

SECTION 5

NV200 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.



A.u.S. Spielgeräte GesmbH
Scheydgasse 48 A-1210 Wien
Tel. +43-1-271 66 00 Fax.+43-1-271 66 00 75
E-mail: verkauf@aus.at
<http://www.aus.at>
Öffnungszeiten: Mo-Fr. 9-18 Uhr

NV200 MANUAL SET – SECTION 5

5.	SOFTWARE IMPLEMENTATION GUIDE	3
5.1	Communication Protocols	3
5.2	SSP and eSSP	6
5.3	ccTalk	12
5.4	Connection Options	15



5. SOFTWARE IMPLEMENTATION GUIDE

5.1 Communication Protocols

The NV200 validator can use several different communication protocols, including eSSP, SIO, ccTalk, MDB, Parallel, Binary and Pulse. Only eSSP, SIO and ccTalk are supported natively – use of the other protocols requires the use of an external interface unit.

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.

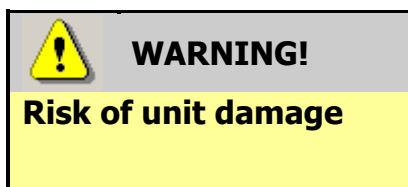
Encrypted Smiley[®] Secure Protocol (eSSP) is an enhancement of SSP. eSSP uses the same 16 bit CRC checksums on all packets as SSP, but also uses a Diffie-Hellman key exchange to allow the host machine and validator to jointly establish a shared secret key over an insecure communications channel. The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

The recommended communication protocol for the NV200 validator is eSSP, as this provides the highest level of data transfer security. A ccTalk interface protocol is also available.

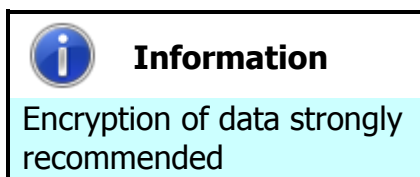
For detailed information and the full protocol specifications please read the following documents, which can be downloaded from the Innovative Technology Ltd website (www.innovative-technology.co.uk):

- SSP Interface Specification (ITL Document number GA00138)
- ITL Bank Note Reader ccTalk Specification (ITL Document number GA00966)

Summaries of the NV200 validator socket connections for the SSP and ccTalk interfaces are shown below:



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.



It is recommended that all transactions with the NV200 validator be encrypted to prevent 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.

NV200 SSP Interface:

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

NV200 ccTalk Interface:

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

Other Protocols:

Multi Drop Bus (MDB) Interface: MDB is a serial bus interface commonly used in electrically controlled vending machines. This is a 9600 Baud Master-Slave system where the NV200 validator is a slave to a master controller.

To use the NV200 with MDB protocol, an **IF5** external interface is required. The IF5 regulates the power supply and opto-isolates the communication lines. The NV200 validator supports the MDB Protocol Version 1, Level 1.

Parallel Interface: To use the NV200 in Parallel mode, an **IF10** external interface is required. When operating in Parallel mode the NV200 will issue a 100ms active LOW pulse on the relevant vend line, and a maximum of 4 channels can be used.

Binary Interface: To use the NV200 in Binary mode, an **IF9** external interface is required. When operating in Binary mode the NV200 will issue a binary pattern on vend lines 1 to 4, and a maximum of 15 channels can be used.

Pulse Interface: To use the NV200 in Pulse mode, an **IF15** external interface is required. When operating in Pulse mode the NV200 outputs a number of pulses on Vend 1. The number of pulses for each channel is different and set to default values within the dataset. The number of pulses and the pulse duration can be modified using the Bank Note Validator Currency Manager Software, and a maximum of 16 channels can be used.

Opto-isolation and RS232 communications is only available on validators with an issue number of 4 or greater. You can check the issue number on the validator as shown here:

Open the NV200 validator lid and check the marking on the PCB where shown in this picture – the marking needs to read **PB00266_4**

If the issue number is less than **4** or is not visible, please contact ITL Support for connection options and information.



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 validator 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 NV200 validator, 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	

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.

Manufacturers Extension:

This command allows the manufacturer of a peripheral to send commands specific to their unit



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	

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

Jammed:

Five-byte response that indicates that the validator is jammed; this is reported until it is un-jammed or reset. It will also become disabled.



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 validator are detailed.

