SMART Payout & SMART Hopper

eSSP <-> ccTalk converter Specification

INTELLIGENCE IN VALIDATION

Innovative Technology
INTELLIGENCE IN VALIDATION

Automated Transactions
INTELLIGENCE IN VALIDATION

Bellis Technology
INTELLIGENCE IN VALIDATION

MAIN HEADQUARTERS

Innovative Technology Ltd
Derker Street – Oldham – England - OL1 4EQ
Tel: +44 161 626 9999 Fax: +44 161 620 2090
E-mail: support@innovative-technology.co.uk
Web site: www.innovative-technology.co.uk

GROUP

Innovative Technology
INTELLIGENCE IN VALIDATION

Automated Transactions
INTELLIGENCE IN VALIDATION

Bellis Technology
INTELLIGENCE IN VALIDATION

EMAIL CONTACTS

BRAZIL
suporte@bellis-technology.com.br

CHINA
support@innovative-technology.co.uk

COLOMBIA
support@automated-transactions.net

GERMANY
support@automated-transactions.de

UNITED KINGDOM
support@innovative-technology.co.uk

SPAIN
soporte@automated-transactions.es

UNITED STATES OF AMERICA
supportusa@bellis-technology.com

REST OF THE WORLD
sales@innovative-technology.co.uk

REVISION HISTORY

Revision	Description	Date
1	First Release	08/04/2010

CONTENTS

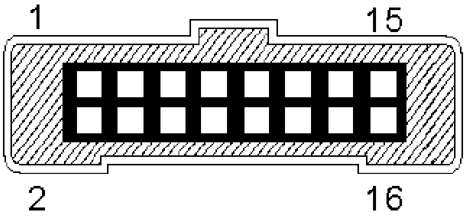
1 INTRODUCTION

This specification documents the commands used by a device to convert the eSSP protocol used by the SMART Hopper and SMART Payout commands in to ccTalk commands transported over an Italian style ccTalk packets. This removes the security provided by the eSSP encryption layer and suitable physical security should be employed to protect the ccTalk bus.

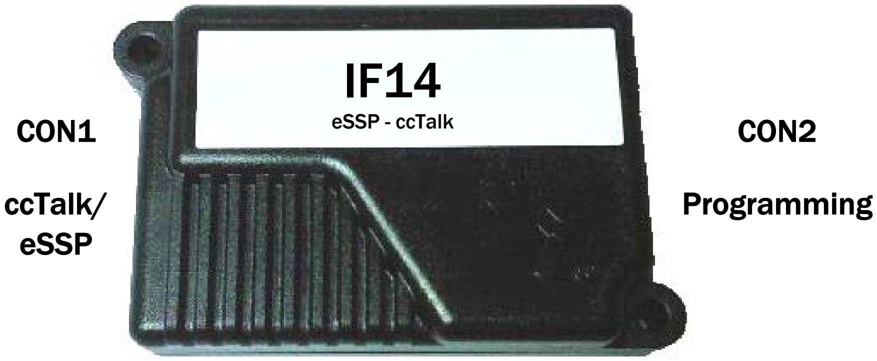
This document should be read in conjunction with the full ccTalk specification. Only the specific commands for the SMART Hopper and SMART Payout are documented. The note validator commands conform to the standard specification, and the SMART Payout commands are an extension to this device on the same address.

2 CONNECTIONS

The pin connections on the IF14 are numbered as looking at the connector:



The connectors are identified as CON1 and CON2.



2.1 CON1

CON1 on the left of the device is the connector used in normal operation of the IF14. The pins are connected as follows:

PIN NUMBER	DESCRIPTION
1 & 5	ccTalk bus connection Pins 1 and 5 should be linked together
2	eSSP TX
6	eSSP RX
15	+12v
16	0v

2.2 CON2

CON2, on the right of the interface is used for programming the IF14. The pin connections are as follows:

PIN NUMBER	DESCRIPTION
1	SSP RX
5	SSP TX
15	+12v
16	0v

Please note the SSP TX & RX pins are inverted with respect to other ITL products.

3 REPRESENTATIONS

3.1 MONATARY VALUES

Values are represented as 32 bit unsigned integer (4 bytes) and in the lowest value of currency. For example €125.65 would be 0x00003115. When sending or receiving a value the Least significant byte is sent first. So in this example [0x15] [0x31] [0x00] [0x00] will be sent.

