GemCore V1.21-Based Reader

Reference Manual

Version 1.0

April 2000

SPECIFIC WARNING NOTICE

All information herein is either public information or is the property of and owned solely by Gemplus who shall have and keep the sole right to file patent applications or any other kind of intellectual property protection in connection with such information.

Nothing herein shall be construed as implying or granting to you any rights, by license, grant or otherwise, under any intellectual and/or industrial property rights of or concerning any of Gemplus 's information.

This document can be used for informational, non-commercial, internal and personal use only provided that:

- **the copyright notice below, the confidentiality and proprietary legend and this full warning notice appear in all copies.**
- **this document shall not be posted on any network computer or broadcast in any media and no modification of any part of this document shall be made.**

Use for any other purpose is expressly prohibited and may result in severe civil and criminal liabilities.

The information contained in this document is provided « AS IS » without any warranty of any kind. Unless otherwise expressly agreed in writing, Gemplus makes no warranty as to the value or accuracy of information contained herein. The document could include technical inaccuracies or typographical errors. Changes are periodically added to the information herein. Furthermore, Gemplus reserves the right to make any change or improvement in the specifications data, information, and the like described herein, at any time.

GEMPLUS HEREBY DISCLAIMS ALL WARRANTIES AND CONDITIONS WITH REGARD TO THE INFORMATION CONTAINED HEREIN, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. IN NO EVENT SHALL GEMPLUS BE LIABLE, WHETHER IN CONTRACT, TORT OR OTHERWISE, FOR ANY INDIRECT, SPECIAL OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER INCLUDING BUT NOT LIMITED TO DAMAGES RESULTING FROM LOSS OF USE, DATA, PROFITS, REVENUES, OR CUSTOMERS, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF INFORMATION CONTAINED IN THIS DOCUMENT.

Smart Cards and Smart Card Readers are patent protected by Innovatron and Bull CP8 and are produced by Gemplus under license.

Printed in France.

GEMPLUS, B.P. 100, 13881 GEMENOS CEDEX, FRANCE.
Tel: +33 (0)4.42.36.50.00   Fax: +33 (0)4.42.36.50.90

Document Reference: DPD14481A00

CONTENTS

PREFACE 1

Audience 1

Conventions 1

Contact for Comments 1

OVERVIEW 2

GEMCORE PRINCIPLES 3

Modules 3

Types of Operations 4
Operation 0 (RUN) 4
Operation 1 (CTRL) 4

The GemCore Kernel 4
Real-Time Emulation 4

Command Exchange Format 5
Format 5
Type of Operation 5
Module Id 5

GEMCORE MEMORY ORGANIZATION 6

Interface Areas 6

GEMCORE APPLICATION MANAGEMENT 8

GEMCORE V1.21-BASED READER PROTOCOLS 9

The Command Layer 10

The Transport Layer 11
TLP224 11
Step 1 11
Step 2 11
The Gemplus Block Protocol (GBP) 12
Examples 13

The Physical Layer 14
Serial Asynchronous Protocol 14

GEMCORE V1.21-BASED READER COMMANDS 15

Command Format 15
GemCore V1.21-Based Reader Configuration Commands 16
Configure SIO line 17
Set Mode 18
Set Delay 19
Read Firmware Version 20
Restart 21
Restart And Run Specified Application 22
Deselect Application Procedure 23
Set Reader In Halt Mode 24

Card Interface Commands 25

Common Card Interface Commands 25
Power Down 26
Define Main Card Type And Card Presence Detection 27
Define Type And Select Auxiliary Card 29
Directory 31
Set Operating Mode 32
Specific Commands for Asynchronous Cards – Generic Operating Mode 33
Power Up - Asynchronous Cards 34
Change Card Communication Parameters – Asynchronous Cards 36
ISO Output – Asynchronous Card 38
ISO Input – Asynchronous Card 39
Exchange APDU – Asynchronous Cards 40
Response Format 41
T=1 IFSC/IFSD 42
Card Status – Asynchronous Card 43
Specific Commands for Asynchronous Cards EMV-Compliant Operating Mode 44
Power Up – EMV-Compliant 45
Exchange APDU – EMV Compliant 46
Response Format 47
T=1 IFSC/IFSD 48
Card Status – EMV Compliant 49
Specific Commands for Asynchronous Cards in Transparent Mode 50
Change Transparent Mode Parameters 51
Power Up – Transparent Mode 53
Exchange Block – Transparent Mode 54
Card Status – Transparent Mode 55
Specific Commands for Synchronous Cards 56
Power Up – Synchronous Cards 57
Read Data From Synchronous Card (ISO Out) 58
Send Data To Synchronous Card (ISO In) 59
Exchange With Synchronous Card (APDU) 60
Card Status – Synchronous Card 61

Reader Memory Management Commands 62
Read Memory 63
Write Memory 64
Erase Flash Memory 66
Select External Memory Page 68
Read CPU Port 69
Write CPU Port 70

LCD Commands 71
Init The LCD 72
Display Character String 73
Display Character 74
Send LCD Command 75

Keypad and Buzzer Commands 76
Set Key Press Timeout 77
Sound Buzzer 78

Real Time Clock Commands 79
Read Date And Time 80
Update Date and Time 81

GemPC410 Control Commands 82
GemPC410 Set Timeout 83
GemPC410 Refresh 84
GemPC410 Power Down 85

GemPC410 LED Management 86
GemPC410 Status 87

USING THE GEMCORE V1.1-BASED READER WITH MICROPROCESSOR CARDS.. 88

Clock Signal...88

Global Interface Parameters...88
TA1..88
TB1 and TB2 88
TC1 89

Communication Protocols..90
T=0 Protocol..90
T=1 Protocol..90

USING THE GEMCORE-BASED READER WITH MEMORY CARDS.. 91

APPENDIX A. STATUS CODES 94

APPENDIX B. INTERPRETED SYNCHRONOUS SMART CARD DRIVER... 96

Card Type 01h..96

8051 Interpreter...96

Initialization 97

Card Presence...97

Card Withdrawal 97

Short Circuit...97
Instructions 98

Modified Instructions..100
RET...100
RETI..100

Macro-Commands..100
Example 105

TERMINOLOGY...106

Abbreviations...106

Glossary 107

INDEX 108

List of Figures

Figure 1. The GemCore Kernel and the Modules..3
Figure 2. The GemCore Kernel...4
Figure 3. GemCore V1.21-Based Reader Memory Mapping. 7
Figure 4. User Application AIA2 8
Figure 5. Three-Layer GemCore V1.21-Based Reader Protocol.........................9

List of Tables

Table 1. TA1 Values Handled in Standard Mode
(Frequency: 3.6864 MHz)...89
Table 2. TA1 Values Handled in Double-Speed Mode
(Frequency: 7.3728 Mhz)...89
Table 3. Summary of the Memory Card Commands..93

PREFACE

This document provides information about the GemCore V1.21-Based Reader software. Refer to the *GemCore Preliminary Technical Data Sheet* for a detailed description of the GemCore V1.21-Based Reader hardware.

Audience

This document is intended for anyone wishing to develop electronic systems using a smart card interface.

Conventions

By default, a numeric is expressed in decimal.

A hexadecimal number is followed by the h character. For example, the decimal value 13 expressed in hexadecimal becomes **0Dh**.

A byte B consists of eight bits $b_7b_6b_5b_4b_3b_2b_1b_0$: b_7 is the most significant (the highest) bit and b_0 the least significant (the lowest) bit:

One byte	b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0

A string of bytes consists of n concatenated bytes $B_0B_1….B_{n-1}$: B_0 is the most significant (the highest) byte and B_{n-1} the least significant (the lowest) byte:

A string of n bytes	B_0	B_1	B_2	B_3	B_4	…	B_{n-1}

S stands for “status” in command results.

Contact for Comments

We welcome your feedback to help us provide high quality documentation. Contact us at:

E-mail TOLDgem@gemplus.com

Postal Technical & Online Documentation (TOLD),
Gemplus,
B.P. 100,
13881 Gémenos Cedex,
FRANCE

Please remember to quote the document reference, your job function and your company.

OVERVIEW

This document describes the interface between the application and the GemCore V1.21-Based Reader.

The GemCore V1.21-Based Reader, which consists of one programmed controller and up to nine Gemplus IC100 smart card interface chips, is designed to simplify the integration of smart card interfaces in electronic devices and it manages communication with ISO 7816 1-2-3-4 compatible smart cards.

The software inside the reader is compatible with the Gemplus Open Reader Operating System (OROS). It implements communication protocols for the host system (Gemplus Block Protocol (GBP) or Transport Layer Protocol (TLP)) as well as protocols for synchronous and asynchronous smart cards.

Depending on the reader, the software may also manage hardware interfaces with, for instance, a display, a keypad or an external memory. The connection with the host system takes place via a serial asynchronous port at the Transistor-Transistor Logic (TTL) level.

The GemCore V1.21-Based Reader is certified HQL (Plug & Play and PC/SC) and allows for EMV-compliant developments. The reader can be used in two different modes:

- Generic operating mode
- EMV-compliant mode

The **Set Operating Mode** command is used to switch from one mode to another.

Among asynchronous commands, certain are used in generic operating mode, while others are used in the EMV-compliant mode.

***Note**:* *Before an EMV-compliant asynchronous command can be used, the operating mode must have been set accordingly using the Set Operating Mode command.*

GEMCORE PRINCIPLES

Modules

GemCore is built around a real-time kernel that is surrounded with modules. The main function of the GemCore kernel is to manage the modules. These modules can be broken down into two categories:

- *Device handlers* are specific to GemCore. They are embedded.
  Device handlers are described in the chapter *"GemCore-V1.21-Based Reader Commands"*. Each command set (such as the configuration command set or the card interface command set, for example) corresponds to a specific system device handler and each individual command corresponds to an elementary operation.
- *Application modules* are specific to the customer's application.

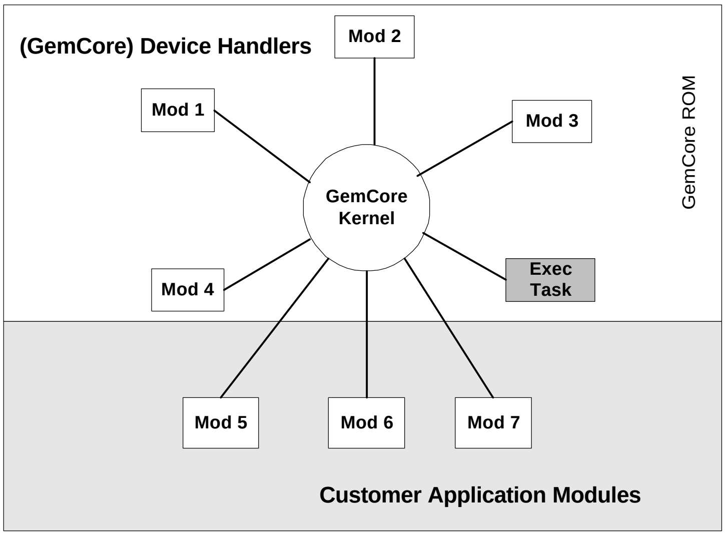


Figure 1. The GemCore Kernel and the Modules

Note: *The Exec Task module is an exception. Indeed, although it is an application module, it is GemCore-specific. It is used for communication between GemCore and the client application.*

Types of Operations

All modules have a common interface called the operation list, which defines the nature of each operation authorized for a particular module. Each module can handle up to eight *operations* (0 to 7), and each operation performs a specific function such as reading data from a card or displaying data on an LCD.

Operations 0 and 1 are described here because their meaning is invariable.

Operation 0 (RUN)

Operation 0 is also known as RUN. GemCore searches for this command at every kernel loop. It is declared void if the module is a device handler. In other words, if it is declared, it means the module is an application module.

Operation 0 does not use any parameters and it cannot be called by the host.

Exec Task is the only GemCore module with which the RUN operation is valid.

Operation 1 (CTRL)

Operation 1 is also known as CTRL.

If the module is a device handler, CTRL is the first operation to be performed by the kernel, provided this operation exists. If it does exist, this operation is used to initialize the device handler.

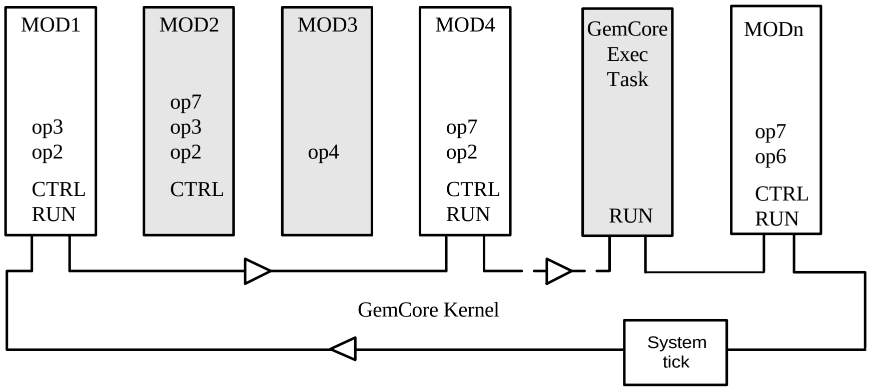
If it is an application module, this operation is not called by the GemCore kernel, but it can be called externally. In this case, Operation 1 can be used to kill the application module.

The GemCore Kernel

The GemCore kernel acts as an endless loop that is used to manage the modules.

When the system starts up, GemCore initializes the CTRL operation for each module (when applicable).

After initializing the CTRL operations, GemCore activates the RUN operation(s) for each application module as shown in the following diagram:



Mod(ule) 1 is an application module (the RUN operation is defined).

Mod(ule) 2 is a device handler (the RUN operation is not defined). It is initialized by the kernel during boot.

Mod(ule) 3 is a device handler which is not initialized during boot (CTRL operation does not exist).

Figure 2. The GemCore Kernel

The GemCore kernel defines whether an operation is run or suspended until the next loop.

Real-Time Emulation

At each loop, GemCore pauses for a period defined in the SYS_TIC data byte. The default value is 20 ms. (The system tick is illustrated in Figure 2).

Command Exchange Format

Modules exchange information using a request/response mechanism.

The requests contain a module number, an operation number, and generally a list of one or more parameters.

Requests use the following format:

Format

<MODULE ID + OPERATION NUMBER> [PARAM1] [PARAM2]…

The responses return a status code and the result(s) of the operation specified in the request. Responses use the following format:

<STATUS CODE> [RESULT1] [RESULT2]…

The module sending the request must wait for the response.

The Module ID and the operation number are coded on one byte.

Type of Operation

The three least significant bits define the type of operation to be performed (see the example in section "*Module Id*").

Module Id

The five most significant bits are used with the last three bits temporarily forced to 0 to determine the module number.

Example:

The **Display Character String** command is found using command 2Bh, which converts to the following in binary:

b7	b6	b5	b4	b3	b2	b1	b0
0	0	1	0	1	0	1	1

The first step consists of identifying the type of operation:

Bits 0 through 2 indicate that the operation type is 3.

The second step consists of identifying which module is concerned by this operation. To do so, the last three bits are forced to zero.

b7	b6	b5	b4	b3	b2	b1	b0
0	0	1	0	1	0	0	0

This indicates that module 28 (LCD) is concerned by this operation. Operation 3 for module LCD corresponds to the **Display Character String** command.

GEMCORE MEMORY ORGANIZATION

GemCore manages different types of memory areas, namely the IDATA, the XDATA and the program memory. These memory areas are either used by the operating system (IDATA and program memory) or dedicated to customer applications (XDATA).

The User DATA #0, User DATA #1, User DATA #2, User Application Interface Area AIA2 memory areas are reserved for customer development when available on the GemCore-V1.21-Based Reader.

A memory page mechanism allows the use of up to 512K for the User DATA #0 area and 512K for the User Application AIA2 area.

Interface Areas

GemCore uses three Interface Areas to store the modules' parameters (list, interruption vectors, etc.).

The System Interface Area (SIA) is reserved for the kernel and the system device handlers.

The first Application Interface Area (AIA1) is reserved for Gemplus extensions.

The second Application Interface Area (AIA2) contains the user modules' parameters. They are 64 bytes (40h) long.

The system scans the application interface areas in the following order:

1. AIA2, which starts at location 8000h (SYS_AIA2).

2. AIA1, which starts at address 4000h (SYS_AIA1).

3. SIA (OS interface area), which starts at location 0000h.

This scanning order enables the user to develop modules with the same number as a system device handler in order to override certain operations.

