Universal "Chip Based Smart Card" Snooping Device

Dimitri Denamany (dmi3@engineer.com)
Patrick Sebastian (patrick_sebastian@petronas.com.my)
A B Sayuti M Saman (sayutis@petronas.com.my)
Universiti Teknologi PETRONAS, Tronoh, Perak, Malaysia

***Abstract*- The paper discusses the design, construction and test of a Universal "Chip Based Smart Card" Snooping Device. This device would function by displaying the actual data communication between the smart card and the smart card reader. The development and design of the snooping device requires that the communication protocols between the smart card and smart card reader be understood and displayed. This snooping device would display all data communications that occur between the smart card and the smart card reader that follow the ISO 7816-3 protocol. The proliferation and application of smart cards in the real world can be furthered by the application of this device. This device would be able to show the actual data communication between the smart card and the smart card reader. The hardware design of the device has been divided into 3 main sections, the snooping module, the communication module and the processing module. The snooping module taps the data being transmitted between the smart card reader and the smart card, the communication module formats, encodes and transfers the data to the processing module, and lastly the processing module that translates the transferred data into the appropriate form and display the data on screen.**

***Index terms*- Smart Card, Snooping Device, Data Transfer, Application Protocol Data Unit**

I. Introduction

Smart cards offer unique features that are attractive such as the ability to store data securely and the ability to run small programs [2]. The ability to store data securely has greatly increased the application and proliferation of smart cards especially in the areas of security and authentication [1]. Apart from the ability to store and retrieve data securely, the smart cards also provide additional functionalities that can be seen in the area of transportation, banking and telecommunications [4].

With the ability of being a multi-purpose card, the task of developing applications that communicate with smart cards becomes a daunting task [2] as there is a need to ensure the communication of data between the smart card and the smart card reader. As such the process of providing smart card technical support for the integration of any customer's system with smart cards will have all kinds of smart card and smart card reader related issues such as incorrect data transmission and wrong operations. These problems are a norm during the implementation of a new system that features smart card technology. Considering this fact, there is a need in many situations where the troubleshooting process would be easier if the data transfer between a smart card and a smart card reader could be monitored for debugging purposes.

The data transfer between the smart card and the smart card reader presents the lowest comprehensible programming level that can be accessed or snooped upon. This is where the problem arises where this type of equipment is either extremely expensive or not unavailable. In developing a smart card snooping device, an understanding of the technology used has to be firstly understood. This includes the type of signals used in transmitting data and the smart card protocols used for communication between the smart card and smart card reader.

By utilizing smart card snooping devices, smart card application developers would be able to see the actual data transmitted and this would assist in the development and troubleshooting of smart card applications. An indirect benefit of the snooping device is cost savings obtained from the ability of developing and troubleshooting applications more effectively. In addition, this type of snooping device would function as a universal analysis tool for any device installed with smart card readers ranging from PC's, ATM Readers to customized smart card operated systems. The development of this device is a further advancement of the work done by Lim et. al [3] that was a PC-based smart card reader whereas the device developed was a portable handheld PALM device.

II. Smart Card Data Transmission

A. *Smart Card Communication Details*

The smart card data communication is conducted through the I/O contacts, where the transmitted data complies with the ISO 7816-3 protocol [1] [2]. Data communication with the smart card is classified as half duplex communication where the data transfer can take place in one direction at any point of time. The data is transferred through the I/O pin using asynchronous serial transmission. Table I shows the asynchronous data transmission details of the card.

There are four contacts tapped from the smart card, Vcc, Gnd, I/O and the RST. The Vcc and ground contacts are tapped to power up the MAX232CPE chip. The I/O and the RST contact are the lines that carry the needed data. The I/O line is kept high when there is no data transfer between the card and the smart card reader.

TABLE I
Asynchronous Data Transmission Details

Baud Rate	9600 *(varies)*
Start Bit	1 Start Bit
Stop Bit	2
Parity Bit	Even parity

The RST line functions as a reset for the smart card and is kept high when the card is functioning. When a card is inserted into the reader, all lines will remain low until all contacts are properly positioned. Once all contacts are positioned properly, the card is powered up as the Vcc contact is brought high. The Vcc contact will remain high until the card is removed.

