

Mifare Application Programming Guide

MIFARE[®] Card Access Scheme

MF5A ActiveX Control Programming Guide

Mifare Application Protocol (MFAP)

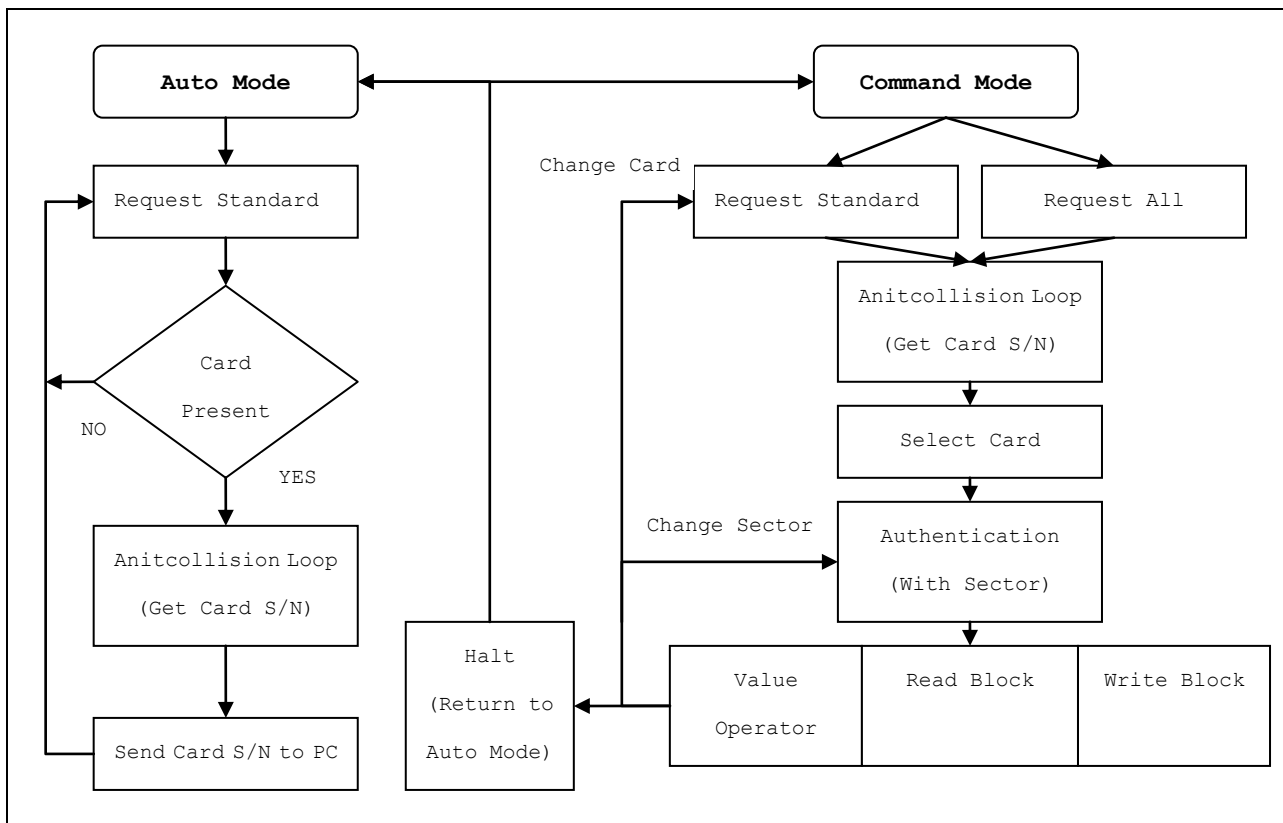
TM970014 REV.H

January 16 2009

Table of Contents

MIFARE® Card Access Scheme (MFAP Flow Chart)	3
MF5A ActiveX Control Programming Guide	5
Overview	5
The Properties, Methods & Events of MF5A ActiveX Control	6
RS232 Properties, Methods and Events	7
Mifare Access Properties, Methods and Events	8
GNetPlus Properties and Method (Common)	14
Mifare Application Protocol	15
MFAP Block:	15
MFAP Query Function Code Table (20h~2Fh)	16
ANNEX. A - GNetPlus Protocol Examples (ASCII Mode)	17
ANNEX. B - Error Code	18
ANNEX. C - WebISP - Firmware Upgrade Utility (Internet Version)	19
ANNEX. D - History	21

MIFARE® Card Access Scheme (MFAP Flow Chart)



Auto Mode / Command Mode:

When the Power is On, MF5 is in Auto Mode and automatically reads the card's Serial No. (when a MIFARE® card is within the reading range) and then sends it ^[note] to the Host. Auto Mode is halted to enter the Command Mode when Host sends the MF5 MIFARE® command for memory operation. MF5 will return to Auto Mode again if Host sends the Halt command.

