E6] [17]**
Coin s. responds: **[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. responds: **[01] [02] [02] [00] [3F] [00] [BC]**
First byte represent 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!

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 into 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 is stored, if coin selector uses CVF (*Coin Value Format*). 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 to fast (opto 2 minimum time not elapsed)	No
19	13	Coin to 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 (Event counter) that is incremented each time a new coin is detected. At the same time data two positions are shifted to the right in coin credit buffer. When counter reaches the value of 255 it toggle 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 (See command 249 Request polling priority). Message format is:
Host sends: **[Dir] [00] [00] [E5] [Chk]**
Coin s. responds: **[01][0B] [Dir] [00] [Ev.cnt.][coin code 1][dir/err] [coin code 2][dirlerr] . . .**
. . . [coin code 5][dirlerr] [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. responds: **[01] [0B] [02] [00] [00][00][00][00][00][00][00][00][00][00][00] [F2]**
The responds after power-up or reset
Coin s. responds: **[01] [0B] [02] [00] [01][01][02][00][00][00][00][00][00][00][00] [EE]**
First event, coin possition 1, sorter path 2 accepted
Coin s. responds: **[01] [0B] [02] [00] [02][02][01][01][02][00][00][00][00][00][00] [EA]**
Second event, coin possition 2, sorter path 1 accepted
Coin s. responds: **[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. responds: **[01] [0B] [02] [00] [04][00][83][00][02][02][01][01][02][00][00] [63]**
Forth event, coin possition 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. responds: **[01] [00] [Dir] [00] [Chk]** ACK
Initially coin selectors are programmed with acceptance enabled. Change is stored in RAM location.
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. responds: **[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 responds 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 has no meaning and always read as "0". Message format is:
Host sends: **[Dir] [00] [00] [E3] [Chk]**
Coin s. responds: **[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. responds: **[01] [01] [02] [00] [01] [FB]** Acceptance enabled (*default*)
Example of message string for coin selector(*address 2*) after activation of master inhibit27 is:
Host sends: **[02] [00] [01] [E3] [1A]**
Coin s. responds: **[01] [01] [02] [00] [00] [FC]** Coin selector inhibited

6.5.2.15 Command 226 [hexE2], Request insertion counter
Coin selector responds 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. responds: **[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. responds: **[01] [03] [02] [00] [00] [00] [00] [FA]**

6.5.2.22 Command 210 [hexD2], Modify sorter paths
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 -(Maximum current consumption for each coil is 500 mA, but recently 5 way sorters with 3 coils are in use3 (ex. 5-way VARIANT sorter from ALBERICI).
Message format is:
Host sends: **[Dir] [02] [01] [D2] [Coin pos.] [Sort.Path] [Chk]**
Coin s. responds: **[01] [00] [Dir] [00] [Chk]** ACK if sorter path is possible to set
Coin s. responds: **[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]!

If host sends command to modify sorter path that is not existent or for coin not programmed, the coin selector will respond with message NAK.
Example of message string for coin selector(*address 2*) redirection of coin **pos. 1** in to **path 2** is:
Host sends: **[02] [02] [01] [D2] [01] [02] [26]**
Coin s. responds: **[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.
It is possible to program a different coil activation schematics by setting the sorter type.

6.5.2.23 Command 209 [hexD1], Request sorter paths
Host send one byte of coin position and coin selector responds with one byte of sorter path. Message format is:
Host sends: **[Dir] [01] [00] [D1] [Coin pos.] [Chk]**
Coin s. responds: **[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. responds: **[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. responds: **[01] [01] [02] [00] [02] [FA]**
If host request sorter path for non programmed coins or non existent position (Position higher than 16)**, the coin selector will respond with message NAK !**

6.5.2.26 Command 196 [hexC4], Request creation date
Coin selector responds 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*) format (For details see cctalk protocol, document cctalk44-2.pdf).
Message format is:
Host sends: **[Dir] [00] [01] [C4] [Chk]**
Coin s. responds: **[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. responds: **[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 responds with two byte of data that represent codified date of last modification of software3 (Up-grade of FLASH program memory). Date of modification is codified also in RTBY format.
Message format is:
Host sends: **[Dir] [00] [01] [C3] [Chk]**
Coin s. responds: **[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. responds: **[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 responds 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. responds: **[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. responds: **[01] [03] [02] [00] [00] [00] [00] [FA]**