After card insertion, the smart card will RESET. The first data transmission upon a card reset is that the card would transfer a string of bytes known as the Answer to Reset (ATR) to the smart card reader The ATR sequence is responsible for initializing the physical communication channel between the reader and the smart card. It facilitates the definition and manipulation of characteristics of the channel. After the ATR sequence is completed, the remaining communication between the smart card and the smart card reader involves only the transfer of Application Protocol Data Units (APDU).

APDU's are commands that are sent to the smart card in order to perform certain operations. The card would respond to the command in order to indicate whether the command had failed or succeeded.

B. Answer-To-Reset (ATR) Standard Format

The ATR is a string of characters returned from the card indicating a successful power-up sequence. The total length of the ATR sequence is limited to 33 bytes and must adhere to the format shown in Table II.

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 (bits 5 - 8) of T0 are designated Y1 and signals the presence of optional characters. The lower 4 bits (bits 1 - 4) are designated K and it indicates the number of historical characters present.

The interface characters are used to select the protocol used for subsequent higher-level communication between the smart card and the reader. The historical characters are usually used to indicate the type, model and use of the specific card where these are manufacturer defined. The check characters (TCK) are used to determine whether a transmission error occurred in sending the ATR from the card to the reader.

TABLE II
ANSWER TO RESET STRUCTURE

Name	Number	Function	Presence
TS	1	Initial Character	Mandatory
T0	1	Format Character	Mandatory
TA_i, TB_i, TC_i, TD_i	<15	Interface Characters	Optional
T1, T2…TK	<15	Historical Characters	Optional
TCK	1	Check Characters	Conditional

C. APDU Standard Format

The APDU is a string of formatted bytes that make up a command to the smart card. The card will reply with a standardized response. The response indicates the success or failure of the command together with requested data.

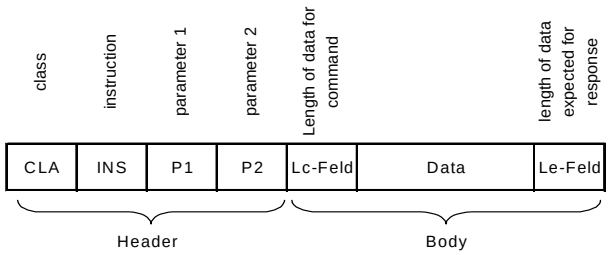


Figure 1. APDU Format

Fig. 1 shows the basic format for the APDU. The 5 bytes shown at the beginning are called the header of the APDU. The CLA and INS bytes are indications of the type of command being invoked. Bytes P1 and P2 on the other hand are bytes that contain the setting for options related to the command. The L_c field indicates the number of bytes that would be trailing the APDU header. The data portion is filled up with the necessary information if there is any data to be sent to the card .The final byte, the L_e field is optional as its use depends if the card reader expects data to be transmitted from the card [5] [6].

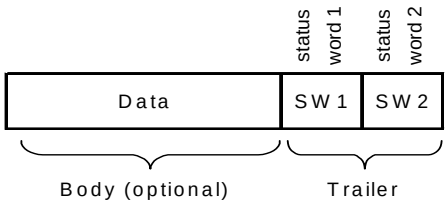


Figure 2. APDU Response Format

Fig. 2 shows the standard APDU response format. For every command (APDU) sent to the card, the SW1 and SW2 (which are known as the *status word)* are two compulsory bytes that must be sent from the smart card to the reader after every command. These are the two bytes that indicate the success or failure of the command. The data field that precedes the SW field in Fig. 2 is the data that will be sent from the card to the reader. The length of this field is stated in the APDU in the Le field [5] [6].

III. HARDWARE DESIGN

A. General Block Diagram of modules

Fig. 3 illustrates the smart card framework architecture that includes the location of the snooping device. The term Card Terminal is used for card-acceptance devices that include card reader, card terminal, interface device and slot that all refer to the same device [2]. The Card Service term is used to represent the various application objects such as electronic signature API. Where the Card Services available are dependent on the availability of those services on the system. The position of the snooping device is placed between the smart card and the Card Terminal where it is positioned to snoop on the data being transmitted between the smart card and the terminal.