[Note]:

1. Data Format that MF5 sends the card's Serial No. to Host in Auto Mode

<STX>CARD-SERIAL NO.<CR><ETX>

STX=02h, CR=0Dh, ETX=03h

2. The MF5 will send 0x1B (ESC) to Host when card removed. (May 23, 2008)

Request Standard / Request All:

When a MIFARE® card is within the reading range of MF5, send [Request Standard] command to establish communications between the card and MF5 (similar to the Polling). [Request All] command enables MF5 to communicate with multiple cards.

Anticollision Loop:

Get the Serial No. from the card that answers the request by [Anticollision] command in order to select the card for operation.

Select Card:

Select an individual card for operation by [Select] command. The operation can be made on only one card at one time. This is a necessary step if there are multiple cards within the reading range of MF5.

Authentication:

After the selection, use the corresponding keys for the Authentication procedure to access the selected Sector/Block of the card. After Authentication, memory operation may be performed.

Note: Use the [Save Key] command to pre-save the corresponding keys of each sector to MF5, which may reduce the risk that the keys being intercepted during communication.

Read/Write Block

As far as MIFARE® Standard Card (1K) is concerned, there are 4 Blocks in each Sector and 16-Byte memory in each Block. [Select] the Block and send [Read/Write] command for memory operation.

Value Operator

Arithmetic operation for electronic purse application. The Command Set includes:

- 1.Format
- 2.Read Value
- 3.Increase Value (with Transfer)
- 4.Decrease Value (with Transfer)

Like the Read/Write Block, select the Block and [Format] it before performing Value Operator command.

Halt:

Use either [Authentication] command to access other Sectors or [Halt] Command to terminate the operation of the card. In the latter case, the card must be withdrawn from the reading range of MF5 in order to perform the next operation.

Summary:

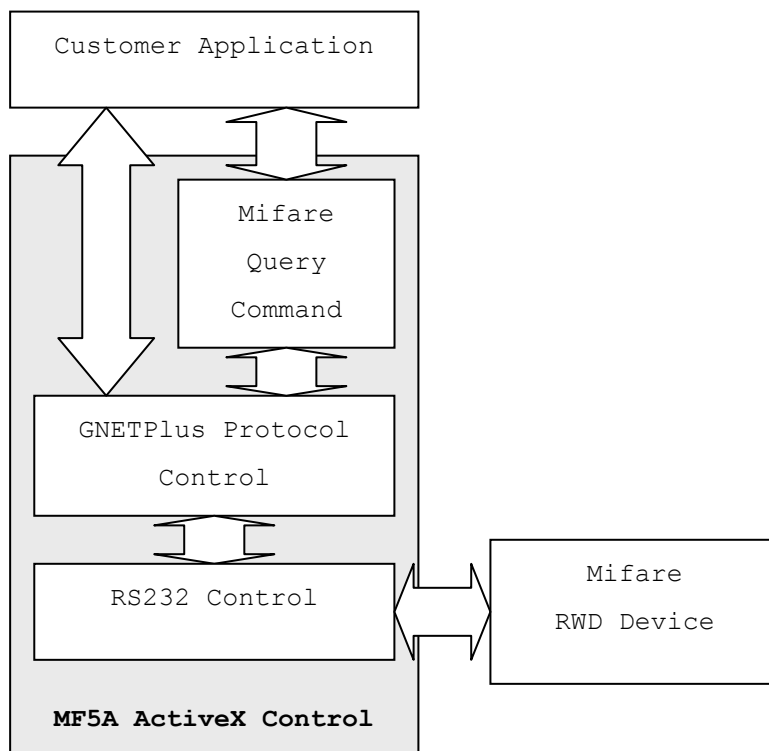
1. It takes 3 steps to pick out a card for operation - Request, Anticollision and Select.
2. An Authentication command has to be carried out before any memory operation.
3. If any mistake occurs during operation, go back to step Request and operate again.
4. Pre-save the Keys of each Sector to MF5 to avoid the risk of interception.
5. The card must be pulled out of the reading range of MF5 after Halt command.

MF5A ActiveX Control Programming Guide

Overview

With MF5A ActiveX Control, it is not necessary to study how to communicate with MF5, neither to write any program for communication protocol to perform the operation of MIFARE® card. MF5 Active Control is automatically registered to the computer when MF5 Demo Software is installed. The file name is MF5Ax ActiveX Control Module.

Note: Reference can be also made to the VB Source Code of MF5 Demo Software in CD.



The Properties, Methods & Events of MF5A ActiveX Control

Properties:

- CommPort
- Settings
- PortOpen
- mfCurrentClass / mfCurrentClassStr
- GNetErrorCode / GNetErrorCodeStr
- Busy
- GetVersion
- CurrentAddr

Methods:

- EnumCommPort
- SetSlaveAddr
- mfRequest / mfRequestEx
- mfAnticollision
- mfSelectCard / mfSelectCardEx
- mfAuthenticate / mfAnticollision2
- mfRead / mfReadEx / mfReadHex
- mfWrite / mfWriteEx / mfWriteHex
- mfGetValue / mfGetValueEx
- mfSetValue / mfSetValueEx
- mfValueSet
- mfHalt
- mfSaveKey
- mfAccessCondition
- Polling
- Reset

Events:

- OnPort
- OnCardEvent

RS232 Properties, Methods and Events

Property	CommPort
Description	Sets and returns the Comm Port number (Default COM1)
Syntax	<i>Object.CommPort [= Integer]</i>
Parameter	Integer , COM PORT number , 1=COM1, 2=COM2 ...

