HID MSR User Manuals

Version: 1.0

Issued on: March 31, 2011

Issued by: GIGA-TMS, Inc.

Issued for: GIGA-TMS, Inc.

Change History

Version	Date	Changes
1.0	March 31, 2011	● Initial version

Table of Contents

Hardware Manuals

This part of the documentation describes the HID MSR hardware supported by GIGA-TMS.

MSR170HK

MSR170HK 43 mm USB HID Magnetic Stripe Swipe Card Read" consists of a high-performance multi-channel fully integrated magnetic stripe decoder chip at a low-profile magnetic read-head. This innovative, yet low-cost card reading solution offers many important advantages over the conventional less-integrated approach. MSR170HK 43 mm USB HID Magnetic Stripe Swipe Card Read" conform to industry specifications including ANSI/ISO Standards 7810, 7811 1/5, 7812 & 7813. It can read both Lo-Co card and Hi-Co cards.

Features

- **Low cost solution for triple track readers**: Available ISO TK1, 2 &3
- **Ultra-compact design**: Low-profile read head contains all needed circuits. Save space!
- **High noise immunity**: No more mill volt-level analog signals to route; no analog signals leave the shielded magnetic head! Withstands noisy PC monitors, cell phones, switching power supplies, etc.
- **High performance decoding**: New design reads badly damaged cards; compensates for poor head mounting
- **AGC (Automatic Gain Control)**: Reads cards from 30% - 200% of ISO 7811 amplitude standard
- No Windows driver needed! "MagStripe Card Reader Configuration Utility" fast and easy to use!

Specification

- **Reference Standards**: ANSI/ISO Standards 7810,7811-1⁄6, 7813
- **Recording Method**: Two-frequency coherent phase (F2F)
- **Decoding Method**:

  ISO: Track1 – IATA, Track2 – ATA, Track3 – THRIFT
- **Card Swiping Direction:** Bi-directional
- **Dimensions:** H 22.40 x W 16.00 x L 43.00 mm

- **Lift:**
    - Electronics: 125,000 hours
    - Head: 1,000,000 passes
- **Environment:**
    - Operating Temp: 0 ~ + 55° C
    - Storage Temp: -10 ~ + 55° C
    - Humidity: 10 ~ 90 relative

MSR220HK

MSR220HK 90 mm USB HID Magnetic Stripe Swipe Card Read" consists of a high-performance multi-channel fully integrated magnetic stripe decoder chip at a low-profile magnetic read-head. This innovative, yet low-cost card reading solution offers many important advantages over the conventional less-integrated approach. MSR220HK 90 mm USB HID Magnetic Stripe Swipe Card Read" conform to industry specifications including ANSI/ISO Standards 7810,7811 1⁄5, 7812 & 7813. It can read both Lo-Co card and Hi-Co cards.

Features

- **Low cost solution for triple track readers**: Available ISO TK1, 2 &3
- **Ultra-compact design**: Low-profile read head contains all needed circuits. Save space!
- **High noise immunity**: No more mill volt-level analog signals to route; no analog signals leave the shielded magnetic head! Withstands noisy PC monitors, cell phones, switching power supplies, etc.
- **High performance decoding**: New design reads badly damaged cards; compensates for poor head mounting
- **AGC (Automatic Gain Control)**: Reads cards from 30% - 200% of ISO 7811 amplitude standard
- No Windows driver needed! "MagStripe Card Reader Configuration Utility" fast and easy to use!

