SECTION 5

SMART PAYOUT MANUAL SET

SOFTWARE IMPLEMENTATION GUIDE

INTELLIGENCE IN VALIDATION

Innovative Technology
INTELLIGENCE IN VALIDATION

Bellis Technology
INTELLIGENCE IN VALIDATION

SMART PAYOUT MANUAL SET – SECTION 5

5. SOFTWARE IMPLEMENTATION GUIDE

5.1 Communication Protocols

The SMART Payout unit can use two different communication protocols – eSSP and ccTalk (CC2).

The recommended communication protocol for the SMART Payout unit is eSSP, as this provides the highest level of data transfer security. A ccTalk (CC2) interface protocol is also available – CC2 builds on the existing ccTalk standard and uses the standard ccTalk packet construction and encryption, but to use the payout features has a different flow of information.

For detailed information and the full protocol specifications please read the following documents, which can be downloaded from the Innovative Technology Ltd website (www.innovative-technology.co.uk):

- SSP Interface Specification (ITL Document number GA138)
- eSSP – ccTalk Converter Specification (ITL Document number GA863)
- ITL Bank Note Reader ccTalk Specification (ITL Document number GA966)

Summaries of the SMART Payout unit socket connections for each of the interfaces are shown below:

SMART Payout SSP Interface:

Pin	Name	Type	Description
1	GND	Input	GND
2	Factory use only		**Do not connect**
3			
4	RxD Opto -	Input	Opto RxD –
5	Factory use only		**Do not connect**
6	RxD Opto +	Input	Opto RxD +
7	Factory use only		**Do not connect**
8	TxD Opto Emitter	Output	Opto isolated TxD Emitter
9	V In	Input	+12 V DC
10	Factory use only		**Do not connect**
11	RxD RS232	Input	RS232 RxD
12	Factory use only		**Do not connect**
13	TxD Opto Collector	Output	Opto Isolated TxD Collector
14	RxD	Input	TTL RxD
15	TxD RS232	Output	RS232 TxD
16	TxD	Output	TTL TxD

SMART Payout ccTalk (CC2) Interface:

Pin	Name	Type	Description
1	GND	Input	GND
2	Factory use only		**Do not connect**
3			
4			
5			
6			
7			
8			
9	V In	Input	+12 V DC
10	Factory use only		**Do not connect**
11			
12			
13			
14	RxD	Input	TTL RxD
15	Factory use only		**Do not connect**
16	TxD	Output	TTL TxD

WARNING!

Risk of unit damage

Do not make any connections to the interface socket pins marked '**Do not connect**' – making connections to these pins could cause severe damage to the unit.

Information

Encryption of data strongly recommended

It is recommended that all credit/dispense transactions with the SMART Payout unit be encrypted to prevent dispense commands being recorded and replayed by an external device. If this is not possible, then other (mechanical) measures should be used to prevent physical bus tapping.

5.2 SSP and eSSP

Smiley® Secure Protocol (SSP) is a secure serial interface specifically designed to address the problems experienced by cash systems in gaming machines. Problems such as acceptor swapping, reprogramming acceptors and line tapping are all addressed.

Encrypted Smiley® Secure Protocol (eSSP) is an enhancement of SSP. eSSP uses the same 16 bit CRC checksums on all packets as SSP, but also uses a Diffie-Hellman key exchange to allow the host machine and SMART Payout unit to jointly establish a shared secret key over an insecure communications channel. The encryption algorithm used is AES with a 128-bit key; this provides a very high level of security.

The encryption of the SSP protocol ensures superior protection and reliability of the data, which is transferred between validator and host machine. The encryption key is divided into two parts:

- The lower 64 bits are fixed and specified by the machine manufacturer allowing control of which devices are used in their machines.
- The higher 64 bits are securely negotiated by the slave and host at power up, ensuring each machine and each session are using different keys.

The interface uses a master-slave model; the host machine is the master and the peripherals (note acceptor, coin acceptor or coin hopper) are the slaves. Data transfer is over a multi-drop bus using clock asynchronous serial transmission with simple open collector drivers. Each SSP device of a particular type has a unique serial number; this number is used to validate each device in the direction of credit transfer before transactions can take place.

Information

200 ms command spacing

When communicating with the SMART Payout unit, poll commands should be sent **at least** 200 ms apart.