Property	Settings
Description	Sets and returns the Settings value (Default 19200,N,8,1)
Syntax	<i>Object.Settings [= String]</i>
Parameter	<i>String; [Baudrate][, Parity][, Data Bits][, Stop Bits]</i>
VB Example	<pre>MF5Ax1.Settings = "19200,N,8,1" MF5Ax1.Settings = "19200" MF5Ax1.Settings = "N,8,1"</pre>

Property	PortOpen
Description	Sets and returns the open status of the Comm Port
Syntax	<i>Object.PortOpen [= Boolean]</i>
Parameter	Boolean, TRUE=Port Open, FALSE=Port Close

Method	EnumCommPort
Description	List all available Comm Ports (including the Virtual ones)
Syntax	<i>String = Object.EnumCommPort(short index)</i>
Parameter	Return String, COM Port Name, (Examples "COM1"...) Index, 0~255
VB Example	<pre>Dim szPort as String, i as integer For i = 0 to 255 szPort = MF5x1.EnumCommPort(i) If szPort <> vbNullString Then ... Else ' If szPort is vbNullString , to exit the for loop Exit For End If Next i</pre>

Event	OnPort
Description	When a USB Virtual Comm Port is removed during connection, this event will inform the program that it is removed and cannot be used any more.
Syntax	<pre>Private Sub MF5x1_OnPort(ByVal Action As MF5AXLib.CommPortEventConstants, ByVal CommPort As Integer) Select Case Action Case comEvPlugin: ShowMsg "COM" & CommPort & " is plug-in" Case comEvRemove: ShowMsg "COM" & CommPort & " is remove" Case comEvRemoveClosed: ShowMsg "COM" & CommPort & " is remove & closed" End Select End Sub</pre>

Mifare Access Properties, Methods and Events

Property	mfCurrentClass (Read Only)
Description	Return current Card Class number
Syntax	<i>Short = Object.mfCurrentClass</i>
Parameter	

Property	mfCurrentClassStr (Read Only)
Description	Return current Card Class description
Syntax	<i>String = Object.mfCurrentClassStr</i>
Parameter	

Property	GNetErrorCode (Read Only).
Description	Return last error number
Syntax	<i>Short = Object.GNetErrorCode</i>
Parameter	

Property	GNetErrorCodeStr (Read Only)
Description	Return last error description.
Syntax	<i>String = Object.GNetErrorCodeStr</i>
Parameter	

Method	mfGetValue / mfGetValueEx mfSetValue / mfSetValueEx
Description	Sets and returns block value.
Syntax	<i>Boolean = Object.mfGetValue(Block, Value)</i> <i>Boolean = Object.mfSetValue(Block, Value)</i>
Parameter	Block, short type for block number. Value, long type for block value.
VB Examples	<pre>(Format Block 1 and Default set to 100) Result = MF5x1.mfSetValue(1 , 100) (Copy the Block 1 value to Block 2) Dim lValue as Long If MF5x1.mfGetValue(0, lValue) Then Result = MF5x1.mfSetValue(2 , lValue) End If</pre>

Method	mfRequest
Description	Send Request command and return the Card Class number.
Syntax	<i>Short = Object.mfRequest</i>
Parameter	Return card class number
Note	Use the method to check card into reader RF range.

Method	mfAnticollision
Description	Send Anticollision command and return the Card S/N (Serial Number).
Syntax	<i>Long = Object.mfAnticollision</i>
Parameter	Return card S/N, It is a long data type.

Method	mfSelectCard
Description	Send Select Card command and return Card memory size (unit kbits)
Syntax	<i>Short = Object.mfSelectCard(Long CardSN)</i>
Parameter	CardSN, card serial number, a long data type
	The Card S/N request from method mfAnticollision.

Property	mfAuthenticate
Description	Select a Sector and Authenticate with key
Syntax	<i>Boolean = Object.mfAuthenticate(Sector, KeyType, szKey)</i>
Parameter	Sector : Short, for Sector number KeyType: Short, for KEY_A(60h) or KEY_B(61h) szKey: HEX String, 12 Hex Codes

Method	mfRead
Description	Read a Block data from Mifare Card.
Syntax	<i>Boolean = Object.mfRead(Block, pBuffer, nSize)</i>
Parameter	<i>Block: Short, Block number</i> <i>pBuffer: Long, Buffer Address pointer.</i> <i>nSize: Short, Buffer size, Max.16 Bytes.</i>
VB Examples	(To Read Block 2 Data from card) Dim blkBuffer(0 to 15) as BYTE, bResult as Boolean bResult = MF5x1.mfRead(2, VarPtr (blkBuffer), lenB(blkBuffer)) <i>Note: If program by VB, you can use the "VarPtr" to got the variable long address pointer.</i>