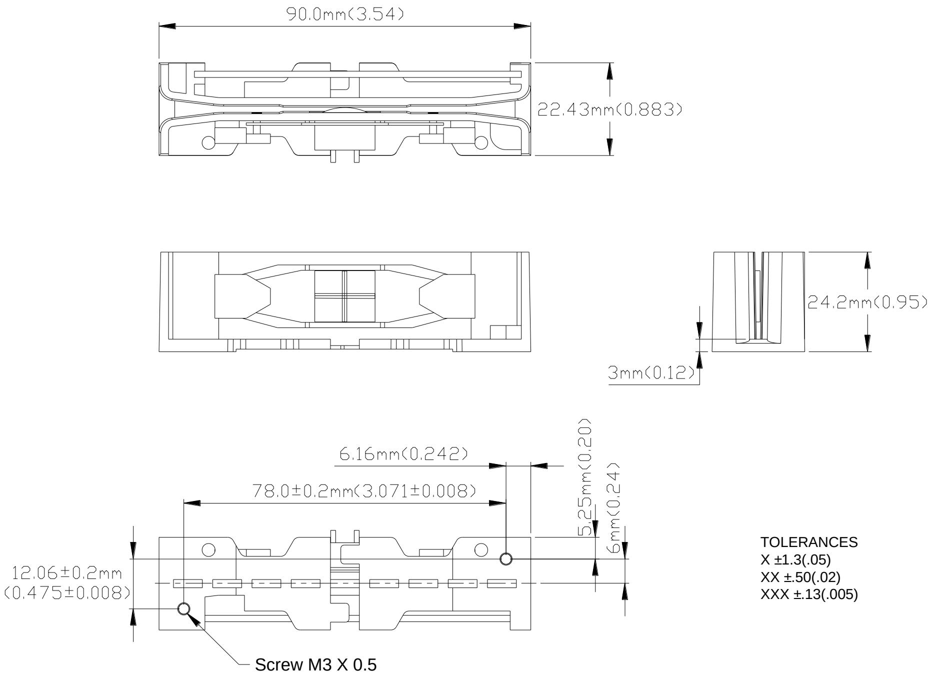
Specification

- **Reference Standards**: ANSI/ISO Standards 7810,7811-1⁄6, 7813
- **Recording Method**: Two-frequency coherent phase (F2F)
- **Decoding Method**:

  ISO: Track1 – IATA, Track2 – ATA, Track3 – THRIFT
- **Card Swiping Direction:** Bi-directional
- **Lift:**
  - Electronics: 125,000 hours
  - Head: 1,000,000 passes
- **Environment:**

  - Operating Temp: 0 ~ + 55° C
  - Storage Temp: -10 ~ + 55° C
  - Humidity: 10 ~ 90 relative
- **Interface**: USB (HID -Keyboard Mode)
- Compatible with USB specification Revision 1.1
- Compatible with HID specification Version 1.1
- **MagStripe Card Reader Configuration Utility**: Support Microsoft Windows98/ME/2000/XP/Vista 32bit.
- **Power Supply**: DC 5V, though USB Interface Cable.
- **Power Consumption**: Less than 12mA maximum total current when no card is being swiped, less than 70 mA maximum total current at 5V while card is being swiped.
- **Card Swiping Speed**: Card speed through the unit may vary from3 ips to 100 ips (7 cm/s to 250 cm/s)
- **Dimensions:**

  Height: 24.20 mm
  Width: 22.43 mm
  Length: 90.00 mm

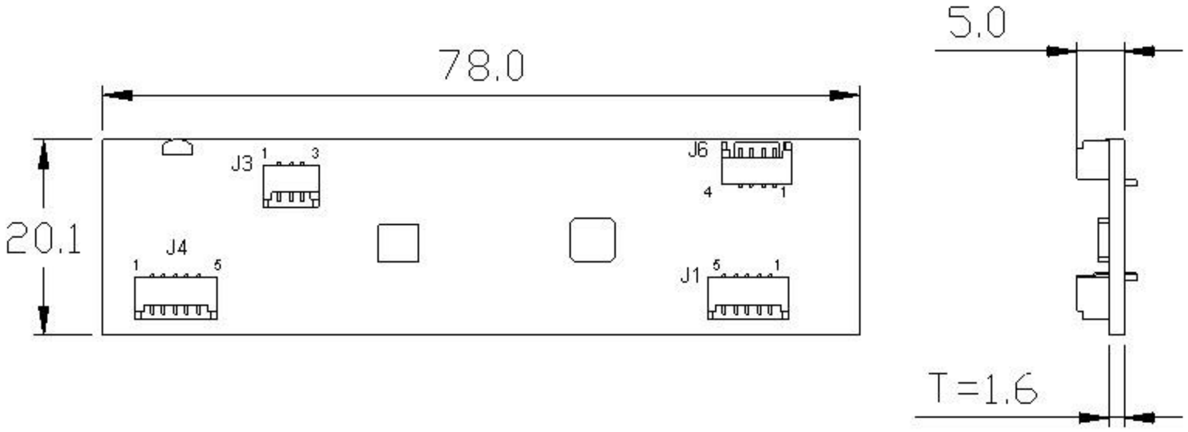


- **Connector:**

P/N	MSR220HK-00
Interface	USB
Pin Numbers	5
Connector	5PIN pitch: 1.25mm
1	VCC
2	D -

3	D +
4	GND
5	HEAD CASE

- **I/O Information:**



MSR Input Connection J1 MSR SHIFT OUT TTL - PIN ASSIGNMENTS	
Pin Numbers	Signal
1	STROBE
2	DATA
3	+5V
4	GND
5	SHIELD

Indication for external Connection J3 LED PIN ASSIGNMENTS	
Pin Numbers	Signal
1	RED
2	+5V
3	GREEN

OUTPUT Connection J4 OUTPUT - PIN ASSIGNMENTS	
Pin Numbers	Signal
1	V.C.C. (+5V)
2	D -
3	D +
4	GND
5	SHIELD

Message Format

- USB interface (HID -Keyboard Mode): HID Keyboard device

- Tracks and Serial Number Structure

I S O

IATA	%	Track 1, 210 bpi, 79 alphanumeric characters, 7 bits/char	?
ABA	;	Track 2, 75 bpi, 40 numeric characters, 5 bits/char	?
THRIFT	+	Track 3, 210 bpi, 107 numeric characters, 5 bits/char	?