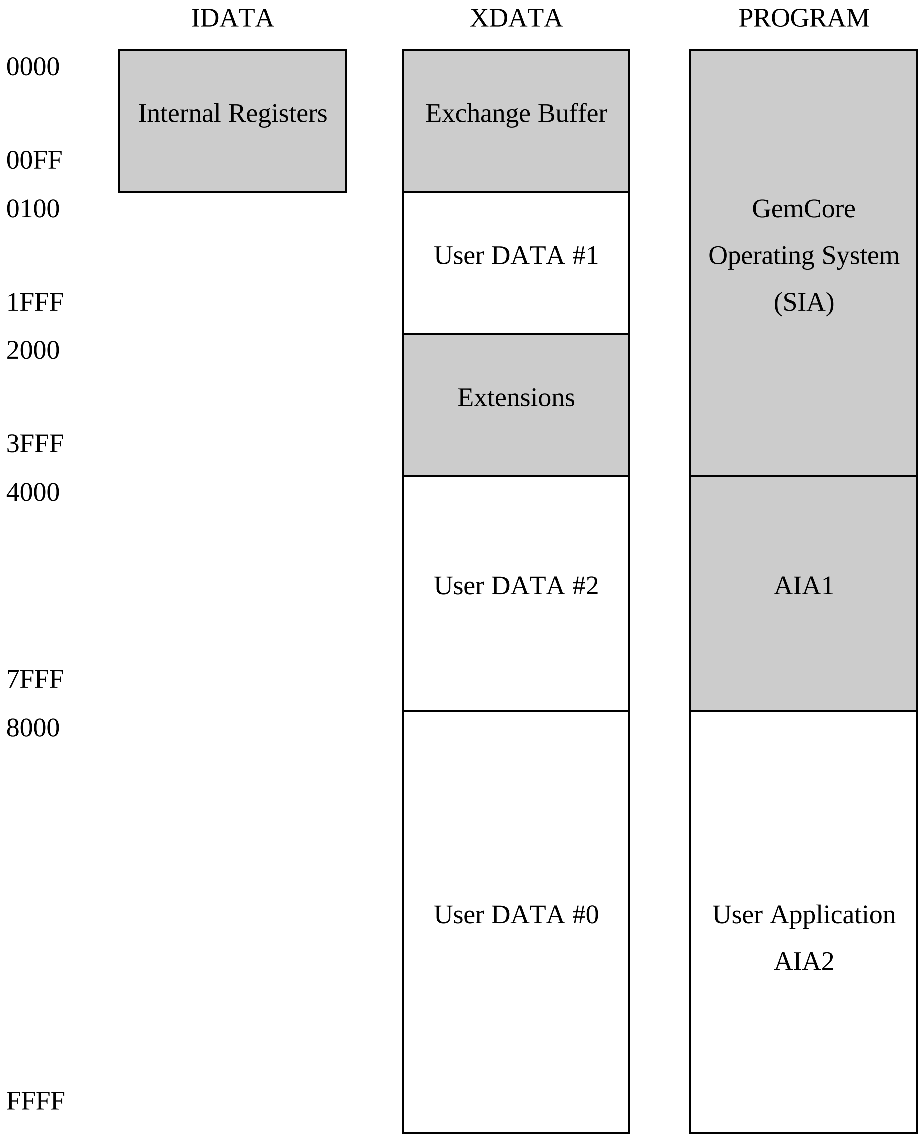


Figure 3. GemCore V1.21-Based Reader Memory Mapping.

User Application Interface Area 2 appears as a 32K window. However, due to GemCore’s page memory system, it actually contains 512K of application code and data (see Figure 4).

Warning*:* *The Extensions area is reserved for GemCore operations. Users should not map these locations with memory.*

GEMCORE APPLICATION MANAGEMENT

GemCore can manage up to four customer applications. These applications are stored in User Application Area AIA2 and can use external memory located in the User DATA #0, User DATA #1 and the User DATA #2 areas.

As a general rule, User DATA #1 and User DATA #2 areas are used as working RAM or to store intermediate data.

Note: *Depending on the reader used, the User DATA #1 and User DATA #2 areas may not be available.*

512 Kb of memory are available for the User Application AIA2 area and 512 Kb are available for the User DATA #0 area.

The size of the applications is predefined:

Application 1:	Program 64 Kb, DATA 64 Kb
Application 2:	Program 64 Kb, DATA 64 Kb
Application 3:	Program 128 Kb, DATA 128 Kb
Application 4:	Program 256 Kb, DATA 256 Kb

When the system is powered up, GemCore looks for the applications defined and automatically runs the first application it finds.

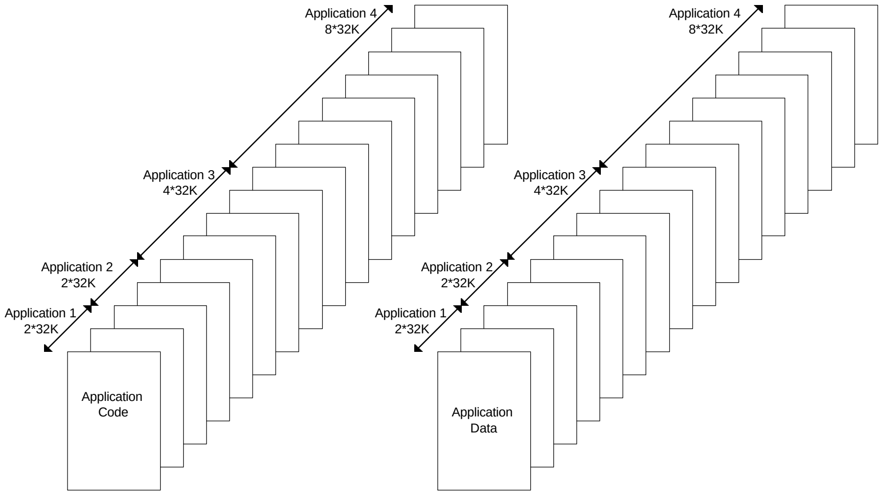


Figure 4. User Application AIA2

When the system is powered up, GemCore looks for the defined applications and automatically runs the first application it finds.

GEMCORE V1.21-BASED READER PROTOCOLS

All transmissions with the GemCore V1.21-Based Reader are handled by three protocol layers:

- The command layer
- The transport layer
- The physical layer

The command layer handles and interprets the GemCore V1.21-Based Reader commands. The commands are made up of a command code, data, and parameters.

The transport layer handles the message addressing, specifies the transmission type, and validates each transmission. The transport layer can use one of two protocols: the TLP224 protocol or the GEMPLUS Block Protocol.

The physical layer handles the data transmission itself.

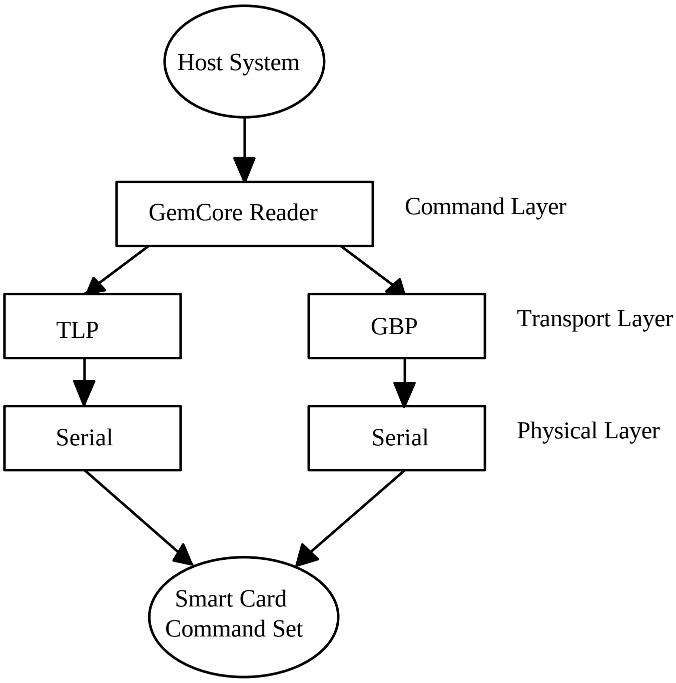


Figure 5. Three-Layer GemCore V1.21-Based Reader Protocol.

The following paragraphs describe the protocol layers in more detail.

The Command Layer

The command layer handles and interprets the GemCore V1.21-Based Reader commands. The commands are made up of a command code, data, and parameters.

Commands are sent in the following format:

```
|CommCode|Parameters|Data|
```

Where:

`CommCode`	Is the command code.
`Parameters`	Are the parameters sent with the command.
`Data`	Is the data accompanying the command, where appropriate.

The *"GemCore V1.21-Based Reader Commands"* section describes the `CommCode`, `Parameters`, and `Data` field values for each command.

The GemCore V1.21-Based Reader replies to every command it receives with a status code formatted as follows:

```
|S|Data|
```

Where:

`S`	Is the status code identifier.
`Data`	Is the data returned with the status code, where appropriate.

The Transport Layer

The transport layer handles message addressing, specifies the transmission type, and validates each transmission. The GBR (GemCore V1.21-Based Reader) transport layer can use one of two protocols: the TLP224 protocol or the Gemplus Block Protocol (GBP). The following paragraphs describe these protocols.

TLP224

TLP protocol processing consists of two steps.

Step 1

The first step is to construct the message to be transmitted. Under the TLP224 protocol, the messages have the following format:

For messages transmitted without errors:

`<ACK><LN><MESSAGE><LRC>`

Where:

`ACK`	60h, indicating that the previous command or status code was transmitted without errors.
`LN`	Is the length of the message (command or status code).
`MESSAGE`	Is the command or status code.
`LRC`	Is the result of an EXCLUSIVE OR (XOR) between the `ACK`, `LN`, and `MESSAGE` characters.

For messages transmitted with errors:

`<NACK><LN><LRC>`

Where:

`NACK`	E0h, indicating that there was an error in the message transmission.
`LN`	00h
`LRC`	E0h

Step 2

During the second step, the source performs the following processes:

- Conversion of each byte to be transmitted into two ASCII characters. For example, to transmit byte 3Ah, the source will transmit the values 33h and 41h. This prevents other equipment from interpreting the control characters.
- Add an End Of Transmission (EOT) byte at the end of the transmission. This byte is assigned the value 03h.

For example, to transmit under the TLP224 protocol the **Power Down** command which uses the command code 4Dh and no parameter, the following sequence would be sent:

	ACK	LN	Message	LRC	EOT
Command	60	01	4D	2C	
TLP Protocol Transmission	36 30	30 31	34 44	32 43	03

The timeout between each character is 100 ms.

The Gemplus Block Protocol (GBP)

The Gemplus Block Protocol (GBP) is a simplified version of the T=1 card protocol. Under the GBP, data is transmitted in blocks between the source and the destination. There are three types of blocks:

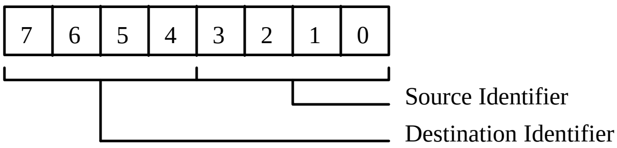
- I-Blocks (Information Blocks). I-Blocks hold the data to be exchanged between the source and the destination.
- R-Blocks (Receive Ready Block). R-Blocks hold positive or negative acknowledgments to transmissions.
- S-Blocks (Supervisory Block). S-Blocks synchronize transmissions between the source and the destination.

The data is exchanged in the following format :

NAD	PCB	LEN	DAT	EDC

Where:

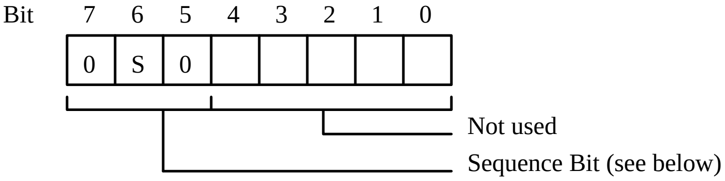
- NAD is the source and the destination identifier formatted on one byte as follows:



The GemCore V1.21-Based Reader identifier is 4 and the host system identifier is 2.

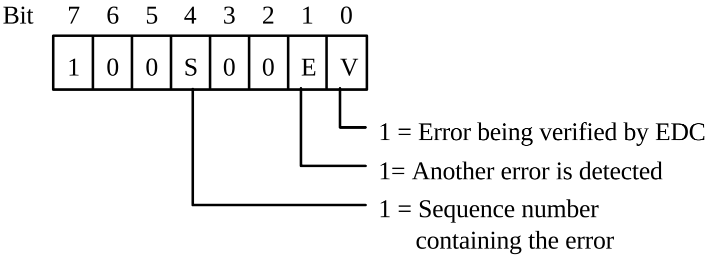
- PCB indicates the block type, as described below:

I-Block PCBs take the following format:



The sequence bit is set to 0 at power up. The source sends the first I-Block that it transmits with the sequence bit set to 0. It increments the sequence bit by one every time it sends an information block. The GemCore V1.21-Based Reader and the host system generate sequence bit values independently.

R-Block PCBs take the following format:


Bit
7
6
5
4
3
2
1
0
1
0
0
S
0
0
E
V
1 = Error being verified by EDC
1= Another error is detected
1 = Sequence number containing the error

S-Blocks request the destination to set the sequence bits to 0 and return a response to the source indicating that the transmission is completed.

S-Block PCBs use the following format:

Bit	7	6	5	4	3	2	1	0	
	1	1	0	0	0	0	0	0	Resynch request

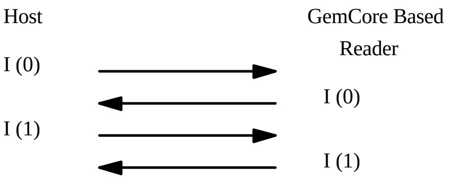
Bit	7	6	5	4	3	2	1	0	
	1	1	1	0	0	0	0	0	Resynch response

- LEN specifies on one byte the number of bytes in the DAT field.
- DAT holds the data being transmitted.
- EDC is the result of an exclusive OR performed on the NAD, PCB, LEN, and DAT bytes.

Examples

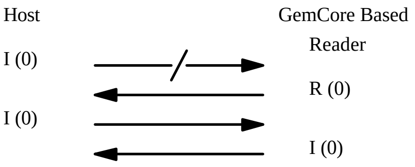
The following examples illustrate different types of transmissions under the GBP protocol.

Transmission without errors:

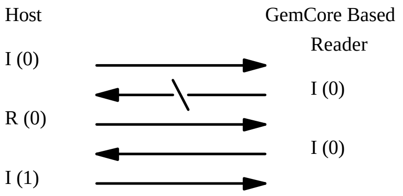


Transmission with errors:

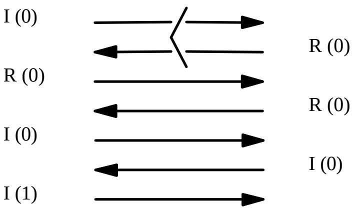
Case 1.



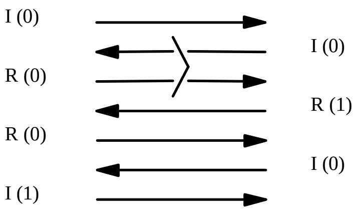
Case 2.



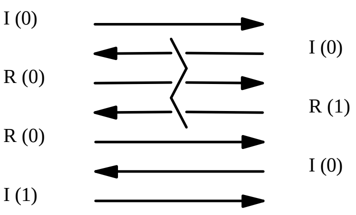
Case 3.



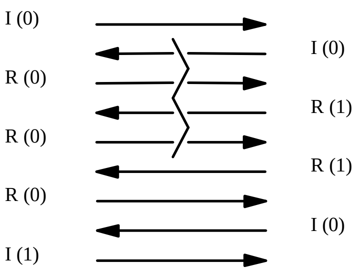
Case 4.



Case 5.



Case 6.



The Physical Layer

The physical layer handles the data transmission itself. The physical layer uses the Serial protocol.

Serial Asynchronous Protocol

The Serial Asynchronous Protocol can be sent directly over the serial line.

The bytes are sent over the line by an UART whose transmission characteristics (such as speed and parity) are determined by the configuration of the GemCore V1.21-Based Reader.

The default configuration is 9,600 baud, eight bits, no parity and one stop bit.

GEMCORE V1.21-BASED READER COMMANDS

This section describes the GemCore V1.21-Based Reader commands. For each command it indicates:

- The function it performs
- The syntax
- The data it returns

Command Format

Commands are sent to the reader in the following format:

```
|CommCode|Parameters|Data|
```

Where:

`CommCode`	Is the command code.
`Parameters`	Are the parameters sent with the command.
`Data`	Is the data accompanying the command, where appropriate.

The reader replies to every command it receives with a response formatted as follows:

```
|S|Data|
```

Where:

`S`	Is the status code identifier.
`Data`	Is the data returned with the status code, where appropriate.

"Appendix A – Status Codes" lists the status codes and their meanings.

GemCore V1.21-Based Reader Configuration Commands

The GemCore V1.21-Based Reader configuration commands are used to define the reader settings.

GemCore V1.21-Based Reader configuration commands are:

- **Configure SIO Line**
- **Set Mode**
- **Set Delay**
- **Read Firmware Version**
- **Restart**
- **Restart And Run Specified Application**
- **Deselect Application**
- **Set Reader In Halt Mode**

Each command is described in the following pages.

See "*Appendix A – Status Codes*" for a description of status codes.

Configure SIO line

This command sets the SIO line parity, baud rate, and number of bits per character. After a power up, the line defaults to no parity, eight bits per character, 9,600 baud and one stop bit.

Note: *The line is reconfigured as soon as this command is executed. The response is returned with the new parameters.*

Format

0Ah CB

Where:

CB	Is the configuration byte. The configuration flag settings are defined in the following table:

Bit	Value	Option Selected
7 to 5		Not used
4	0	No parity
	1	Even parity
3	0	Eight bits per character
	1	Seven bits per character
2 to 0	xxx	Sets the baud with the following values: 000 RFU 001 76,800 010 38,400 011 19,200 100 9,600 101 4,800 110 2,400 111 1,200

Response

S (Status)

Set Mode

This command enables the user to disable ROS command compatibility and define the reader operation mode (TLP or Normal). The reader defaults to ROS command compatibility enabled and TLP mode.

***Notes**:* *Disabling ROS command compatibility disables this command. ROS command compatibility can only be enabled once again by performing a hardware reset on the reader so that the default configuration is restored.*

Disabling ROS command compatibility also disables TLP mode, regardless of the value of bit 3 (see below).