Method	mfWrite
Description	Write a Block data to Mifare Card.
Syntax	<i>Boolean = Object.mfWrite(Block, pBuffer, nSize)</i>
Parameter	<i>Block: Short, Block Number</i> <i>pBuffer: Long, Buffer Address pointer.</i> <i>nSize: Short, Buffer Size, Max.16 Bytes.</i>
VB Examples	(Write a string to Block 2) Dim szName as String, bResult as Boolean szName = "GIGA-TMS INC." bResult = MF5x1.mfWrite(2, VarPtr(szName), len(szName)) <i>Note: If programming by VB, you can use the "VarPtr" to get the variable long address pointer.</i>

Method	mfValueSet
Description	Operate the value block for increase or decrease in old value.
Syntax	<i>Boolean = Object.mfValueSet(Block, Opt, Value)</i>
Parameter	<i>Block : Short, Block Number</i> <i>Opt : Short, MF_INC(Increase) or MF_DEC(Decrease)</i> <i>Value : Long, operate value.</i>

Method	mfHalt
Description	To halt the current selected card, this card must leave the reader's RF range.
Syntax	<i>Boolean = Object.mfHalt</i>
Parameter	

Method	mfAutoMode
Description	To halt the current selected card, this card must leave the reader's RF range.
Syntax	<i>Boolean = Object.mfAutoMode (Boolean)</i>
Parameter	TRUE=Enable, FALSE=Disable

Method	mfSaveKey
Description	Save the sector key to reader.
Syntax	<i>Boolean = Object.mfSaveKey (KeyType, nSector, szKey)</i>
Parameter	<i>KeyType : Short, For KEY_A or KEY_B</i> <i>nSector : Short, For Sector number</i> <i>szKey : String, 12 HEX Codes.</i>

Method	mfAccessCondition / mfAccessConditionEx																																																																																																																		
Description	Change the card access condition bits																																																																																																																		
Syntax	<pre>Boolean = Object.mfAccessCondition(szKeyA, szKeyB, CB0, CB1, CB2, CB3)</pre> <pre>Boolean = Object.mfAccessConditionEx(szKeyA, szKeyB, CB0, CB1, CB2, CB3, GPB)</pre>																																																																																																																		
Parameter	<p><i>szKeyA</i> : String, Key A, 12 HEX Codes.</p> <p><i>szKeyB</i> : String, Key B, 12 HEX Codes.</p> <p><i>CB0/CB1/CB2/CB3</i>: Short, Block0~3 access bits.</p> <p><i>GPB</i> : MAD General purpose byte.</p>																																																																																																																		
Remark	<p>Access condition for data blocks (CB0~CB2)</p> <table border="1"> <thead> <tr> <th>CBn</th> <th>read</th> <th>write</th> <th>incr</th> <th>decr</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>KEY A B</td> <td>KEY A B</td> <td>KEY A B</td> <td>KEY A B</td> </tr> <tr> <td>1</td> <td>KEY A B</td> <td>KEY B</td> <td>Never</td> <td>Never</td> </tr> <tr> <td>2</td> <td>KEY A B</td> <td>Never</td> <td>Never</td> <td>Never</td> </tr> <tr> <td>3</td> <td>KEY A B</td> <td>KEY B</td> <td>KEY B</td> <td>KEY A B</td> </tr> <tr> <td>4</td> <td>KEY A B</td> <td>Never</td> <td>Never</td> <td>KEY A B</td> </tr> <tr> <td>5</td> <td>KEY B</td> <td>Never</td> <td>Never</td> <td>Never</td> </tr> <tr> <td>6</td> <td>KEY B</td> <td>KEY B</td> <td>Never</td> <td>Never</td> </tr> <tr> <td>7</td> <td>Never</td> <td>Never</td> <td>Never</td> <td>Never</td> </tr> </tbody> </table> <p>Access condition for the Sector Trailer (CB3)</p> <table border="1"> <thead> <tr> <th rowspan="2">CB3</th> <th colspan="2">KEY_A</th> <th colspan="2">ACCESS BITS</th> <th colspan="2">KEY_B</th> </tr> <tr> <th>read</th> <th>write</th> <th>read</th> <th>write</th> <th>read</th> <th>write</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Never</td> <td>KEY A</td> <td>KEY A</td> <td>Never</td> <td>KEY A</td> <td>KEY A</td> </tr> <tr> <td>1</td> <td>Never</td> <td>KEY B</td> <td>KEY A B</td> <td>Never</td> <td>Never</td> <td>KEY B</td> </tr> <tr> <td>2</td> <td>Never</td> <td>Never</td> <td>KEY A</td> <td>Never</td> <td>KEY A</td> <td>Never</td> </tr> <tr> <td>3</td> <td>Never</td> <td>Never</td> <td>KEY A B</td> <td>Never</td> <td>Never</td> <td>Never</td> </tr> <tr> <td>4</td> <td>Never</td> <td>KEY A</td> <td>KEY A</td> <td>KEY A</td> <td>KEY A</td> <td>KEY A</td> </tr> <tr> <td>5</td> <td>Never</td> <td>Never</td> <td>KEY A B</td> <td>KEY B</td> <td>Never</td> <td>Never</td> </tr> <tr> <td>6</td> <td>Never</td> <td>KEY B</td> <td>KEY A B</td> <td>KEY B</td> <td>Never</td> <td>KEY B</td> </tr> <tr> <td>7</td> <td>Never</td> <td>Never</td> <td>KEY A B</td> <td>Never</td> <td>Never</td> <td>Never</td> </tr> </tbody> </table> <p>Note: "KEY A B" means KEY A or KEY B</p>	CBn	read	write	incr	decr	0	KEY A B	KEY A B	KEY A B	KEY A B	1	KEY A B	KEY B	Never	Never	2	KEY A B	Never	Never	Never	3	KEY A B	KEY B	KEY B	KEY A B	4	KEY A B	Never	Never	KEY A B	5	KEY B	Never	Never	Never	6	KEY B	KEY B	Never	Never	7	Never	Never	Never	Never	CB3	KEY_A		ACCESS BITS		KEY_B		read	write	read	write	read	write	0	Never	KEY A	KEY A	Never	KEY A	KEY A	1	Never	KEY B	KEY A B	Never	Never	KEY B	2	Never	Never	KEY A	Never	KEY A	Never	3	Never	Never	KEY A B	Never	Never	Never	4	Never	KEY A	KEY A	KEY A	KEY A	KEY A	5	Never	Never	KEY A B	KEY B	Never	Never	6	Never	KEY B	KEY A B	KEY B	Never	KEY B	7	Never	Never	KEY A B	Never	Never	Never
CBn	read	write	incr	decr																																																																																																															
0	KEY A B	KEY A B	KEY A B	KEY A B																																																																																																															
1	KEY A B	KEY B	Never	Never																																																																																																															
2	KEY A B	Never	Never	Never																																																																																																															
3	KEY A B	KEY B	KEY B	KEY A B																																																																																																															
4	KEY A B	Never	Never	KEY A B																																																																																																															
5	KEY B	Never	Never	Never																																																																																																															
6	KEY B	KEY B	Never	Never																																																																																																															
7	Never	Never	Never	Never																																																																																																															
CB3	KEY_A		ACCESS BITS		KEY_B																																																																																																														
	read	write	read	write	read	write																																																																																																													
0	Never	KEY A	KEY A	Never	KEY A	KEY A																																																																																																													
1	Never	KEY B	KEY A B	Never	Never	KEY B																																																																																																													
2	Never	Never	KEY A	Never	KEY A	Never																																																																																																													
3	Never	Never	KEY A B	Never	Never	Never																																																																																																													
4	Never	KEY A	KEY A	KEY A	KEY A	KEY A																																																																																																													
5	Never	Never	KEY A B	KEY B	Never	Never																																																																																																													
6	Never	KEY B	KEY A B	KEY B	Never	KEY B																																																																																																													
7	Never	Never	KEY A B	Never	Never	Never																																																																																																													