The device's programmable configuration options affect the format of the card data.
The card data format for the default configuration is as follows:

%	Track 1 Data	?	;	Track 2 Data	?	+	Track 3 Data	?

Status Indicator

ERROR INDICATOR (Red color)
When encountering erroneous input, defective card, misread, or
Incorrectly encoded data, the device will turn on the ERROR indicator.

READY INDICATOR (Green color)
Indicating the reader is ready to accept new inputs.

STATUS	**GREEN LED**	**RED LED**	**BUZZER**
POWER ON	ON	ON	Be-
READY	ON	OFF	**X**
READ OK	BLINK 1 TIME	OFF	Be
READ ERROR	OFF	ON	Be- Be- Be-

MSR250HK

MSR250HK 100 mm USB HID Magnetic Stripe Swipe Card Read" consists of a high-performance multi-channel fully integrated magnetic stripe decoder chip at a low-profile magnetic read-head. This innovative, yet low-cost card reading solution offers many important advantages over the conventional less-integrated approach. MSR250HK 100 mm USB HID Magnetic Stripe Swipe Card Read" conform to industry specifications including ANSI/ISO Standards 7810, 7811 1⁄5, 7812 & 7813. It can read both Lo-Co card and Hi-Co cards.

Features

- **Low cost solution for triple track readers**: Available ISO TK1, 2 &3
- **Ultra-compact design**: Low-profile read head contains all needed circuits. Save space!
- **High noise immunity**: No more mill volt-level analog signals to route; no analog signals leave the shielded magnetic head! Withstands noisy PC monitors, cell phones, switching power supplies, etc.
- **High performance decoding**: New design reads badly damaged cards; compensates for poor head mounting
- **AGC (Automatic Gain Control)**: Reads cards from 30% - 200% of ISO 7811 amplitude standard
- No Windows driver needed! "MagStripe Card Reader Configuration Utility" fast and easy to use!

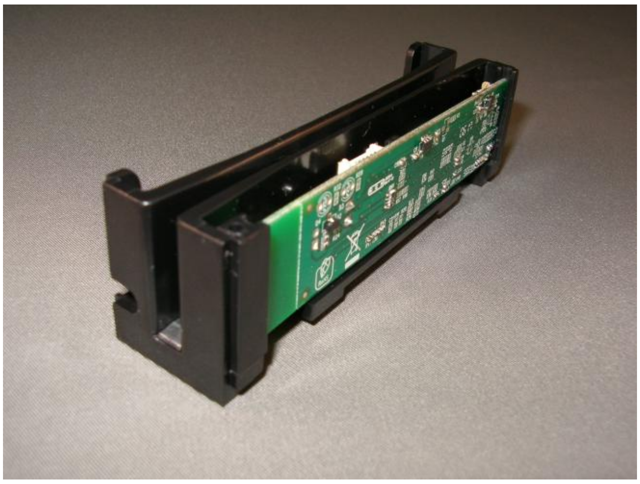
Specification

- **Reference Standards**: ANSI/ISO Standards 7810,7811-1⁄6, 7813
- **Recording Method**: Two-frequency coherent phase (F2F)
- **Decoding Method**:

  ISO: Track1 – IATA, Track2 – ATA, Track3 – THRIFT
- **Card Swiping Direction:** Bi-directiona**l**
- **Lift:**
  - ■ Electronics: 125,000 hours
  - ■ Head: 1,000,000 passes
- **Environment:**

  - Operating Temp: 0 ~ + 55° C
  - Storage Temp: -10 ~ + 55° C
  - Humidity: 10 ~ 90 relative
- **Interface**: USB (HID -Keyboard Mode)
- Compatible with USB specification Revision 1.1
- Compatible with HID specification Version 1.1
- **MagStripe Card Reader Configuration Utility**: Support Microsoft Windows98/ME/2000/XP/Vista 32bit.
- **Power Supply**: DC 5V, though USB Interface Cable.
- **Power Consumption**: Less than 12mA maximum total current when no card is being swiped, less than 70 mA maximum total current at 5V while card is being swiped.
- **Card Swiping Speed**: Card speed through the unit may vary from3 ips to 100 ips (7 cm/s to 250 cm/s)
- **Dimensions**: L98.3 x W30.3 x H31.2 mm with cover