6.5.2.29 Command 193 [hexC1], Request fraud counter
Coin selector responds 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 (Coins with close recognition parameters sometime called “killer coin or channel”). Message format is:
Host sends: **[Dir] [00] [00] [C1] [Chk]**
Coin s. responds: **[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. responds: **[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. responds: **[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 sends one byte data of coin position and coin selector responds 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. responds: **[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. responds: **[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. responds: **[01] [06] [02] [00] [2E][2E][2E][2E][2E][2E] [E3]** Coin not programed

6.5.2.33 Command 176 [hexB0], Request alarm counter
Coin selector responds with one bytes of alarm counter data.
Alarm counter is set to zero after power up or reset command. It is incremented each time a coin acceptor detect any type of erroneous coin acceptance. Possible alarms: Coin direction error(Jojo), coin too slow (Coin jam) or coin too fast
Message format is:
Host sends: **[Dir] [00] [00] [B0] [Chk]**
Coin s. responds: **[01] [01] [Dir] [00] [Cunt] [Chk]**
Example of message string for coin selector(*address 2*) after power-up is:
Host sends: **[02] [00] [01] [B0] [4D]**
Coin s. responds: **[01] [03] [02] [00] [00] [FC]**

6.5.2.34 Command 173 [hexAD], Request thermistor reading
Some coin selectors AL66 (Coin selectors for use in extreme ambient temperature conditions, i.e. for external use) include a built-in linear temperature sensor.
Using this command is possible to read temperature on surface of coin selector PCB.
If temperature sensor is not built in coin selector will not responds to this command.
Temperature sensor is linear type, with 1 unit change for one degree Celsius change.
For 0°C value will be dec**[50]**, for ie. 25°C it will be dec**[75]**, for -10°C it will be dec**[40]** and for 50°C it will be dec**[100]**. Message format is:
Host sends: **[Dir] [00] [00] [AD] [Chk]**
Coin s. responds: **[01] [01] [Dir] [00] [Temp] [Chk]**
Example of message string for coin selector(*address 2*) at ambient temperature of 25°C is:
Host sends: **[02] [00] [01] [AD] [50]**
Coin s. responds: **[01] [01] [02] [00] [4B] [B1]**

6.5.2.35 Command 170 [hexAA], Request base year
Coin selector responds 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. responds: **[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. responds: **[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 bus 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 uses address 1 and address 0 is for broadcast message all commands that changes the address should not accept this settings.
All changes are stored in non-volatile memory, EEPROM !

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 responds 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]** Brodcast message
Coin s. responds: **Dly -> [Address]**
Example of message string for coin selector(*address 2*) is:
Host sends: **[00] [00] [01] [FD] [02]**
Coin s. responds: **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. responds: **Dly=1 s -> [FA]** Address is 250

6.5.3.2 Command 252 [hexFC], Address clash
Command Address clash has same responds from coin selector but host issue this command with specific device address. Coin selector responds with only one byte (*nonstandard 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. responds: **Random Dly -> [Address]**
Example of message string for coin selector(*address 2*) **AL06V-c** is:
Host sends: **[02] [00] [01] [FC] [01]**
Coin s. responds: **Random Dly -> [02]** Address is 2

6.5.3.3 Command 251 [hexFB], Address change (not supported by coin acceptors)
Command Address change is issued to a specified device only. Coin selector responds with ACK message. Message format is:
Host sends: **[Dir] [01] [01] [FB] [Address] [Chk]**
Coin s. responds: **[01] [00] [02] [00] [FD]** ACK
Example of message string for coin selector(*address 2*) change to address to 20:
Host sends: **[02] [01] [01] [FB] [14] [ED]**
Coin s. responds: **[01] [00] [02] [00] [FD]** ACK Address is now 20
Coin selector does not responds to attempt of change an address to 0 or 1.

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 programing!
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 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)!

6.5.4.1.1 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 some time 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 respond with **NAK** message.
If number of blocks to read exceed block range, coin selector will also responds 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 extend this number coin selector will responds 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. responds: **[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.
represent the number of data to 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. responds: **[01][10][02][00][89][87][B4][77][A7][9F][08][08][0C][16][0A][04][08][01][09][C8][52]**

6.5.4.1.3 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.