Host	Slave	Comments
> SYNC	< OK	Synchronisation command
> SET_GENERATOR, <i>[64 bit prime number]</i>	< OK	Set the encryption key generator
> SET_MODULUS, <i>[64 bit prime number]</i>	< OK	Set the encryption key modulus
> REQUEST_KEY_EXCHANGE <i>[64 bit host intermediate key]</i>	< OK, <i>[64bit slave intermediate key]</i>	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 < <i>[SERIAL NUMBER]</i>	NV200 Serial Number
> SETUP_REQUEST	< OK < <i>[SETUP INFORMATION]</i>	NV200 Setup
> SET_ROUTING, 01 14 00 00 00	< OK	Route notes of value 0020 to the NV200 Cashbox
> 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
> POLL	< OK < STACKING	Stack the note
> POLL	< OK < CREDIT < 03 < STACKING < STACKED	Credit given for channel 3 (£20), note stacked
> 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

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.

Each communication sequence (a command or request for information) consists of 2 message packets structured in one of the formats detailed below. The first packet will go from the master device to the slave device and then a reply will be sent from the slave device to the master device.

Commands can have 3 primary formats:

- 8 Bit Checksum – No Encryption
- 16 Bit CRC – No Encryption
- 16 Bit CRC – BNV Encryption

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 NV200 validator, Read Buffered Bill events (command 159) should be sent **at least** 200 ms apart.



ccTalk Command Summary

Command	Header	Parameters	Example
Reset Device	001	None	ACK
Request Comms Revision	004	None	X.Y
Read Barcode Data	129	None	ACK
Store Encryption Code	136	None	ACK
Switch Encryption Code	137	3 bytes Encryption key	ACK
Request Currency Revision	145	None or Country Code (2 digit)	'GBP02113'
Operate Bi-directional Motors	146	None	ACK
Stacker Cycle	147	None	ACK
Request Bill Operating Mode	152	None	3
Modify Bill Operating Table	153	Escrow & Stacker	ACK
Route Bill	154	0/1	ACK/254
Request Bill Position	155	Country Code (2 digit)	00000111 00000000
Request Country Scaling	156	Country Code (2 digit)	100
Request Bill ID	157	None	'GB0010A'
Read Buffered Bill Events	159	None	100000000000
Request Address Mode	169	None	1
Request Base Year	170	None	2006
Request Build Code	192	None	161209
Request Last Mod Date	195	None	00
Calculate ROM Checksum	197	None	4 byte checksum
Request Option Flags	213	None	3 (stacker & escrow)
Request Data Storage Av.	216	None	00000
Enter Pin	218	Pin1, Pin2, Pin3, Pin4	ACK
Enter New Pin	219	Pin1, Pin2, Pin3, Pin4	ACK
Request Accept Count	225	None	3
Request Insertion Count	226	None	7
Request Master Inhibit	227	None	1



Command	Header	Parameters	Example
Set Master Inhibit	228	Bit Mask	ACK
Request Inhibits	230	None	Inhibit Low, Inhibit High
Set Inhibits	231	Channels	ACK
Perform Self Check	232	None	0
Request Software Version	241	None	XX.YY
Request Serial Number	242	None	3 byte serial number
Request Product Code	244	None	'NV200'
Request Equipment Category	245	None	'Bill Validator'
Request manufacturer ID	246	None	'ITL'
Request Polling Priority	249	None	200
Simple Poll	254	None	ACK

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 NV200 validator has two connectors that are used to allow interfacing and programming; these connectors are easily accessible at the back of the validator.



Information

Power always required regardless of connection type.

Power is always required on pins 15 and 16 of the 16 way connector.



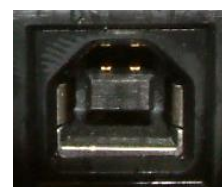
The first connector is a 16 pin socket used to interface the NV200 to the host machine. The pin numbering of the socket is shown below, as well as an overview of the socket connections:



Pin	Description
6	Serial Data Out (Tx)
7	Serial Data In (Rx)
15	+ V
16	0V / Ground Connection

The function of pins 1 to 9 can change depending on which machine interface is being used with the NV200. Typically, the validator will be using SSP, ccTalk or SIO interfaces. MDB, Parallel, Binary and Pulse interfaces are only supported with the use of an external interface.

The USB connector is a standard Type 'B' USB socket, and can be used for interfacing to the host machine – in this case, power must be provided through the 16 way connector. The USB socket can also be used for programming the NV200 – 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.



Further details of the cables needed to interface and program the NV200 validator can be found in Section 4 of this manual set (subsection 4.7).