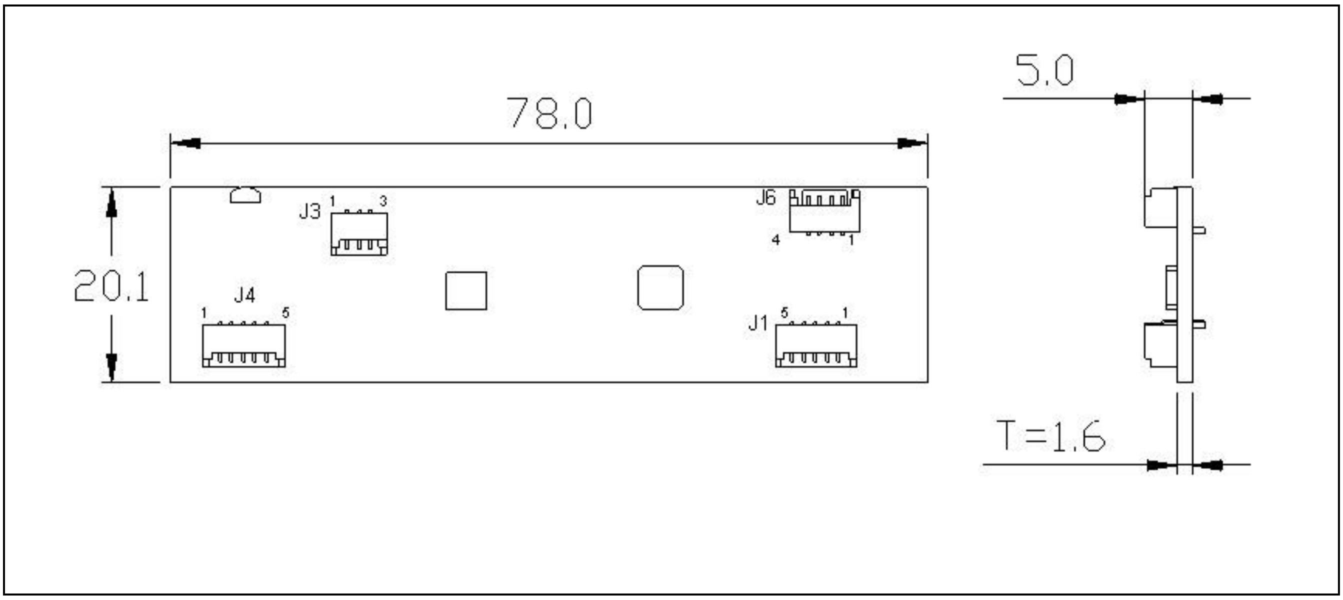
Module



- **Dimensions:** 98.3 x W 27.1 x H 29.5 mm without cover
- **Connector:**

P/N	MSR220HK-00
Interface	USB
Pin Numbers	5
Connector	5PIN pitch: 1.25mm
1	VCC
2	D -
3	D +
4	GND
5	HEAD CASE

I/O Information:



MSR Input Connection

J1 MSR SHIFT OUT TTL - PIN ASSIGNMENTS

Pin Numbers	Signal
1	STROBE
2	DATA
3	+5V
4	GND

Indication for external Connection

J3 LED PIN ASSIGNMENTS

Pin Numbers	Signal
1	RED
2	+5V
3	GREEN

OUTPUT Connection

J4 OUTPUT - PIN ASSIGNMENTS

Pin Numbers	Signal
1	V.C.C. (+5V)
2	D -
3	D +
4	GND
5	SHIELD

Message Format

- USB interface (HID -Keyboard Mode): HID Keyboard device
- Tracks and Serial Number Structure

I S O

IATA	%	Track 1, 210 bpi, 79 alphanumeric characters, 7 bits/char	?
ABA	;	Track 2, 75 bpi, 40 numeric characters, 5 bits/char	?

THRIFT	+	Track 3, 210 bpi, 107 numeric characters, 5 bits/char	?

The device's programmable configuration options affect the format of the card data.
The card data format for the default configuration is as follows:

%	Track 1 Data	?	;	Track 2 Data	?	+	Track 3 Data	?

Status Indicator

ERROR INDICATOR (Red color)
When encountering erroneous input, defective card, misread, or
Incorrectly encoded data, the device will turn on the ERROR indicator.

READY INDICATOR (Green color)
Indicating the reader is ready to accept new inputs.

STATUS	GREEN LED	RED LED	BUZZER
POWER ON	ON	ON	Be-
READY	ON	OFF	**X**
READ OK	BLINK 1 TIME	OFF	Be
READ ERROR	OFF	ON	Be- Be- Be-

Firmware Manuals

This part of the documentation describes the communication protocol of firmware related to MSR readers.

Vender ID and Product ID

Before starting the communication, it needs to get the control handle for the later data transferring. The MSR readers using the same vender ID, the value is 5735 (0x1667 in hex format). The product IDs are listed as below:

Model	Product ID (decimal)
MSR170HK	9
MSR220HK	9
MSR250HK	9

Programming

Using feature report function to tranfer the HID USB command and data. The **HidD_SetFeature**[1] function is used to send out the data. The **HidD_GetFeature**[2] function is used to retrieve the incoming data. The command data size is 131 bytes. Below is the command and data format:

1st byte	2nd byte	3rd ~ 131st
Report ID	Command Code	Data

The value of Report ID is always to be 0x00.

Note:

1. The **HidD_SetFeature** routine sends a feature report to HID control. For more details, please refer to Microsoft MSDN website.
2. The **HidD_GetFeature** routine returns a feature report from HID control. For more details, please refer to Microsoft MSDN website.