6.5.4.1.4 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. responds: **[01] [6] [Dir] [00] [data 1] [data 2] . . . [data 6] [Chk]**
42 20 x 3 byte of memory!
43 Special case write command
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. responds: **[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 our coin programming software.

6.5.4.1.5 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 knead some new function or improvement, which where 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 are 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!

6.5.4.1.6 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, inputloutput 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. responds: **[01][00][02][00] [FD]** ACK
After that, coin selector must be switched off/on!

6.6 Error Codes

NOTA BENE: I codici d'errore non sono presenti per default. E' possibile predisporre la gettoniera per trasmetterli, SE E SOLO SE CIO' VIENE RICHIESTO AL MOMENTO DELL'ORDINE..

ELENCO DEI CODICI D'ERRORE

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	08	Second close coin	Yes/both
11	0B	Sorter not ready	Yes
13	0D	Validation sensor not ready	Yes
16	10	Credit sequence error (Jojo)	No
18	12	Coin to fast (opto 2 minimum time not elapsed)	No
19	13	Coin to 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	-

7. Implementazione del Protocollo Criptato AES

I comandi ccTalk implementati in questa periferica sono quelli presenti sul documento "Elenco comandi del protocollo ccTalk per il mercato Italiano", legge 289 – comma 6" che specifica il pacchetto di comandi ccTalk Italy attualmente in uso (cfr. più sotto "ccTalk Italy communication protocol"), ma modificati in modo da rendere la periferica conforme ai nuovi requisiti di sicurezza stabiliti con il documento "Technical Table Report 2012, 3.3, 2nd edition - Peripherals 27.02.2013", allegato, a cui si rimanda per i dettagli.

COMANDI DA IMPLEMENTARE per L'OMOLOGAZIONE delle SCHEDE GIOCO "COMMA 6A +" (normativa NEWSLOT)
***COMMANDS TO BE IMPLEMENTED ON THE GAME-BOARDS for the* "COMMA 6A +" (NEWSLOT) CERTIFICATION**

E' riportato qui sotto lo schema riassuntivo dei comandi specifici del protocollo di cifratura Diffie-Hellmann AeS256, previsto dalla normativa New Slot. Inoltre, per la convenienza del programmatore, la descrizione dei comandi 235, 234, 206, e 200 viene fornita sia in lingua italiana che in lingua inglese.

For the convenience of the designer, this data sheet provides the sum-up of the specific commands relevant to the Diffie-Hellmann AeS256 encryption protocol, as required by the Italian New Slot regulations. The description of the non-encrypted commands 235, 234, 206 e 200 is given in both italian and english languages.

NOTA: per informazioni specifiche sul protocollo Diffie Hellmann come definito dal tavolo Tecnico 3.3, vedere il documento allegato "Technical table rev 3 3-2nd edition – Peripheric.pdf".
PLEASE NOTICE: the details concerning the Diffie Hellmann protocol can be found in the attached document "Technical table rev 3 3-2nd edition – Peripheric.pdf"

Code	Command header

HEX	HEADER	DESCRIPTION
EB	***235***	***Read DH public key***
EA	***234***	***Send DH public key***
CE	***206***	***Switch encryption AES key***
C8	***200***	***Request product parameters***
DE	***223***	***Modify inhibit and override registers***

DEFINIZIONE DEI COMANDI NON CRIPTATI

HEADER 235 (0xEB) "READ DH PUBLIC KEY"

Struttura del messaggio per la richiesta della chiave pubblica:
[Destination Address] [1] [Source Address] [235] [Mode] [CHK]

[mode]
0 - request status
1 - request public key

La risposta dello slave sara una delle seguenti a seconda del parametro sopra indicato:
- mode 0:[Destination Address] [Number Byte] [Source Address] [0] [Status] [CHK]
- mode 1:[Destination Address] [Number Byte] [Source Address] [0] [key 1] [key 2]… [key N] [CHK]
dove:
[Number Byte]= N/16 dove N e: 16 o 32 o 64 o 128 o 256 o 512

[status]
0 - shared key calculation in progress
1 - shared key ready

[key]: LSB first
La chiave pubblica inviata dallo slave sara di 256bit, quindi 32 byte trasmessi.