Remark:

If KEY_B may be read (all gray marked lines) the memory space for KEY_B is used for data storage and it shall not be used for authentication because all further memory access operations will fail.

Method	mfGetAccessCondition
Description	The reader sends the event to the host when the card inserted or removed.
Syntax	Boolean = Object.mfGetAccessCondition(*CB0, *CB1, *CB2, *CB3, *GPB)
Parameter	CB0, CB1, CB2, CB3: Point to a short variable, that receives the CB0~CB3 access conditions value. GPB: Point to a short variable, that receives the MAD GPB value.

Remark (For the Access Conditions) :

Mifare Standard 1K: For the 16 sectors the access conditions can be set individually for a data area sized one block.

Mifare Standard 4K: For the first 32 sectors the access conditions can be set individually for a data area sized one block. For the last 8 sectors the access conditions can be set individually for a data area sized 5 blocks.

Event	OnCardEvent (CardEventConstants iEvent)
Description	The reader sends the event to the host when the card inserted or removed. (For Auto Mode only)
Syntax	
Parameter	iEvent, short type: MF_CARD_REMOVE = Card Remove MF_CARD_PRESENT = Card Insert
VB Examples	<pre>Private Sub MF5x1_OnCardEvent(ByVal iEvent As MF5AXLib.CardEventConstants) Select Case iEvent Case CardEventConstants.MF_CARD_PRESENT ShowMsg "Card Present" Case CardEventConstants.MF_CARD_REMOVE ShowMsg "Card Remove" End Select End Sub</pre>

GNetPlus Properties and Method (Common)

Property	Busy (Read Only)
Description	Return communication status.
Syntax	Boolean = Object.Busy
Parameter	

Property	CurrentAddr (Read/Write Only)
Description	Sets or return current machine communication ID.
Syntax	Object.CurrentAddr [=Integer]
Parameter	

Method	SetSlaveAddr
Description	Return current machine firmware version.
Syntax	Boolean = Object.SetSlaveAddr(Integer Address)
Parameter	