Commands

The commands are used to control the keyboard device and can be issued through the USB port. Below table list all available commands:

Command Code	Description
0x30	Set Language Key Map
0x40	Set MSR Prefix/Suffix
0x50	Set MSR Output Mode
0x60	Get Macro Key definition of Layer #1
0x70	Get Macro Key definition of Layer #2
0x80	Get Language Key Map
0x90	Get MSR Prefix/Suffix

0xA0	Get MSR Output Mode
0xC0	Get Firmware Version
0xD0	Set MSR reader into Firmware Upload Mode

Setting Language Key Map

Function: Defines the **control value** and HID Usage ID of ASCII code

Data format: **R$C_iI_iC_{i+1}I_{i+1}$…**, where **R** is the report index, **C_i** is the binary value that indicates the used **control value** of following key Usage ID, I_i is the binary value that indicates the key Usage ID. The index value i is the ASCII code that related the **control value** **C_i** and the Usage ID **I_i**.

Details:

Because the data size of the key map can be up to 512 bytes, so it needs to send 4 reports to MSR reader for the updating, so the value of report index **R** is from 0 to 3.

Each ASCII code is related with the **control value** and the Usage ID, and different language platform (Windows/Linux ...) may use different **control value** and Usage ID for a certain ASCII code. For example, for the English platform, the Usage ID of letter "x" is 0x1C, but for Germany platform, the "x" Usage ID is 0x1D.

1st report:

.	Rpt ID	Cmd	Rpt Idx	Key Map						
No.	1st	2nd	3rd	4th	5th	6th	7th	…	130 th	131th
Value	0x00	0x30	0x00	C_0	I_0	C_1	I_1	…	C_{63}	I_{63}

2nd report:

.	Rpt ID	Cmd	Rpt Idx	Key Map						
No.	1st	2nd	3rd	4th	5th	6th	7th	…	130 th	131th
Value	0x00	0x30	0x01	C_{64}	I_{64}	C_{65}	I_{65}	…	C_{127}	I_{127}

3rd report:

.	Rpt ID	Cmd	Rpt Idx	Key Map						
No.	1st	2nd	3rd	4th	5th	6th	7th	…	130 th	131th
Value	0x00	0x30	0x02	C_{129}	I_{129}	C_{130}	I_{130}	…	C_{191}	I_{191}

4th report:

.	Rpt ID	Cmd	Rpt Idx	Key Map						
No.	1st	2nd	3rd	4th	5th	6th	7th	…	130 th	131th
Value	0x00	0x30	0x03	C_{192}	I_{192}	C_{193}	I_{193}	…	C_{255}	I_{255}

Control value tells MSR reader how to hit the control keys (Shift/Alt/Ctrl) or the way of hitting the key while sending out the Usage ID. For example, the lower case letter "a", the control value is 0. For the upper case letter "A", the control value is 1. Below shows all the **control values**:

- **0**: No control key hit, direct to send out the Usage ID.
- **1**: Press Shift key, and then send out the Usage ID.
- **6**: Press right Alt key, and then send out the Usage ID.
- **16**: Double hit the key button.
- **17**: Press Shift key and double hit the key button.

Not all the ASCII code needs to be defined by the control value and Usage ID. For example, before ASCII code

32, only the value 13 (for Enter key) is needed to be defined. For the undefined ASCII code, just leave 0 value in the buffer.

Getting Language Key Map

Function: Returns the **control value** and HID Usage ID of ASCII code

Data format: **R**, where **R** is the report index

Details:

-

Example: here is a sample to show how to the retrieve language key map with specified report index. The report ID is 0x00. The command code is 0x80, and the report index number is 3. Then the command data will be as follow:

.	Rpt ID	Cmd	Rpt Idx	Remains
No.	1st	2nd	3rd	4th -131th
Value	0x00	0x80	0x03	0x00

Visual Basic 6 Sample Code:

```
Redim bCmdData(131 - 1) as Byte

bCmdData(0)=&H0          'Report ID
bCmdData(1)=&H80         'Command Code for retrieve key map
bCmdData(2)=&H03         'Report Index

if (HidD_SetFeature(HidDevice, bCmdData(0), 131)=1 then
   HidD_GetFeature(HidDevice, bCmdData(0), 131)
End if
```

HidDevice is the handle to the device path of specified vendor ID and product ID of HID USB MSR reader. The retrieved data will put in the later 4th byte of bCmdData buffer. The retrieved data format is same to the data format of updating language key map command.

Setting MSR Prefix/Suffix

Function: Defines the prefix or suffix of MSR output data

Data format: **KL*dd…d***, where **K** is the binary value that indicates which type of prefix or suffix to be updated., **L** is the binary value that indicates the length of macro key string, ***dd…d*** is the content of updated prefix or suffix string

Details:

The prefix and suffix string may contain the letters, numbers, function keys, control keys or special keys.