SSP Commands and Responses

a. Commands

Action	Command Code (Hex)	Command Set
Reset	0x01	Generic
Host Protocol Version	0x06	
Poll	0x07	
Get Serial Number	0x0C	
Synchronisation command	0x11	
Disable	0x09	
Enable	0x0A	
Program Firmware / currency	0x0B (Programming Type)	
Manufacturers Extension	0x30 (Command, Data)	
Set inhibits	0x02	Validator
Display On	0x03	
Display Off	0x04	
Set-up Request	0x05	
Reject	0x08	
Unit data	0x0D	
Channel Value data	0x0E	
Channel Security data	0x0F	
Channel Re-teach data	0x10	
Last Reject Code	0x17	
Hold	0x18	

Action	Command Code (Hex)	Command Set
Enable Protocol Version Events	0x19 (made obsolete in protocol version 6)	Validator
Get Bar Code Reader Configuration	0x23	
Set Bar Code Reader Configuration	0x24	
Get Bar Code Inhibit	0x25	
Set Bar Code Inhibit	0x26	
Get Bar Code Data	0x27	
Enable Payout Device	0x5C	Payout
Disable Payout Device	0x5B	
Set Routing	0x3B	
Get Routing	0x3C	
Payout Amount	0x33	
Get Note amount	0x35	
Halt Payout	0x38	
Float Amount	0x3D	
Get Minimum Payout	0x3E	
Payout by denomination	0x46	
Float by denomination	0x44	
Empty All	0x3F	
SMART empty	0x52	
Cashbox Payout Operation Data	0x53	

Notes:

Action	Comments
Reset:	Single byte command, causes the slave to reset
Host Protocol Version:	Dual byte command, the first byte is the command; the second byte is the version of the protocol that is implemented on the host.
Poll:	Single byte command, no action taken except to report latest events.
Get Serial Number:	Single byte command, used to request the slave serial number. Returns 4-byte long integer.
Sync:	Single byte command, which will reset the validator to expect the next sequence ID to be 0.
Disable:	Single byte command, the peripheral will switch to its disabled state, it will not execute any more commands or perform any actions until enabled, any poll commands will report disabled.
Enable:	Single byte command, the peripheral will return to service.
Manufactures Extension:	This command allows the manufacturer of a peripheral to send commands specific to their unit
Enable Payout Device:	Single byte command to enable the Payout module.
Disable Payout Device:	Single byte command to disable the Payout module. All notes accepted will be routed to the NV200 cashbox and payout commands will not be accepted.
Set Routing:	Six-byte command to set the routing of each note value. Notes can either be routed to the NV200 cashbox, or to the Payout module and used for payouts. By default all note values are stacked.
Payout Amount:	Five-byte command to set the value to payout.
Get Note Amount:	Five-byte command that will return the note counter for a given value in the Payout module.
Float:	Nine-byte command to set the minimum payout and the value to float to.
Get Minimum Payout:	Single byte command that returns the minimum payout value.
Empty:	Single byte command that will cause all notes to be sent to the stacker for removal.

b. Responses

Action	Command Code (Hex)	Command Set
OK	0xF0	Generic
Command not known	0xF2	
Wrong number of parameters	0xF3	
Parameter out of range	0xF4	
Command cannot be processed	0xF5	
Software Error	0xF6	
FAIL	0xF8	
Key Not Set	0xFA	
Slave Reset	0xF1	Validator
Read, n	0xEF, Channel Number	
Credit, n	0xEE, Channel Number	
Rejecting	0xED	
Rejected	0xEC	
Stacking	0xCC	
Stacked	0xEB	
Safe Jam	0xEA	
Unsafe Jam	0xE9	
Disabled	0xE8	
Fraud Attempt, n	0xE6, Channel Number	
Stacker Full	0xE7	
Note cleared from front at reset	0xE1, Channel Number	