Format

01h 00h [OB]

Where:

[OB] Is the option selection byte. The flag settings are described in the following table:

	Native	ROS	TLP
xxxx1xx1	✓	✓	✓
xxxx0xx1	✓	✓	
xxxx0xx0 or xxxx1xx0	✓		

Note: *If this byte is not sent, the reader operation mode stays unchanged, but the result is still returned.*

Response

S <mode>

Where:

<mode> Is the mode the reader is operating in. The mode is returned on one byte that indicates the operating mode as shown the following table:

	Native	ROS	TLP
00001001	✓	✓	✓
00000001	✓	✓	
00000000	✓		

***Note**:* *In TLP mode, the GemCore V1.21-Based Reader adds the TA1, TB1, TC1, TD1 bytes if they are not present in the asynchronous card Answer To Reset.*

Set Delay

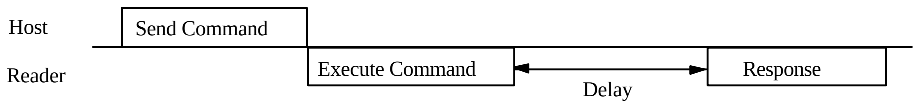
If a slow host computer is used with the GemCore V1.21-Based Reader, this command can be used to delay responses.

Format

23h 01h 00h 67h 01h Delay

Where:

Delay — Is the response delay in ms. Enter a value between 0 and 255. When the system is powered up, the delay time defaults to 0.



Response

None.

Read Firmware Version

Returns the version of the firmware installed in the reader.

Format **22h 05h 3Fh E0h 10h**

Response **S** Version

Where:

Version (GemCore1.21-xy)	Is the installed software version in ASCII, where x is the custom indicator, for example, G for Gemplus, and where y can be any character.

OROS-COMPATIBLE COMMAND:

Format **22h 05h 3Fh F0h 10h**

Response S <OROS-R2.99-R1.00>

Restart

This command is used to reset the GemCore operating system.

All the parameters are initialized with the default values.

Format **0Ch 00h 00h 00h**

Response None.

Restart And Run Specified Application

This command is used to run the specified application.

If the application does not exist, an error code is returned.

Format

0Ch 00h 00h APP

Where:

APP	Is the number of the application to be run (1, 2, 3, or 4).

Response

S (03h if the specified application does not exist).

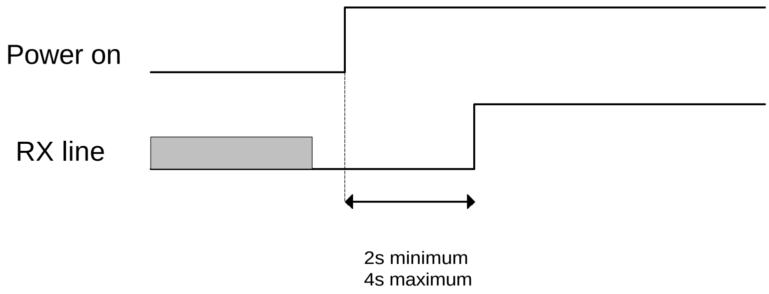
Deselect Application Procedure

This procedure instructs GemCore to run with no active application. It can be used before downloading a new customer application for a stand-alone device.

This procedure is used when disabling applications using the GemCore Tool Kit Loader.

The GemCore Reader's RX line must be set to 0 during a minimum of two seconds and a maximum of four seconds, while GemCore is powered up.

The RX line should be set to 0 before the reader is powered on.



Applications are disabled until a **Restart** command is performed or until the next power-up sequence takes place.

This procedure does not affect external memories such as the customer’s application memories.

Set Reader In Halt Mode

This command is used to switch the reader to halt mode.

When switching to this mode, all the card interfaces are powered down, the reader is in low power consumption mode, and all commands are ignored.

The only way to move out of the halt mode is to reset the GemCore micro-controller.

Format **0Eh 00h 00h 00h**

Response **00h**

Note: *The reader goes into halt mode at most 2 milliseconds after sending the result.*

Card Interface Commands

The card interface commands manage the communication with smart cards.

The GemCore operating system can simultaneously manage up to nine smart card interfaces. In order to limit the command set, the smart card interfaces are organized in one main smart card interface and eight auxiliary smart card interfaces.

The auxiliary smart card interface required should be selected before use.

The behavior of certain commands changes depending on the selected card type. Therefore, some commands are common to all types and others are redefined or disabled according to the card type.

Four groups of commands are defined:

- Common card interface commands,
- Specific commands for asynchronous cards, generic operating mode
- Specific commands for asynchronous cards, EMV-compliant operating mode
- Specific commands for asynchronous cards in transparent mode,
- Specific commands for synchronous cards.

Common Card Interface Commands

These commands are valid regardless of the selected type.

Common card interface commands are:

- **Power Down**
- **Define Main Card Type**
- **Define Type And Select Auxiliary Card**
- **Directory**
- **Set Operating Mode**

Each command is described in the following pages. When a comparable ROS command existed previously, its format is described along with the GemCore format to help with the transition from one operating system to the other.

See “*Appendix A. Status Codes*” for a description of status codes.

Power Down

This command is used to power down the card. The GemCore V1.21-Based Reader is powered down automatically when the card is removed.

GBR Format

11h Main Card
19h Selected Auxiliary Card

ROS Format

4Dh 00h 00h 00h Main Card only

Response

S

The **Power Down** command always ends normally if a card is present in the reader.

If no card is inserted, the command returns the FBh "card missing" error.

Define Main Card Type And Card Presence Detection

The GemCore V1.21-Based Reader does not have a smart card recognition algorithm. It therefore is necessary to define the type of card used. This command sets the card type.

Notes: *ROS and GBR versions of this command are different. The two formats are described below.*

When the GemCore V1.21-Based Reader is reset or powered up, the card type defaults to standard microprocessor card mode (type 2).

GBR Format

17h T [00h [P]]

ROS Format

02h T P

Where:

T Is the card type selection byte. The card type codes are as follows:

Enter This Code	To Use This Card
01h	Other synchronous smart cards; interpreted driver.
02h	Standard speed mode (clock frequency = 3.6864 MHz) ISO 7816-3 T=0 and T=1 microprocessor cards.
12h	Double speed mode (clock frequency = 7.3728 MHz) ISO 7816-3 T=0 and T=1 microprocessor cards.
03h	GPM256 (read only).
04h	GPM416/GPM896 in Standard Mode.
06h	GFM2K/GFM4K.
07h	GPM103.
08h	GPM8K(SLE4418/4428).
09h	GPM2K(SLE4432/4442 or PCB2032/2042).
0Dh	GPM276/GAM275.
0Eh	GPM271/GAM273.
0Fh	GAM226.
0EFh	Microprocessor cards used in transparent mode at standard speed (clock frequency = 3.6864 MHz). Protocol management is not handled by the reader.
0FFh	Microprocessor cards used in transparent mode at standard speed (clock frequency = 7.3728 MHz). Protocol management is not handled by the reader.

If the command sent with a family number which does not match the current main card, the current main card is powered down unless the following changes occur:

From	To
02	EF
EF	02
12	FF
FF	12

P Is the card presence byte. This optional parameter is used to modify the card presence indication options. When this parameter is not specified card presence is not indicated.

Enter this code	To indicate card presence:
00h	On P1.6 (SCL line), card present = 1
01h	On P1.6 (SCL line), card present = 0
02h	On P3.1 (TxD line), card present = 1
03h	On P3.1 (TxD line), card present = 0
04h or 06h	On P1.7 (SDA line), card present = 1
05h or 07h	On P1.7 (SDA line), card present = 0
08h	On P1.6 (SCL line), positive pulse (5ms) upon card insertion/withdrawal
09h	On P1.6 (SCL line), negative pulse (5ms) upon card insertion/withdrawal
0Ah	On P3.1 (TxD line), positive pulse (5ms) upon card insertion/withdrawal
0Bh	On P3.1 (TxD line), negative pulse (5ms) upon card insertion/withdrawal
0Ch or 0Eh	On P1.7 (SDA line), positive pulse (5ms) upon card insertion/withdrawal
0Dh or 0Fh	On P1.7 (SDA line), negative pulse (5ms) upon card insertion/withdrawal

Response **S**

Define Type And Select Auxiliary Card

The GemCore V1.21-Based Reader does not have a smart card recognition algorithm. It therefore is necessary to define the type of card currently used. This command sets the auxiliary card type and selects the auxiliary card number.

***Note**: When the GemCore V1.21-Based Reader is reset or powered up, the auxiliary card type defaults to standard microprocessor card mode (type 2) and auxiliary card number 1 is selected.*

The commands sent to the auxiliary card will be executed by the auxiliary card currently selected.

Switching from one auxiliary card to another one does not affect the status of the unselected auxiliary cards.

GBR Format

1Fh T N

Where:

T Is the card type selection byte. The card type codes are as follows:

Enter This Code	To Use This Card
01h	Other synchronous smart cards; interpreted driver.
02h	Standard speed mode (clock frequency = 3.6864 MHz) ISO 7816-3 T=0 and T=1 microprocessor cards.
12h	Double speed mode (clock frequency = 7.3728 MHz) ISO 7816-3 T=0 and T=1 microprocessor cards.
03h	GPM256 (read only).
04h	GPM416/GPM896 in Standard Mode.
06h	GFM2K/GFM4K.
07h	GPM103.
08h	GPM8K(SLE4418/4428).
09h	GPM2K(SLE4432/4442 or PCB2032/2042).
0Dh	GPM276/GAM275.
0Eh	GPM271/GAM273.
0Fh	GAM226.
0EFh	Microprocessor cards used in transparent mode at standard speed (clock frequency = 3.6864 MHz). Protocol management is not handled by the reader.
0FFh	Microprocessor cards used in transparent mode at standard speed (clock frequency = 7.3728 MHz). Protocol management is not handled by the reader.

If the command is sent with a family number that does not match the current auxiliary card, the current auxiliary card is powered down, unless the following changes occur:

From	To
02	EF
EF	02
12	FF
FF	12

N Is the auxiliary Card Number (1 to 8).

Response

S

***Notes**:* *When the type changes to asynchronous card in transparent mode, the transparent mode parameters are reset to default settings for this mode (see **Change Transparent Mode Parameters** command).*

When the following change occurs:

From	*To*
EF	*02*
FF	*12*

*the communication parameters for the asynchronous card must be reset using the **Change Card Communication Parameters** command.*

Directory

This command is used to obtain the types of cards handled, the release number and the characteristics for each card driver.

GBR Format **17h 00h**

Response **S** <TYPE, CMD, REV> ... <TYPE, CMD, REV>

Where:

Type	**Card Type (for example: 02h Asynchronous Card)**
CMD	00: ISO IN/OUT
	01: APDU
	02: ISO IN/OUT and APDU
REV	Card driver release (2 bytes)

Set Operating Mode

This command selects the operating mode of treatment of an asynchronous card.

Two modes exist:

- Generic mode (the default mode selected at GBR reset)
- EMV-compliant mode

Some commands are not allowed in EMV mode, while others show changes in their behavior.

The operating mode applies to main card and to all auxiliary cards.

GBR Format

17h 00h [Mode]

Where:

Mode	Is the operating mode to be selected 47h selects the generic mode 45h selects the EMV-compliant mode 00h returns the mode currently selected

Response

S Mode

Where:

Mode	Is the mode currently selected 47h = generic mode 45h = EMV-compliant mode

Specific Commands for Asynchronous Cards – Generic Operating Mode

These commands, which are used with a card selected as asynchronous (type = 0x02 or 0x12), have a specific behavior.

Commands that are valid with these types are:

- **Power Up – Asynchronous Cards**
- **Change Card Communication Parameters – Asynchronous Cards**
- **ISO Output – Asynchronous Cards**
- **ISO Input – Asynchronous Cards**
- **Exchange APDU – Asynchronous Cards**
- **Card Status – Asynchronous Cards**

See “*Appendix A. Status Codes*” for a description of status codes.

Power Up - Asynchronous Cards

This command powers up and resets a card.

GBR Format

12h [CFG][PPS0,PPS1,PPS2,PPS3][PCK] Main Card

1Ah [CFG][PPS0,PPS1,PPS2,PPS3][PCK] Selected Auxiliary Card

ROS Format

6Eh 00h 00h 00h Main Card only

If the CFG parameter is not specified, the card is powered with 5V, there is no PTS management and the operating mode is compatible with OROS2.2X

If the CFG parameter is specified:

X0XXXX01	Class A: Vcc for Card is 5V
X0XXXX10	Class B: Vcc for Card is 3V
X0XXXX11	Class AB: Vcc for Card is 5V or 3V
0000XXXX	Operation is compatible with OROS2.2X
0001XXXX	Reset and no PPS management. The reader stays at 9,600 baud if the card is in negotiable mode.
0010XXXX	Reset and automatic PPS management. The reader uses the highest speed proposed by the card. Change to T=1 protocol if there is a choice between T=0 and T=1.
1111XXXX	Manual PPS management. This command does not reset the card. It must be preceded by a **Power Up** command with the CFG parameter set to 0001XXXX. The parameters from PPS0 to PCK are sent to the card at 9,600 baud. If PCK is omitted, it is computed and added by the GBR. If the card responds with PPS RESPONSE, the reader is configured using the parameters returned.
00001000	Valid only if T=1 is the current protocol, otherwise no action occurs. An S-IFS block exchange is initiated by the GBR. The IFSD (maximum length of INF field accepted by the GBR) sent to the card is the value of parameter PPS0. No other parameters are allowed.
X0XX1XXX or 11111XXX	If the selected protocol after the ATR or the PPS exchange is T=1, the GBR initiates an S-IFS block exchange. The IFSD value indicated to the card is 0FEh. After a command reset with no PPS and with IFSD exchange, a command of manual PPS management is invalid.

Response

S <card response>

Where:

<card response>	Is the card Answer To Reset (ATR).

With the ROS command, if TLP compatibility is enabled, the ATR is preceded by three bytes R1, R2, R3.

R1: compatibility mode: 28h for TLP and 01h for ROS

R2: current card type

R3: ATR length

Note: *When TLP compatibility is enabled (see "Set mode" command) the TA1, TB1, TC1 and TD1 bytes missing in the ATR are returned with their default values:*

TA1	TB1	TC1	TD1
11h	25h	00h	00h

Note: *When TLP compatibility is enabled, the missing bytes are returned but **T0 is not modified**. The syntax of the ATR is therefor not valid.*

Example:

```
3B  A0  00  81  71  27  42  00  35
```

becomes:

```
3B  A0  11  00  00  81  71  27  42  00  35
```

TCK and T0 are not valid.

Change Card Communication Parameters – Asynchronous Cards

This command dynamically changes the parameters of the communication with the card. This command is mainly used to switch the speed or the protocol when the card uses a proprietary mode to switch these parameters.

GBR Format

12h `PRT CNF1 CNF2 CNF3 CNF4` Main Card

1Ah PRT CNF1 CNF2 CNF3 CNF4 Selected Auxiliary Card

Where:

PRT Selects the protocol. The format of this byte is:

0	1	0	C	P	E	S	R

C= 0; bits P E S R are significant.

1; bits P E S R are not taken into account.

P selects the protocol to be used.

When P = 0, the protocol is T=0.

When P = 1, the protocol is T=1.

E selects the computing mode for EDC. It is significant only if T=1 is selected.

When E = 0, EDC is LRC.

When E = 1, EDC is CRC.

S initializes the sequence number of the last I-block sent.

When S = 0, the next I-bock will be sent with sequence number 1.

When S = 1, the next I-block will be sent with sequence number 0.

R is the sequence number for the next I-block to be received.

When R = 0, the next I block is expected to have sequence number 0.

When R = 1 the next I block is expected to have sequence number 1.

CNF1 Selects the new TA1 (FI/DI=speed) to be used.

CNF2 Selects the new TC1 (N=extra guard time) to be used.

CNF3 If protocol T=0 is selected, this indicates the new TC2 (WI= waiting time) to be used.
If protocol T=1 is selected, this indicates the new TA3 (IFSC= maximum length of information field of blocks which can be received by the card) to be used.

CNF4 If protocol T=0 is selected, reserved for future use.
If protocol T=1 is selected, indicates the new TB3 (BWI/CWI= block and character waiting time) to be used.

Response S

***Note**: No check is performed on parameters `PRT`, `CNF1`, `CNF2`, `CNF3` and `CNF4`.*

ISO Output – Asynchronous Card

This command sends ISO OUT commands, that is, commands which retrieve data from the asynchronous card.

This command can return up to 252 data bytes in one operation. Two operations are required to obtain 256 bytes.

GBR Format

13h CLA INS A1 A2 LN Main Card

1Bh CLA INS A1 A2 LN Selected Auxiliary Card

ROS Format

DBh CLA INS A1 A2 LN Main Card only

Where:

CLA, INS, A1, A2, and LN	Are the five ISO header bytes. For more details about ISO header contents, refer to the documentation concerning the card being used. The ISO header is transmitted directly to asynchronous cards.

Response

S <data> SW1 SW2

Where:

<data>	Is the data returned by the card. If a smart card error or GemCore V1.21-Based Reader error is detected (S<>0 and S<>E7h), the GBR does not return any valid data. The card may return any number of bytes up to LN.

If the number of data bytes to be returned is greater than 252, the first 252 bytes are contained in the <data> field. In order to obtain the rest of the response, the following command must be sent:

GBR Format

13h FFh FFh FFh FFh FFh Main card
1Bh FFh FFh FFh FFh FFh Selected auxiliary card

Response

S <data> SW1 SW2

Where:

<data>	Is the rest of the response (LN-252 bytes).