3.2 IDENTIFING COINS/NOTES

Each type of coin is identified by its value and represented using the standard format outlined above. So the values for euro coins are:

COIN / NOTE	HEX VALUE	DATA TO SEND
€0.01	0x00000001	[0x01] [0x00] [0x00] [0x00]
€0.02	0x00000002	[0x02] [0x00] [0x00] [0x00]
€0.05	0x00000005	[0x05] [0x00] [0x00] [0x00]
€0.10	0x0000000A	[0x0A] [0x00] [0x00] [0x00]
€0.20	0x00000016	[0x16] [0x00] [0x00] [0x00]
€0.50	0x00000032	[0x32] [0x00] [0x00] [0x00]
€1.00	0x00000064	[0x64] [0x00] [0x00] [0x00]
€2.00	0x000000C8	[0xC8] [0x00] [0x00] [0x00]
€5.00	0x000001F4	[0xF4] [0x01] [0x00] [0x00]
€10.00	0x000003E8	[0xE8] [0x03] [0x00] [0x00]
€20.00	0x000007D0	[0xD0] [0x07] [0x00] [0x00]
€50.00	0x00001388	[0x88] [0x13] [0x00] [0x00]
€100.00	0x00002710	[0x10] [0x27] [0x00] [0x00]
€200.00	0x00004E20	[0x20] [0x4E] [0x00] [0x00]
€500.00	0x0000C350	[0x50] [0xC3] [0x00] [0x00]

4 COMMAND SUMMARY

COMMAND	HEADER	PARAMETERS	REPLY	SH	SP
Simple Poll	254	None	ACK	Y	Y
Request Equipment Category	245	None	'SMART_HOPPER' 'SMART_PAYOUT'	Y	Y
Request Product Code	244	None	'SH3' or 'SP1'	Y	Y
Request manufacturer ID	246	None	'ITL'	Y	Y
Request Software Version	241	None	XX.YY	Y	Y
Request Comms Revision	004	None	X.Y	Y	Y
Reset Device	001	None	ACK	Y	Y
Request Serial Number	242	None	3 byte serial No	Y	Y
Enter New Pin	219	Pin1, Pin2, Pin3, Pin4	ACK	Y	Y
Enter Pin	218	Pin1, Pin2, Pin3, Pin4	ACK	Y	Y
Request Data Storage Av.	216	None	00000	Y	Y
Request Option Flags	213	None	3 (stacker & escrow)	N	Y
Modify Bill Operating Tbl	153	Escrow & Stacker	ACK	N	Y
Request Inhibits	230	None	InhibitLow, InhibitHigh	N	Y
Request Build Code	192	None	161209	Y	Y
Request Last Mod Date	195	None	00	Y	Y
Request Address Mode	169	None	1	Y	Y
Read Buffered Bill Events	159	None	10000000000	Y	Y
Route Bill	154	0/1	ACK/254	N	Y
Switch Encryption Code	137	3 bytes Encryption key	ACK	Y	Y
Set Routing	020	Route, Value	ACK	Y	Y
Get Routing	021	Value	Route for value.	Y	Y
Payout Amount	022	Value	ACK	Y	Y
Float	023	Min Payout, Value	ACK	Y	Y
Empty	024	None	ACK	Y	Y
Get Minimum Payout	025	None	Min Payout Value	Y	Y
Get Coin/Note Amount	026	Value	Count of coin/note	Y	Y
Set Coin/Note Amount	027	Value, Count to add	ACK	Y	N
Get Device Setup	028	None		Y	N
Request Status	029	None	Status	Y	Y

5 COMMAND DETAILS

5.1 SIMPLE POLL [0XFE = 254]

Send: [Addr] [00] [01] [FE] [Chk]
Reply: [01] [00] [Addr] [00] [Chk]
Where: Addr = Address of SMART Hopper or SMART Payout

Command to check the correct operation of the communication and to confirm the presence in the bus of a Smart Hopper or SMART Payout. If no reply is received to the request sent (Reception timeout in Machine), it will indicate that the SMART Hopper or SMART Payout is faulty or not connected. All the ccTalk peripherals must respond to a 'Simple Poll', regardless of the ccTalk communication protocol level that has been implemented.

