* **Protocol Type selection (PTS)**

Protocol type selection is used by the interface device to change the communications protocol and/or the default values of FI and DI. The PTS command must be issued immediately after the answer to reset and only applies when the IC card is in the negotiable mode.

The interface device may choose to operate by using the first indicated protocol after the answer to reset and by using the default values of F and D. This results in an implicit selection of the protocol and the communication parameters. Should the interface device wish to effect any change to this situation then it must issue the PTS command.

The PTS request consists of an initial character PTSS (coded FFhex), followed by a format character PTSO, and three optional characters PTS1, PTS2, PTS3 and PCK the check character. This is shown in [figure 17](http://www.sat-digest.com/SatXpress/SmartCard/info/FIG17.GIF). The response from the ICC follows the same format as the request.

The PTS0 format character is encoded as shown in [figure 17](http://www.sat-digest.com/SatXpress/SmartCard/info/FIG17.GIF). The bit map is used to indicate the presence or otherwise of PTS1, PTS2 and PTS3. These are encoded by bits 5, 6 and 7 respectively where a logic `1' level indicates the presence of the character. The protocol type is indicated by bits 1, 2, 3 and 4 which are binary encoded for T=0 to T=15.

The PTS1 character when present is used to define the values for FI as coded for TA1. These parameters are used for defining the work etu (elementary time unit).

The check character PCK is computed such that the exclusive OR (XOR) of all the characters from PTSS to PCK inclusive is equal to zero.

When the ICC implements the PTS request message correctly it replies by echoing the same request as the response message. If bit 5 of the PTS1 response character is set to zero then the default values of F and D will be used.

**The T=0 communication protocol**

The interface device always initiates the command for the T=0 protocol. Interaction between the interface device and the ICC results in successive commands and responses. For this protocol, data can only flow in one direction for the command response pair. In other words, either the command message contains data for the ICC or the command request data from the ICC which is then included in the response. The direction of data flow is implicit on the definition of the command and hence both the interface device and the ICC need to have the necessary a-priori knowledge. When it is required to transfer data in both directions for a particular command then a "get response" command may be used after the primary command to recover the response data.

The command message consists of a 5 character header which the interface device sends to the ICC. The ICC then replies with a procedure byte after which either data is sent to the ICC, or from the ICC, depending on the particular command. This procedure byte is to allow the interface device to control the Vpp EPROM programming voltage. In the case of EEPROM memory this procedure byte is effectively redundant. The message flow for the T=0 protocol is shown in [figure 18](http://www.sat-digest.com/SatXpress/SmartCard/info/FIG18.GIF). The command header consists of the following 5 bytes:

* + CLA - the instruction class (FF is reserved for PTS)
  + INS - the instruction code (e.g read memory)
  + P1 - instruction code qualifier (e.g memory address)
  + P2 - additional INS code qualifier
  + P3 - the length of the data block

When P3 is equal to zero the data from the card will be 256 bytes. When data is to be transferred into the card then a zero data transfer is implied.

The normal condition for the ACK procedure byte is for this byte to echo the instruction byte (INS). Other options allow the interface devices to control the Vpp programming voltage as required. The card may optionally send a NULL procedure byte (60hex) which allows further time for the processing of the command. In this situation the IFD should await a further procedure byte. The ISO standard also allows the card to send the first status byte (SW1) as the procedure byte.

There are two status bytes SW1 and SW2. These bytes are sent from the ICC to the interface device on completion of the command to indicate the current card status. The normal response is:

SW1, SW2 = 90hex, 00hex

When SW1 = 6X or 9X various error conditions are reported by the card. ISO 7816-3 defines 5 such error conditions:

* + SW1 = 6E - Card does not support instruction class

= 6D - Invalid INS code

* + SW1 = 6B - Incorrect reference

= 67 - incorrect length

= 6F - no particular diagnosis

**The T = 1 comms protocol**

The T = 1 communication is an asynchronous half duplex block transmission protocol. In terms of the OSI model this protocol operates at layer 2, the data link layer. The physical layer (layer 1) operates in the same way as for the T = 0 protocol except for the error detection and correction. In essence this protocol puts an envelope around a block of characters which allows:

* + flow control
  + block chaining
  + error correction.

The choice of communication protocol for the ICC is still a hot topic and one has to consider what advantages can be offered by the block protocol and then to examine the price that must be paid.

The most obvious advantage of the T = 1 protocol is the ability to manage data flow in both directions. In our discussion of the T = 0 protocol it was shown that for a particular command that the data is either sent to or received from the ICC. This limitation was really due to the use of a single byte for defining the length of the data related to the command.

The T = 1 protocol also removes the T = 0 restriction of the master slave relationship where the interface device (IFD) always initiates a command to which the ICC responds. For this block protocol a command may be initiated by either the IFD or the ICC albeit within the restrictions of the protocol.

A further advantage of the T = 1 protocol is the ability to chain the blocks of data such that an arbitrarily large block of data may be transferred as the result of a single command by the transmission of the appropriate number of frames chained in sequence.