Note*:* *The GBR returns error code 1Bh if a card interface command other than the above is sent.*

ISO Input – Asynchronous Card

This command sends ISO IN commands, that is, commands which send data to an asynchronous card.

This command can send up to 248 data bytes in one operation. Two operations are required to send 255 data bytes.

GBR Format

14h CLA INS A1 A2 LN <data> Main Card
1Ch CLA INS A1 A2 LN <data> Selected Auxiliary Card

ROS Format

DAh CLA INS A1 A2 LN <data> Main Card Only

Where:

CLA, INS, A1, A2, and LN	Are the five ISO header bytes. For more details about the ISO header contents, refer to the documentation concerning the card being used. The ISO header is transmitted directly to microprocessor cards (asynchronous cards).
<data>	Represents the LN data bytes transmitted to the card after the ISO header. The maximum length of the data is 248 bytes.

Response

S SW1 SW2

The SW1 and SW2 bytes hold the standard status codes returned by the card. Their respective values are 90h and 00h if the operation is successful.

If the number of data bytes to be transmitted is greater than 248, the command below containing the last data bytes must be sent before the ‘normal’ **ISO Input** command containing the first 248 data bytes.

GBR Format

14h FFh FFh FFh FFh (LN-248) <data248.dataLN>Main Card
1Ch FFh FFh FFh FFh (LN-248) <data248.dataLN> Selected Auxiliary Card

Response

S SW1 SW2

Exchange APDU – Asynchronous Cards

Sends a command Application Data Protocol Unit (APDU) to a card, and retrieves the response APDU.

GBR Format

15h APDU Main Card
1Dh APDU Selected Auxiliary Card

Where:

APDU	Is the command APDU. If the T=1 protocol is selected and the APDU command length is greater than the card information field size, it is truncated and sent to the card in several chained blocks. If the T=0 protocol is selected, the APDU transportation in T=0 TPDU (Transport Protocol Data Unit) is handled by the GBR. Please refer to the documentation concerning the card currently used for APDU command details. Up to three operations are required to perform a maximum length ISO short APDU exchange (261 bytes for APDU and 258 bytes for APDU responses).

Response

S Response APDU

Where:

Response APDU	Is the response APDU to the command. If the T=1 protocol is selected and the card replies in chained blocks, they are concatenated. If the T=0 protocol is selected, the T=0 TPDU of the response is mapped in the APDU response format by the GBR. Refer to the documentation concerning the card currently used for APDU response details.

If the command APDU length (LA) exceeds 254 bytes, the command below containing the last part of the command APDU must be sent before the “normal” APDU exchange command containing the first 254 bytes.

GBR Format

15h FFh FFh FFh FFh (LA-254) <apdu255.apduLA> main card.

1Dh FFh FFh FFh FFh (LA-254) <apdu255.apduLA> selected auxiliary card.

If the response APDU length (Lr) exceeds 254 bytes, the first 254 bytes of the response are returned with the status code 1Bh indicating that the command below must be sent to retrieve the last bytes of the response.

GBR Format

15h FFh FFh FFh FFh XX Main card.

1Dh FFh FFh FFh FFh XX Selected auxiliary card.

Where:

XX can be any dummy byte value.

APDU Format

The APDU format is defined by the ISO 7816-4 standard.

APDUs can belong to one of several cases, depending on the length and contents of the APDU. The GemCore V1.21-Based Reader handles the following cases

Case 1	No command or response data.
Case 2	Short format: no command data, response data between 1 and 256 bytes.
Case 3	Short format: command data between 1 and 255 bytes and no response data.
Case 4	Short format: command data between 1 and 255 bytes, response data between 1 and 256 bytes.

These cases are referred to as 1, 2S, 3S, and 4S, respectively.

Command Format

Commands are sent in the following format:

Header	Body		
CLA INS P1 P2	Lc	Parameters/data	Le

The fields are described below.

Header Fields

Header fields are mandatory. They are as follows:

Field Name	Length	Description
CLA	1	Instruction class.
INS	1	Instruction code. This is given in the command descriptions.
P1	1	Parameter 1.
P2	1	Parameter 2.

Body Fields

The command body is optional. It includes the following fields:

Field Name	Length	Description
Lc	1	Length of the data field.
Data	Lc	Command parameters or data.
Le	1	Expected length of the data to be returned.

For full details about the header and body field contents refer to the documentation concerning the card currently used.

Response Format

Responses to commands are received in the following format.

Body	Trailer
Data	SW1, SW2

The body is optional and holds any data returned by the card.

The trailer includes the following two mandatory bytes:

SW1: Status byte 1 which returns the command processing status
SW2: Status byte 2 which returns the command processing qualification

For full details about the response field contents refer to the documentation concerning the card currently used.

T=1 IFSC/IFSD

If the T=1 protocol is used, when block chaining occurs, the buffers’ length is determined by IFSC and IFSD parameters.

The default values for the GBR buffer (IFSD) and the card buffer (IFSC) are 32 bytes.

The smart card can indicate a specific value of IFSC during the ATR. GBR takes into account this new value instead of the default one.

To specify an IFSD value other than the default one to the card, see the **Power Up – Asynchronous Card** command.

Card Status – Asynchronous Card

This command is used to obtain the status of the main card interface or of the auxiliary card. It returns information indicating:

- The type of card currently used
- Card presence
- The power supply value
- The card power status
- The communication protocol (T=0 or T=1)
- The speed parameters between the card and the reader

GBR Format

17h Main Card
1Fh Selected Auxiliary Card

Response

S STAT TYPE CNF1 CNF2 CNF3 CNF4

Where:

STAT	NNNNXXXX	Card number 0000XXXX=Card#0 0001XXXX=Card#1
	XXXXXXX0	Power supply = 5V
	XXXXXXX1	Power supply = 3V
	XXXXXX0X	Card not powered
	XXXXXX1X	Card powered
	XXXXX0XX	Card not inserted
	XXXXX1XX	Card inserted
	XXXX0XXX	T=0 protocol
	XXXX1XXX	T=1 protocol
TYPE	Activated Card type	

CNF1
CNF2
CNF3
CNF4
	CNF1=TA1 (FI/DI)
	CNF2=TC1 (EGT)
	CNF3=WI
	CNF4=00
		T=0 Card as per ISO 7816/3

CNF1
CNF2
CNF3
CNF4
	CNF1=TA1 (FI/DI)
	CNF2=TC1 (EGT)
	CNF3=IFSC
	CNF4=TB3 (BWI/CWI)
		T=1 Card as per ISO 7816/3

Specific Commands for Asynchronous Cards EMV-Compliant Operating Mode

These commands behave specifically when the EMV-compliant mode is selected.

- Power Up – EMV-compliant
- Exchange APDU – EMV-compliant
- Card Status – EMV-compliant

When in EMV-compliant operating mode, the GBR rejects the following commands:

- Change Card Communication Parameters – Asynchronous cards
- ISO Ouput – Asynchronous cards
- ISO Input – Asynchronous cards

When in EMV-compliant operating mode, the transparent type can be selected for a card. However, the card cannot be used because all commands are rejected.

See “*Appendix A. Status Codes*” for a description of status codes.

Power Up – EMV-Compliant

This command powers up and resets the card.

The card response is transmitted using the EMV criteria, and the card behavior is EMV-compliant.

The card can be:

- Accepted
- Accepted after a warm reset
- Rejected

If protocol T=1 is selected, an automatic IFSD exchange is performed.

GBR Format	**12h**	Main card
	1Ah	Selected Auxiliary Card

Response

S <card response>

Where:

<card response>	Is the card Answer To Reset (ATR)

Exchange APDU – EMV Compliant

Sends a command Application Data Protocol Units (APDU) to a card, and retrieves the response APDU.

Behavior obeys to EMV requirements. For example, deactivation on timeout.

GBR Format

15h APDU Main Card
1Dh APDU Selected Auxiliary Card

Where:

APDU	Is the command APDU. If the T=1 protocol is selected and the APDU command length is greater than the card information field size, it is truncated and sent to the card in several chained blocks. If the T=0 protocol is selected, the APDU transportation in T=0 TPDU (Transport Protocol Data Unit) is handled by the GBR. Please refer to the documentation concerning the card currently used for APDU command details. Up to three operations are required to perform a maximum length ISO short APDU exchange (261 bytes for APDU and 258 bytes for APDU responses).

Response

S Response APDU

Where:

Response APDU	Is the response APDU to the command. If the T=1 protocol is selected and the card replies in chained blocks, they are concatenated. If the T=0 protocol is selected, the T=0 TPDU of the response is mapped in the APDU response format by the GBR. Refer to the documentation concerning the card currently used for APDU response details.

If the command APDU length (LA) exceeds 254 bytes, the command below containing the last part of the command APDU must be sent before the "normal" APDU exchange command containing the first 254 bytes.

GBR Format

15h FFh FFh FFh FFh (LA-254) <apdu255.apduLA> main card.

1Dh FFh FFh FFh FFh (LA-254) <apdu255.apduLA> selected auxiliary card.

If the response APDU length (Lr) exceeds 254 bytes, the first 254 bytes of the response are returned with the status code 1Bh indicating that the command below must be sent to retrieve the last bytes of the response.

GBR Format

15h FFh FFh FFh FFh XX Main card.

1Dh FFh FFh FFh FFh XX Selected auxiliary card.

Where:

XX can be any dummy byte value.

APDU Format

The APDU format is defined by the ISO 7816-4 standard.

APDUs can belong to one of several cases, depending on the length and contents of the APDU. The GemCore V1.21-Based Reader handles the following cases

Case 1 No command or response data.
Case 2 Short format: no command data, response data between 1 and 256 bytes.
Case 3 Short format: command data between 1 and 255 bytes and no response data.
Case 4 Short format: command data between 1 and 255 bytes, response data between 1 and 256 bytes.

These cases are referred to as 1, 2S, 3S, and 4S, respectively.

Command Format

Commands are sent in the following format:

Header	Body		
CLA INS P1 P2	Lc	Parameters/data	Le

The fields are described below.

Header Fields

Header fields are mandatory. They are as follows:

Field Name	Length	Description
CLA	1	Instruction class.
INS	1	Instruction code. This is given in the command descriptions.
P1	1	Parameter 1.
P2	1	Parameter 2.

Body Fields

The command body is optional. It includes the following fields:

Field Name	Length	Description
Lc	1	Length of the data field.
Data	Lc	Command parameters or data.
Le	1	Expected length of the data to be returned.

For full details about the header and body field contents refer to the documentation concerning the card currently used.

Response Format

Responses to commands are received in the following format.

Body	Trailer
Data	SW1, SW2

The body is optional and holds any data returned by the card.

The trailer includes the following two mandatory bytes:

SW1: Status byte 1 which returns the command processing status
SW2: Status byte 2 which returns the command processing qualification

For full details about the response field contents refer to the documentation concerning the card currently used.

T=1 IFSC/IFSD

If the T=1 protocol is used, when block chaining occurs, the buffers’ length is determined by IFSC and IFSD parameters.

The default values for the GBR buffer (IFSD) and the card buffer (IFSC) are 32 bytes.

The smart card can indicate a specific value of IFSC during the ATR. GBR takes into account this new value instead of the default one.

Card Status – EMV Compliant

This command is used to obtain the status of the main card interface or of the auxiliary card. It returns information indicating:

- The type of card currently used
- Card presence
- The power supply value
- The card power status
- The communication protocol (T=0 or T=1)
- The speed parameters between the card and the reader

GBR Format

17h Main Card
1Fh Selected Auxiliary Card

Response

S STAT TYPE CNF1 CNF2 CNF3 CNF4

Where:

STAT	NNNNXXXX	Card number 0000XXXX=Card#0 0001XXXX=Card#1
	XXXXXXX0	Power supply = 5V
	XXXXXXX1	Power supply = 3V
	XXXXXX0X	Card not powered
	XXXXXX1X	Card powered
	XXXXX0XX	Card not inserted
	XXXXX1XX	Card inserted
	XXXX0XXX	T=0 protocol
	XXXX1XXX	T=1 protocol
TYPE	Activated Card type	

CNF1
CNF2
CNF3
CNF4
	CNF1=TA1 (FI/DI)
	CNF2=TC1 (EGT)
	CNF3=WI
	CNF4=00
		T=0 Card as per ISO 7816/3

CNF1
CNF2
CNF3
CNF4
	CNF1=TA1 (FI/DI)
	CNF2=TC1 (EGT)
	CNF3=IFSC
	CNF4=TB3 (BWI/CWI)
		T=1 Card as per ISO 7816/3

Specific Commands for Asynchronous Cards in Transparent Mode

These commands are designed for use with an asynchronous card in transparent mode (type = EFh or FFh). They have a specific behavior.

Commands that are valid in this mode are:

- **Change Transparent Mode Parameters**
- **Power Up – Transparent Mode**
- **Exchange Block – Transparent Mode**
- **Card Status – Transparent Mode**

See “*Appendix A. Status Codes*” for a description of status codes.

Change Transparent Mode Parameters

This command is used to set the working parameters of the transparent mode.

GBR Format

12h CFG ETU EGT CWT BWT Main Card

1Ah CFG ETU EGT CWT BWT Selected Auxiliary Card

Where:

CFG specifies the card characteristics and selects the operating mode.

XXXXXXX0	Vcc for the card is 5V.
XXXXXXX1	Vcc for the card is 3V.
XXXX0XXX	The format of the blocks received is not defined: the end of a received block is determined the CWT timeout.
XXXX1XXX	The format of the blocks received is comparable to that of the T=1 protocol. The third byte of the block indicates the length of the data to be received before the EDC field.
XX0XXXXX	The direct convention is used to transfer byte.
XX1XXXXX	The inverse convention is used to transfer byte.
X0XXXXXX	During the ATR, a check is performed for the T0 and TDI characters to compute the number of characters to be received.
X1XXXXXX	No check or computation is performed during the ATR. The ATR is complete upon CWT timeout.
0XXXXXXX	Significant only if bit 3 of the CFG is set. EDC is one byte long.
1XXXXXXX	Significant only if bit 3 of the CFG is set. EDC is two bytes long.

ETU ETU duration, coded in clock period number minus one and divided into three: [(FI/DI)-1]/3

SAMPLES FOR COMMON TA1:

TA1	FI/DI	ETU Parameter
0x11	372	124 = 7Bh
0x12	186	61 = 3Dh
0x13	93	30 = 1Eh
0x14	41.5	15 = 0Fh
0x18	31	10 = 0Ah
0x58	124	41 = 29h
0x95	32	10 = 0Ah

EGT — Defines the extra guard time etus between characters sent by the GemPC. The total duration of this character is (11+EGT)*(etu duration)

CWT — Defines the maximum waiting time between the leading edges of two consecutive characters in the same direction.

Timeout duration is (11+2^CWT)*(etu duration)

The maximum value for CWT is 15.

BWT — Defines the maximum waiting time between the leading edges of two consecutive characters sent in opposite directions.

Timeout duration is [(11+960*21 BWT)]*(etu duration)

Response **S**

Note*:* *Default parameter values are*

CFG = 40h

ETU = 7Bh

EGT = 02h

CWT = 0Dh

BWT = 04h

Power Up – Transparent Mode

This command powers up and resets an asynchronous card in transparent mode.

GBR Format

12h Main card

1Ah Selected auxiliary card

Response

S <card response>

Where:

<card response> is the card Answer To Reset (ATR).

***Note**: No verification is performed on characters returned by the card, in particular with respect to TS and TCK.*

Exchange Block – Transparent Mode

This command sends a block to a card and receives a block back in response.

GBR Format

15h BLOCK	Main card
1Dh BLOCK	Selected auxiliary card

Where:

BLOCK is the block to be sent.

Response

S Response BLOCK

Where:

Response BLOCK is the block received in response.

Up to three operations are required in order to perform an exchange of blocks of maximum length (259 bytes).

If the length (LB) of the block to be sent to the card exceeds 254 bytes, the command below containing the last part of the block must be sent before the “normal” exchange block command containing the first 254 bytes.

GBR Format

14h <block 255.blockLB> Main card

1Ch <block 255.blockLB> Selected auxiliary card

If the length of the block received in response (LR) exceeds 254 bytes, the first 254 bytes are returned with the status code 1Bh, indicating that the command below must be sent to retrieve the last bytes of the response.

GBR Format

13h	Main card
1Bh	Selected auxiliary card

***Note**: If no block is given in the command, the GBR waits for the response block.*

Card Status – Transparent Mode

This command is used to obtain the current transparent mode parameters. It returns information regarding:

- The transparent mode selected
- The card presence
- The power supply value
- The card power status
- The speed and timeout parameters

GBR Format

17h Main card

1Fh Selected auxiliary card

Response

S STAT TYPE ETU EGT CWT BWT

Where:

STAT	NNNNXXXX	Card number 0000XXXX = card # 0 0001XXXX = card # 1
	XXXXXXX0	Power supply = 5V
	XXXXXXX1	Power supply = 3V
	XXXXXX0X	Card not powered on.
	XXXXXX1X	Card powered on
	XXXXX0XX	Card not inserted
	XXXXX1XX	Card inserted
TYPE	Activated card type	EFh = Transparent mode at normal speed FFh = Transparent mode at double speed
ETU	etu duration	
EGT	Extra guard time requested	
CWT	Character waiting time	
BWT	Response block waiting time	

Specific Commands for Synchronous Cards

These commands are used with a synchronous card (type = 03h up to 0Fh). Their behavior is specific.