There are four kinds of prefix and suffix, which are package, Track 1, Track 2 and Track 3.

Below is the list of type value for prefix and suffix:

0 (Prefix for package)	The prefix string will append in the start of whole track data.
1 (Suffix for package)	The suffix string will append in the end of whole track data.
2 (Prefix for Track 1)	The prefix string will append in the start of track 1 data.
3 (Suffix for Track 1)	The suffix string will append in the end of track 1 data.
4 (Prefix for Track 2)	The prefix string will append in the start of track 2 data.
5 (Suffix for Track 2)	The suffix string will append in the end of track 2 data.
6 (Prefix for Track 3)	The prefix string will append in the start of track 3 data.
7 (Suffix for Track 3)	The suffix string will append in the end of track 3 data.

Example: here is a sample to show how to define the prefix of Track 1.

The report ID is 0x00. The command code is 0x40, and the type value to be programmed is 2. The prefix string will be “<TK1>=”.

The length of macro key string is 6. Then the command data will be as follow:

.	**Rpt ID**	**Cmd**	**Typ Val**	**Length**	**Macro Key Definition**						
No.	1st	2nd	3rd	4th	5th	6th	7th	8th	9th	10th	11th-131th
Value	0x00	0x40	0x02	0x06	0x3c	0x54	0x4B	0x31	0x3D	0x3E	0x00
Char.					<	T	K	1	=	>	

Visual Basic 6 Sample Code:

```
Redim bCmdData(131 - 1) as Byte

bCmdData(0)=&H0         ‘Report ID
bCmdData(1)=&H40        ‘Command Code
bCmdData(2)=&H02        ‘Type value for the Prefix of Track 1
bCmdData(3)=&H6         ‘The string length of “<TK1=>”
bCmdData(4)=&H3C        ‘ASCII code of ”<”
bCmdData(5)=&H54        ‘ASCII code of ”T”
bCmdData(6)=&H4B        ‘ASCII code of ”K”
bCmdData(7)=&H31        ‘ASCII code of “1”
bCmdData(8)=&H3D        ‘ASCII code of “=”
bCmdData(9)=&H3E        ‘ASCII code of “>”
HidD_SetFeature(HidDevice, bCmdData(0), 131)
```

HidDevice is the handle to the device path of specified vendor ID and product ID of HID USB MSR reader.

Getting MSR Prefix/Suffix

Function:	Returns the prefix or suffix of MSR output data
Data format:	**K**, where **K** is the binary value that indicates which type of prefix or suffix to be updated.

Details:

The prefix and suffix string may contain the letters, numbers, function keys, control keys or special keys. There are four kinds of prefix and suffix, which are package, Track 1, Track 2 and Track 3.

Below is the list of type value for prefix and suffix:

0 (Prefix for package)	The prefix string will append in the start of whole track data.
1 (Suffix for package)	The suffix string will append in the end of whole track data.
2 (Prefix for Track 1)	The prefix string will append in the start of track 1 data.
3 (Suffix for Track 1)	The suffix string will append in the end of track 1 data.
4 (Prefix for Track 2)	The prefix string will append in the start of track 2 data.
5 (Suffix for Track 2)	The suffix string will append in the end of track 2 data.
6 (Prefix for Track 3)	The prefix string will append in the start of track 3 data.
7 (Suffix for Track 3)	The suffix string will append in the end of track 3 data.

Example: here is a sample to show how to retrieve the prefix of Track 1.

The report ID is 0x00. The command code is 0x90, and the type value to be retrieved is 2.

The length of macro key string is 6. Then the command data will be as follow:

.	Rpt ID	Cmd	Typ Val	Remains
No.	1st	2nd	3rd	4th-131th
Value	0x00	0x90	0x02	0x00

Visual Basic 6 Sample Code:

```
Redim bCmdData(131 - 1) as Byte

bCmdData(0)=&H0          ‘Report ID
bCmdData(1)=&H90         ‘Command Code
bCmdData(2)=&H02         ‘Type value for the Prefix of Track 1

if (HidD_SetFeature(HidDevice, bCmdData(0), 131)=1 then
   HidD_GetFeature(HidDevice, bCmdData(0), 131)
End if
```

HidDevice is the handle to the device path of specified vendor ID and product ID of HID USB MSR reader. The retrieved data will put in the later 4th byte of bCmdData buffer. The retrieved data format is same to the data format of updating MSR prefix/suffix command.

Setting MSR Output Mode

Function:	Defines the tracks output sequence, required tracks to be output and the Start Sentinel/End Sentinel for each track
Data format:	**B$D_1D_2D_3S_1E_1S_2E_2S_3E_3$O**, where **B** is the fixed binary value and the value is always to be 0, **D_1**, **D_2** and **D_3** is the binary value of **decode mode** for Track1, Track2 and Track3 separately. **S_1E_1** is the ASCII of Start Sentinel and End Sentinel for Track1. **S_2E_2** is the ASCII of Start Sentinel and End Sentinel for Track2. **S_3E_3** is the ASCII of Start Sentinel and End

Sentinel for Track3. **O** is the binary value that indicates the output order.