Action	Command Code (Hex)	Command Set
Note cleared into cash box at reset	0xE2, Channel Number	Validator
Cash Box Removed	0xE3	
Cash Box Replaced	0xE4	
Bar Code Ticket Validated	0xE5	
Bar Code Ticket Acknowledge	0xD1	
Note path open	0xE0	
Channel Disable	0xB5	
Dispensing	0xDA, Current value dispensed	Payout
Dispensed	0xD2, value dispensed	
Jammed	0xD5, value dispensed	
Halted	0xD6, value dispensed	
Floating	0xD7, value to cashbox	
Floated	0xD8, value to cashbox	
Time Out	0xD9, value dispensed	
Incomplete Payout	0xDC, value dispensed, value requested	
Incomplete Float	0xDD, value to cashbox, value requested	
Emptying	0xC2	
Empty	0xC3	
Note stored in payout	0xDB	
SMART Emptying	0xB3	
SMART Emptied	0xB4	

Notes:

Action	Comments
Command Not Known:	Returned when an invalid command is received by a peripheral.
Wrong Number Of Parameters:	A command was received by a peripheral, but an incorrect number of parameters were received.
Parameter Out Of Range:	One of the parameters sent with a command is out of range.
Command Cannot Be Processed:	A command sent could not be processed at that time.
Software Error:	Reported for errors in the execution of software e.g. Divide by zero. This may also be reported if there is a problem resulting from a failed remote firmware upgrade, in this case the firmware upgrade should be redone
Key Not Set:	The slave is in encrypted communication mode but the encryption keys have not been negotiated
Dispensing:	Five-byte response reporting the value of notes that have been dispensed at the point when the poll was received.
Dispensed:	Five-byte response that indicates when the payout has finished a dispense operation; also reports the value of notes that have been dispensed.
Jammed:	Five-byte response that indicates that the payout is jammed; this is reported until it is un-jammed or reset. It will also become disabled. Also reports the value of notes that have been dispensed before the jam.
Time Out:	This is given if a search for a note in the payout store fails after a time-out period and there is no way to pay that value with any others - the event will be given along with the value paid out up to the time out point.
Incomplete Payout / Float:	This event is given when the payout starts up if a payout or float operation was in progress when the power was removed. Reports the value that was dispensed and the value that was originally requested.
Note stored in payout:	This event is given when notes paid in to the payout system are routed to the payout store.
Emptying:	This event is given while the payout is being emptied of notes into the cashbox by the EMPTY command.
Empty:	This event is given at the end of the empty process.

Example SSP Communications

Here is an example of the communication between host and slave. Both the typical commands from the host and responses from the payout are detailed.

Host	Slave	Comments
> SYNC	< OK	Synchronisation command
> SET_GENERATOR, *[64 bit prime number]*	< OK	Set the encryption key generator
> SET_MODULUS, *[64 bit prime number]*	< OK	Set the encryption key modulus
> REQUEST_KEY_EXCHANGE *[64 bit host intermediate key]*	< OK, *[64bit slave intermediate key]*	Host sends the host intermediate key, slave responds with the slave intermediate key. The encryption key is then calculated independently by both host and slave.
> GET_SERIAL	< OK < [*SERIAL NUMBER*]	NV200 Serial Number
> SETUP_REQUEST	< OK < [*SETUP INFORMATION*]	NV200 Setup
> SET_ROUTING, 00 05 00 00 00	< OK	Route notes of value 0005 to the SMART Payout
> SET_ROUTING, 00 0A 00 00 00	< OK	Route notes of value 0010 to the SMART Payout
> SET_ROUTING, 01 14 00 00 00	< OK	Route notes of value 0020 to the NV200 Cashbox
> ENABLE_PAYOUT_DEVICE	< OK	Enable SMART Payout
> SET_INHIBIT > 07 > 00	< OK	Enable channels 1,2 and 3
> ENABLE	< OK	Enable NV200
> POLL	< OK < DISABLED	
> POLL	< OK	
> POLL	< OK < NOTE READ < 00	NV200 currently reading a note
> POLL	< OK < NOTE READ < 03	Note has been recognised as channel 3 (£20)
> HOLD	< OK	Hold the note in escrow
> HOLD	< OK	Hold the note in escrow
> POLL	< OK < STACKING	Stack the note
> POLL	< OK < CREDIT < 03 < STACKING < STACKED	Credit given for channel 3 (£20), note stacked
> POLL	< OK	
> PAYOUT_AMOUNT > 0F > 00 > 00 > 00	< OK	Payout £15

Host	Slave	Comments
> POLL	< OK < DISPENSING < 00 < 00	Dispensing, £0 dispensed so far
> POLL	< OK < DISPENSING < 0A < 00	Dispensing, £10 dispensed so far
> POLL	< OK < DISPENSED < 0F < 00	Dispensed £15
> POLL	< OK	

Full support is available from ITL and local support offices for implementing eSSP - this support includes libraries and example applications. When requesting this information, please specify your preferred language(s) and operating system.

