

SECTION 5

SMART HOPPER MANUAL SET

SOFTWARE IMPLEMENTATION GUIDE

INTELLIGENCE IN VALIDATION

Innovative Technology assume no responsibility for errors, omissions, or damages resulting from the use of information contained within this manual.

SMART HOPPER MANUAL SET – SECTION 5

5.	SOFTWARE IMPLEMENTATION GUIDE	3
5.1	Communication Protocols	3
5.2	SSP and eSSP	4
5.3	Example SSP Communications	11
5.4	Connection Options	13
5.5	Frequently Asked Questions	15



5. SOFTWARE IMPLEMENTATION GUIDE

5.1 Communication Protocols

The SMART Hopper unit uses eSSP communication protocol.

The recommended communication protocol for the SMART Hopper unit is eSSP, as this provides the highest level of data transfer security

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)

Summaries of the SMART Hopper unit socket connections for the eSSP interfaces are shown below:

SMART Hopper SSP Interface:

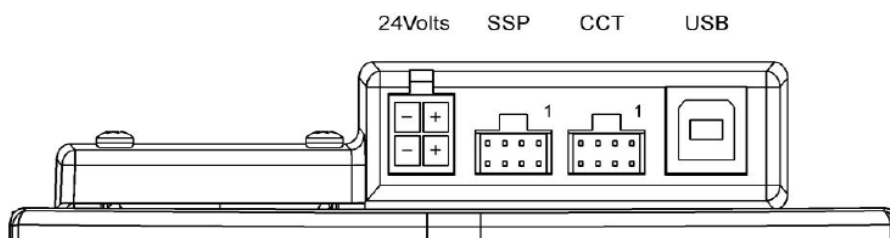


Figure 2 - Interface Connections location

eSSP Connections (Host Machine)

Pin	Function
1	eSSP Tx
2	eSSP Rx
8	Comms GND

Table 8 – eSSP Host Connector Pin Details

CCT Connections (coin acceptor)

Pin	Function
1 and 2	Data
7	+12V
8	0V

Table 9 - CCT Coin Acceptor Connections



WARNING!

Risk of unit damage

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

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 Hopper 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 Hopper unit, poll commands should be sent **at least** 200 ms apart.



eSSP 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	Hopper
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	

Commands marked **RED** must be encrypted.



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 Hopper Device	0x5C	Hopper
Disable Hopper Device	0x5B	
Set Coin Routing	0x3B, route, value	
Get Coin Routing	0x3C, value	
Payout Amount	0x33, value	
Get Coin amount	0x35, value	
Halt Payout	0x38	
Float Amount	0x3D, min payout, float value	
Get Minimum Payout	0x3E	
Set Individual Coin Accept Inhibit	0x40, state, value	
Payout Coins by denomination	0x46	
Float by denomination	0x44	
Empty All	0x3F	
Host Serial Number Request	0x14, Serial No	

Commands marked **RED** must be encrypted.

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 Hopper module.
Disable Payout Device:	Single byte command to disable the Hopper module.
Set Routing:	Six-byte command to set the routing of each coin note value.
Payout Amount:	Five-byte command to set the value to Pay out.
Get Coin Amount:	Five-byte commands that will return the coin counter for a given value in the Hopper module.
Float:	Nine-byte command to set the minimum Hopper and the value to float to.
Get Minimum Payout:	Single byte command that returns the minimum payout value.
Empty:	Single byte command that will empty all coins.



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	Hopper
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	
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 coin that has 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 coin that has 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.

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 Hopper are detailed. For more details on generating the encryption key, refer to SSP Protocol specification, document number GA138.