Commands valid in this mode are:

- **Power Up – Synchronous Card**
- **Read Data From Synchronous Card (ISO Out)**
- **Send Data to Synchronous Card (ISO In)**
- **Exchange with Synchronous Card (APDU)**
- **Card Status – Synchronous Card**

See “*Appendix A. Status Codes*” for a description of status codes.

Power Up – Synchronous Cards

This command powers up and resets a card.

GBR Format

12h Main Card
1Ah Selected Auxiliary Card

***Note**: The card is powered on with 5V.*

Response

S <card response>

Where:

<card response>	Is the card Answer To Reset (ATR).

***Note:** If the memory card does not return an ATR, a default ATR is returned (3Bh 00h 00h 00h 00h 00h)*

Read Data From Synchronous Card (ISO Out)

This command reads data from a memory card the T=0 ISO Out format. See "*Using the GemCore-Based Reader with Memory Cards*" for a list of the respective memory card commands.

This command can return up to 249 bytes.

GBR Format

13h CLA INS A1 A2 LN Main Card
1Bh CLA INS A1 A2 LN Selected Auxiliary Card

ROS Format

DBh CLA INS A1 A2 LN Main Card only

Where:

CLA, INS, A1, A2, and LN — Are the five ISO-like header bytes. The ISO header is interpreted by the GemCore-Based Reader.

Response

S <data> SW1 SW2

Where:

<data> — Is the data returned by the card. If a smart card error or GemCore V1.21-Based Reader error is detected (S<>0 and S<>E7h), the GBR does not return any data.
The card may return any number of bytes up to LN.

SW1, SW2 — Are status bytes added by the GBR to simulate an ISO format. These bytes specify an error if they differ from 90h 00h.

Send Data To Synchronous Card (ISO In)

This command sends data to a memory card, using a format command in the same way as ISO commands do. See "*Using the GemCore-Based Reader with Memory Cards*" for a list of the respective memory card commands.

The length of data sent to the card must not exceed 249 bytes.

GBR Format

14h CLA INS A1 A2 LN <data> Main Card
1Ch CLA INS A1 A2 LN <data> Selected Auxiliary Card

Where:

CLA, INS, A1, A2, and LN	Are the five ISO-like header bytes. The ISO header is interpreted by the GemCore-Based Reader.
<data>	Represents the LN data bytes transmitted to the card. The maximum length of the data is 249 bytes.

Response

S SW1 SW2

SW1 and SW2 are added by the GBR to emulate an ISO format. These bytes specify an error if they differ from 90h 00h.

Exchange With Synchronous Card (APDU)

This command is used to exchange data with a synchronous card using a format command in the same way as APDU ISO commands do.

GBR Format

15h APDU Main Card
1Dh APDU Selected Auxiliary Card

Where:

APDU Is the command APDU. The command APDU must not exceed a length of 254 bytes.

Response

S Response APDU

Where:

Response APDU Is the response APDU to the command.

The response APDU must not exceed a length of 251 bytes. Please refer to the documentation concerning the card currently used for APDU response details.

APDU Format

For APDU command format information, refer to the command “**Exchange APDU – Asynchronous Cards**”.

Response Format

Responses to commands are received in the following format.

Body	Trailer
Data	SW1, SW2

The body is optional and holds any data returned by the card.

The trailer includes the following two mandatory bytes:

SW1: Status byte 1 is added by the GBR to specify an error. It is 90h if there is no error.
SW2: Status byte 2 is added by the GBR to specify an error. It is 00h if there is no error.

Card Status – Synchronous Card

This command is used to obtain the status of the main card interface or of the auxiliary card. It returns information indicating:

- The type of card currently used
- Card presence
- The power supply value
- The card power status

GBR Format

17h Main Card
1Fh Selected Auxiliary Card

Response

S STAT TYPE CNF1 CNF2 CNF3 CNF4

Where:

STAT	NNNNXXXX	Card number 0000XXXX=Card#0 0001XXXX=Card#1
	XXXXXXX0	Power supply = 5V
	XXXXXXX1	Power supply = 3V
	XXXXXX0X	Card not powered
	XXXXXX1X	Card powered
	XXXXX0XX	Card not inserted
	XXXXX1XX	Card inserted
TYPE	Activated Card type	

CNF1
CNF2
CNF3
CNF4
	CNF1= 00 (RFU)
	CNF2= 00 (RFU)
	CNF3= 00 (RFU)
	CNF4= 00 (RFU)	

Reader Memory Management Commands

Reader memory management commands are:

- **Read Memory**
- **Write Memory**
- **Erase Flash Memory**
- **Select External Memory Page**
- **Read CPU Port**
- **Write CPU Port**

See “*Appendix A. Status Codes*” for a description of status codes.

Read Memory

Reads the contents of all the memory areas which can be addressed by the reader. This command is only operative provided that the memory under consideration is not read-protected.

Format

22h Type [Page] ADH ADL LN

Where:

Type Is the type of memory to be read, mapped as follows:

b7	b6	b5	b4	b3	b2	b1	b0
0	P	0	0	T	T	T	T

P Is the page parameter flag. If set, this bit specifies that the optional Page parameter is present.

TTTT Is the type of memory read.

Value	Memory Type
0001	IDATA (Internal CPU data memory)
0010	XDATA (External data memory)
0101	Code memory
0110	XDATA (External data memory, FLASH Atmel)

Page Is the optional byte indicating the XDATA and CODE page to be selected before reading can occur. If this parameter is not present, the page currently selected is read. See *"Select External Memory Page"* for further details.

Note: *The current page is not modified.*

ADH,ADL Is the 16-bit address of the first byte to be read. ADH is the most significant byte and ADL is the least significant byte.

LN Is the length of data to be read in bytes.

Response

S <data bytes>

Write Memory

Writes in all memory areas which can be addressed by the reader. This command is only operative when the memory under consideration is not write-protected.

Format

23h Type [Page] ADH ADL LN <data>

Where:

Type — Is the type of memory to be written, mapped as follows:

b7	b6	b5	b4	b3	b2	b1	b0
E	P	0	0	T	T	T	T

E — Is the FLASH/EEPROM memory type flag (only for XDATA or CODE memory types). If set, this bit indicates a FLASH-type memory.

P — Is the page parameter flag. If set, this bit specifies that the optional Page parameter is present.

TTTT — Is the type of memory to be read.

Value	**Memory Type**
0001	IDATA (Internal CPU data memory)
0010	XDATA (external data memory)
0101	CODE memory
0110	Xdata (External data memory, Flash Atmel)

Examples :

1) 23h 02h 80h 00h 01h <Data>: Writes one byte in the RAM memory, in the XDATA area, at location 8000h.

2) 23h 85h 80h 00h 01h <Data>: Writes one byte in the FLASH memory, in the CODE area, at location 8000h.

3) 23h 86h 80h 00h 01h <Data>: Writes one byte in the FLASH Atmel memory, in the XDATA area, at location 8000h.

Page — Is the optional byte indicating the XDATA and CODE page to be selected before the **Write** command can be performed. If this parameter is absent, the page currently selected is written. See *"Select External Memory Page"* for details.

Notes: *The current page is not modified.*
Writing to the FLASH Atmel memory can only take place on readers with 256 bytes of RAM at XDATA address 100h.
The maximum length of data written with a single command is 250 bytes.

ADH,ADL — Define the 16-bit address of the first byte of memory to be written. ADH is the most significant byte and ADL is the least significant byte.

LN — Is the length of data to be written in bytes.

<data> — Is the data to be written.

Response

S

Memory Read And Write Protection

Both the program memory and the data memory can be protected against read or write access. Two 8-byte codes are used for this purpose: the first code protects the program memory, and the second protects the data memory.

When the program memory is protected:

- The 22 01 ADH ADL LNG command returns error code 1F.
- The 22 X5 ADH ADL LNG command returns error code 1F.
- The 23 01 ADH ADL LNG <DATA> command returns error code 1F.
- The 23 X5 ADH ADL LNG <DATA> command returns error code 1F.
- The 26 85 ADH ADL DATA command returns error code 1F.

When the data memory is protected:

- The 22 X2 ADH ADL LNG command returns error code 1F.
- The 23 X2 ADH ADL LNG <DATA> command returns error code 1F.
- The 26 82 ADH ADL DATA command returns error code 1F.

In order to be efficient, the data memory protection must be used along with a protected program memory.

Memory protection codes are located in the application program memory area and must be downloaded with the application software.

The program memory protection code is located between address FFB0 and address FFB7.

The data memory protection code is located between address FFA0 and address FFA7.

In order to be validated, the eight bytes of the protection code must be followed by eight bytes representing the complementary code.

Example :
FFA0: 11 22 33 44 55 66 77 88 FF FF FF FF FF FF FF FF
FFB0: 01 02 03 04 05 06 07 08 FE FD FC FB FA F9 F8 F7

The access to the data memory is free (that is, its code is not validated). The program memory is protected.

In order to enable read or write access to a protected area, the following write command must be used:

Code memory: 23 X5 FF B0 08 < 8 byte code >
Data memory: 23 X2 FF A0 08 < 8 byte code >

In all cases, the reader response is the status code 1F.

If the code presented is correct, the next read or write command will be executed.

Erase Flash Memory

Erases all or part of the contents of the Flash memory. This command is only operative provided that the memory is not considered write-protected.

***Note**: Execution of this command can last up to one minute.*

Format

26h Type [Page] ADH ADL <CODE>

Where:

Type Is the type of memory to be written, mapped as follows:

b7	b6	b5	b4	b3	b2	b1	b0
1	P	0	0	T	T	T	T

P Is the page parameter flag. If set, this bit specifies that the optional Page parameter is present.

TTTT Is the type of memory to erase.

Value	Memory Type
0010	XDATA memory
0101	CODE memory
0110	XDATA memory (Flash Atmel)

Page Is an optional byte indicating the XDATA and CODE page to be selected before writing can take place. If this parameter is absent, the page currently selected is erased. See the *"Select External Memory Page"* command for details.

***Notes**: The current page is not modified.*

For Flash Atmel, it is not necessary to erase the memory before writing. If memory erasing is requested, the entire memory must be erased.

ADH,ADL Define the 16-bit start address for the erase command. ADH is the most significant byte and ADL is the least significant byte.

<CODE> Is the erase command code.

It is 10h if the whole memory is to be erased (the address should then be D555h), or 30h if one sector only is to be erased (the address should then be the sector address).

Response

S

***Warning**: Executing this command can require up to ten seconds. The host timeout parameter must be set accordingly.*

Example 1:

The following commands erase the data stored in the AMD 29F010 Flash memory used for program storage, starting from address 8000h. This memory is organized into eight sectors of 16 Kbytes each.

Memory configuration:

	Page#0	Page#1	Page#2	Page#3
8000h BFFFh	Sector 1	Sector1	Sector 1	Sector 1
C000h FFFFh	Sector 2	Sector 2	Sector 2	Sector 2

Erase chip command:	26h 85h D5h 55h 10h
Erase first sector command:	26h 85h 80h 00h 30h
Erase second sector command:	26h 85h C0h 00h 30h
Erase first sector in code page 2 command:	26h C5h 20h 80h 00h 30h

Example 2:

The following commands erase the data stored in the AMD 29F040 Flash memory used for program storage starting from address 8000H. This memory is organized into eight sectors of 64 Kbytes each.

Erasing one sector erases two pages of 32 Kbytes each.

Memory configuration:

	Page#0	Page#1		Page#6	Page#7
8000h FFFFh	Sector 0		. . .	Sector 3	

Erase chip command:	26h 85h D5h 55h 10h
Erase sector 0 command:	26h 85h 80h 00h 30h

Example 3:

The following command erases the entire Flash Atmel memory:

26h 86h D5h 55h 10h

Select External Memory Page

GemCore can manage up to sixteen 32-Kbyte pages of CODE memory, and sixteen 32-Kbyte pages of XDATA memory. This command selects the active page.

When 512 Kbytes of memory are used, the physical memory is split into two blocks of eight pages each.

- Application #1 is mapped on pages 0 and 1 of the first block.
- Application #2 is mapped on pages 2 and 3 of the first block.
- Application #3 is mapped on pages 4, 5, 6 and 7 of the first block.
- Application #4 is mapped on pages 0 to 7 of the second block.

By default, Application #1, CODE Page 0 and XDATA Page 0 are selected after **Power up**.

Format

27h Page [App]

Where:

Page Is the byte indicating the XDATA and CODE page to select in the following format:

b7	b6	b5	b4	b3	b2	b1	b0
0	C	C	C	0	D	D	D

Bits 2 to 0: Indicate the XDATA page to select.
Bit 3: Not used.
Bits 6 to 4: Indicate the CODE page to select.
Bit 7: Not used.

App Is an optional byte indicating the application number.

Memory organization:

8000h						
FFFFh	Page#0	Page#1	Page#2	Page#3	...	Page#7
XDATA	xxxxx000	xxxxx001	xxxxx010	xxxxx011	...	xxxxx111
CODE	x000xxxx	x001xxxx	x010xxxx	x011xxxx	...	x111xxxx

Response

S

Read CPU Port

Reads the state of a CPU port.

Format

24h PORT

Where:

PORT Is the number of the port to be read as defined in the following table:

Value	Port
00	Port0- P0
01	Port1-P1
02	Port2-P2
03	Port3-P3

Response

S Value

Where:

Value Is the value read from the specified CPU port.

Write CPU Port

Writes to a CPU port.

Format

25h PORT VALUE

Where:

PORT Is the number of the port to be written to, as defined in the following table:

Value	**Port**
00	Port0-P0
01	Port1-P1
02	Port2-P2
03	Port3-P3

VALUE Is the value to be written to the CPU port.

Response

S

LCD Commands

The LCD commands are used to control the LCD. They must be used with LCD modules that are compatible with the HITACHI HD 44780 LCD Controller.

LCD commands are:

- **Init The LCD**
- **Display Character String**
- **Display Character**
- **Send LCD Command**

See “*Appendix A. Status Codes*” for a description of status codes.

Init The LCD

Initializes the LCD.

Format	**2Ah**
Response	**S**

Display Character String

Displays a string of characters on the LCD.

Format

2Bh [POS] CHARS

Where:

[POS]	Is the beginning of the character string. This parameter starts at 80h for character 1 line 1, 81h for character 2 line 1, C0h for character 1 line 2 and so on. If this byte is omitted, the character string is displayed at the current cursor position. Bit 7 of this byte must always be set to 1.
CHARS	Is the character string to be displayed in ASCII.

Response

S

Display Character

Displays a character on the LCD at the current cursor position.

Format **2Ch** CHAR

Where:

CHAR	Is the character to be displayed in ASCII.

Response **S**

Send LCD Command

Sends an LCD control command.

Format

2Dh COMCODE

Where:

COMCODE Is one of the command codes listed below:

Command Code	Action
01h	Clears the LCD.
02h	Cursor home.
04h	Moves the cursor to the left after a **Display Character** command.
05h	Moves the text to the right after a **Display Character** command.
06h	Moves the cursor to the right after a **Display Character** command.
07h	Moves the text to the left after a **Display Character** command.
08h	LCD off.
0Ch	LCD on and no cursor.
0Dh	LCD on and blink character at cursor position.
0Eh	LCD on and display fixed cursor.
0Fh	LCD on and display blinking cursor.
10h	Moves the cursor to the left.
14h	Moves the cursor to the right.
18h	Moves the text to the left.
1Ch	Moves the text to the right.

Response

S

Keypad and Buzzer Commands

GemCore can control a 4x4 keypad and a buzzer with the following commands:

- **Set Key Press Timeout**
- **Sound Buzzer**

See “*Appendix A. Status Codes*” for a description of status codes.

Set Key Press Timeout

Sets the number of seconds the reader waits for a key to be pressed and switches the 25 milliseconds key tone on and off.

Format

32h TIME BEEP

Where:

TIME — Is the number of seconds the reader waits for a key to be pressed, in units of 100 ms. For example, 07h specifies 700 ms.

BEEP — Switches the key tone on/off. 00h switches it off and 01h switches it on.

Response

S KEY

Where:

KEY — Is the code of the key pressed (before the timeout). The following table lists the key codes:

Key	**Code**	**Key**	**Code**	**Key**	**Code**	**Key**	**Code**
1	11h	2	21h	3	31h	F1	41h
4	12h	5	22h	6	32h	F2	42h
7	13h	8	23h	9	33h	F3	43h
<	14h	0	24h	>	34h	F4	44h

Sound Buzzer

Sounds the buzzer and specifies its frequency and duration.

Format

33h TIME [FREQ]

Where:

TIME	Is the duration of the buzzer. The units of time are a function of the frequency.
[FREQ]	Is the frequency of the buzzer (between 1,183 Hz and 68,267 Hz). This is an optional parameter. If it is omitted, the sound frequency defaults to 3,600 Hz.

The following formulas can be used to determine approximate values for these parameters:

T*N/36000

[FREQ] = 307200 / N – 4

Where:

T = TIME in ms
N = Frequency in Hz

Example:

For a 2KHz beep to last 200 ms,

Time = T(200)*N(2000)/36000 = 11

[FREQ] = 307200 / N(2000) – 4 = 150

Response

S

Real Time Clock Commands

GemCore can read and update the date and time stored in the reader's clock with the following commands:

- **Read Date and Time**
- **Update Date And Time**

Each command is described in the following pages.

Read Date And Time

Reads the real-time clock date and time.