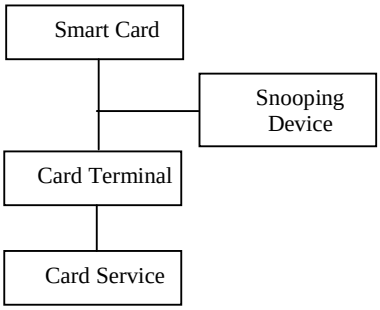


Figure 3. Smart Card Framework Architecture

Fig. 4 illustrates the 3 main modules that make up the snooping device. The snooping module is the module that consists of a connector and a custom designed dummy smart card. The communication module is comprised of a MAX232 chip and a serial connector. The processing module accepts the serial data and processes it to display the information that is gathered.

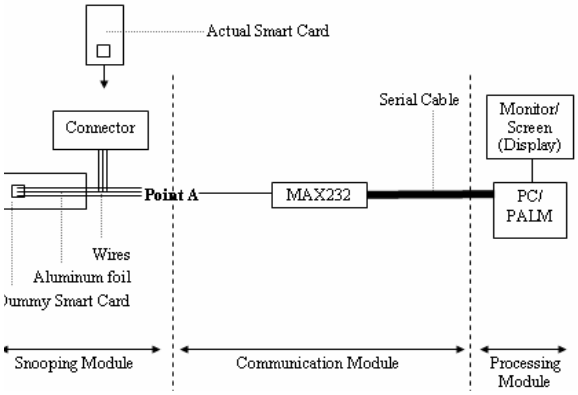


Figure 4. Device Block Diagram

Snooping Module

The dummy smart card is a specially fabricated PCB board that has the same dimensions as an actual smart card and eight contacts at the exact same position as an actual smart card. These eight contacts are all linked to the edge of the card where connector pins are connected to it. The connector pins will be used to link the contacts of the dummy smart card to the contacts of the actual smart card. The purpose of the connector is to make the design more robust and to make snooping of various smart cards convenient.

The connector has a slot where an actual smart card is inserted. The pins of the smart card would then come into contact with a copper contact plate that is physically linked to the 8 pins on the dummy smart card. All signals to and from the smart card will be transmitted to the smart card reader through the physical connection set up through the dummy smart card and the connector. With the physical connections done, the snooping circuitry begins from certain contacts of the smart card that are tapped in parallel with the existing circuitry.

A smart card has a total of eight contacts. In order to snoop on data transfer, four main contacts are tapped which are the I/O contact, V_{cc} contact, Gnd contact and the RST (RESET) contact. The I/O contact contains the data that is transferred from the card the reader and vice versa. The RST contact on the other hand is the reset pin for the smart card chip. These two lines are linked to the Communication module.

B.

C. Communication Module

The communications module is the portion of the circuit that handles the transferring and data formatting from the snooping module to the processing module. The module consists of two main components, the MAX232CPE chip and the RS232 Connector. From the snooping module, four contacts are tapped from the card and brought into the communication module.

The Ground contact is connected to both the MAX232 and the RS232 connector (GND-Pin 5) for the sake of standardization. The Vcc contact is connected to the Vcc of the MAX232 chip. The MAX232 chip obtains its power source from the smart card 5V V_{cc} contact. The I/O contact and the RST contacts are connected to the MAX232 chip.

The signals on these contacts are represented as 5V (HIGH logic) and 0V (LOW logic) are converted to a representation understood by the serial port of the processing module by the MAX232 chip. The HIGH logic on the serial port is represented as a voltage between –3V and –12V and a LOW logic is represented as a voltage between 3V and 12V. After the signals are converted, the I/O signal is sent to the Receive (RX – Pin 2) Line of the RS232 connector. The output of the RST line from the MAX232 on the other hand is sent to the Data Carrier Detect (DCD-Pin 1) Pin.

D. Processing Module

The processing module serves two main purposes; the first is to process the data sent through the serial port. The second function is to organize the information properly and display it in a meaningful manner. In the final outcome, the processing module used was a PALM device and the serial connector from the communication module was connected to the HotSynch cable of the Palm.

