SIM CARD PROTOCOLS

Though rarely thought about by most users their mobile phone contains a remarkable computing device that enables them to go about their business of making calls, text messaging or playing online games. The Subscriber Identity Module (SIM) inside the mobile phone is actually a fully fledged microcomputer with an operating and file system. The protocols used between the mobile equipment or card accepting device and the SIM are a combination of protocol types that should be familiar to anyone who has studied the Open Systems Interconnection (OSI) Reference Model-that is to say the data link and application layers (though the T=0 model mixes layers).

This paper attempts in broad strokes to outline the construction of these protocols and how they are used.

Card Power-Up

When a smart card, or in our discussion SIM card, is inserted into a card reader or mobile handset the card is not immediately powered up. Doing so without out at first testing if the card is properly inserted can destroy the card.

Once the proper insertion of the card has been detected, the card accepting device (CAD) or mobile equipment (ME) the CAD or ME first powers the card to a normal operating voltage and the card is brought to an idle state. Then a RST command is sent over the contact line (C2) on the face of the SIM card. The RST command remains in a low state for a set period of time and then is set to a high state. This is a signal to the SIM card to begin its initialization sequence.

The initialization sequence ends with the SIM card sending an Answer to Reset (ATR). Its primary purpose is to indicate the status of the smart card power-up sequence. It also conveys information which the reader requires in order to optimize the speed of communication between the reader and the card. The total length of the ATR sequence is limited to 33 bytes (The ATR must adhere to the structure specified by ISO 7816-3). The table shown below illustrates the ATR as outlined in ISO 7816-3.

Character ID	**Definition**
Initial Character Section	
TS	Mandatory initial character
Format Character Section	
T0	Indicator for presence of interface characters
Interface Character Section	
TA_1	Global, codes F1 and D1
TB_1	Global, codes 11 and Pl1
TC_1	Global, code N

TD_1	Codes Y_2 and T
TA_2	Specific
TB_2	Global, code Pl2
TC_2	Specific
TD_2	Codes Y_3 and T
TA_3	TA_i, TB_i and TC_i are specific
...TD_i	Codes Y_{i+1} and T
Historical Character Section	
T1	Card specific information
...TK	(Maximum of 15 characters)
Check Character Section	
TCK_2	Optional check character

Table 1: ATR structure[1].

TS and T0 are the only mandatory bytes in the ATR sequence. The initial character TS is used to establish bit-signaling and bit-ordering conventions. T0 is used to indicate the presence or absence of subsequent interface or historical characters. The upper 4 bits or nibble (bits 5 - 8) is designated Y1 and signals the presence of optional characters based on a logic 1 in the following bit positions:

- Bit 5 indicates TA_1 is present
- Bit 6 indicates TB_1 is present
- Bit 7 indicates TC_1 is present
- Bit 8 indicates TD_1 is present

The lower nibble (bits 1 - 4) is designated K and is interpreted as a numeric value in the range 0 - 15. It indicates the number of historical characters present.

The interface characters are used to select the protocol and parameters used for subsequent higher-level communication between the smart card and the reader. The TA_1 interface character is the most significant to this paper as it provides the necessary information to achieve optimal communication speeds between the reader and the card. The value of Y1 indicates whether TA_1, TB_1, TC_1 or TD_1 is present and the value of Yi (encoded in TD_{i-1}) determines the presence of TA_i, TB_i, TC_i and TD_i. The purpose of a few of the most relevant of these is explained below.

- TA1: Encodes FI in the upper nibble (bits 5 - 8) and DI the lower nibble (bits 1 - 4). These are used to determine the clock conversion factor (F) and the bit rate adjustment factor (D) and maximum supported clock rate according to the tables below. The default values of F and D if unspecified are: F = 372 and D = 1.

FI	0000	0001	0010	0011	0100	0101	0110	0111
F	Internal clock	372	558	744	1116	1488	1860	RFU
***f* max (MHz)**	-	5	6	8	12	16	20	-

FI	1000	1001	1010	1011	1100	1101	1110	1111
F	RFU	512	768	1024	1536	2048	RFU	RFU
***f* max (MHz)**	-	5	7.5	10	15	20	-	-

Table2: Values of the upper nibble of the TA1 byte of the ATR[1].

DI	0000	0001	0010	0011	0100	0101	0110	0111
D	RFU	1	2	4	8	16	32	RFU

DI	1000	1001	1010	1011	1100	1101	1110	1111
D	12	20	1/2	1/4	1/8	1/16	1/32	1/64

Table 3: Values of the lower nibble of the TA1 byte of the ATR[1].