Format **3Ah**

Response **S** YEAR MONTH DAY HOUR MINUTE SECOND

Where:

YEAR	Is the year value, in BCD.
MONTH	Is the month value, in BCD.
DAY	Is the day value, in BCD.
HOUR	Is the hour value, in BCD.
MINUTE	Is the minute value, in BCD.
SECOND	Is the second value, in BCD.

For example, November 25, 1999, 17:15:00 is coded 99 11 25 17 15 00.

Update Date and Time

Updates the real-time clock date and time.

Format

3Bh YEAR MONTH DAY HOUR MINUTE SECOND

Where:

YEAR	Is the new year value, in BCD.
MONTH	Is the new month value, in BCD.
DAY	Is the new day value, in BCD.
HOUR	Is the new hour value, in BCD.
MINUTE	Is the new minute value, in BCD.
SECOND	Is the new second value, in BCD.

***Note**: A value must be entered for all the above fields.*

Response

S

GemPC410 Control Commands

The following commands are only available on GemPC410 readers. Some of the commands have no effect but are required in order to ensure compatibility with GemPC400 products.

The GemPC410 reader simulates a GCR400 with an external power supply.

GemPC410 commands are:

- **GemPC410 Set Timeout**
- **GemPC410 Refresh**
- **GemPC410 Power Down**
- **GemPC410 LED Management**
- **GemPC410 Status**

See “*Appendix A. Status Codes*” for a description of status codes.

GemPC410 Set Timeout

This command is only available on GemPC410 readers.

This command has no effect; it is only used for GCR400 compatibility.

Format 52h T

Response **S** = B0h

GemPC410 Refresh

This command is only available on GemPC410 readers.

It has no effect; it is only used for GCR400 compatibility.

Format **53h**

Response **S**

GemPC410 Power Down

This command is only available on GemPC410 readers.

It has no effect; it is only used for GCR400 compatibility.

Format **54h**

Response **S** = B0h

GemPC410 LED Management

This command is only available on GemPC410 readers.

It controls the LED.

Format

55h LED

Where:

LED = 00h : LED Off

LED = 01h : LED On

LED = 02h : Default value (the LED blinks when the smart card is powered down and comes on when the smart card is powered up).

Response

S

GemPC410 Status

This command is only available on GemPC410 readers.

It returns the GemPC410 status.

Format **56h**

Response **S** Status 00h

Where:

Status Is the current reader status

b7	b6	b5	b4	b3	b2	b1	b0
0	0	0	0	1	0	LED1	LED0

Bits 1 and 0: indicate the LED status
00 : LED Off
01 : LED On
10 : Default value (the LED blinks when the smart card is powered down and comes on when the smart card is powered up).

USING THE GEMCORE V1.1-BASED READER WITH MICROPROCESSOR CARDS

The GemCore V1.1-Based Reader handles ISO 7816-3 T=0 and T=1 protocol microprocessor cards. The following section describes the implementation of these standards.

Clock Signal

The GemCore V1.1-Based Reader can transmit one of two clock frequency values to the card, depending on the previously selected operating mode:

- 3.6864 MHz for the standard mode (ISO compliance),
- 7.3728 MHz for the double-speed mode (that is above ISO specifications, for cards which can operate at this frequency).

The operating mode is specified while selecting the card type with the **Define card type** command. Card type 02h should be selected for standard mode and card type 12h for double-speed mode.

Global Interface Parameters

These parameters are returned by the microprocessor card during the ATR. For more information on these parameters, refer to the ISO 7816-3 standard document.

TA1

The GemCore V1.1-Based Reader interprets this parameter to match its communication rate with that of the card, according to the clock rate conversion factor F. F is coded on the most significant nibble and the bit rate adjustment factor D is coded on the least significant nibble.

The initial communication rate used during the ATR is 9909.68 baud in the standard mode and 19819.35 baud in double-speed mode.

After receiving the ATR, the GemCore V1.1-Based Reader selects the communication rate according to TA1. Tables 1 and 2 show the clock rate conversion factors, the bit rate conversion factors, and the selected baud according to TA1 values for both the standard mode and the double-speed mode.

***Note**: The TA1 values handled by the GemCore V1.1-Based Reader are shaded in Tables 1 and 2.*

TB1 and TB2

The Vpp option is not available on the GemCore V1.1-Based Reader. TB1 and TB2 parameters are ignored and the Vpp default value is set to 5V.

TC1

This parameter defines the extra guardtime N, required by the card. This parameter is processed when sending characters to the card, to ensure a delay of at least (12+N) etu between two characters.

D=	1		2		4		8		12		16		20		32	
F=	TA1	Rate (bd)	TA1	Rate (bd)	TA1	Rate (bd)	TA1	Rate (bd)	TA1	Rate (bd)	TA1	Rate (bd)	TA1	Rate (bd)	TA1	Rate (bd)
372	**01**	9 909.68	**02**	19 819.35	**03**	39 638.71	**04**	79 277.42	**08**	-	**15**	158 554.84	**09**	-	**06**	-
372	**11**	9 909.68	**12**	19 819.35	**13**	39 638.71	**14**	79 277.42	**18**	118 916,13	**15**	158 554.84	**19**	-	**16**	-
558	21	-	**22**	13 212.90	**23**	26 425.81	**24**	52 851.61	**28**	79 277,42	**25**	105 703.23	**29**	-	**26**	-
744	31	-	**32**	9 909.68	**33**	19 819.35	**34**	39 638.71	**38**	-	**35**	79 277.42	**39**	-	**36**	-
1116	41	-	42	-	**43**	13 212.90	**44**	26 425.81	**48**	39 638,71	**45**	52 851.61	**49**	-	**46**	-
1488	51	-	52	-	**53**	9 909.68	**54**	19 819.35	**58**	-	**55**	39 638.71	**59**	-	**56**	-
1860	61	-	62	-	63	-	**64**	15 855.48	**68**	-	**65**	31 710.97	**69**	39 638,71	**66**	-
512	91	-	**92**	14 400.00	**93**	28 800.00	**94**	57 600.00	**98**	86 400,00	**95**	115 200.00	**99**	-	**96**	-
768	A1	-	A2	-	**A3**	19 200.00	**A4**	38 400.00	**A8**	57 600,00	**A5**	76 800.00	**A9**	-	**A6**	-
1024	B1	-	B2	-	**B3**	14 400.00	**B4**	28 800.00	**B8**	-	**B5**	57 600.00	**B9**	-	**B6**	115 200,00
1536	C1	-	C2	-	C3	-	**C4**	19 200.00	**C8**	28 800,00	**C5**	38 400.00	**C9**	-	**C6**	76 800,00
2048	D1	-	D2	-	D3	-	**D4**	14 400.00	**D8**	-	**D5**	28 800.00	**D9**	36 000,00	**D6**	57 600,00

Table 1. TA1 Values Handled in Standard Mode (Frequency: 3.6864 MHz)

D=	1		2		4		8		12		16		20		32	
F=	TA1	Rate (bd)	TA1	Rate (bd)	TA1	Rate (bd)	TA1	Rate (bd)	TA1	Rate (bd)	TA1	Rate (bd)	TA1	Rate (bd)	TA1	Rate (bd)
372	**01**	19819.35	**02**	39 638.71	**03**	79 277.42	**04**	158 554.84	**08**	-	**05**	-	**09**	-	**06**	-
372	**11**	19819.35	**12**	39 638.71	**13**	79 277.42	**14**	158 554.84	**08**	-	**15**	-	**19**	-	**16**	-
558	**21**	13 212.90	**22**	26 425.81	**23**	52 851.61	**24**	105 703.23	**18**	-	**25**	-	**29**	-	**26**	-
744	**31**	9 909.68	**32**	19 819.35	**33**	39 638.71	**34**	79 277.42	**28**	-	**35**	-	**39**	-	**36**	-
1116	41	-	**42**	13 212.90	**43**	26 425.81	**44**	52 851.61	**38**	-	**45**	-	**49**	-	**46**	-
1488	51	-	**52**	9 909.68	**53**	19 819.35	**54**	39 638.71	**48**	-	**55**	-	**59**	-	**56**	-
1860	61	-	62	-	**63**	15 855.48	**64**	31 710.97	**58**	-	**65**	-	**69**	-	**66**	-
512	**91**	14 400.00	**92**	28 800.00	**93**	57 600.00	**94**	115 200.00	**68**	-	**95**	-	**99**	-	**96**	-
768	A1	-	**A2**	19 200.00	**A3**	38 400.00	**A4**	76 800.00	**98**	-	**A5**	-	**A9**	-	**A6**	-
1024	B1	-	**B2**	14 400.00	**B3**	28 800.00	**B4**	57 600.00	**A8**	-	**B5**	-	**B9**	-	**B6**	-
1536	C1	-	C2	-	**C3**	19 200.00	**C4**	38 400.00		-	**C5**	-	**C9**	-	**C6**	-
2048	D1	-	D2	-	**D3**	14 400.00	**D4**	28 800.00		-	**D5**	-	**D9**	-	**D6**	-

Table 2. TA1 Values Handled in Double-Speed Mode (Frequency: 7.3728 Mhz)

Communication Protocols

The least significant nibble of the TD1 parameter in the ATR defines the protocol to be used by the reader (T=0 or T=1), according to the following table:

Value	Protocol
0	T=0
1	T=1

If the reader does not receive a TD1 value, it defaults to the T=0 protocol.

T=0 Protocol

The specific TC2 interface parameter is interpreted to set the value of the work waiting time, W. If this parameter is absent, a maximum of 960xD etu elapses before timing-out on a character sent by the card. Otherwise a maximum of 960xDxW etu elapses before timing-out.

To send instructions to a T=0 microprocessor card, the **ISO Input** and **ISO Output** or the exchange **APDU** commands are used.

T=1 Protocol

To send instructions to a T=1 microprocessor card, the **Exchange APDU** command is used. The T=1 specific interface bytes are interpreted as per clause 9 of the ISO 7816-3 standard. These bytes are TA3, TB3, TC3.

TA3 codes the Information Field Size of the card (IFSC). The default value is 32 bytes.

TB3 codes the BWI (Block Writing Time Integer) and the CWI (Character Waiting Time Integer).

TC3 defines the Error Detection Code (EDC) type.

USING THE GEMCORE-BASED READER WITH MEMORY CARDS

Memory cards cannot interpret smart card instructions in the same way as ISO 7816-3 microprocessor cards can.

T=0 formatted instructions are therefore interpreted and converted into the appropriate timing sequences required to control the memory cards listed in the tables below. For further details, refer to the relevant card documentation.

These instructions are send to the reader using the **ISO Input**, **ISO Output**, or **APDU Exchange** commands.

GPM256	**Card Type = 03h**
Read Byte	(ISO OUT) 00h B0h 00h (Start Address) (Read Length)

GPM416	**Card Type = 04h**
Read Byte	(ISO OUT) 00h B0h 00h (Start Address) (Read Length)
Write Byte	(ISO IN) 00h D0h 00h (Start Address) (Write Length) (Data, .., Data)
Erase Word	(ISO IN) 00h DEh (Number of Word) (Start Address) 00h
Present Card Secret Code	(ISO IN) 00h 20h 04h 08h 02h (Code2, Code1)
Present Erase Secret Code	(ISO IN) 00h 20h 40h 28h 04h (Code4, .., Code1)
Change Fuse State	(ISO IN) 00h D4h 00h 00h 00h

GPM896	**Card Type = 04h**
Read Byte	(ISO OUT) 00h B0h 00h (Start Address) (Read Length)
Write Byte	(ISO IN) 00h D0h 00h (Start Address) (Write Length) (Data, .., Data)
Erase Word	(ISO IN) 00h DEh (Number of Word) (Start Address) 00h
Present Card Secret Code	(ISO IN) 00h 20h 04h 0Ah 02h (Code2, Code1)
Present Erase Secret Code #1	(ISO IN) 00h 20h 00h 36h 06h (Code6, .., Code1)
Present Erase Secret Code #2	(ISO IN) 00h 20h 80h 5Ch 04h (Code4, .., Code1)
Change Fuse State	(ISO IN) 00h D4h 00h 00h 00h

GPM103	**Card Type = 07h**
Read Byte	(ISO OUT) 00h B0h 00h (Start Address) (Read Length)
Write Byte	(ISO IN) 00h D0h 00h (Start Address) (Write Length) (Data, .., Data)
Read Counter Value	(ISO OUT) 00h B2h 05h 08h 02h
Write New Counter Value	(ISO IN) 00h D2h 05h 08h 02h (Value MSB, Value LSB)
Erase and Write Carry	(ISO IN) 00h E0h 01h (Counter Address) 00h

GAM226	**Card Type = 0Fh**
Read Byte	(APDU) 00h B0h 00h (Address) (Read Length)
Write Byte	(APDU) 00h D0h 00h (Address) (Write Length) (Data, .., Data)
Erase and Write Carry	(APDU) 00h E0h 01h (Address)
Present Card Secret Code	(APDU) 00h 20h 00h 00h 03h (Code3, Code2, Code1)
Authenticate	(APDU) 00h 88h 01h A0h 06h (Alea6, .. , Alea1) 02h
Restore	(APDU) 00h D4h 00h 00h

GPM271	**Card Type = 0Eh**
Read Byte	(APDU) 00h B0h 00h (Address) (Read Length)
Write Byte	(APDU) 00h D0h 00h (Address) (Write Length) (Data, .., Data)
Erase and Write Carry	(APDU) 00h E0h 01h (Address)
Present Card Secret Code	(APDU) 00h 20h 00h 00h 03h (Code3, Code2, Code1)
Restore	(APDU) 00h D4h 00h 00h
Blow Fuse	(APDU) 00h DAh 00h 00h

GAM273	**Card Type = 0Eh**
Read Byte	(APDU) 00h B0h 00h (Address) (Read Length)
Write Byte	(APDU) 00h D0h 00h (Address) (Write Length) (Data, .., Data)
Erase and Write Carry	(APDU) 00h E0h 01h (Address)
Present Card Secret Code	(APDU) 00h 20h 00h 00h 03h (Code3, Code2, Code1)
Authenticate	(APDU) 00h 88h 00h 00h 04h (Alea4, Alea3, Alea2, Alea1) 01h
Restore	(APDU) 00h D4h 00h 00h
Blow Fuse	(APDU) 00h DAh 00h 00h

GPM276	**Card Type = 0Dh**
Read Byte	(APDU) 00h B0h 00h (Address) (Read Length)
Write Byte	(APDU) 00h D0h 00h (Address) (Write Length) (Data, .., Data)
Erase and Write Carry	(APDU) 00h E0h 01h (Address)
Present Card Secret Code	(APDU) 00h 20h 00h 00h 03h (Code3, Code2, Code1)
Restore	(APDU) 00h D4h 00h 00h
Blow Fuse	(APDU) 00h DAh 00h 00h

GAM275	**Card Type = 0Dh**
Read Byte	(APDU) 00h B0h 00h (Address) (Read Length)
Write Byte	(APDU) 00h D0h 00h (Address) (Write Length) (Data, .., Data)
Erase and Write Carry	(APDU) 00h E0h 01h (Address)
Present Card Secret Code	(APDU) 00h 20h 00h 00h 03h (Code3, Code2, Code1)
Authenticate	(APDU) 00h 88h 00h 00h 04h (Alea4, Alea3, Alea2, Alea1) 01h
Restore	(APDU) 00h D4h 00h 00h
Blow Fuse	(APDU) 00h DAh 00h 00h

GAM326	**Card Type = 0Fh**
Read Byte	(APDU) 00h B0h 00h (Address) (Read Length)
Write Byte	(APDU) 00h D0h 00h (Address) (Write Length) (Data, .., Data)
Erase and Write Carry	(APDU) 00h E0h 01h (Address)
Present Card Secret Code	(APDU) 00h 20h 00h 00h 03h (Code3, Code2, Code1)
Authenticate	(APDU) 00h 88h 11h A0h 06h (Alea6, .. , Alea1) 02h
Restore	(APDU) 00h D4h 00h 00h

GFM2K/4K	**Card Type = 06h**
Read Byte Area	(ISO OUT) 00h B0h (AddressH) (AddressL) (Read Length)
Write Byte Area	(ISO IN) 00h D0h (AddressH) (AddressL) (Write Length) (Data, .., Data)

GPM2K	**Card Type = 09h**
Read Data Area	(ISO OUT) 00h B0h 00h (Address) (Read Length)
Write Data Area	(ISO IN) 00h D0h 00h (Address) (Write Length) (Data, .., Data)
Read Protection Area	(ISO OUT) 00h B0h 80h 00h 04h
Write Protection Area	(ISO IN) 00h D0h 80h (Address) (Write Length) (Data, .., Data)
Read Security Area	(ISO OUT) 00h B0h C0h 00h 04h
Write Security Area	(ISO IN) 00h D0h C0h (Address) (Write Length) (Data, .., Data)
Present Card Secret Code	(ISO IN) 00h 20h 00h 00h 03h (Code3, Code2, Code1)

GPM8K	**Card Type = 08h**
Read Data Area	(ISO OUT) 00h B0h (AddressH) (AddressL) (Read Length)
Write Data Area	(ISO IN) 00h D0h 00h (AddressH) (AddressL) (Write Length) (Data,..,Data)
Present Card Secret Code	(ISO IN) 00h 20h 00h 00h 02h (Code2, Code1)
Read Protection Area	(ISO OUT) 00h B0h (80h + AddressH) 00h 20h
Write Protection Area	(ISO IN) 00h D0h (80h + AddressH) (AddressL) 01h (Data)
Read Security Area	(ISO OUT) 00h B0h C0h 00h 03h
Write Security Area	(ISO IN) 00h D0h C0h (Address) (Write Length) (Data, .., Data)

Table 3. Summary of the Memory Card Commands

APPENDIX A. STATUS CODES

The status codes returned the cards are listed in the table below.