For the PALM device to receive the serial data and to process the data, a Palm application was developed. The initial stage of the PALM application execution involves the setup of the serial port of its respective hardware. Parameters such as baud rate, start bit and stop bit have to be set to establish a successful communication channel. The serial port is opened and the receive buffer is checked periodically for incoming data. Every time data is received, memory is allocated for the data for temporary data storage. The data is then processed to display the transmitted bytes on screen. To display the data on screen, the data must first be translated in to its ASCII representation.

IV. RESULTS

A. PALM program

The PALM snooping program is the most important component in the processing module. The program handles the establishing of a connection to the serial port, accepting serial data, converting it into ASCII format and finally displaying the information in a user-friendly manner. Palm programs are event-based applications that run in an infinite loop. Upon execution of the program, the application is set up where several loop handlers are available and each of these handlers performs certain functions. When anything is done to the Palm, an event is dispatched into the loop and the

event handler will take the appropriate action in order to respond to the incoming event. Once the event has been completed, it is discarded and the program goes back to looping till the next event is dispatched into the loop.

The program starts off by automatically opening the serial port according to configuration information that indicated the baud rate, number of start bit, number of stop bits and parity bits. Upon completion of opening the serial port, the PALM device is ready to accept data. After the port is opened and the data received, each byte of data is sent into a byte conversion subroutine. The byte conversion subroutine is a portion of the program that converts the bytes received into its ASCII representation to be properly displayed. For instance if a bytes 0x00 is received, in order to display 0x00, it has to be converted in to 0x30 and 0x30 where 0 is represented as hex 30 in ASCII. The conversion is done byte by byte and the output is immediately displayed upon exiting the subroutine.

B. Program Output analysis

Fig. 5 is a sample output obtained during the insertion of a smart card into the smart card reader. The Palm emulator displays a portion of the data communication that took place during the insertion of the smart card. In illustrating the data transmission, focus will be given to the first few lines. Data transmission begins with the card transmitting its ATR. The remaining transmitted data are commands that are APDU's and responses from the smart card. The following is a copy of the data in Fig. 5 with numbered lines for ease of illustration.

Line 1 : 3B951840FF6201020104 F0220700
Line 2 : 009000 C0A4000002A43F006114 C0
Line 3 : A4000002A40000610FC0A4000002
Line 4 : A43F006114 C0A400C0A4000002A4
Line 5 : 3F006114 C0A4000002A43F116114

Line 1 (beginning) is the first transmission that is from the card. The first transmission is the ATR. Hence 3B951840FF6201020104 is actually the ATR of the card. After the transfer of the ATR bytes, the program proceeds with the command (APDU) and response sequence. In the log shown, Line 4 onwards basically corresponds to the APDU format. The bytes in Line 1(ending) and 2 (beginning) are F0220700009000. Breaking the bytes into two portions, we can see that the command APDU is actually F022070000 where else the response status word (SW) is 9000. Refer to Table III to check up on the returned status word.

TABLE III
INTERPRETATION OF LINE 1(ENDING) AND LINE 2 (BEGINNING)

Command APDU					Response SW	
CLA	**INS**	**P1**	**P2**	**L_c**	**SW1**	**SW2**
F0	22	07	00	00	90	00

The data in Line 2 (middle) (C0A4000002A43F006114) can also be classified using the same manner as stated for the command in Line 1(ending) and 2 (beginning) above. However, there are certain differences between Line 2 (middle) and Line 1(ending) and 2 (beginning). First of all, since the L_c field is 02, this means that there are two data bytes that will trail the APDU header. The second difference is that sometimes, there are some extra bytes that are transferred to the smart card in order to act as a separator. These bytes are not actually part of the data packet.

TABLE IV
INTERPRETATION OF LINE 2 (MIDDLE)

Command APDU							Response SW	
CLA	**INS**	**P1**	**P2**	**L_c**	**Data**		**SW1**	**SW2**
C0	A4	00	00	02	3F	00	61	14

From Table IV, it can be seen that there is one byte that was missed out, the A4 byte after the L_c parameter. This is the special byte that acts as a separator. By referring to Line 3, 4 and 5, it can be seen that this byte and probably a C0 byte is repeated in each and every APDU transfer. The status words that are returned are rather straightforward. All the communication from line 2 onward (after the ATR in line 1) complies with the standard APDU format. Hence, each and every APDU send to the card can be logged and the response sent from the card to the reader can also be documented. This repeats until the card is disconnected or reset. If the card is reset, the data transmission starts all over again with the transmission of the ATR and the corresponding APDU's as shown in Fig. 5.