The block protocol also has a more sophisticated error management system. This allows the use of a block error detection code (EDC) and the ability to re-transmit blocks that are subject to some error condition. By comparison the T = 0 protocol has a primitive character error detection and correction scheme.

Clearly there is a price to be paid for this higher layer protocol. Apart from the more complex software in both the ICC and the IFD the protocol is more demanding on the RAM memory of the ICC which needs to maintain the last sent block in case retransmission is required. In general the T = 1 protocol offers advantages where the application is managing large blocks of data, particularly when it is required to pass data in both directions as part of a particular command. The efficiency of the protocol is only really apparent for larger data transmissions since the underlying physical layer is still operating in character mode as for the T = 0 protocol. The reduction of the character frame to 11 etu (elementary time units) compared with the 12 etu demanded by T = 0 has to be balanced against the administrative overhead of the frame structure which has both a prologue and epilogue.

There can be no doubt that the error control is significantly improved over the T = 0 protocol but at the lower speed of 9600 bit/second operated by many ICC's over very short transmission paths the probability of communication errors is much reduced. However it is clear that there is a move towards the use of the T = 1 protocol and it seems highly likely that this will become the predominant protocol of the future. We should not however dismiss the use of the T = 0 protocol which in some situations may well offer a more optimum technical solution. The T = 1 protocol is specified in the ISO standard ISO 7816 - 3 / AMD.1

* The T=0 protocol also includes an error detection and correction mechanism. which relies on the receiver detecting a parity error upon which it takes the I/O line to the low logic level within the first etu guard time (10.5 + 0.2 etu) for a minimum of 1 etu and a maximum of 2 etu. The transmitter looks for this condition and retransmits the corrupt character.
* **The block frame**

The frame consists of three fields:

* + prologue field
  + information field (optional)
  + epilogue field

as shown below:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Prologue Field | | | Information Field | Epilogue Field |
| Node Address  NAD | Protocol Control Byte  PCB | Length  LEN | Optional  INF | Error Detection LRC or CRC  EDC |
| 1 Byte | 1 Byte | 1 Byte | 0-254 Bytes | 1/2 Bytes |

The prologue field consists of three bytes:

* + NAD the node address
  + PCB protocol control byte
  + LEN the data length

The NAD byte uses bits 3 -1 to identify the source address and bits 7 - 5 to identify the destination address. The bits 4 and 8 are used for Vpp control which will not be discussed further here. The node address byte allows the use of multiple logical channels where required otherwise both addresses should be set to zero.

The PCB byte allows the identification of three types of block frame,

* + An information block (I - block)
  + A receive ready block (R - block)
  + A supervisory block (S - block)

The information block is the frame which is used to transmit application commands and data between the ICC and the IFD. The receive - ready block is used as an acknowledgment when the protocol is sending data as a sequence of chained blocks. The supervising block is used to establish control parameters and to effect a resynchronisation or abort status as the result of some error condition. The information block also acts as an acknowledgement byte in the non chaining mode.

The LEN byte indicates the number of bytes (if any) in the information field of the frame. Its allowed range of values are from 00 - FE hex. This allows a maximum information field of 254 bytes.

The information field is used to convey the application commands and data.

The epilogue field contains the block error detection code which may be either an LRC (longitudinal redundancy check) or a CRC (cyclic redundancy check). The LRC is 1 byte while the CRC occupies 2 bytes. This option is defined by the specific interface characters.

**Specific interface characters.**

We have previously discussed the specific interface characters given by the answer to reset (ATR). The T = 1 protocol uses three of these characters to establish the necessary options before communication can take place. These bytes are assigned as follows (where i > 2),

* + TAi = IFSC (default = 32)
  + TBi  
    (bit 4 - 1) = CWI (default = 13)  
    (bit 8 - 5) = BWI (default = 4)
  + TCi  
    (bit 1 = 1) = CRC option  
    (bit 1 = 0) = LRC option (default)

The IFSC is the information field size for the card. There is also an IFSD which is the information field size for the interface device. This has a default value of 32 bytes and can only be changed by means of an S - block request from the IFD to the ICC.

**Waiting times**

The T = 1 protocol uses two waiting time parameters to help flow control,

* + Character Waiting Time (CWT)
  + Block Waiting Time (BWT)

The character waiting time is the maximum time between successive characters in a block while the block waiting time is the maximum time between the leading edge of the last character in a block sent by the IFD and the leading character of the next block sent by the card.

The character waiting time may be used to detect an error in the length of a block while the block waiting time may be used to detect an unresponsive card. There is also a block guard time (BGT) which is defined as the minimum time between the leading edge of the last character of one block and the leading edge of the first character in the new block to be sent in the alternative direction. The CWT and BWT are calculated from the values of CWI and BWI coded as shown previously in the specific interface bytes by means of the following equations:

* + CWT = (2BWI + 11) etu
  + BWT = (2BWI X 960 X 372 / f) Sec + 11 etu
  + Where f is the clock frequency.