Code	Meaning
01h	Unknown driver or command.
02h	Operation impossible with this driver.
03h	Incorrect number of arguments.
04h	Reader command unknown. The first byte of the command is an invalid command code.
05h	Response too long for the buffer.
10h	Response error at the card reset. The first byte of the response (TS) is not valid.
12h	Message too long. The buffer is limited to 254 bytes, of which 248 bytes are for the data exchanged with the card.
13h	Byte reading error returned by an asynchronous card.
15h	Card powered down. A power up command must be sent to the card before any other operation.
1Bh	A command has been sent with an incorrect number of parameters.
1Ch	Overlap on writing to the Flash memory.
1Dh	The TCK check byte is incorrect in a microprocessor card. Answer To Reset.
1Eh	An attempt has been made to write to write-protected external memory.
1Fh	Incorrect data has been sent to the external memory. This error is returned after a write check during a downloading operation. Can occur if the memory is protected.
A0h	Error in the card reset response, such as unknown exchange protocol, or TA1 byte not recognized. The card is not supported. The card Answer To Reset is nevertheless returned.
A1h	Card protocol error (T=0/T=1).
A2h	Card malfunction. The card does not respond to the reset or has interrupted an exchange by timing-out.
A3h	Parity error during a microprocessor exchange. This error only occurs after several unsuccessful attempts to resend.
A4h	Card has aborted chaining (T=1).
A5h	Reader has aborted chaining (T=1).
A6h	RESYNCH successfully performed by GemCore.
A7h	Protocol Type Selection (PTS) error.
B0h	GemPC410 command not supported.
CFh	Other key already pressed.

E4h	The card has just sent an invalid "Procedure Byte" (*see* ISO 7816-3).
E5h	The card has interrupted an exchange (the card sends an SW1 byte but more data remains to be sent or received).
E7h	Error returned by the card. The SW1 and SW2 bytes returned by the card are different than 90h 00.
F7h	Card removed. The card has been withdrawn during the execution of a command. Check that the card instruction is not partially completed.
F8h	The card is consuming too much electricity or is short-circuiting.
FBh	Card missing. There is no card in the smart card interface. The card may have been removed when it was powered up, but no command has been interrupted.

APPENDIX B. INTERPRETED SYNCHRONOUS SMART CARD DRIVER

Card Type 01h

This type enables you to use commands designed to handle synchronous card protocols which are not supported by GemCore. The protocol is defined by the parameters given in the command sent to GemCore. These parameters are specified in 8051 assembler code.

The 8051 assembler (INTEL ASM51) generates the commands to be executed and the GemCore software interprets the bytes as 8051 operation codes.

The GemCore interpreter can execute most 8051 instructions along with a few macro commands dedicated to synchronous cards.

Format

`16h` `CLA INS A1 A2 Lin <DATA IN> Lout Lcode <CODE>`Main Card

`1EH CLA INS A1 A2 Lin <DATA IN> Lout Lcode <CODE>`Selected Auxiliary Card

Where:

`CLA, INS, A1, A2`	Are the command parameters.
`Lin`	Is the number of bytes present in the `DATA IN` field.
`DATA IN`	Is the data to be sent to the card.
`Lout`	Is the length of the expected response.
`Lcode`	Is the number of bytes present in the `CODE` field.
`CODE`	Is the 8051 executable code.

Response

`S` `<data byte>`

8051 Interpreter

The GemCore interpreter handles the following functions:

- An accumulator (A)
- Eight registers (R0 to R7)
- A carry (C)
- A program counter (PC)

All instructions concerning the IDATA or XDATA RAM memories, also have an incidence on the XDATA memory. The XDATA memory starts at address 0000h and ends at address 00FFh.

The instruction to be executed is registered in this memory area (command 16h).

Only relative jumps can be used.

Initialization

Upon reception of a 16h command, the interpreter registers are initialized as follows:

PC points to the first `<CODE>` byte.
C = 0
A = `CLA`
R0 and R4 point to the address following the last `<CODE>`byte.
R1 points to the address of the first `<DATA IN>` byte.
R2 = `Lin`
R3 = `Lout`
R5 = `INS`
R6 = `A1`
R7 = `A2`

```
16h CLA=A INS=R5 A1=R6 A2=R7 Lin=R2 <DATAIN> Lout=R3 Lcode <CODE>
                                    ↑ R1                  R0/R4 ↑
```

Card Presence

Before executing a 16h command, the software checks that a card is actually present in the smart card connector.

If the card is missing, the following error message is returned: “CARD ABSENT” (S = FBh).

Card Withdrawal

As soon as the smart card is powered up, the GemCore card withdrawal interruption is activated.

If the card is withdrawn, the interpreted program is interrupted, all contacts with the smart card are deactivated and the following error message is returned: “CARD WITHDRAWN” (S = F7h).

Short Circuit

The card power up instructions check for short circuits between pins C1 (VCC) and C5 (GND).

If a short-circuit is detected, the following error message is returned: “TOO MUCH CONSUMPTION” (S = F8h).

Instructions

The following table is used to obtain a hexadecimal instruction code. The line number defines the four most significant bits and the column number defines the least significant bits (for example, INC A = 04h).

***Note**: Instructions in italics are macro-commands. See the "Macro-Commands" section in "Appendix B" for more details.*

	0	*1*	2	3	4	5	6	7
0	NOP 1/12	*VCC_OFF* 1/		RR A 1/16	INC A 1/18		INC @R0 1/22	INC @R1 1/22
1		*VCC_ON* 1/	*RESET* 1/	RRC A 1/19	DEC A 1/18		DEC @R0 1/22	DEC @R1 1/22
2		*CLR_RST* 1/13	RET (*) 1/	RL A 1/14	ADD A,#data 2/21		ADD A,@R0 1/26	ADD A,@R1 1/26
3		*SET_RST* 1/13	RETI (*) 1/	RLC A 1/21	ADDC A,#data 2/24		ADDC A,@RO 1/29	ADDC A,@R1 1/29
4	JC rel 2/15/19	*CLR_IO* 1/13	*RET_0K* 1/	*RDH_L* 1/	ORL A,#data 2/17		ORL A,@R0 1/22	ORL A,@R1 1/22
5	JNC rel 2/15/20	*SET_IO* 1/13	*RET_NOK* *2/*	*RDH_R* 1/	ANL A,#data 2/17		ANL A,@R0 1/22	ANL A,@R1 1/22
6	JZ rel 2/15/19	*CLR_CLK* 1/13	*RET_ERR* *3/*	*WRL_L* 1/	XRL A,#data 2/17		XRL A,@R0 1/22	XRL A,@R1 1/22
7	JNZ rel 2/17/20	*SET_CLK* 1/13	*CLK_INC* 1/14/XXX	*CLK_INC8* 1/14/XXX	MOV A,#data 2/19		MOV @R0, #data 1/27	MOV @R1, #data 1/27
8	SJMP rel 2/16	*CLR_C4* 1/13	*RDL_R* 1/	*RDL_L* 1/				
9		*SET_C4* 1/13	*WRH_L* 1/	*WRH_R* 1/	SUBB A,#data 2/		SUBB A,@R0 1/29	SUBB A,@R1 1/29
A		*CLR_C8* 1/13	*RST_PUL* 1/24	*WRL_R*				
B		*SET_C8* 1/13	*CLK_PUL* 1/24	CPL C 1/14	CJNE A,#data,rel 3/27/38		CJNE @R0,#data,rel 3/33	CJNE @R1,#data,rel 3/33
C			*WAIT_US* 20/5100	CLR C 1/14	SWAP A 1/15		XCH A,@R0 1/27	XCH A,@R1 1/27
D			*WAIT_MS* 1ms/255ms	SETB C 1/14			XCHD A,@R0 1/25	XCHD A,@R1 1/25
E		*IO_TO_C* 1/16	*GET_D* 1/1100/1s	*GET_I* 1/1100/1s	CLR A 1/14		MOV A,@R0 1/25	MOV A,@R1 1/25
F		*C_TO_IO* 1/15	*SEND_D* 1/1100/1s	*SEND_I* 1/1100/1s	CPL A 1/14		MOV @R0,A 1/25	MOV @R1,A 1/25

Table 4. Hexadecimal Instruction Codes

1/12/23 means: instruction over one byte/12 µs minimum/23 µs maximum. (For the jump instructions, the time taken is maximum when the jump is executed).

(*) Instruction already exists in 8051 Assembler code but with a different function for the interpreter.

	8	9	A	B	C	D	E	F
0	INC R0 1 / 19	INC R1 1 / 19	INC R2 1 / 19	INC R3 1 / 19	INC R4 1 / 19	INC R5 1 / 19	INC R6 1 / 19	INC R7 1 / 19
1	DEC R0 1 / 19	DEC R1 1 / 19	DEC R2 1 / 19	DEC R3 1 / 19	DEC R4 1 / 19	DEC R5 1 / 19	DEC R6 1 / 19	DEC R7 1 / 19
2	ADD A,R0 1 / 24	ADD A,R1 1 / 24	ADD A,R2 1 / 24	ADD A,R3 1 / 24	ADD A,R4 1 / 24	ADD A,R5 1 / 24	ADD A,R6 1 / 24	ADD A,R7 1 / 24
3	ADDC A,R0 1 / 27	ADDC A,R1 1 / 27	ADDC A,R2 1 / 27	ADDC A,R3 1 / 27	ADDC A,R4 1 / 27	ADDC A,R5 1 / 27	ADDC A,R6 1 / 27	ADDC A,R7 1 / 27
4	ORL A,R0 1 / 20	ORL A,R1 1 / 20	ORL A,R2 1 / 20	ORL A,R3 1 / 20	ORL A,R4 1 / 20	ORL A,R5 1 / 20	ORL A,R6 1 / 20	ORL A,R7 1 / 20
5	ANL A,R0 1 / 20	ANL A,R1 1 / 20	ANL A,R2 1 / 20	ANL A,R3 1 / 20	ANL A,R4 1 / 20	ANL A,R5 1 / 20	ANL A,R6 1 / 20	ANL A,R7 1 / 20
6	XRL A,R0 1 / 20	XRL A,R1 1 / 20	XRL A,R2 1 / 20	XRL A,R3 1 / 20	XRL A,R4 1 / 20	XRL A,R5 1 / 20	XRL A,R6 1 / 20	XRL A,R7 1 / 20
7	MOV R0,#data 2 / 22	MOV R1,#data 2 / 22	MOV R2,#data 2 / 22	MOV R3,#data 2 / 22	MOV R4,#data 2 / 22	MOV R5,#data 2 / 22	MOV R6,#data 2 / 22	MOV R7,#data 2 / 22
8	-	-	-	-	-	-	-	-
9	SUBB A,R0 1 / 26	SUBB A,R1 1 / 26	SUBB A,R2 1 / 26	SUBB A,R3 1 / 26	SUBB A,R4 1 / 26	SUBB A,R5 1 / 26	SUBB A,R6 1 / 26	SUBB A,R7 1 / 26
A	-	-	-	-	-	-	-	-
B	CJNE R0, #data, rel 3 / 32 / 43	CJNE R1, #data, rel 3 / 32/ 43	CJNE R2, #data, rel 3 / 32 / 43	CJNE R3, #data, rel 3 / 32 / 43	CJNE R4, #data, rel 3 / 32 / 43	CJNE R5, #data, rel 3 / 32 / 43	CJNE R6, #data, rel 3 / 32 / 43	CJNE R7, #data, rel 3 / 32 / 43
C	XCH A,R0 1 / 21	XCH A,R1 1 / 21	XCH A,R2 1 / 21	XCH A,R3 1 / 21	XCH A,R4 1 / 21	XCH A,R5 1 / 21	XCH A,R6 1 / 21	XCH A,R7 1 / 21
D	DJNZ R0,rel 2 / 24 / 28	DJNZ R1,rel 2 / 24 / 28	DJNZ R2,rel 2 / 24 / 28	DJNZ R3,rel 2 / 24 / 28	DJNZ R4,rel 2 / 24 / 28	DJNZ R5,rel 2 / 24 / 28	DJNZ R6,rel 2 / 24 / 28	DJNZ R7,rel 2 / 24 / 28
E	MOV A,R0 1 / 20	MOV A,R1 1 / 20	MOV A,R2 1 / 20	MOV A,R3 1 / 20	MOV A,R4 1 / 20	MOV A,R5 1 / 20	MOV A,R6 1 / 20	MOV A,R7 1 / 20
F	MOV R0,A 1 / 19	MOV R1,A 1 / 19	MOV R2,A 1 / 19	MOV R3,A 1 / 19	MOV R4,A 1 / 19	MOV R5,A 1 / 19	MOV R6,A 1 / 19	MOV R7,A 1 / 19

Table 4. Hexadecimal Instruction Codes (continued)

1/12/23 means: instruction over one byte / 12 µs minimum/23 µs maximum.

(*) Instruction already exists in 8051 Assembler code but with a different function for the interpreter.

Modified Instructions

RET

When the interpreter finds the RET code, the program is ended. GemCore returns the XDATA RAM memory data, R4 pointing to the first byte to be returned and R0 to the byte following the last response byte.

RETI

When the interpreter finds the RETI code, the program is ended. GemCore returns the contents of the registers in the following order:

```
PC A R0 R1 R2 R3 R4 R5 R6 R7 C
```

This instruction is used for software development.

Macro-Commands

%RET_OK

When the interpreter finds the RET_OK code, the program is ended. GemCore returns the last contents of the XDATA RAM memory, R4 pointing to the first byte to be returned and R0 to the byte following the last response byte.
S = 00h and the two status bytes SW1 = 90h and SW2 = 00H are added at the end of the message.

%RET_NOK (ERROR)

When the interpreter finds the RET_NOK instruction, the program is ended. GemCore returns the last contents of the XDATA RAM memory, R4 pointing to the first byte to be returned and R0 to the byte following the last response byte.
S = E7h, SW1 = 92h and SW2 returns an error code. These two bytes are added at the end of the message.

%RET_ERR (ERR1,ERR2)

Same as %RET_NOK but with SW1 = ERR1 and SW2 = ERR2.

%VCC_OFF

This command powers down all the smart card contacts as per ISO 7816-3 standard specifications.

%VCC_ON

This command initializes the smart card contacts.
If a card is present and is not short circuited, the following steps are carried out:

- VCC contact set at 5V.
- VPP contact set at 5V.
- RESET contact set to level 0.
- CLOCK contact set to level 0.
- I/O contact set to level 1 (high impedance).
- C4 contact set to level 0.
- C8 contact set to level 0.

%CLR_RST

This instruction sets the smart card's RESET contact to 0.

%SET_RST

This instruction sets the smart card's RESET contact to 1. It is only operative if the smart card is powered up.

%CLR_IO

This instruction sets the smart card's I/O contact to 0.

%SET_IO

This instruction sets the smart card's I/O contact to 1. It is only operative if the smart card is powered up.

%CLR_CLK

This instruction sets the smart card's CLK contact to 0.

%SET_CLK This instruction sets the smart card's CLK contact to 1. It is only operative if the smart card is powered up.

%CLR_C4 This instruction sets the smart card's C4 contact to 0.

%SET_C4 This instruction sets the smart card's C4 contact to 1. It is only operative if the smart card is powered up.

%CLR_C8 This instruction sets the smart card's C8 contact to 0.

%SET_C8 This instruction sets the smart card's C8 contact to 1. It is only operative if the smart card is powered up.

%IO_TO_C This instruction copies the state of the I/O contact into the C bit.

%C_TO_IO This instruction copies the level held in C to the smart card's I/O contact. It is only operative if the smart card is powered up.

%CLK_INC This instruction allows pulses to be generated on CLK. The total number of packets is indicated in A (0 to 255).
CLK is set to 0 for 10 ms then to 1 for 10 ms. At the end of the sequence, CLK is set to 0.

%CLK_IN_C8 This instruction allows eight pulse packets to be generated on CLK. The total number of packets is indicated in A (0 to 255).
CLK is set to 0 for 10 ms then to 1 for 10 ms. At the end of the sequence, CLK is set to 0.

%GET_D When the 3.68 MHz asynchronous clock is activated on CLK, this command reads eight bits from the I/O in asynchronous mode and classes them in A using the direct convention.
The configuration is 9,600 baud, 8 bits, even parity, one stop bit, 1s timeout.

%GET_I Same as GET_D, but the eight bits read are classed in A using the inverse convention.

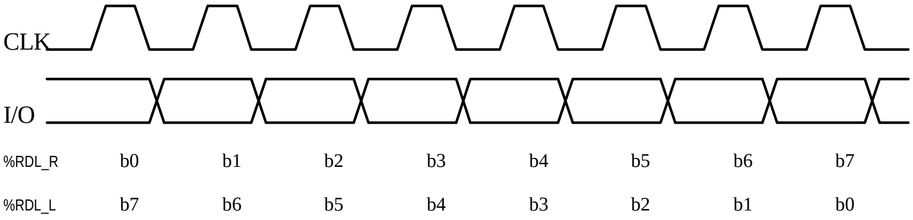
%SEND_D When the 3.68 MHz asynchronous clock is activated on CLK, this command writes the contents of A on the I/O in asynchronous mode using the direct convention.
The configuration is 9,600 baud, 8 bits, even parity, one stop bit, 1s timeout.

%SEND_I Same as SEND_D, but the eight bits are written to the I/O using the inverse convention.