Details:

Decode mode is used to limit which track needs to be or not to be output. The **decode mode** value has below three values:

0 (Disabled)	The specified track will not be output, regardless of the specified track is decoded or not.
1 (Required)	The specified track needs to be decoded. If the specified track is not decoded, the MSR will not output any data even other tracks are decoded.
2 (Enabled)	The specified track can be output. If the specified track data is not decoded, the track filed will leave blank.

The output order defines the sequence of track1-3 to be output, which can be six kinds of orders. The values are:

0	The track order is Track1-Track2-Track3.
1	The track order is Track1-Track3-Track2.
2	The track order is Track2-Track1-Track3.
3	The track order is Track2-Track3-Track1.
4	The track order is Track3-Track1-Track2.
5	The track order is Track3-Track2-Track1.

Example: here is a sample to show how to define the MSR output mode.

The report ID is 0x00. The command code is 0x50, and the B value fixed to be 0. The Track1, Track2 and Track3 decode modes are enabled, required and disabled separately. The Track1 SS (Start Sentinels) is “%”, the Track2 SS is “;” and the Track3 SS is “;”. The ES (End Sentinel) is “?” for all tracks. The output order value is 5 (Track3-2-1). Then the command data will be as follow:

.	**Rpt ID**	**Cmd**	**B**	**D1**	**D2**	**D3**	**S1**	**E1**	**S2**	**E2**	**S3**	**E3**	**O**	**Remains**
No.	1st	2nd	3rd	4th	5th	6th	7th	8th	9th	10th	11th	12th	13th	14th-131th
Value	0x00	0x50	0x00	0x02	0x01	0x00	0x25	0x3F	0x3B	0x3F	0x3B	0x3F	0x05	0x00
Char.							%	?	;	?	;	?		

Visual Basic 6 Sample Code:

```
Redim bCmdData(131 - 1) as Byte

bCmdData(0)=&H0          ‘Report ID
bCmdData(1)=&H40         ‘Command Code
bCmdData(2)=&H00         ‘Fixed to 0
bCmdData(3)=&H2          ‘Track1 decode mode - enabled
bCmdData(4)=&H1          ‘Track2 decode mode - required
bCmdData(5)=&H0          ‘Track3 decode mode - disabled
bCmdData(6)=&H25         ‘Track 1 SS - %
bCmdData(7)=&H3F         ‘Track 1 ES - ?
bCmdData(8)=&H3B         ‘Track 2 SS - ;
bCmdData(9)=&H3F         ‘Track 2 ES - ?
bCmdData(8)=&H3B         ‘Track 3 SS - ;
bCmdData(9)=&H3F          'Track 3 ES - ?
bCmdData(8)=&H5           'Output order – Track3-2-1

HidD_SetFeature(HidDevice, bCmdData(0), 131)
```

HidDevice is the handle to the device path of specified vendor ID and product ID of HID USB MSR reader.

Getting MSR Output Mode

Function:	Returns the tracks output sequence, required tracks to be output and the Start Sentinel/End Sentinel for each track
Data format:	**B**, where **B** is the fixed binary value and the value is always to be 0

Details:

The retrieving data content can refer to Updating MSR Output Mode.

Example: here is a sample to show how to retrieve the MSR output mode settings.

The report ID is 0x00. The command code is 0x50.

.	Rpt ID	Cmd	B Val	Remains
No.	1st	2nd	3rd	4th-131th
Value	0x00	0x50	0x00	0x00

Visual Basic 6 Sample Code:

```
Redim bCmdData(131 - 1) as Byte

bCmdData(0)=&H0           'Report ID
bCmdData(1)=&H50          'Command Code
bCmdData(2)=&H00          'Always fixed to 0

if (HidD_SetFeature(HidDevice, bCmdData(0), 131)=1 then
   HidD_GetFeature(HidDevice, bCmdData(0), 131)
End if
```

HidDevice is the handle to the device path of specified vendor ID and product ID of HID USB keyboard. The retrieved data will put in the later 4th byte of bCmdData buffer. The retrieved data format is same to the data format of updating MSR output mode command.

Getting Firmware Version

Function:	Returns the firmware version of this firmware
Data format:	**V**, where **V** is the fixed binary value and the value is always to be 0

Details:

Get Firmware Version command returns current firmware version string.

The version string contains model name, firmware ROM number and version which separated by space character. For example: KB200-USB-TS ROM-T1033 V1.0R0

Example: here is a sample to show how to retrieve the firmware version string.

The report ID is 0x00. The command code is 0x50.