HEADER 234 (0xEA) "SEND DH PUBLIC KEY"

Struttura del messaggio per l'invio della chiave pubblica da parte dell'host:
[Destination Address] [Number Byte] [Source Address] [234] [key 1]
[key 2]… [key N] [CHK]
dove:
[Number Byte]= N/16 dove N e: 16 o 32 o 64 o 128 o 256 o 512
[key]:LSB first
La chiave pubblica inviata dall'host sara di 256bit quindi 32 byte trasmessi.
La risposta dello slave sara un ACK alla ricezione della chiave.

HEADER 200 (0xC8) "REQUEST PRODUCT PARAMETERS"

Comando per richiedere i parametri della periferica e la sua identificazione.
L'host invia la sua massima lunghezza della chiave come parametro del comando:
[Destination Address] [1] [Source Address] [200] [Maximum DH key length] [CHK]
Lo slave risponde:
[Destination Address] [42] [Source Address] [0] [Maximum DH key length] [Maximum Baud rate supported] [FW rev 1] …[FW rev 8] [ACMI protocol rev 1] [ACMI protocol rev 2] [Serial number 1]…[Serial number 4] [Manuf 1]…[Manuf 16] [Product code 1]…[Product code 8] [DH counter 1] [DH counter 2] [CHK]

- [Maximum DH key length]: 2 (256 bit);
- [Maximum Baud rate supported]: 1 (9600);
- [FW rev 1] …[FW rev 8]: "4.02 ";
- [ACMI protocol rev 1] [ACMI protocol rev 2]: 33;
- [Serial number 1]…[Serial number 4]: 0-4294967296, LSB->MSB;
- [Manuf 1]…[Manuf 16]: "ALBERICI ";
- [Product code 1]…[Product code 8]: "AH MNMCR";
- [DH counter 1] [DH counter 2]: 0-65535, LSB->MSB;

STRUTTURA DEI MESSAGGI CRIPTATI

Struttura dei messaggi con dati e senza dati:
[Destination address]
[Nr. of AES bytes]
[Source address]
[Header]
[AES 1]
...
[AES n]
[Checksum]
Il numero di AES byte e un multiplo di 16 , l'header in questo caso e una costante: '220' per la richiesta, che identifica che il blocco e criptato, e '0' per la risposta, il checksum semplice a 8 bit.

Il blocco dei dati AES per un messaggio con dati , prima della criptazione, ha la seguente struttura:
[nr. bytes] [header] [challenge lsb] [challenge msb][data 1] [data 2] ... [data n] [filler 1] [filler 2]... [filler k] [crc lsb] [crc msb]

nr. bytes = number of data bytes **n**
header = ccTalk command
challenge = number sent by the host
data = ccTalk data
filler = 0 for byte padding (*ignore*)
crc = error-detecting code (CRC-CCITT,polinomial x16+x12+x5+1,initial 0x0000)

Il blocco dei dati AES per un messaggio senza dati , prima della criptazione, ha la seguente struttura:
[nr. bytes] [header] [challenge lsb] [challenge msb] [filler 1] [filler 2] ... [filler k] [crc lsb] [crc msb]
1. nr. bytes = **0**
2. header = ccTalk command or "0=ACK", "5=NACK", "6=BUSY"
3. challenge = number sent by the host
4. filler = 0 for byte padding (ignore)
5. crc = error-detecting code (CRC-CCITT,polinomial x16+x12+x5+1,initial 0x0000)

NEW COMMANDS SPECIFICATION

Header 235 - Read DH public key
Not encrypted command
The command sent by the host:
[Destination Address] [1] [Source Address] [235] [Mode] [CHK]
[mode]
0 - request status
1 - request public key
The answer sent by slave when [Mode]= 0 can be:
[Destination Address] [Number Byte] [Source Address] [0] [Status] [CHK]
Where: [status]
0 - shared key calculation in progress
1 - shared key ready
The answer sent by slave when [Mode]= 1 can be:
[Destination Address] [Number Byte] [Source Address] [0] [Status] [CHK]
Where: [status]
0 - shared key calculation in progress
Or
[Destination Address] [Number Byte] [Source Address] [0] [key 1] [key 2]… [key N] [CHK]
Where:
[Number Byte]= N/16 where N is: 32 or 64 or 128 or 256 or 512
[key]
The key length depends on the maximum prime size supported by the hardware.
If mode = 1 then the host can request the DH public key from the peripheral. This is returned LSB first. The length will depend on the processing capability of the peripheral e.g. 128 bytes for 1024 bit DH; it may be much shorter though. Once the host sends the DH host public key using the command below, it can poll the peripheral status to see when it has performed all the calculations necessary to create the shared key for AES-256 encryption.
The host sends [mode]
= 0 and the peripheral returns a status of 1 when it is ready.
The host should not attempt any other command until the shared key is ready.