5.2 REQUEST EQUIPTMENT CATEGORY ID [0XF5 = 245]

Send: [Addr] [00] [01] [F5] [Chk]
Reply: [01] [12] [Addr] [00] ['S'] ['M'] ['A'] [...] ['E'] ['R'] [Chk] **or**
Reply: [01] [12] [Addr] [00] ['S'] ['M'] ['A'] [...] ['U'] ['T'] [Chk]
Where: Addr = Address of SMART Hopper or SMART Payout

This command permits reception from the Hopper/Payout of the chain of characters that identifies the type of device in question. In the case of the SMART Hopper 'SMART_HOPPER' is received and 'SMART_PAYOUT' for the SMART Payout.

5.3 REQUEST PRODUCT CODE [0XF4 = 244]

Send: [Addr] [00] [01] [F4] [Chk]
Reply: [01] [01] [Addr] [00] ['S'] ['H'] ['3'] [Chk] or
Reply: [01] [01] [Addr] [00] ['S'] ['P'] ['1'] [Chk]
Where: Addr = Address of SMART Hopper or SMART Payout

With this command, the corresponding Hopper/Payout sends the machine the product code of the device. The complete identification of the product can be determined by the use of the [**Request product code**] command followed by the [**Request build code**] command. In the case of the SMART Hopper, it responds: "SH3" and "SP1" for the SMART Payout.

5.4 REQUEST MANUFACTURER ID [0XF6 = 246]

Send: [Addr] [00] [01] [F6] [Chk]
Reply: [01] [03] [Addr] [00] ['I'] ['T'] ['L'] [Chk]
Where: Addr = Address of SMART Hopper or SMART Payout

With this command, the corresponding Hopper/Payout sends the Machine the identification of the device manufacturer; 'ITL' is sent in response.

5.5 REQUEST SOFTWARE REVISION [0XF1 = 241]

Send: [Addr] [00] [01] [F1] [Chk]
Reply: [01] [02] [Addr] [00] [XX] [YY] [Chk]
Where: XX = before point, YY = after point
Addr = Address of SMART Hopper or SMART Payout

With this command, the corresponding product sends the machine the present software version. In the case of the SMART Hopper and Payout, the format XX.YY is used.

5.6 REQUEST COMMS REVISION [0X04 = 4]

Send: [Addr] [00] [01] [04] [Chk]
Reply: [01] [03] [Addr] [00] [Data 1] [Data 2] [Data 3] [Chk]
Where:
Data 1 = Level of the communication protocol
Data 2 = Part prior to the dot of the communication software version
Data 3 = Part following the dot of the communication software version.
Addr = Address of SMART Hopper or SMART Payout

As a reply to this command, the SMART Payout or Hopper sends the implementation level of the **ccTalk** protocol and the communication software version. If this version is 3.6, the `main revision' will be 3 and the 'minor revision' will be 6.

5.7 RESET DEVICE [0X01 = 1]

This command causes the device to carry out software reset. The device sends a positive acknowledgement string immediately before making the reset.

5.8 REQUEST SERIAL NUMBER [0XF2 =242]

Send: [Addr] [00] [01] [F2] [Chk]
Reply: [01] [03] [Addr] [00] [Serial 1 - LSB] [Serial 2] [Serial 3 - MSB] [Chk]
Where: Addr = Address of SMART Hopper or SMART Payout

In reply to this command, the device sends the serial number in a 3-byte code. This command has been implemented in order for the machine to include the constant safety code necessary to be able to make the coin payments.

5.9 ENTER NEW PIN NUMBER [0XDB = 219]

Send: [Addr] [04] [01] [DB] [PIN 1] [PIN 2] [PIN 3] [PIN 4] [Chk]
Reply: [01] [03] [Addr] [00] [Chk]
Where: Addr = Address of SMART Hopper or SMART Payout

All the commands implemented in the SMART Hopper or SMART Payout can be protected by a PIN. The present PIN number can be changed with this command, but as it is a command that is also protected by a PIN, the present PIN number must first be introduced. If a ccTalk command is executed without having entered the correct PIN number (including this command), there will be no reply from the device. The PIN number is a 32-bit binary code (4,294,967,296 combinations). Modifying the PIN number to the value 0 deactivates the protection by PIN.