%RDL_R This command reads eight bits and classes them in A with a right rotation.

%RDL_L This command is the same as RDL_R but with a left rotation

The sequence for these two commands is as follows:


CLK
I/O
%RDL_R b0 b1 b2 b3 b4 b5 b6 b7
%RDL_L b7 b6 b5 b4 b3 b2 b1 b0

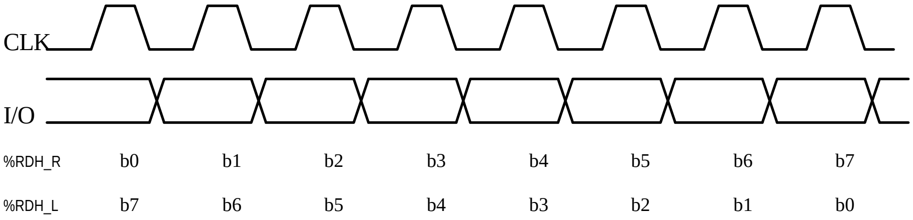
- CLK contact set to 0 for 10 µs.
- CLK contact set to 1 for 10 µs.

The I/O line is read 5µs **before** the CLK rising edge.

%RDH_R This command reads eight bits and classes them in A, with a right rotation.

%RDH_L This command is the same as RDH_R but with a left rotation

The sequence for these two commands is as follows:


CLK
I/O
%RDH_R b0 b1 b2 b3 b4 b5 b6 b7
%RDH_L b7 b6 b5 b4 b3 b2 b1 b0

- CLK contact set to 0 for 10 µs.
- CLK contact set to 1 for 10 µs.

The I/O line is read 5µs **after** the rising edge of the clock.
The first bit to be read is b0 of A. The last bit to be read is b7 of A.
At the end of the command, CLK is set to level 0.

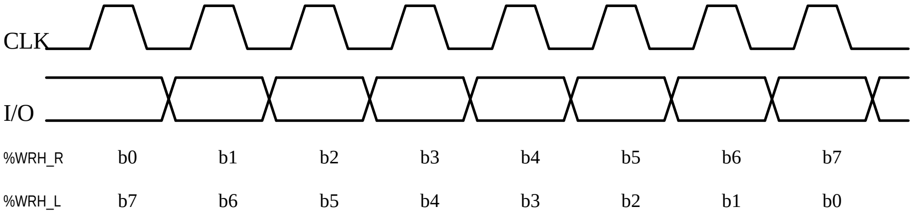
%WRH_R

This command writes the contents of A on the I/O contact, with a right rotation.

%WRH_L

This command is the same as WRH_R but with a left rotation (bit b7 of A is the first bit to be sent and bit b0 is the last).

The sequence for these two commands is as follows:


CLK
I/O
%WRH_R b0 b1 b2 b3 b4 b5 b6 b7
%WRH_L b7 b6 b5 b4 b3 b2 b1 b0

- CLK contact set to 0 for 10 µs.
- CLK contact set to 1 for 10 µs

The bit to be sent on I/O is set 5 µs **before** the rising edge of CLK.
Bit b0 of A is the first bit to be sent and bit b7 the last.
At the end of the command, CLK is set to level 0 and the I/O line is set to a high impedance level.

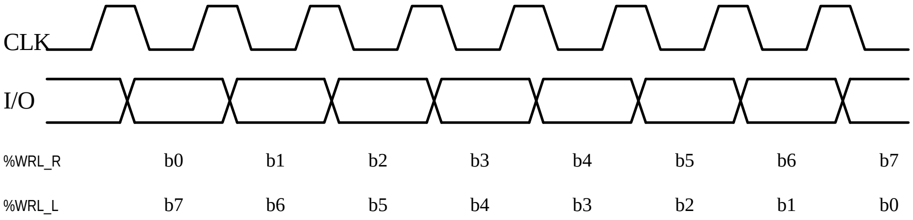
%WRL_R

This command writes the contents of A on the I/O contact, with a right rotation.

%WRL_L

Same as WRL_R but with a left rotation (b7 of A is the first bit to be sent and bit b0 is the last).

The sequence for these two commands is as follows:


CLK
I/O
%WRL_R b0 b1 b2 b3 b4 b5 b6 b7
%WRL_L b7 b6 b5 b4 b3 b2 b1 b0

- CLK contact set to 0 for 10 µs.
- CLK contact set to 1 for 10 µs.

The bit to be sent on I/O is set 5 µs **before** the falling edge of CLK.
Bit b0 of A is the first bit to be sent and b7 the last.
At the end of the command, CLK is set to level 0 and the I/O line is set to a high impedance level.

%RST_PUL

This command generates a logical pulse 1 for 10 µs on the RESET line and then resets the line to level 0.

%CLK_PUL

This command generates a logical pulse 1 for 10 µs on the CLK line and then resets the line level to 0.

%WAIT_US (TIME)

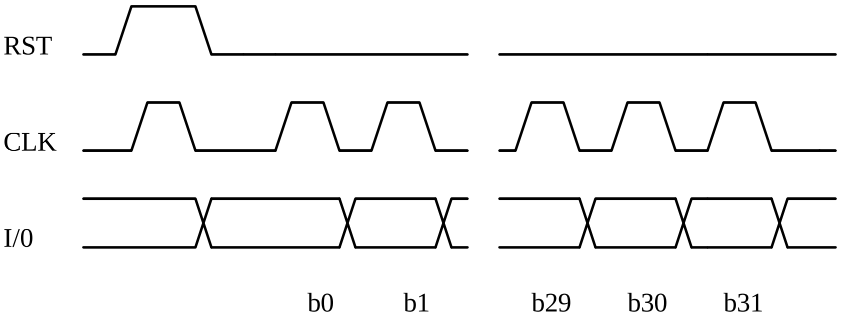
This command waits for the length of time specified in the TIME parameter.
The waiting time equals TIME * 10 µs.

%WAIT_MS (TIME)

This command waits for the length of time specified in the TIME parameter. The waiting time equals TIME* 1ms.

%RESET

This command executes the RESET synchronous card sequence with the GPM2K/8K protocol. GemCore returns the 32 bit ATR.
Executing the command interrupts the current program.



The RST and CLK signals are forced to level 0 for 10µs.
The CLK signal rises 5µs after the RST rising edge and remains at 1 for 40µs.
The RST signal falls 5µs after CLK and remains at 0 until the end of the sequence.
The CLK high and low levels remain constant for 10µs while the ATR is read, and the data is read 5µs after the rising edge of the CLK.
b0 is the least significant bit of the first byte returned by GemCore, b7 being the most significant bit.
b8 is the least significant bit of the second byte returned by GemCore, b15 being the most significant bit.
b16 is the least significant bit of the third byte returned by GemCore, b23 being the most significant bit.
b24 is the least significant bit of the third byte returned by GemCore, b31 being the most significant bit.

Example

GPM256 Read Command

Interpreted GPM256 source code:

```
                                ;Initialization:
                                ;CLA, INS, A1: not used
                                ;A2 = R7: location of first byte to be read
                                ;Lout = R3: number of byte to read

81       %CLR_C4                ;
71       %SET_CLK               ;Clears the internal counter
61       %CLR_CLK               ;

91       %SET_C4                ;
EF       MOV A,R7               ;Selects the first byte to be read
73       %CLK_INC8              ;

82       READ:RDL_BYTE          ;Reads one byte

F6       MOV@R0, A              ;Puts the byte in the output buffer
08       INC R0                 ;
DB FB    DJNZR3, READ           ;Reads the next byte

42       %RET_OK                ;Returns the result and adds 90h 00h when
                                ;all the bytes are read
```

Formatted GemCore Command

`16h CLA INS A1 A2 Lin <DATA IN> Lout Lcode <CODE>`

`CLA` = 00h	not used.
`INS` = B0h	not used. Only for card driver compatibility.
`A1` = 00h	not used.
`A2` = XXh	location of the first byte to be read.
`Lin` = 00h	no byte to be sent to the card.
`DATA IN`	not used, empty field.
`Lout` = YYh	number of bytes to be read.
`Lcode` = 0Ch	number of bytes in the code

`CODE` = 81h 71h 61h 91h EFh 73h 82h F6h 08h DBh FBh 42h

Command:

16h 00h B0h 00h XXh 00h YYh 0Ch 81h 71h 61h 91h EFh 73h 82h F6h 08h DBh FBh 42h

Response:

`S <YY bytes DATA READ> 90h 00h`

TERMINOLOGY

Abbreviations

ACK	Acknowledgement byte
ADH	Used in the **Read Memory** and **Write** Memory commands, ADH is the most significant byte of the 16-bit address of the first byte to be read or written.
ADL	Used in the **Read Memory** and **Write** Memory commands, ADL is the least significant byte of the 16-bit address of the first byte to be read or written.
AIA	Application Interface Area
APDU	Application Protocol Data Unit
BWI	Block Waiting time Integer
CB	Configuration Byte
CLK	Clock
CRC	Cyclic Redundancy Check
CWI	Character Waiting time Integer
DAT	DATa (being transmitted)
EDC	Error Detection Code
EEPROM	Electrically Erasable Programmable Read Only Memory
EOT	End Of Transmission
etu	elementary time unit
GBP	Gemplus Block Protocol
GBR	GemCore-Based Reader
I-Block	Information Block
IFSC	Information Field Size of the Card
IFSD	Information Field Size of the Device
ISO	International Standards Organization
LCD	Liquid Crystal Display
LEN	Length of the Data field
LN	Length of the message (command or status code)
LR	Length of APDU response
LRC	Result of an EXCLUSIVE OR (XOR) between the ACK, the LN and the MESSAGE characters.
N	Auxiliary card number
NAD	Node Address
OROS	Open Reader Operating System
PCB	Printed Circuit Board
PTS	Protocol Type Selection
R-Block	Receive Ready Block

ROS	Reader Operating System
S	Status
S-Block	Supervisory Block
SIA	System Interface Area
TLP	Transport Layer Protocol
TTL	Transistor-Transistor-Logic
XOR	EXCLUSIVE-OR operation

Glossary

APDU
Application Protocol Data Unit; data exchange protocol between a card and a reader. The APDU can be changed to ensure that it meets reader requirements for the user's site. For example, in the GCR400 card reader, the APDUMAXIN is 248 and the APDUMAXEXP is 251.

Baud
Rate of signals per second transmitted over a communication channel.

Block
Logically contiguous data memory that is allocated when requested for data field.

Command Layer
The command layer handles and interprets the GemCore V1.1-Based Reader commands.

Physical Layer
The physical layer handles the data transmission.

T = 0 Protocol
Character-oriented asynchronous half duplex transmission protocol.

T = 1 Protocol
Block-oriented asynchronous half duplex transmission protocol.

TA1
Interface byte that defines the rate of transmission.

Transport Layer
The transport layer handles message addressing, specifies the transmission type, and validates each transmission. The transport layer can use one of two protocols: the TLP224 protocol or the Gemplus Block Protocol.

INDEX

%C_TO_IO................................101
%CLK_IN_C8...........................101
%CLK_INC...............................101
%CLK_PUL 103
%CLR_C4 101
%CLR_C8 101
%CLR_CLK 100
%CLR_IO..................................100
%CLR_RST...............................100
%GET_D...................................101
%GET_I 101
%IO_TO_C................................101
%RET_ERR (ERR1,ERR2).........100
%RET_NOK (ERROR)..............100
%RET_OK 100
%RST_PUL 103
%SET_C4.................................101
%SET_C8..................................101
%SET_CLK...............................101
%SET_IO 100
%SET_RST 100
%VCC_OFF 100
%VCC_ON................................100
8051 assembler commands............96
AIA1... 6
AIA2... 6
APDU
  command format.......................40
  response format............ 41, 47, 60
Application.................................... 3
  deselecting to download a new
    application............................23
  management by GemCore........... 8
  running a specific 22
  size 8
Application modules 3
Auxiliary card type and selection...29
Buzzer
  defining frequency and duration.**78**
Card driver characteristics
  reading......................................31
Card interface commands..............25
Card power status............. 43, 49, 61
Card presence................... 43, 49, 61
Card Status – EMV Compliant 49
Card Status – Synchronous Card....61
Card Status – Transparent Mode....55
Card Status–Asynchronous Card ...43
Change Card Communication
  Parameters – Asynchronous Cards
 36
Change Transparent Mode
  Parameters 51
Command format..........................10
Command layer............................10

Commands
  8051 assembler 96
  card interface 25
  format.......................................15
  GemCore V1.21-Based Reader
    configuration.........................16
  GemPC410 ....**82**
  keypad and buzzer.....................76
  LCD...71
  reader management 62
  real time clock...........................**79**
  sending command APDUs... 40, 60
  sending ISO IN 39, 59
  sending ISO OUT 38, 58
*Communication protocol (T=0 or
  T=1)* 43, 49
Communication protocols................9
Configure SIO Line.......................17
CPU port
  reading......................................69
  writing......................................70
Define Main Card Type and Card
  Presence Detection....................27
Define Type and Select Auxiliary
  Card...29
Deselect Application Procedure.....23
Device handlers 3
Directory 31
Display Character 74
Display Character String 73
Erase Flash Memory 66
Exchange APDU – Asynchronous
  Cards 40
Exchange APDU– EMV Compliant
 46
Exchange Block – Transparent Mode
 54
Exchange With Synchronous Card
  (APDU) 60
External memory
  selecting pages..........................68
Firmware version
  reading......................................20
First Application Interface Area 6
Flash memory
  erasing 66
Format
  command 15
  command message 10
  GBP message 12
  response............................. 10, 15
  TLP224 message.......................11
GBP protocol................................12
GemCore applications.....................3
GemCore interpreter 96

  - card presence 97
  - card withdrawal 97
  - hexadecimal instruction codes 98
  - initialization 97
  - macro instructions 100
  - modified instructions 100
  - short circuit 97
- GemCore kernel 4
- GemCore V1.21-Based Reader commands 15
- GemCore V1.21-Based Reader configuration commands 16
- GemPC410 commands **82**
- GemPC410 LED Management **86**
- GemPC410 Power Down **85**
- GemPC410 Refresh **84**
- GemPC410 Set Timeout **83**
- GemPC410 Status **87**
- Gemplus Block Protocol 12
- I-Blocks (Information Blocks) 12
- IDATA 7
- Init the LCD 72
- Interface areas 6
- Interpreter functions 96
- ISO Input–Asynchronous Card 39
- ISO Output–Asynchronous Card 38
- Kernel 4
- Key press timeout
  - setting 77
- Keypad and buzzer commands 76
- LCD
  - commands 71
  - displaying a character 74
  - displaying a string 73
  - initializing 72
  - sending control commands 75
- Main card type and presence 27
- Memory
  - read/write protection 65
  - reading 63
  - selecting external memory pages 68
  - writing 64
- Memory cards
  - summary of the commands **91**
  - using the GemCore V1.1-Based Reader with **91**
- Memory mapping 7
- Memory organization 6
- Microprocessor cards
  - clock frequencies **88**
  - interface parameters **88**
  - TA1 parameter (communication rate) **88**
  - TB1 and TB2 parameters (Vpp option) **88**
  - TC1 parameter (extra guardtime) **89**
  - TD1 parameter (communication protocol) **90**
  - using the GemCore V1.1-Based Reader with **88**
- Operations 4
- Overriding operations 6
- Physical layer 14
- Power Down 26
- *Power supply value reading* 43, 49, 61
- Power Up – Asynchronous Cards 34
- Power Up – EMV Compliant 45
- Power Up – Synchronous Cards 57
- Power Up – Transparent Mode 53
- PPS management 34
- PROGRAM memory 7
- Protocol
  - command layer 10
  - GBP 12
  - TLP224 11
  - transport layer 11
- Protocol Type Selection 34
- Protocols 9
- R-Blocks (Receive Ready Block) 12
- RDH_L 102
- RDH_R 102
- RDL_L 102
- RDL_R 102
- Read CPU Port 69
- Read Data From Synchronous Card (ISO Out) 58
- Read Date and Time **80**
- Read Firmware Version 20
- Read Memory 63
- Reader management commands 62
- Reader operation mode
  - defining 18
- Real time clock
  - reading date and time **80**
  - updating date and time **81**
- Real time clock commands **79**
- Real-Time Emulation 4
- Release number
  - reading 31
- RESET 104
- Resetting the operating system 21
- Responses
  - delaying 19
- Restart 21
- Restart and Run Specified Application 22
- ROS command compatibility
  - disabling 18
- S-Blocks (Supervisory Block) 12
- Second Application Interface Area 6
- Select External Memory Page 68
- Send Data To Synchronous Card (ISO In) 59
- Send LCD Command 75
- SEND_D 101
- SEND_I 101
- Serial asynchronous protocol 14

Set Delay 19
Set Key Press Timeout 77
Set Mode.......................................18
Set Operating Mode 32
Set Reader In Halt Mode 24
SIA.. 6
SIO line settings
  defining.....................................17
Sound Buzzer................................78
Speed parameters....................43, 49
Status codes 10, 94
Synchronous card protocols...........96
System Interface Area 6
TLP224...11
Transport layer..............................11
*Type of card currently used*43, 49, 61
Types of cards handled
  reading......................................31
Update Date and Time **81**
User Application Interface Area
  AIA2 6
User DATA #0 6
User DATA #1 6
User DATA #2 6
WAIT US (TIME) 104
WAIT_MS (TIME).....................104
WRH_L.....................................103
WRH_R.....................................103
Write CPU Port.............................70
Write Memory 64
WRL_L 103
WRL_R 103
XDATA...7