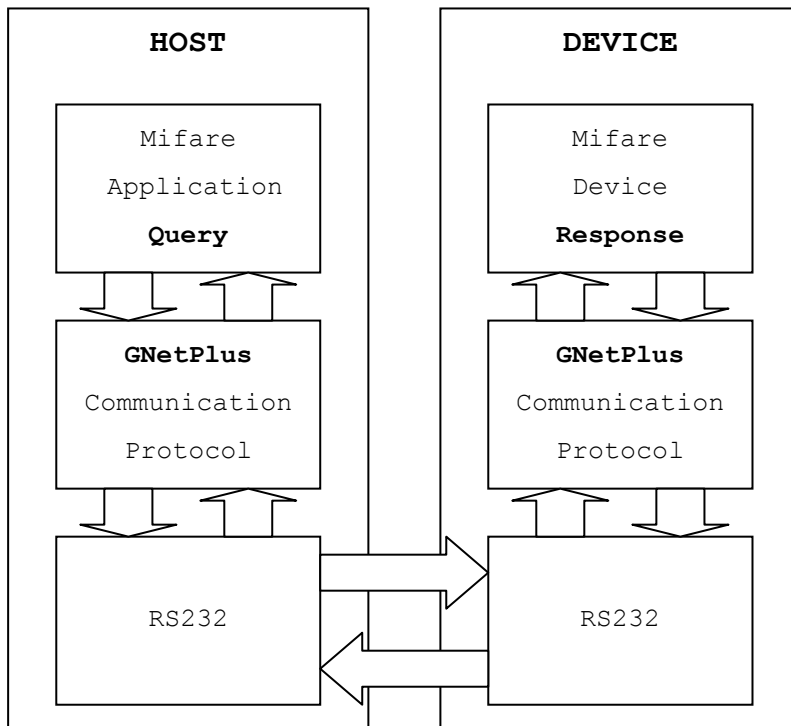
Method	GetVersion
Description	Return current machine firmware version.
Syntax	String = Object.GetVersion
Parameter	

Method	Polling
Description	Poll all machines with ID.
Syntax	<i>Boolean = Object.Polling(short ID)</i>
Parameter	ID, short type, 0~255
Remake	Any command must to poll machine first. If Id=0, Any machine id will response the command.

Method	Reset
Description	Reset current machine.
Syntax	Boolean = Object.Reset()
Parameter	

Mifare Application Protocol

MFAP Block:



Please refer to another document "**GNetPlus® Communication Protocol**"(GNetPlus.pdf) first.

MFAP Query Function Code Table (20h~2Fh)

Desc	Query (Master/Host)			Response (Slave/Device)		
	Func	Len	Data Bytes	Func	Len	Data Bytes
Request	20h	0		ACK	2	Card Class Type (Integer)
Anti-collision	21h	0		ACK	4	Card Serial Number (Long)
Select Card	22h	4	Card Serial Number (Long)	ACK	1	Card Memory Size
Authenticate	23h	2	KEY_TYPE ¹ + SECTOR	ACK	0	
Read a Block	24h	1	Block# ²	ACK	16	Block Data
Write a Block	25h	17	Block# + Block Data	ACK	0	
Set Value	26h	6	Block# + OPT ³ + Value (Long)	ACK	0	with Transfer
Read Value	27h	1	Block#	ACK	4	Value (Long)
Create a Value Block	28h	1	Block#	ACK	0	
Access Condition	29h	16	KEYA ⁴ +CB0+CB1+CB2+CB3+KEYB ⁴	ACK	0	
Access Condition	29h	17	KEYA ⁴ +CB0+CB1+CB2+CB3+GPB+KEYB ⁴	ACK	0	
Halt	2Ah	0		ACK	0	
Save Key	2Bh	8	KEY_TYPE + SECTOR + KEY ⁴	ACK	0	
Get Second S/N	2Ch	0		ACK	4	Card second S/N (Long)
Get Access Condition	2Dh	0		ACK	5	CB0, CB1, CB2, CB3, GPB
Authenticate + Key	2Eh	8	KEY_TYEP + SECTOR + KEY ⁴	ACK	0	
RequestAll	2Fh	0		ACK	2	Card Class Type (Integer)
SetValueEx	0x32	6	Block# + OPT ³ +Value (Long)	ACK	0	without Transfer
Transfer	0x33	1	Block#			
Restore	0x34	1	Block#			
GetSector	0x3D	2	AID ⁵ #	ACK	1	Return Sector# By AID
RF Power On/Off	0x3E	1	0=OFF, 1=ON	ACK	0	
AutoMode	3F	1	0:Disable, 1:Enable	ACK		Current Mode

Note:

1. KEY_TYPE: KEY_A=60h, KEY_B=61h
2. Block#: Block Number
3. OPT: Increase=C1h, Decrease =C0h
4. KEYA, KEYB, KEY: KEY VALUE, Size=6 Bytes. Example: (LSB) 2C 1B 30 26 3A B7 (MSB)
5. AID (Application Id) is for Multi Application.

Remark:

1. Authenticate (23h): Must save key (2Bh) before authenticating the sector.
2. Get Second S/N (2Ch): For Mifare Ultra-Light card only.
3. Set Value (26h): Must create a value block(28h) before setting values.
4. **Access Condition: About CB0, CB1, CB2 and CB3, please see page 11.**

ANNEX. A - GNetPlus Protocol Examples (ASCII Mode)

The ASCII Mode is easy to send command from HyperTerminal (or other communication Terminal) to Reader, and easy to learn the MIFARE® operation.