Host	Slave	Comments
> SYNC	< OK	
> 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, the slave responds with the slave intermediate key. The encryption key is then calculated independently by both the host and the slave.
> GET_SERIAL	< OK < [SERIAL NUMBER]	
> GET_DEVICE_SETUP	< OK < [SETUP INFORMATION]	
> GET_COIN_AMOUNT, 0A 00 00 00	< OK < 2c 01	300 coins of 0010 value
> GET_COIN_AMOUNT, 14 00 00 00	< OK < 00 96	150 coins of 0020 value
> SET_ROUTING, 01 0A 00 00 00	< OK	Route coins of value 0010 to the cash box
> SET_ROUTING, 00 14 00 00 00	< OK	Recycle coins of value 0020 for future payouts
> SET_COIN_ACCEPT_INHIBIT, 00 01 00	< OK	Set the coin mech to Inhibit the acceptance of coins of value 0001
> SET_COIN_ACCEPT_INHIBIT, 01 14 00	< OK	Set the coin mech to accept coins of value 0020
> ENABLE	< OK	
> POLL	< OK	
> POLL	< OK	
> POLL	< OK, COIN_CREDIT, 0100	Credit from coin mech of 0100 value
> POLL	< OK	
> POLL	< OK, COIN_CREDIT, 0050	Credit from coin mech of 0050 value
> POLL	< OK	
> PAYOUT_AMOUNT, 28 00 00 00	< OK	Payout a value of 0040
> POLL	< OK < DISPENSING, 14 00 00 00	Currently dispensed 0020
> POLL	< OK < DISPENSED, 28 00 00 00	Dispense value of 0040 completed
> POLL	< OK	

Table 11 - eSSP example for the SMART hopper with coin mech connected via the hopper

Host	Slave	Comments
> SYNC	< OK	
> 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, the slave responds with the slave intermediate key. The encryption key is then calculated independently by both the host and the slave.
> GET_SERIAL	< OK < [SERIAL NUMBER]	
> GET_DEVICE_SETUP	< OK < [SETUP INFORMATION]	
> GET_COIN_AMOUNT, 0A 00 00 00	< OK < 2c 01	300 coins of 0010 value
> GET_COIN_AMOUNT, 14 00 00 00	< OK < 00 96	150 coins of 0020 value
> SET_ROUTING, 01 0A 00 00 00	< OK	Route coins of value 0010 to the cash box
> SET_ROUTING, 00 14 00 00 00	< OK	Recycle coins of value 0020 for future payouts
> ENABLE	< OK	
> POLL	< OK	
> POLL	< OK	
> SET_COIN_AMOUNT, 03 00 32 00	< OK	3 coins of value 0050 added to hopper (total value of 0150)
> POLL	< OK	
> POLL	< OK	
> PAYOUT_AMOUNT, 28 00 00 00	< OK	Payout a value of 0040
> POLL	< OK < DISPENSING, 14 00 00 00	Currently dispensed 0020
> POLL	< OK < DISPENSED, 28 00 00 00	Dispense value of 0040 completed
> POLL	< OK	

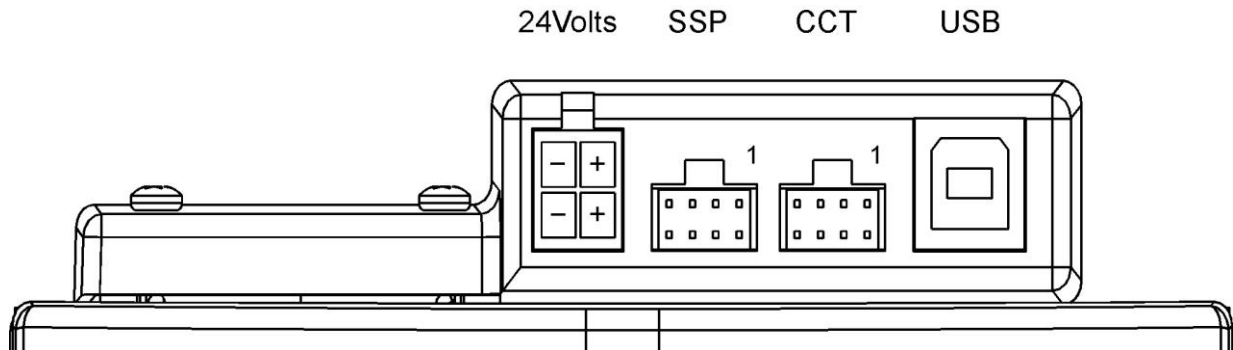
Table 12 – eSSP example for the SMART hopper with coin mech connected separately