- TB_1, TB_2: Used to encode information concerning the programming voltage and current.
- TC_1: Interpreted as an 8-bit unsigned integer representing the extra guard time required between characters (N). The default value of N is 0.
- TD_1: Encodes Y1 in upper nibble and the protocol type in the lower nibble. ISO 7816-3 defines two protocols: the T=0 protocol and the T=1 protocol. T=0 is an asynchronous *character-oriented protocol* where an acknowledgement must be received for every byte that is sent. In contrast, T=1 is an asynchronous *block-oriented protocol* where a number of bytes can be sent before an acknowledgement must be received . If TD_1 is not present, then T=0 is used implicitly.

The historical characters are usually used to indicate the type, model and use of the specific card. These are generally defined by the manufacturer or card issuer. There is no established standard for the data in these historical bits. The check character (TCK) is used to determine whether a transmission error occurred in sending the ATR from the card to the reader. TCK is a checksum calculated such that performing a bit-wise exclusive-or (XOR) operation on all bytes in the ATR from T0 to TCK results in a answer of zero.

Transmission Protocol Data Units

Transmission Protocol Data Units (TPDUs) are the data structures passed back and forth by the SIM and the CAD or ME. The two protocols commonly used for this communication are the T=0 protocol and the T=1 protocol.

T=0

This protocol is a byte oriented protocol, meaning that the minimum unit of information transferred across the I/O channel is a byte. Error handling for this protocol is the same. This protocol tends to mix elements from the link-level and application-level protocol layers.

The TPDU of the T=0 protocol is comprised of two data structures-one sent from a reader or device to the card and one sent back from the card(similar to command and response). The command header of the T=0 protocol includes the following five fields.

CLA-this is a one byte field that is a collection of instructions

INS-This is a one byte field that specifies a specific instruction to the card from within the set of instructions in the CLA

P1-This is a one byte field that used to specify the addressing used by the INS and CLA instruction

P2-this is a one byte field also used in addressing

P3-this is a one byte field that is used to specify the number of bytes transferred to or from the card as part of the instruction execution[2].

The below table lists some of the instructions that can be contained in the CLA byte.

CLA Byte	Instruction Set
0X	ISO 7816-4 instructions (files and security)
10 to 7F	Reserved for future use
8X or 9X	ISO 7816-4 instructions
AX	Application/vendor specific instructions
B0 to CF	ISO 7816-4 instructions
D0 to FE	Application/vendor specific instructions
FF	Reserved for protocol type selection

Table 4: CLA instruction set definitions[2].

The next table shows the instructions set in the INS byte for security and file system access on the card.

INS Value	Command Name	INS Value	Command Name
0E	Erase Binary	C0	Get Response
20	Verify	C2	Envelope
70	Manage Channel	CA	Get Data
82	External Authenticate	D0	Write Binary
84	Get Challenge	D2	Write Record
88	Internal Authenticate	D6	Update Binary
A4	Select File	DA	Put Data
B0	Read Binary	DC	Update Record
B2	Read Record(s)	E2	Append Record

Table 5: ISO 7816-4 INS codes[2].

As was mentioned before the T=0 protocol tends to mix elements from other levels in the OSI model. The P1 and P2 parameters, although defined at the link level protocol level are in truth dependant on the instruction that is specified. In other words they are dependant on the application protocol information. These two parameters provide control or addressing for application-specific instructions such as the **Select File** instruction which involves the selection of a specific file in the file system which then allows for other operations such as writing or reading.

P3 is also an application-level parameter. P3 generally specifies the number of bytes to be transmitted during the INS specified instruction execution. Data is either outgoing-moving from card to reader or incoming-from reader to card.

After each command TPDU, a response TPDU is returned. This TPDU is made up of a number of procedure bytes. This TPDU contains three Mandatory fields and one optional one.

- ACK: indicates that the card has received the [CLA, INS] command
- NULL: used for flow control on the I/O channel by the card. It signals (to the reader) that the card is still processing the command and so the reader must wait before sending another command
- SW1: status response of the current command
- SW2: (optional) also conveys a status response to the reader[2]

The ACK byte is a repeat of the INS byte that was sent by the command TPDU. The second byte NULL, is a way to mark the time period for processing the command. If a response is not received within a timeout period the reader of device may send out a RST sequence to reinitialize the protocol. A least one NULL response received by the reader will prevent this from occurring.

