

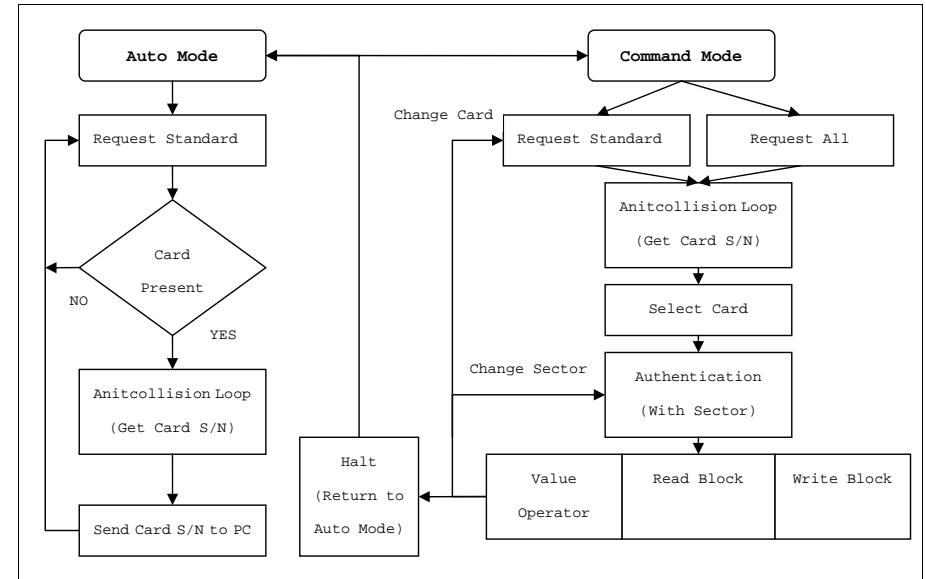
MF5 Programming Guide

MIFARE® Card Access Scheme

MF5 ActiveX Control Programming Guide

MF5 Communication Protocol

MIFARE® Card Access Scheme (MF5 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]: Data Format that MF5 sends the card's Serial No. to Host in Auto Mode

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

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

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
- 4.Decrease Value

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:

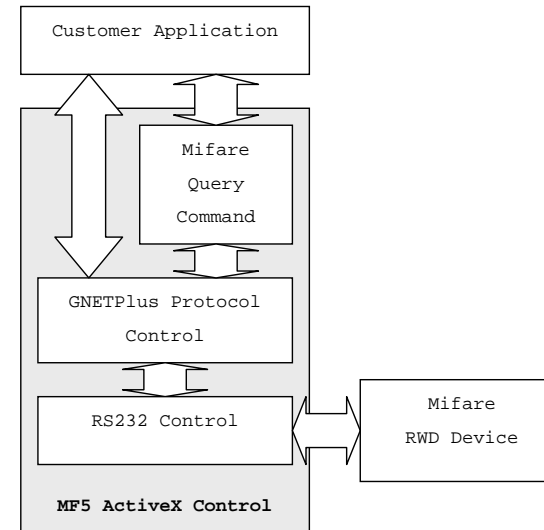
1. It takes 3 steps to pick out a card for operation - Request, Anticollision and Select.
2. An Authentication command has to be carried out before any memory operation.
3. If any mistake occurs during operation, go back to step Request and operate again.
4. Pre-save the Keys of each Sector to MF5 to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

MF5 ActiveX Control Programming Guide

Overview

With MF5 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 MF5x 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 MF5 ActiveX Control

Properties:

CommPort
 Baudrate
 PortOpen
 mfCurrentClass / mfCurrentClassStr
 mfLastError / mfLastErrorStr
 mfValue / mfValueEx
 gnetBusy
 gnetVersion
 gnetMachineld

Methods:

EnumCommPort
 mfRequest / mfRequestEx
 mfAnticollision
 mfSelectCard / mfSelectCardEx
 mfAuthenticate / mfAnticollision2
 mfRead / mfReadEx / mfReadHex
 mfWrite / mfWriteEx / mfWriteHex
 mfValueSet
 mfHalt
 mfSaveKey
 mfAccessCondition
 gnetPolling
 gnetReset

Events:

PortRemoved
 CardPresent

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	Baudrate
Description	Sets and returns the Baudrate value (Default 19200)
Syntax	<i>Object.Baudrate [= Integer]</i>
Parameter	Integer; Baudrate value (examples 19200, 9600, 4800...)