Header 234 - Send DH public key
Not encrypted command
The command sent by the host:
[Destination Address] [Number Byte] [Source Address] [234] [key 1] [key 2]… [key N] [CHK]
Where:
[Number Byte]= N/16 where N is: 32 or 64 or 128 or 256 or 512
[key] is sent LSB first
Received data : ACK
This command transfers the DH host public key to the peripheral. **The host, for an optimized security level, should transfer the key with the maximum length supported either the host and the slave (use the command 200 to handshake the DH key length), but also a DH key length smaller is acceptable.**
An ACK is returned immediately and the calculation of a shared key in the peripheral is then started. The host can use the previous command in a status polling mode to find out when the calculation is complete.

Header 223 - Modify inhibit and override registers ***(valid only for coin acceptor - comando valido solo per gettoniera)***
New encrypted command
[Destination Address] [32] [Source Address] [220] [AES data 1]……[AES data 32] [CHK]
The format of Data Blocks [AES data 1]…..[AES data 32] when decrypted has this format:
[Number Data Bytes= 13] [223] [Challenge LSB] [Challenge MSB]
[Inhibit mask 1] [Inhibit mask 2] [Inhibit mask 3] [Inhibit mask 4]
[Cash value 1] [Cash value 2] [Cash value 3] [Cash value 4]
[Coin count 1] [Coin count 2] [Coin count 3] [Coin count 4]
[Sorter override mask]
[Filler 1= 0]…[Filler 13= 0] [CRC LSB] [CRC MSB]
Received data : ACK
This command controls coin acceptance in the coin acceptor; both whether a coin is accepted or rejected, and where the coin is routed (e.g. hopper or cashbox).

The inhibit mask controls which coins are enabled or disabled. The cash value sets a maximum value that can be accepted (coins are disabled by the peripheral accordingly). The coin count sets a maximum number of coins that can be accepted (coins are disabled by the peripheral accordingly). The sorter override mask forces subsequent hopper coins to cashbox.
[inhibit mask]
B0 - Coin Type 1 (0 - coin inhibited, 1 - coin enabled)
B1 - Coin Type 2
B2 - Coin Type 3
…
B31 - Coin Type 32
Each bit controls a coin inhibit. Inhibit mask 1 is the LSB and controls coin types 1 to 8; inhibit mask 4 controls coin types 25 to 32. There is support for up to 32 coin types. Refer to the product manual to see how many coin types are supported. The default operation is to have all inhibit mask bytes set to 255. There is no problem enabling unsupported coin types.
[cash value]
0 - disabled
1 or more - maximum cash value which can be accepted
This is a single value in 4 bytes (0 to 4,294,967,295). Cash value 1 is the LSB.
The cash value is specified in terms of the lowest monetary unit in that currency e.g. cents, pence, pesos etc.
The coin acceptor will accept coins up to the value specified (and using the inhibit mask supplied). Inserted coins which take the cumulative total above this amount will be rejected regardless of the inhibit status. Event code 1,'Reject coin', will be returned for these coins. Every time the host sends this command the cumulative total is reset to zero.
The use of this field alleviates the problems with latency while waiting for a serial credit and trying to apply an inhibit mask before the next coin is entered. Latency is usually worse when high-strength encryption is being used.
[coin count]
0 - disabled
1 or more - maximum number of coins which can be accepted
This is a single value in 4 bytes (0 to 4,294,967,295). Coin count 1 is the LSB.
The coin acceptor will accept coins up to the total number specified (and using the inhibit mask supplied) and then reject all subsequent coins. Event code 1, 'Reject coin', will be returned for these coins. Every time the host sends this command the cumulative coin count is reset to zero.
The use of this field alleviates the problems with latency while waiting for a serial credit and trying to apply an inhibit mask before the next coin is entered. Latency is usually worse when high-strength encryption is being used.
[sorter override mask]
B0 - Sorter Path 1 (0 - cashbox sorting, 1 - normal sorting)
B1 - Sorter Path 2
B2 - Sorter Path 3
B3 - Sorter Path 4
B4 - Sorter Path 5
B5 - Sorter Path 6
B6 - Sorter Path 7
B7 - Sorter Path 8
Each bit controls a sorter path divert; not a coin type. Up to 8 sorter paths (= hoppers) are supported. If a sorter path has an override then the coin will be sent to an auxiliary or default path; usually the cashbox.
The default operation is to have the sorter override mask set to 255.