The SW1 is a status byte from the card to the reader that will tell the result of the sent instruction; in certain instructions the card may return data bytes to the reader or equipment. If this is the case the SW2 status byte is returned. This is a trigger for the reader to now execute another command-the **GetResponse** command, which actually returns the data from the previous executed instruction.

Error checking occurs in the T=0 protocol with a parity check. For every byte transferred 11 bits must be used. The parity bit is cleared or set to make the total number of bits set be even. The receiving side of the channel can look at the bits transferred prior to the parity bit and determine what parity value to expect. If the actual transmission does not match what was expected it can be assumed that an error has occurred and a recovery procedure must take place. This recovery is initialized on the receiving side and involves the re-transmission of the byte received in error.

T=1

This protocol is block-oriented; meaning that a defined collection of data or block is moved as a whole between the card and reader. This block of data may contain an Application Protocol Data Unit (APDU) defined for a specific application. This shows a good example of the layering between the link and application protocol layers. Moving the information as a single block calls for error free transmission and this error detection and correction is much more complex for the T=1 protocol than in the T=0 protocol.

This error detection is done using a longitudinal redundancy character (LRC)-a slightly more complex parity type check that exists in the T=0 protocol or using a cyclical redundancy check (CRC). The specific CRC algorithm used is defined in ISO 3309.

The T=1 protocol uses three types of blocks:

1. Information block: Used to exchange data between application software on the card and application software on the reader side of the channel.
2. Receive ready block: Used for positive and negative acknowledgements on either end of the channel. A positive acknowledgement indicates that the block was correctly received. Conversely a negative acknowledgement indicates that an error was detected in the block received.
3. Supervisory block: Used to transfer control information between the reader and the card.

Each of the data blocks has the same structure comprising of the following fields.

- Prologue field: Mandatory field with a size of three bytes including the following three elements:
    - NAD: Node Address is used to identify the addresses of the source and the intended destination of the block.
    - PCB: Protocol Control Byte is used to indicate the type of block (either information, receive ready or supervisory)
    - LEN: Length of the block
- Information field: Optional field that may be up to 254 bytes in length and may contain an APDU
- Epilogue field: Mandatory field that is either 1 or 2 bytes in length and is used for error detection

When used the NAD element also contains two subfields: SAD-which is the source address shown by the three low order bits of the NAD and DAD, which is the destination address which is shown in bits five through seven of the NAD.

In the PCB, the two most significant bits (high order) of the byte indicate of which type the block is:

- A high order set to zero indicates and informational block
- The two high order bits set to one indicate a supervisory block
- The high order bit set to one and the other to zero indicates a receive ready block

Application Protocol Data Units

The ISO 7816-4 standard addresses two areas of functionality for application software:

- File system: A set of functions is provided in the form of an API. By using this API application software on the reader side can access files in the file system.
- Security functions: These can be used to limit access to application software or to files on the card.

The T=0 or T=1 protocols are used to support application-level protocols between the smart card application and the reader application. These application protocols exchange data structures called application protocol data units (APDUs). The following diagram illustrates this architecture:

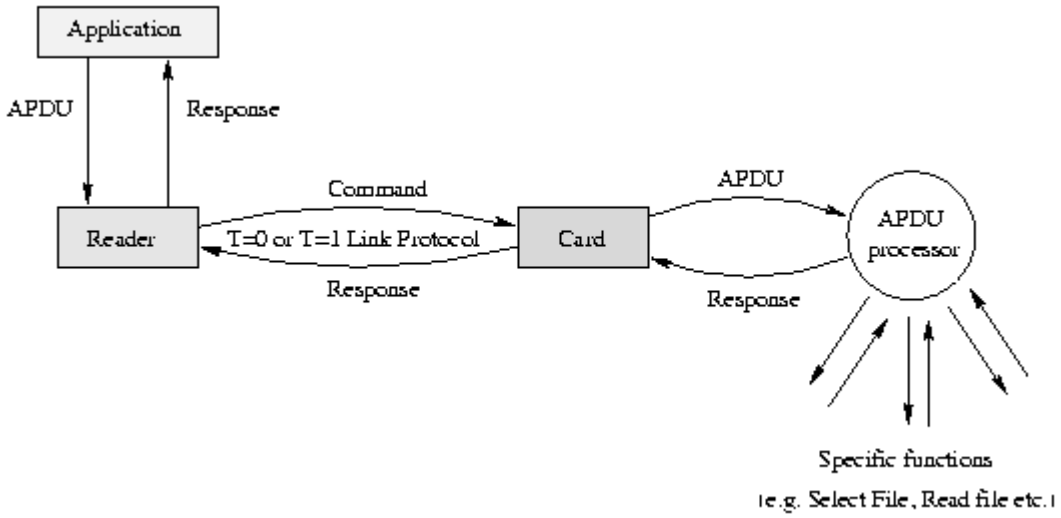

Figure 1: Application communication architecture[2].