The minimum value for the BWT is 100 mS + 11 etu when the card operates with the default frequency of 3.58 MHz. The block guard time has a value of 22 etu such that the delay between the start of the last character of a received block and the start of a transmitted block is greater than BGT but less than BWT. Accordingly the minimum inter block time is 11 etu which is equal to one character time.

**Protocol control byte**

The protocol control byte identifies the different types of block and carries some control information including a single bit sequence number (N) and a block chaining bit (M). Other bits are used to identify transmission errors. The PCB is coded as follows: Type PCB (bits 8-1) Function

The I blocks can occur as independent blocks or as part of a chain. The "More" bit is set to indicate that further blocks are to follow. The sequence number of the sender alternates between '0' and '1' starting with '0'.

The R blocks are used to acknowledge the successful or otherwise receipt of transmitted blocks. The sequence number N carries the value of the next expected value of N where the transmitter alternates the value as mentioned above. While blocks transmitted as part of a chain must be acknowledged by an R block the receipt of a successful stand alone I block may be acknowledged by an I block response. The two correspondents manage the sequence numbers of their I blocks independently alternating between '0' and '1'. The R block has three possible states as shown in the table.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Type** | **PCB (bits 8-1)** | | | | | | | | **Function** |
| I | 0 | N | 0 | 0 | 0 | 0 | 0 | 0 | Standard I Block |
| I | 0 | N | 1 | 0 | 0 | 0 | 0 | 0 | Chain bit set |
| R | 1 | 0 | 0 | N | 0 | 0 | 0 | 0 | No errors |
| R | 1 | 0 | 0 | N | 0 | 0 | 0 | 1 | EDC / Parity error |
| R | 1 | 0 | 0 | N | 0 | 0 | 1 | 0 | Other errors |
| S | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | Resynch request |
| S | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | Resynch response |
| S | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | IFS request |
| S | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | IFS response |
| S | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | Abort request |
| S | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | Abort response |
| S | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | WTX request |
| S | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | WTX response |

The S blocks are used to invoke four control states as shown in the table. The resynch request is used by the IFD (only) to force a reset of the block transmission parameters to their initial values. A chain may be aborted by either the IFD or ICC perhaps due to some physical error such as memory corruption. The ICC may send an IFS request to invoke a change in the IFSC it can support. Similarly the IFD may send an IFS request to indicate a new IFSD it can support. The S block control also allows the ICC to request an extension to the block waiting time (BWT) that may be necessary to execute a command received in an I block. The INF field in this block contains a single byte integer value which is to be calculated as a multiple of the BWT value. In all cases the receiver of an S block should send the appropriate response block.

**The T = 1 Protocol in operation**

Using the notation of the ISO 7816 standard we can show the basic operation of the protocol. A more complete definition can be obtained from the standard.

* + I Blocks; I (N,M)
  + Where  
    N = Sequence number (alternately`0' and `1' )

M = More data bit

The More data bit is set when an additional I block is to follow in the chain

* + R Block; R (N)

Where N = Sequence number of next expected block

The protocol defines that the IFD and the ICC each have the right to transmit in turn where communication commences with transmission of a block by the IFD.

[Figure 19](http://www.sat-digest.com/SatXpress/SmartCard/info/FIG19.GIF)

**Normal I block transmission**

[Figure 20](http://www.sat-digest.com/SatXpress/SmartCard/info/FIG20.GIF)

**I Block Transmission with chaining**

Note that an I block was used by the ICC to acknowledge the last block in the chain sent by the IFD. The ICC may send chained blocks in the same way as shown for the IFD.

[Figure 21](http://www.sat-digest.com/SatXpress/SmartCard/info/FIG21.GIF)

**Error handling in I block transmission**

Error in I block receipt

[Figure 22](http://www.sat-digest.com/SatXpress/SmartCard/info/FIG22.GIF)

Error in I block chain response

[Figure 23](http://www.sat-digest.com/SatXpress/SmartCard/info/FIG23.GIF)

In both cases the transmitter is notified to retransmit the block received in error. There are of course a large number of possible error scenarios but they are all based on the simple concepts shown above.

**Summary**

In this article we have given a brief overview of the technology of Smart cards. We have looked at the basic components that make up the Smart Card and have explored the elements of the chip which are at the centre of this technology.

Many poeple have questioned why the introduction of Smart cards have been so slow for which the typical response has always related to the lack of standards. We have tried to show here that the standards do exist and that the central core necessary to build cards and devices are fairly well defined. It is particularly appropriate at this time to comment that there are several major commercial projects using Smart Cards such as "Mondex" (Global Electronic Cash) that are based on these ISO standards. Similarly we can now also report on the major payment schemes of Visa, Mastercard and Europay who have worked together to produce specifications based on the ISO standards for the introduction of chip cards in their varioius systems.

It has been said many times before but now at last we really are seeing the day of the Smart Card.