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 Connection Options

SMART Hopper

All the connectors needed to set up the SMART Hopper unit are easily accessible on the bottom base: there are four connectors that are used to allow interfacing and programming:

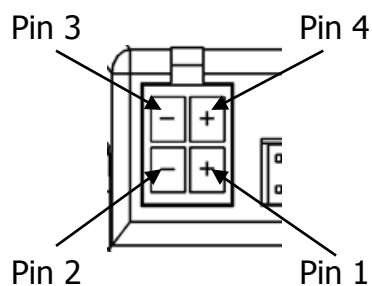


Information

Power always required regardless of connection type.

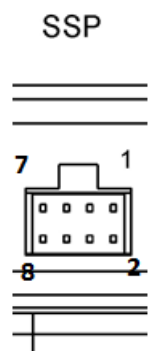
Power is always required on pins 3 and 4 of the 4 way connector.

The first connector is a 4 pin socket used to power up the SMART Hopper.



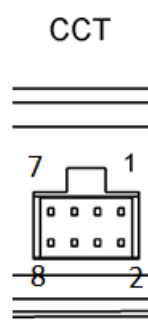
Pin	Description
1	+24V DC
2	0V / Ground Connection
3	N/C
4	N/C

Interface communication from the SMART Hopper unit to the host machine can communicate by SSP and CCT coin mech. The SSP pin numbering of the socket is shown below, as well as an overview of the socket connections:



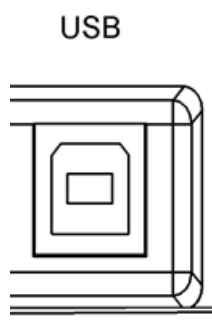
Pin	Description
1	Serial Data Out (Tx)
2	Serial Data In (Rx)
7	N/C
8	0V / Ground Connection

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



Pin	Description
1	Serial Data Out (Tx)
2	Serial Data In (Rx)
7	+12 V
8	0V / Ground Connection

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 then our IF17 (TTL to USB) should be used.



5.4 Frequently Asked Questions

a. What currencies does the SMART Hopper support?

- New currency dataset files are published on the ITL website as they are released. To find available datasets visit the Currency Download section within Support. Select the SMART Hopper and the currency you require to see all available dataset options. The blue i icon provides more details to assist selections.

b. I can't find the currency I need, how do I get it created?

- To create a new dataset, ITL requires 100 coins of each denomination. This process will take around 4 weeks to process and then coins can be returned and the dataset will be made available. Please email support@innovative-technology.co.uk for information.

c. Where can I get the software examples for the SMART Hopper?

- please email support@innovative-technology.co.uk for software example

d. 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.

e. What communication protocols does the SMART Hopper support?

- ENCRYPTED SSP (eSSP) 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. 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. The integrity of data transfers is ensured through the use of 16 bit CRC checksums on all packets. A Diffie-Hellman key exchange is used to allow the host machine and SMART hopper 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.

IMPORTANT: All transactions with the SMART Hopper must be encrypted to prevent dispense commands being recorded and replayed by an external device. For detailed information and full protocol specification please refer to SSP Interface Specification (ITL Drawing GA138), this is available from the ITL website www.innovative-technology.co.uk.



f. How fast does the SMART Hopper pay out?

- The SMART Hopper can pay out up to 12 Coins per second.

g. How many coins does the SMART Hopper hold?

- The capacity of the SMART Hopper depends on the size of coins. The table below shows approximate capacity for various coins and assumes all the coins within the hopper are the same coin type.

Type	Diameter	Thickness	Approximate Capacity
UK £1	22.5mm	3.15mm	1300
UK £2	28.4mm	2.5mm	800
Euro €1	23.25mm	2.33mm	1500

- SMART Hopper Weights

Coin Type	Coin Weight	Total Weight
Empty	-	2.60kg
UK £1	9.5g	Approximately 14.95Kg when full (1300)
UK £2	12.0g	Approximately 12.20Kg when full (800)
Euro €1	7.5g	Approximately 13.85Kg when full (1500)



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

**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