5.10 ENTER PIN NUMBER [0XDA = 218]

Send: [Addr] [04] [01] [DA] [PIN 1] [PIN 2] [PIN 3] [PIN 4] [Chk]
Reply: [01] [03] [Addr] [00] [Chk]
Where: Addr = Address of SMART Hopper or SMART Payout

A PIN can protect all the commands implemented in the device. The present PIN number can be introduced with this command. If the present PIN is different from 0, it must be entered after every power-up and reset to be able to execute any command. If a ccTalk command is executed without having entered the correct PIN number, there will be no reply from the device. Whether the PIN number introduced is correct or incorrect, the device sends an affirmative reply.

5.11 SET ROUTING [0X14 = 20]

Send: [Addr] [04] [01] [14] [ROUTE] [Value 0 - LSB] [Value 1] [Value 2]
[Value 3 - MSB] [Chk]
Reply: [01] [03] [Addr] [00] [Chk]
Where: Addr = Address of SMART Hopper or SMART Payout

This command sets the route for a coin/note, the coin/note is identified by its value.
The coin/note route can be set to be recycled and used for payouts, or the coin/note can be set to always be routed into the cashbox. The value of the route is chosen from the table below:

Route	Code (HEX)
Coin/Note recycled and used for payouts	0x00
Coin/Note routed to cashbox	0x01

5.12 GET ROUTING [0X15 = 21]

Send: **[Addr] [04] [01] [15] [Value 0 - LSB] [Value 1] [Value 2] [Value 3 - MSB] [Chk]**
Reply: **[01] [01] [Addr] [00] [ROUTE] [Chk]**
Where: Addr = Address of SMART Hopper or SMART Payout

This command returns the route for a coin/note, the coin/note is identified by its value. The value of the route is chosen from the table above:

5.13 PAYOUT AMOUNT [0X16 = 22]

Send: **[Addr] [04] [01] [16] [Value 0 - LSB] [Value 1] [Value 2] [Value 3 - MSB] [Chk]**
Reply: **[01] [00] [Addr] [00] [Chk]** if OK or
Reply: **[01] [01] [Addr] [05] [Err] [Chk]** for an error
Where: Addr = Address of SMART Hopper or SMART Payout
Err is the returned error code.

This command will start the routine to payout the value specified. If the payout is possible the reply will be an ACK. If the command is not possible then the reply will be a NAK with one data byte containing one of the error codes shown below:

Error reason	Error code (HEX)
Not enough value in device	0x01
Cant pay exact amount	0x02
Device busy	0x03
Device disabled	0x04

5.14 FLOAT [0X17 = 23]

Send: **[Addr] [08] [01] [17] [Min_Pay 0 - LSB] [Min_Pay 1] [Min_Pay 2] [Min_Pay 3 - MSB] [Float 0 - LSB] [Float 1] [Float 2] [Float 3 - MSB] [Chk]**
Reply: **[01] [00] [Addr] [00] [Chk]** if OK or
Reply: **[01] [01] [Addr] [05] [Err] [Chk]** for an error
Where: Addr = Address of SMART Hopper or SMART Payout
Min_Pay is the value of the minimum payout required after the float operation.
Float is the value left in the device after the float operation.
Err is the returned error code.

This command will float the value of currency in the device to the level specified in the Value parameter. The value of the required minimum payout must also be specified; this is required to ensure the correct mix of coins/notes is retained. If the device does not have the coins/notes required for the float then an error from the table below will be returned.

Error reason	Error code (HEX)
Not enough value in hopper/payout	0x01
Cant float exact amount	0x02
Hopper Busy	0x03
Hopper Disabled	0x04

5.15 EMPTY [0X18 = 24]

Send: **[Addr] [00] [01] [18] [Chk]**
Reply: **[01] [00] [Addr] [00] [Chk]** if OK or
Where: Addr = Address of SMART Hopper or SMART Payout

This command will empty the SMART Hopper or SMART payout. All coins/notes are sent into the cashbox, the coin/note counters are set to zero. For the value sent to the cashbox the coin/note counters should be read using the command Get note/coin amount (0x1A/0x1B) before the empty command is sent.

5.16 GET MINIMUM PAYOUT [0X19 = 25]

Send: **[Addr] [00] [01] [19] [Chk]**
Reply: **[01] [04] [Addr] [00] [Min_Pay 0 - LSB] [Min_Pay 1] [Min_Pay 2] [Min_Pay 3 - MSB] [Chk]**
Where: Addr = Address of SMART Hopper or SMART Payout
Min_Pay is the value of the minimum payout possible.