Header 206 - Switch encryption AES key

New encrypted command
[Destination Address] [48] [Source Address] [220] [AES data 1]……[AES data 48] [CHK]
The format of Data Blocks [AES data 1]…..[AES data 32] when decrypted has this format:
[Number Data Bytes= 32] [206] [Challenge LSB= 0] [Challenge MSB= 0]
[New key 1] …[New key 32]
[Filler 1= 0]…[Filler 10= 0] [CRC LSB] [CRC MSB]
Received data : ACK
The device changes the key to the new value only when it decrypts the packet and confirms the CRC is correct.

Header 200 - Request product parameters
Not encrypted command
The command sent by the host
[Destination Address] [1] [Source Address] [200] [Maximum DH key length] [CHK]
The answer sent by slave is:
[Destination Address] [42] [Source Address] [0] [Maximum DH key length] [Maximum Baud rate supported] [FW rev 1]…[FW rev 8] [ACMI protocol rev 1] [ACMI protocol rev 2] [Serial number 1]…
[Serial number 4] [Manuf 1]…[Manuf 16] [Product code 1]…[Product code 8] [DH counter 1] [DH counter 2] [CHK]
Where:

[Maximum DH key length] follows the table below:

DH key lenght	Maximum DH key length value
256	2
512	3
1024	4
2048	5
4096	6

[Maximum Baud rate supported] follows the table below:

Baud rate	Maximum Baud rate value
9600	1
14400	2
19200	3
38400	4

[FW rev] is 8 x ASCII characters
[ACMI protocol] is a 2 bytes encoding of the release number of this document (e.g. referring the document 3.0 the [ACMI protocol rev 1]= 3 and [ACMI protocol rev 2]= 0)
[Serial Number] is 4 bytes binary. Range 0-4294967296. Serial number 1 = LSB, Serial number 4 = MSB.
The serial number is numerical only and must be unique for a given manufacturer / product name.
[Manuf] is 16 x ASCII characters. Manufacturer name. Upper case only.
[Product code] is 8 x ASCII characters. This is the product name. It may be shortened to fit within the 8 characters. Left justified and right padded with spaces (ASCII 32). May contain upper / lower case.
[DH counter] is the counter of the total DH key exchange procedure completed with success in the whole lifecycle of the device. Range 0-65535. DH counter 1 = LSB, DH counter 2 = MSB.

8. Smaltimento

IMPORTANTE! Osservare le norme vigenti per lo smaltimento degli imballi e per la rottamazione del prodotto!
Rif.: D. Lgs. 151/2005 - Direttiva 2002/96/EC

INFORMAZIONE AGLI UTENTI SUL DL 151/2005
ai sensi dell'art. 13 del Decreto legislativo 25 Luglio 2005, n. 151 "Attuazione delle Direttive 2002/95/CE, 2002/96/CE, relative alla riduzione dell'uso di sostanze pericolose nelle apparecchiature elettriche ed elettroniche, nonché allo smaltimento dei rifiuti":

In adempimento a quanto stabilito dall'Art. 13 del citato decreto, si trasmettono all'Utente del prodotto le informazioni seguenti, attinenti alla salvaguardia dell'ambiente, e relative allo smaltimento dei RAEE (Rifiuti di Apparecchiature Elettriche ed Elettroniche).

Questa apparecchiatura è classificabile come Distributore Automatico di denaro contante o gettoni, e quindi appartiene alla categoria 10.2 prevista dal decreto legislativo sopra citato, allegato 1B.

1. Alcune parti della presente apparecchiatura, a causa di specifiche sostanze presenti nei componenti elettronici, potrebbero arrecare effetti potenzialmente dannosi per l'ambiente e per la salute umana, se non smaltite conformemente alle norme in oggetto o se usate in maniera impropria.