Figure 5. Sample output of the snooping program

This device was successfully designed and implemented using various devices and methods. The hardware configuration can be classified into three modules.

The first module, the snooping module is a combination of two specially engineered components that are used in order to tap on the required signal lines between the card and the reader. The second module, the communication module functions to encode the signals into the required voltage levels and format for the serial port. Lastly, the processing module accepts the data through the serial port, processes it and displays the byte exact byte transfer in the screen. The processing module for the final design was a PALM device

that was an affordable and, more importantly, portable device.

The development of this snooping device is further advancement on the work done by Lim, et. al.[3] that was a PC-based smart card reader. The data log that was created in the processing module complies with the IS0-7816 smart card communication protocols. This standard is the basic byte transfer format for the communication between the smart card and the smart card reader. The development of this device will allow for faster development of smart card applications due to the details of smart card data transmission that can be snooped and displayed.

REFERENCES

D. Husemann, "The smart card: Don't leave home without it," IEEE Concurrency, Vol. 7, Apr. – June 1999.

D. Husemann and R. Hermann, "Open Card: Talking to your Smart Card," IEEE Concurrency, Vol. 7, July – Sep. 1999.

Lim C.H., Dan Y.H., Lau K.T. and Choo K.Y, "Smart Card Reader," IEEE Transactions on Consumer Electronics, Vol. 29, Feb. 1993.

M.R. Carr, "Smart Card Technology with case studies," Proceedings of 36th Annual 2002 International Conference on Security Technology, Oct. 2002.

William Stallings, "Data and Computer Communication," Sixth Edition 2003,Prentice Hall International Inc.

Henry Dreifus, "Smart Cards:A guide to building and managing smart card applications," 1st Edition 1996, John Wiley & Sons Inc.

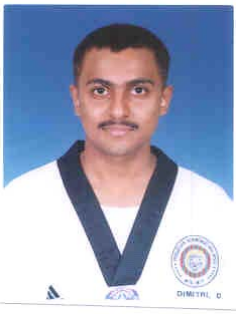
Jose Luis Zoreda, "Smart Cards," 1st Edition 1994, ARTECH HOUSE Inc.

Robert MykLand, "PALM OS Programming from the Ground Up," 1st Edition 2000, McGraw-Hill

Steve Mann, "Advanced PALM Programming," 1st Edition 2001, John Wiley & Sons Inc.

)] Eric Giguere, "Palm Database Programming," 1st Edit
1999, John Wiley & Sons Inc.

l] PALM OS SDK Documentation



Dimitri Denamany graduated with a B.Eng (Hons.) in Electrical and Electronics with a major in Computer Systems from Universiti Teknologi PETRONAS, Malaysia in 2004.

He is currently on contract as a Smart Card Engineer (R&D) with AXALTO, an international company that does business and development of smart cards. He is currently awaiting placement with SCHLUMBERGER as an International Field Engineer (Wireline). His interests are in the areas of Power Electronics, Power Systems and Software Engineering.

Patrick Sebastian received his B.Eng (Hons.) in Electrical Engineering with a major in Electronics from Universiti Teknologi Malaysia, Malaysia in 1993, and the M.Sc. in Electrical Engineering from University of Minnesota, USA in 2003.

From 1993 to 1997, he was attached to Carsem(M), Malaysia as an Equipment Engineer. And from 1997 to 2001, he was attached to Seagate Technologies, Malaysia as a Senior Process Engineer and Six Sigma Black Belt. He is currently a Lecturer in the Department of Electrical and Electronics Engineering at Universiti Teknologi PETRONAS, Malaysia. His interests are in the areas of computer systems architecture, power dissipation of microprocessors, digital image processing and real-time systems.

A B Sayuti M Saman received his B.Eng in Electrical, Electronics & Systems from Universiti Kebangsaan Malaysia (UKM) in 1991, and M.Sc. in Computer Systems Design from University of Manchester Institute of Science and Technology (UMIST), UK in 2003. He is currently a Lecturer in the Department of Electrical and Electronics Engineering at Universiti Teknologi PETRONAS, Malaysia. His interests are in the areas of r