This command will return the value of the minimum payout that is possible with the coins/notes that are currently in the device. This is effectively the value of the lowest coin/note in the device.

5.17 GET COIN/NOTE AMOUNT [0X1A = 26]

Send: **[Addr] [04] [01] [1A] [Value 0 - LSB] [Value 1] [Value 2] [Value 3 - MSB] [Chk]**
Reply: **[01] [02] [Addr] [00] [Count - LSB] [Count - MSB] [Chk]**
Where: Addr = Address of SMART Hopper or SMART Payout
Value is the value of the coin/note to get the counter for.
Count is the number of coins/notes of the specified value.

This command will return the count of the number of coins/notes of the value specified. The count is 16 bits long and is returned LSB first.

5.18 SET COIN/NOTE AMOUNT [0X1B = 27]

Send: **[Addr] [06] [01] [1B] [Value 0 - LSB] [Value 1] [Value 2] [Value 3 - MSB]**
[Count - LSB] [Count - MSB] [Chk]
Reply: **[01] [00] [Addr] [00] [Chk]**
Where: Addr = Address of SMART Hopper
Value is the value of the coin for the counter to increment.
Count is the number of coins of the specified value to add.

This command will **add** the number of coins specified in Count to the internal coin counter for the value specified in Value. If the count specified is Zero then the counter will be reset. It is not possible to set the absolute value to anything except zero in a single command. This command should be used each time the coin acceptor routes a coin into the SMART Hopper, or when the SMART Hopper has coins manually added. This command is invalid for the SMART Payout as the notes are automatically added to the counter by the note validator.

5.19 GET DEVICE SETUP [0X1C = 28]

Send: **[Addr] [00] [01] [1C] [Chk]**
Reply: **[01] [4+n*4] [Addr] [00] [Data 0] [Data 1] [….] [Data (4+n*4)] [Chk]**
Where: Addr = Address of SMART Hopper/SMART Payout.
.n is the number of coin/note types used.

This command will return the setup of the device, containing the currency, the number of different types of coin/note and the value of each coin/note that the device can handle. The length of the returned data will be 4+(n*4) bytes long, where n is the number of notes/coins that can be used. The format of the data is shown in the table below:

Data	Size / type	Notes
Country Code	3 bytes, string	See Country Code Table. E.g. EUR
Number of Coin Values (n)	1 byte, integer	
Coin Value 1	4 bytes, long	e.g. 0x01, 0x00, 0x00, 0x00 = 0.01
……….	……….	Repeat for each coin value
Coin Value n	4 bytes, long	

5.20 REQUEST STATUS [0X1D = 29]

Send: **[Addr] [00] [01] [1D] [Chk]**
Reply: **[01] [n] [Addr] [00] [Data 0] [Data 1] [....] [Data n] [Chk]**
Where: Addr = Address of SMART Hopper / SMART Payout.
.n is the number of data bytes from the table below

This command will return the status of the device and the progress of the current requested operation. After issuing any action commands this command should be used to track the status. The possible status codes are shown in the table below

EVENT/ STATE	EVENT CODE	DATA BYTES (N)
Idle	0x00	1
Dispensing	0x01, Current value dispensed	5
Dispensed	0x02, value dispensed	5
Coins Low	0x03	1
Empty	0x04	1
Jammed	0x05, value dispensed	5
Halted	0x06, value dispensed	5
Floating	0x07, value to cashbox	5
Floated	0x08, value to cashbox	5
Time Out	0x09, value dispensed	5
Incomplete Payout	0x0A, value dispensed, value requested	9
Incomplete Float	0x0B, value to cashbox, value requested	9
Cashbox Paid	0x0C, value to cashbox	5
Coin Credit	0x0D, value received	5
Emptying	0x0E	1
Emptied	0x0F	1
Fraud Attempt	0x10, Fraud Code	2
Disabled	0x11	1
Note Stored	0x12	1
Slave Reset	0x13	1
Note Read	0x14	5
Note Credit	0x15	5
Note Rejecting	0x16	1
Note Rejected	0x17	1
Note Stacking	0x18	1
Note Stacked	0x19	1
Note Path Jam	0x1A	1
Note Stack Jam	0x1B	1
Note from front at start	0x1C	5
Note stacked at start	0x1D	5
Cashbox full	0x1E	1
Cashbox removed	0x1F	1
Cashbox replaced	0x20	1