2. Sulla presente apparecchiatura è applicato un simbolo che rappresenta il cassonetto barrato: esso indica che il prodotto, una volta giunto alla fine della sua vita utile, deve essere raccolto separatamente dagli altri rifiuti. E' fatto espresso divieto di smaltire la presente apparecchiatura come rifiuto urbano: essa deve essere tassativamente sottoposta a procedura di raccolta separata.

3. A questo scopo, sono stati disposti appositi centri di raccolta delle apparecchiature elettriche ed elettroniche a cui consegnare i prodotti da smaltire (RAEE).

4. Il Rivenditore, a fronte dell'acquisto di un nuovo prodotto equipollente, ritirerà gratuitamente l'apparecchiatura dismessa per inoltrarla ad uno dei centri di raccolta, trattamento e smaltimento ecologicamente compatibili.

5. La negligenza nell'applicazione della Direttiva comporta l'applicazione delle segg.

SANZIONI AMMINISTRATIVE:

i. Il Rivenditore che, nell'ipotesi di cui all'articolo 6, comma 1, lettera b), non ritira a titolo gratuito un'apparecchiatura elettrica o elettronica, è punito, per ogni apparecchiatura non ritirata o ritirata a titolo oneroso con sanzione amministrativa pecuniaria da euro 150 ad euro 400.

ii. Il produttore che, senza avere provveduto alla iscrizione presso la Camera di Commercio ai sensi dell'articolo 14, comma 2, immette sul mercato AEE, è punito con la sanzione amministrativa pecuniaria da euro 30.000 ad euro 100.000.

iii. Il produttore che, entro il termine stabilito col decreto di cui all'articolo 13, comma 8, non comunica al registro nazionale dei soggetti obbligati allo smaltimento dei RAEE le informazioni, relative ai dati di vendita ecc, di cui all'articolo 13, commi 6 e 7, ovvero le comunica in modo incompleto o inesatto, è punito con la sanzione amministrativa pecuniaria da € 2.000 ad € 20.000.

iv. Il produttore che non provvede ad organizzare il sistema di raccolta separata dei RAEE professionali di cui all'articolo 6 - comma 3 ed i sistemi di ritiro ed invio, di trattamento e di recupero dei RAEE, di cui agli articoli 8 - comma 1 e 9 - comma 1, 11 - comma 1 e 12 - commi 1, 2 e 3, e fatti salvi, per tali ultime operazioni, gli accordi eventualmente conclusi ai sensi dell'articolo 12 - comma 6, è punito con la sanzione amministrativa pecuniaria da euro 30.000 ad euro 100.000.

v. Il produttore che, dopo il 13 agosto 2005, nel momento in cui immette una apparecchiatura elettrica od elettronica sul mercato, non provvede a costituire la garanzia finanziaria di cui agli articoli 11 – comma 2, o 12 - comma 4, è punito con la sanzione amministrativa pecuniaria da euro 200 ad euro 1.000 per ciascuna apparecchiatura immessa sul mercato.

vi. Il produttore che non fornisce, nelle istruzioni per l'uso di AEE, le informazioni agli Utenti di cui all'articolo 13 - comma 1, è punito con la sanzione amministrativa pecuniaria da € 200 ad € 5.000.

vii. Il produttore che, entro un anno dalla immissione sul mercato di ogni tipo di nuova AEE, non mette a disposizione dei centri di re-impiego e degli impianti di trattamento e di riciclaggio le informazioni di cui all'articolo 13 - comma 3, è punito con la sanzione amministrativa pecuniaria da euro 5.000 ad euro 30.000.

viii. Il produttore che, dopo il 13 agosto 2005, immette sul mercato AEE prive della indicazione o del simbolo del "cassonetto sbarrato" di cui all'articolo 13, commi 4 e 5, è punito con la sanzione amministrativa pecuniaria da euro 200 ad euro 1.000 per ciascuna apparecchiatura immessa sul mercato. La medesima sanzione amministrativa pecuniaria si applica laddove i suddetti indicazione o simbolo non siano conformi ai requisiti stabiliti all'articolo 13, commi 4 e 5.

ix. Fatte salve le eccezioni di cui all'articolo 5, comma 2, chiunque, dopo il 1° luglio 2006, immette sul mercato AEE nuove contenenti le sostanze di cui all'articolo 5, comma 1 o le ulteriori sostanze individuate ai sensi dell'articolo 18, comma 1, è punito con la sanzione amministrativa pecuniaria da euro 50 ad euro 500 per ciascuna apparecchiatura immessa sul mercato oppure da euro 30.000 ad euro 100.000.