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	PortRemoved
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	
Parameter	

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

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

Property	mfValue (Read/Write)
Description	Sets and returns block value.
Syntax	<i>Object.mfValue(Block)[= Value]</i>
Parameter	Block, short type for block number. Value, long type for block value.
VB Examples	(Format Block 1 and Default set to 100) MF5x1.mfValue(1) = 100 (Copy the Block 1 value to Block 2) Dim newValue as Long newValue = MF5x1.mfValue(1) MF5x1.mfValue(2) = newValue

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	Boolean = Object.mfAuthenticate(Sector, KeyType, szKey)
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 pBuffer: Long, Buffer Address pointer. 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 pBuffer: Long, Buffer Address pointer. 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 Opt : Short, MF_INC(Increase) or MF_DEC(Decrease) 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	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 nSector : Short, For Sector number szKey : String, 12 HEX Codes.</i>

Method	mfAccessCondition																																																																																																																						
Description	Change the card access condition bits																																																																																																																						
Syntax	<i>Boolean = Object.mfAccessCondition(szKeyA, szKeyB, CB0, CB1, CB2, CB3)</i>																																																																																																																						
Parameter	<i>szKeyA : String, Key A, 12 HEX Codes.</i> <i>szKeyB : String, Key B, 12 HEX Codes.</i> <i>CB0/CB1/CB2/CB3: Short, Block0~3 access bits.</i>																																																																																																																						
Remark	<p>For Data block (block 0~2, CB0~CB2) access bits</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>For Block 3 (Sector Trailer, CB3) access bits:</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																																																																																																																	

Event	CardEvent(short nEvent)
Description	The reader sends the event to the host when the card inserted or removed.
Syntax	
Parameter	nEvent, short type: 0 = Card Remove 1 = Card Insert

GNetPlus Properties and Method (Common)

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

Property	gnetMachineID
Description	Sets or return current machine communication ID.
Syntax	Object.gnetMachineId [=Integer]
Parameter	

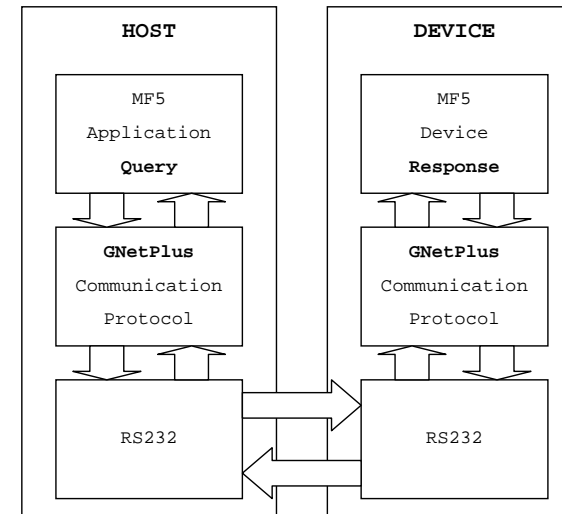
Property	gnetVersion
Description	Return current machine firmware version.
Syntax	String = Object.gnetVersion
Parameter	

Method	gnetPolling
Description	Poll all machines with ID.
Syntax	<i>Boolean = Object.gnetPolling(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	gnetReset
Description	Reset current machine.
Syntax	Boolean = Object.gnetReset()
Parameter	

MF5 Communication Protocol

MF5 Communication Block:



Please refer to another document "**GNetPlus Programmer Guide**" first.

MF5 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	
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	
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)
Reserve	2Dh					
Authenticate + Key	2Eh	8	KEY_TYEP + SECTOR + KEY ⁴	ACK	0	
RequestAll	2Fh	0		ACK	2	Card Class Type (Integer)

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)

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 10.**