The APDU structure defined by ISO 7816-4 is very similar to the TPDU structure used in the T=0 protocol. In fact, when an APDU is transported by the T=0 protocol, the elements of the APDU directly overlay the elements of the TPDU.

The APDU as defined by ISO 7816 is link-level protocol independent and also defined at the application level.

APDU Structure

There are two types of messages used to support the ISO 7816-4 application protocols: the *command APDU* (sent from the reader to the card) and the *response APDU* (sent from the card to the reader).

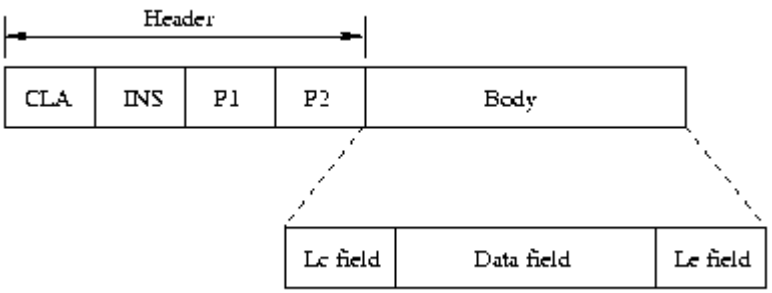

Figure 2: The command APDU structure[2].

The command APDU consists of a header and a body (shown above). The header includes CLA, INS, P1 and P2 fields. As in the T=0 protocol, CLA and INS specify an application class and instruction. P1 and P2 are used to qualify specific instructions and are given specific definitions by each [CLA, INS] instruction. The body of the APDU can vary in size and is used to transmit data to the card's APDU processor as part of a command or to convey a response from the card to the reader. The Lc field specifies the number of bytes to be transmitted to the card as part of the instruction-length of the data field. The data field contains information that must be sent to the card to allow its APDU processor to execute the command specified in the APDU. The Le field specifies the number of bytes that will be returned to the reader in the response APDU.

The body of the APDU can appear in four forms:

1. No data is transferred to or from the card, so the APDU only contains the header.
2. No data is transferred to the card, but data is returned from the card. The body of the APDU only contains a non-null Le field.
3. Data is transferred to the card, but none is returned from it. The body of the APDU includes the Lc and data fields.
4. Data is transferred to the card and is also returned from the card as a result of the command. The body of the APDU includes the Lc, data and Le fields.

Figure 3: The response APDU structure.

The response APDU consists of a body and a trailer- considerably simpler than that of a command APDU. The body is either null or it includes a data field – this depends on the specific command and its success or failure by the APDU processor. If the APDU contains a data field its length is determined by the Le field in the corresponding command APDU.

The trailer consists of up to two fields of status information called SW1 and SW2. These fields return a status code in which one byte is used to specify an error category and the other is used to specify a command-specific status or error indication.

The error codes follow the ISO 7816 numbering scheme in which one byte is used to determine error category and the other shows the specific error.

RETURN CODE						
PROCESS COMPLETED				PROCESS INTERRUPTED		
Normal	Warning			Execution Error		Checking Error
61XX or 9000	62XX	63XX		64XX	65XX	67XX to 6FXX

Table 6: ISO 7816 Error Return Codes

Lastly, in the ADPU the CLA byte has two additional noteworthy features:

- The two low order bits can indicate a logical communication channel between the card APDU processor and the reader application.
- The next two high order bits can indicate that secure messaging is being used between the reader application and the card APDU.

The goal of this paper was to outline the general communication protocols that the mobile phone SIM card-which inherits these from smart cards-uses to communicate with card reader or the mobile equipment. The TPDU protocols T=0 and T=1 were discussed as well as the APDU structure. The examiner is encouraged to examine the cited works further for more in depth information regarding SIM card protocols and commands.

CITATIONS

1. Shillington Nicole, Waker Travers, *The Design of a Smart Card Interface*, http://www.cs.uct.ac.za/Research/DNA/SOCS/projectpage.html
2. Guthery Scott B., Jurgensen Timothy M., *Smart Cards: The Developers Toolkit*, Prentice Hall 2002, 78-113