GNetPlus Communication Package

Mode	Header	Address ³	Function	Byte Count	DATA BYTES	Error Check	Trailer
ASCII	Colon ²	2BYTE	2BYTE	2BYTE	STRING	None	CR
BINARY¹	SOH	1BYTE	1BYTE	1BYTE	BINARY	CRC16	None

Note:

1. About Binary Mode , Please see the GNetPlus Communication Protocol.
2. Colon = 3Ah = ':'
3. Address = Reader ID

ASCII Mode Examples:

```

:002000<CR>          HOST      :Send Request (20h)
:0006020400          READER    :Response 400h (Card Class)

:002100<CR>          HOST      :Send Anti-Collision (21h)
:000604CB4540A2      READER    :Response Card S/N=CB4540A2 (MSB First)

:002204CB4540A2<CR>  HOST      :Send Select Card (22h) with Card S/N
:00060108            READER    :Response ACK and Card Memory Size (8K bits)

:0023026000<CR>     HOST      :Send Authenticate(23h),KEY_A=60h Sector=00h
:00060100            READER    :Response ACK and No Error (00h)

:00240100<CR>       HOST      :Read a Block (24h), Block Number=00h
                        READER    :Response Block 0 Data
:000610A24045CB6C88040046DAF20532363031

:002A00<CR>          HOST      :Send Halt Command (2Ah) for Selected Card
:00060100            READER    :Response ACK and No Error (00h)

```

P.S:

1. ACK = 06h
2. NAK = 15h

ANNEX. B - Error Code

Error Code	Description	Note
03h	EMPTY	
04h	AUTHENTICATE ERROR	
09h	KEY ERROR	
0Ah	NOT AUTHENTICATE ERROR	
0Eh	TRANSFER ERROR	
0Fh	WRITE ERROR	
10h	INC ERROR	
11h	DEC ERROR	
12h	READ ERROR	
1Ch	ACCESS TIMEOUT	
1Fh	NO TAG ERROR	
27h	WRONG PARAMETER VALUE	
2Dh	HOST AUTHENTICATE ERROR	
2Fh	WRONG DESKEY	
33h	DES KEY LOAD ERROR	
E0h	GNET COMMAND DENY	
E1h	GNET COMMAND ILLEGAL	
E2h	GNET COMMAND OVERRUN	
E3h	GNET PACKAGE CRC ERROR	
E4h	GNET OUT OF MEMORY	
E5h	GNET OUT OF FRAME	
E6h	GNET UNKNOW COMMAND	

Example (ASCII Mode)

:00**2000**<CR>

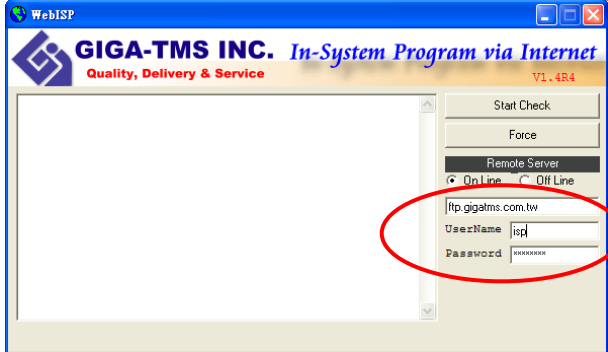
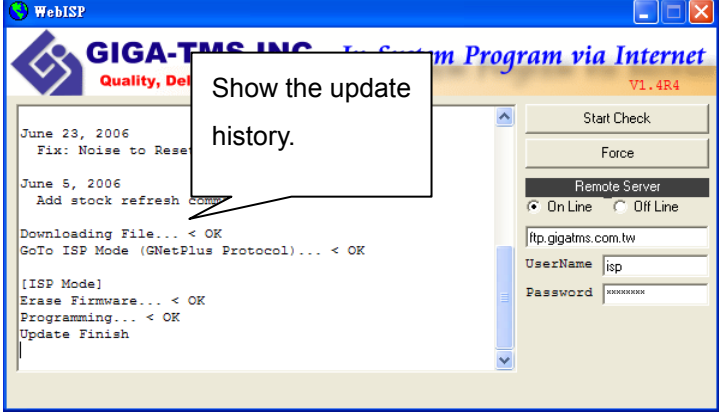
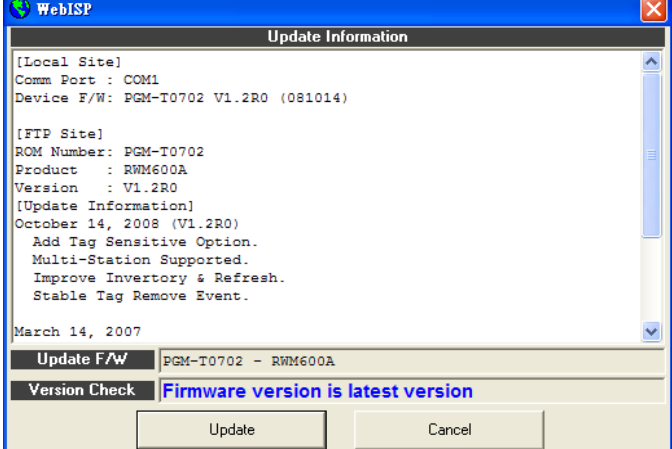
Host Send Request Command (20h)