9. Condizioni di Garanzia

Il Cliente della Alberici S.p.A. beneficia di 12 mesi di garanzia, che decorrono dalla settimana di produzione del prodotto. Qualunque comunicazione al riguardo va accompagnata dal numero di matricola dell'apparecchio e da copia della fattura di vendita.
Per ottenere la riparazione in garanzia, il prodotto deve essere inviato, franco destino, alla sede della Alberici S.p.A., accompagnato dai seguenti documenti:
- copia della fattura di vendita
- documento di trasporto (DDT) riportante chiaramente la motivazione "reso per riparazione in garanzia"
- una relazione dettagliata del tipo di problema riscontrato e delle circostanze in cui si verifica.

Prima di inviare il prodotto, raccomandiamo comunque di contattarci al numero (+39) 051 944300; è spesso possibile risolvere le eventuali anomalie per via telefonica, evitando inutili perdite di tempo e costi aggiuntivi.
La Alberici S.p.a. si riserva di verificare che i termini di garanzia siano applicabili, ovvero che il problema non derivi da:
- danni da trasporto
- danni derivati da installazione impropria o configurazione errata
- installazione in complessi non a norma civile o elettrica
- manomissione intenzionale o involontaria
- utilizzazione o manutenzione errate o negligenti del prodotto
- inosservanza delle Precauzioni d'uso (cfr. Cap. 4)
- calamità naturali, atti vandalici, azioni dolose o colpose

La mancanza di entrambe le etichette (esterna ed interna) annulla il diritto alla garanzia.

Il prodotto, una volta riparato, viene reso in porto assegnato o con trasporto pagato in anticipo.

10. Servizio al Cliente

La Alberici S.p.a. è lieta di fornire tutto il necessario supporto informativo con riguardo sia all'uso e manutenzione ordinaria, sia all'assistenza tecnica.
Se la richiesta riguarda un problema tecnico, raccomandiamo di annotare e comunicare i seguenti dati:
- modello dell'apparecchio
- n° di serie
- versione della scheda di controllo (l'etichetta identificativa si trova sul retro della scheda stessa)
- descrizione precisa dell'anomalia

Per ottenere supporto telefonico, vi preghiamo di chiamare il numero (+39) 051 944300, specificando se si tratta di un quesito tecnico o di una richiesta relativa all'uso del prodotto, e di precisare il prodotto oggetto della richiesta.

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,**

D I C H I A R A

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

Modello	Configurazione	Protocollo	N° di serie e/o matricola
		☐ **ccTalk**	
☐ **AL66 FG**	☐ **V** ☐ **S** ☐ **I**	☐ **Pulse**	_____
		☐ **Aes256 DH**	

Essendo realizzato conformemente al modello prototipo campione denominato AL66 FG V, finito di testare positivamente ai fini EMC e LVD (rapporto 7037CE-AL66.doc) il 25/09/2015, ~~da~~lla STP S.r.l., con sede legale in via P.F. Andrelini, 42, 47121 Forlì (FC), Italia e sede operativa in via San Donnino, 4, 40127 Bologna (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

Alberici S.p.A.
Progettazione e produzione sistemi di pagamento, accessori per videogames e vending machines
Via Ca' Bianca, 421, 40024 Castel San Pietro Terme (BO), Italia.
Telefono:+39-(0)51-944300 r.a. – Fax:+39-(0)51-944594 – P.Iva:00627531205
E-mail: info@alberici.net – Url: http://www.alberici.net

NOTA

La Alberici S.p.A. si riserva il diritto di apportare modifiche alle specifiche tecniche dell'apparecchiatura descritta in qualunque momento e senza preavviso, nell'ambito del perseguimento del miglioramento continuo del proprio prodotto.

Alberici

Progettazione e produzione di sistemi di pagamento, accessori per videogames e macchine vending
Design and manufacture of payment systems, accessories for videogames and vending machines

Via Ca' Bianca 421
40024 Castel San Pietro
Terme (BO) – ITALY

Tel. + 39 051 944 300
Fax. + 39 051 944 594

http://www.alberici.net

info@alberici.net