5.3 ccTalk (CC2)

This section should be read in conjunction with the full ccTalk specification, which can be downloaded from the internet (www.cctalk.org).

ccTalk is a serial communications protocol in widespread use throughout the money transaction industry. Peripherals such as coin acceptors, note validators and hoppers found in a diverse range of automatic payment equipment use ccTalk to communicate with the host controller.

The protocol uses an asynchronous transfer of character frames in a similar manner to RS232. The main difference is that it uses a single two-way communication data line for half-duplex communication rather than separate transmit and receives lines. It operates at TTL voltages and is 'multi-drop' (peripherals can be connected to a common bus and are logically separated by a device address) - each peripheral on the ccTalk bus must have a unique address.

CC2 is Innovative Technology Ltd's extended version of ccTalk, and is used with the SMART Payout unit - the note validator commands conform to the standard specification, and the SMART Payout commands are an extension to this device on the same address.

As it is possible to use the ccTalk protocol without encryption, suitable physical security should be employed to protect the ccTalk bus.

Information

200 ms command spacing

When communicating with the SMART Payout unit, Read Buffered Bill events (command 159) should be sent **at least** 200 ms apart.

CC2 Command Summary

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

Monetary Values

Values are represented as 32 bit unsigned integers (4 bytes) and in the lowest value of currency. For example:

€50.00 would be 0x00001388

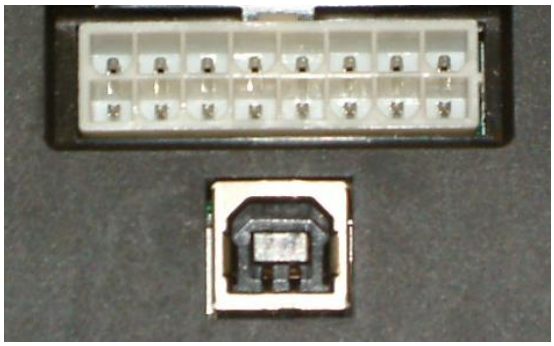
When sending or receiving a value the least significant byte is sent first. So in this example [0x88] [0x13] [0x00] [0x00] will be sent.

Each type of note is identified by its value and represented using the standard format outlined above. As an example, the values for Euro notes are:

Note (€)	Hex value	Data to Send
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]

5.4 Connection Options

The SMART Payout unit has two connectors that are used to allow interfacing and programming. The first connector is a 16 pin socket used to interface the SMART Payout unit to the host machine.

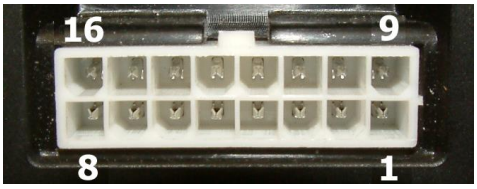


Information

Power always required regardless of connection type.

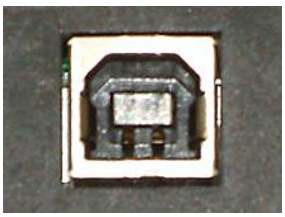
Power is always required on pins 1 and 9 of the 16 way connector.

The pin numbering of the socket is shown below, as well as an overview of the socket connections:



Pin	Description
1	0V / Ground Connection
9	+12V DC
14	Serial Data In (Rx)
16	Serial Data Out (Tx)

The USB connector is a standard Type B USB socket. The USB socket can be used for programming the SMART Hopper unit and also bench testing – a USB 2.0 compliant Type 'A' to 'B' lead can be used to do this. USB cables should be electrically shielded and less than 5 metres long. **Please note:** Direct USB should **NOT** be used for Host communications. If USB is required than our IF17 (TTL to USB) should be used.



Further details of the cables needed to interface and program the SMART Payout unit can be found in Section 4 of this manual set (subsection 4.7).