:001501**1F**

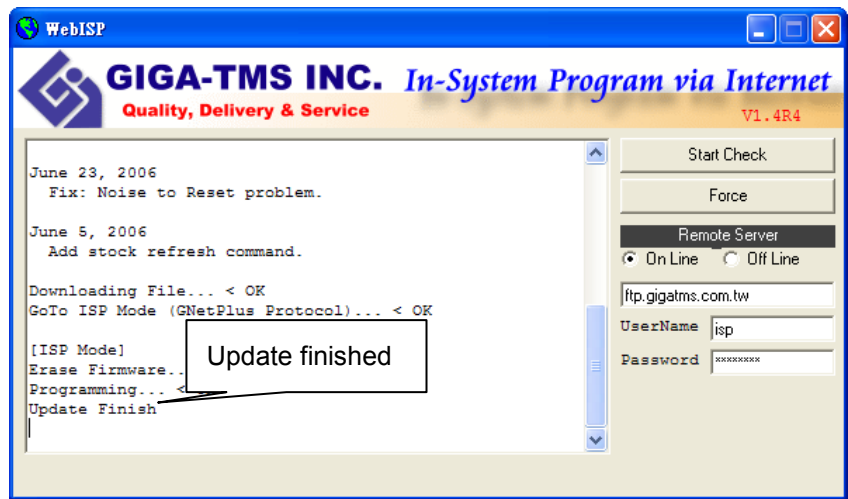
Reader Response NAK with Error Code (1Fh=No Tag on Reader)

ANNEX. C - WebISP - Firmware Upgrade Utility (Internet Version)

Install the WebISP (include in CD-ROM) in your Windows System first (It may need to reboot your system) and follow the steps as below: (First of all, you need to connect the reader or programmer to PC, and make sure they were power-on)

<p>Step 1: Input your account (UserName and Password)</p> <p>Note: Contact us to get your account when needed.</p>	
<p>Step 2: Click [Start Check] to automatically check the firmware version from our FTP server.</p> <p>Note:</p> <ol style="list-style-type: none"> 1. The WebISP will auto scan all Comm ports to search the reader or programmer. 2. The WebISP will show the [Update Information] and list the update history. 	
<p>Step 3: If your reader's or programmer's firmware out of date, then WebISP will prompt you to update the firmware. Click [Update] to begin Updating the firmware.</p>	

Step 4: Wait for the updating to finish. And repeat step 2 to update other readers or programmers.



ANNEX. D - History

REV.A October 17, 2003

1. Initial MF5

REV.B May 11, 2004

1. Append GNetPlus ASCII Mode Instruction. (ANNEX.A)
2. Append GNetPlus Error Code Instruction. (ANNEX.B)

REV.C June 7, 2004

1. Append WinISP51 instruction. (ANNEX. C)
2. New ActiveX method [mfAccessConditionEX](#). (Page 11)
3. New ActiveX Method [mfGetAccessCondition](#). (Page 12)
4. New Query Function Code 2Dh: Get Access Bit (Page 15)
5. New Query Function Code 29h: Set Access Bit to support MAD-GPB when query **parameter length=17**. (Page 15)
6. Firmware Version:
MF5 : PGM-T0499 V1.1R0 (040607)
PCR310/PRW106 : PGM-T0487 V1.3R1 (060607)

REV.D June 30, 2004

1. Remark the EY_B limits (Page 11).
2. Add WebISP Instruction (ANNEX C)

REV.E December 19, 2006

1. Add a command to Disable or Enable the Auto Mode.

REV.F May 20, 2008

1. Add SetValueEx command and without Transfer (Page 15)
2. Add Transfer Command (Page 15)
3. Add Restore Command (Page 15)
4. Add GetSector Command with AID# (Page 15)
5. Add RF Power On/Off command (Page 15)

REV.G November 26, 2008

1. Fix Data Format that MF5 sends without <LF> (Page 2)

REV.H January 16 2009

1. Change MF5 ActiveX Control Programming Guide to MF5A ActiveX Control Programming Guide.

Comparison	
MF5 ActiveX	MF5Ax ActiveX
Properties	
Baudrate	Settings
mfLastError / mfLastErrorStr	GNetErrorCode / GNetErrorCodeStr
mfValue / mfValueEx	mfGetValue / mfGetValueEx (Method) mfSetValue / mfSetValueEx (Method)
gnetBusy	Busy
gnetVersion	GetVersion
gnetMachineld	CurrentAddr SetSlaveAddr (Method)
Methods	
gnetPolling	Polling
gnetReset	Reset
Events	
PortRemoved	OnPort
CardPresent	OnCardEvent