5.5 Frequently Asked Questions

a. What settings should I use on the DIP switches on the rear of the unit?

- Look at the DIP switch tables in Section 1 of this manual set (subsection 1.4)

b. How do I use the encryption key?

- The encryption key is made up of two parts – this is explained in subsection 5.2. The two parts of the encryption key are:
    a) A variable key (one that is exchanged at start up by the host machine – read subsection 5.2 for more information)
    b) A fixed key (which can be set using the PiPS software as described in Section 3 of this manual set). The default key value is 0x0123456701234567

c. My notes are always stacked in the cashbox even though I have chosen for them to go into the payout unit

- Check that the Green LED on the rear of the SMART Payout unit is flashing – see the Flash Codes in Section 1 of this manual set (subsection 1.9) if this is not the case.
- Make sure the diverter is in the correct position – with the unit powered up turn DIP switch 8 ON and OFF to make sure.
- The Payout module might be disabled in software - send an enable payout command.
- The Payout module might be full – check how many notes are stored.
- The notes might be detected as damaged or not straight – in this case they will be stacked in the cash box so that they will not jam the payout module.

d. My payout module is communicating in ccTalk but I want to update it. How can I do this?

- To do this, the validator head needs to be removed – you can find out how to do this by following the procedure in Section 1 of this manual set (subsection 1.1).
    a) Provide power to the NV200 validator. Don't worry that the bezel lights are flashing as this is normal.
    b) Turn DIP switch 8 on the rear of the validator up then down. The bezel will quickly flash then the unit will reset.
    c) Remove the power and refit the validator to the payout module.
    d) Update the unit using the PiPS software as described in Section 3 of this manual set.

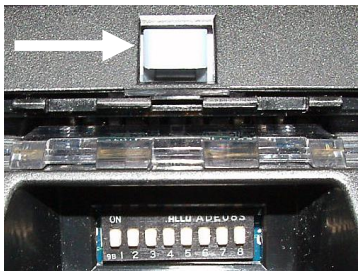
e. My payout module has stopped functioning and I want to return it for repair - however it has bank notes inside

- All bank notes that are inside payout modules returned to ITL are handled with the highest security and carefully tracked internally until their return to the customer - if you do not want to ship the unit with the bank notes inside, please follow the instructions for manual payout in Section 4 of this manual set (subsection 4.10).
- If manual payout is not possible please contact ITL technical support.

f. Is my NV200 validator compatible with the payout module?

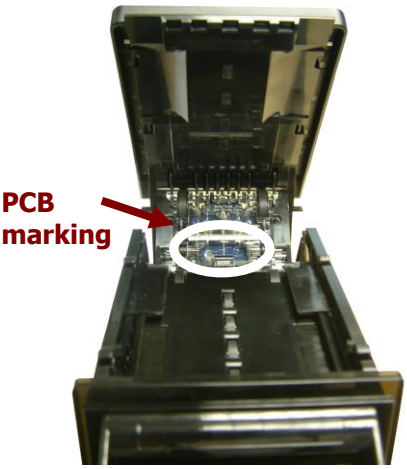
- Early revisions of the NV200 did not support the payout module. Check for all of the following features to ensure compatibility:

A grey diverter plunger on the rear of the NV200 validator head (just above the DIP switches)



Mounting brackets on the rear of the cash box housing

Open the NV200 validator lid and check the marking on the PCB where shown in this picture – the marking needs to read **PB266_4**



If any one of these features is not present, a new NV200 validator will be needed

g. Can I connect to the Host machine via USB?

- The direct USB port is for on the bench testing/Programming only. If a USB connection is desired, we recommend going through our IF17. The IF17 is a TTL to USB conversion box which filters out any noise and provides a smooth signal between the SMART Hopper and Host machine.

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

Innovative Technology
INTELLIGENCE IN VALIDATION

Bellis Technology
INTELLIGENCE IN VALIDATION

AUSTRALIA
support@innovative-technology.com.au

BRAZIL
suporte@bellis-technology.com.br

CHINA
support@innovative-technology.co.uk

GERMANY
supportDE@innovative-technology.eu

ITALY
supportIT@innovative-technology.eu

SPAIN
supportES@innovative-technology.eu

UNITED KINGDOM
support@innovative-technology.co.uk

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