.	Rpt ID	Cmd	Remains
No.	1st	2nd	3th-131th
Value	0x00	0xC0	0x00

Visual Basic 6 Sample Code:

```
Redim bCmdData(131 - 1) as Byte

bCmdData(0)=&H0         'Report ID
bCmdData(1)=&C50        'Command Code

if (HidD_SetFeature(HidDevice, bCmdData(0), 131)=1 then
   HidD_GetFeature(HidDevice, bCmdData(0), 131)
End if
```

.	Rpt ID	Cmd	Length	Firmware String							
No.	1st	2nd	3rd	4th	5th	6th	7th	8th	9th	…	Remains
Value	0x00	0xC0	0x1D	0x4B	0x42	0x32	0x30	0x30	0x2D		0x00
				K	B	2	0	0	-	…	

HidDevice is the handle to the device path of specified vendor ID and product ID of HID USB MSR reader. The retrieved data will put in the later 4th byte of bCmdData buffer.

Appendix A: Keyword list

Keyboard/Keypad

Usage Name	Keyword
Keyboard ESCAPE	<Esc>
Keyboard F1	<F1>
Keyboard F2	<F2>
Keyboard F3	<F3>
Keyboard F4	<F4>
Keyboard F5	<F5>
Keyboard F6	<F6>
Keyboard F7	<F7>

Keyboard F8	<F8>
Keyboard F9	<F9>
Keyboard F10	<F10>
Keyboard F11	<F11>
Keyboard F12	<F12>
Keyboard BackSpace	<BackSpace>
Keyboard Tab	<Tab>
Keyboard Caps Lock	<CapsLock>
Keyboard Enter	<Enter>
Keyboard Left Shift	<LShift>
Keyboard Right Shift	<RShift>
Keyboard Left Ctrl	<LCtrl>
Keyboard Left WinKey	<LGU>
Keyboard Space	<SpaceBar>
Keyboard Right Alt	<RAlt>
Keyboard Right WinKey	<RGUI>
Keyboard Right Ctrl	<RCtrl>
Keyboard Print Screen/SysRq	<PrintScreenSysRq>
Keyboard Scroll Lock	<ScrollLock>
Keyboard Pause/Break	<PauseBreak>
Keyboard Insert	<Insert>
Keyboard Home	<Home>
Keyboard Page Up	< PageUp >
Keyboard Delete	<Delete>
Keyboard End	< End >
Keyboard Page Down	<PageDown>
Keyboard Up Arrow	<UpArrow>
Keyboard Insert	<Insert>
Keyboard Left Arrow	<LeftArrow>
Keyboard Down Arrow	<DownArrow>
Keyboard Right Arrow	<RightArrow>
Keypad Num Lock	<NumLock>
Keypad /	<Pad/>
Keypad *	<Pad*>
Keypad -	<Pad ->
Keypad 7	<Pad7>
Keypad 8	<Pad8>
Keypad 9	<Pad9>
Keypad +	<Pad+>
Keypad 4	< Pad4>
Keypad 5	<Pad5>

Keypad 6	<Pad6>
Keypad 1	<Pad1>
Keypad 2	<Pad2>
Keypad 3	<Pad3>
Keypad Enter	<PadEnter>
Keypad 0	<Pad0>
Keypad .	<Pad.>
Layer Lock	<LayerLock>

Special Keys

Usage Name	Keyword
Power	<Power>
Sleep	<Sleep>
Wake Up	<WakeUp>
Scan Next Track	<ScanNextTrack>
Scan Previous Track	<ScanPreTrack>
Stop	<Stop>
Play/Pause	<Play/Pause>
Mute	<Mute>
Bass Boost	<BassBoost>
Loudness	<Loudness>
Volume Up	<VolumeUp>
Volume Down	<VolumeDown>
Bass Up	<BassUp>
Bass Down	<BassDown>
Treble Up	<TrebleUp>
Treble Down	<TrebleDown>
Media Select	<MediaSelect>
Mail	<Mail>
Calculator	<Calculator>
My Computer	<MyComputor>
Web Search	<WWWSearch>
Home Page	<WWWHome>
Web Page Back	<WWWBack>
Web Page Forward	<WWWForward>
Web Page Stop	<WWWStop>
Refresh Web Page	<WWWRefresh>
Favorites Web Page	<WWWFavorites>
Keyboard F13	<F13>
Keyboard F14	<F14>

Keyboard F15	<F15>
Keyboard F16	<F16>
Keyboard F17	<F17>
Keyboard F18	<F18>
Keyboard F19	<F19>
Keyboard F20	<F20>
Keyboard F21	<F21>
Keyboard F22	<F22>
Keyboard F23	<F23>
Keyboard F24	<F24>
Delay 100 mini-second	<Delay100ms>
Delay 1 second	<Delay1s>
Reset Shift/Ctrl/Alt key states	<->

Software Manuals

This part of the documentation describes the PC software used to configure the HID MSR reader supported by GIGA-TMS.

MagStripe Card Reader Configuration Utility

The **MagStripe Card Reader Configuration Utility** (**MCRCU**) is used to set up the output format of HID MSR reader (MSR170/220/250 series).

Installation

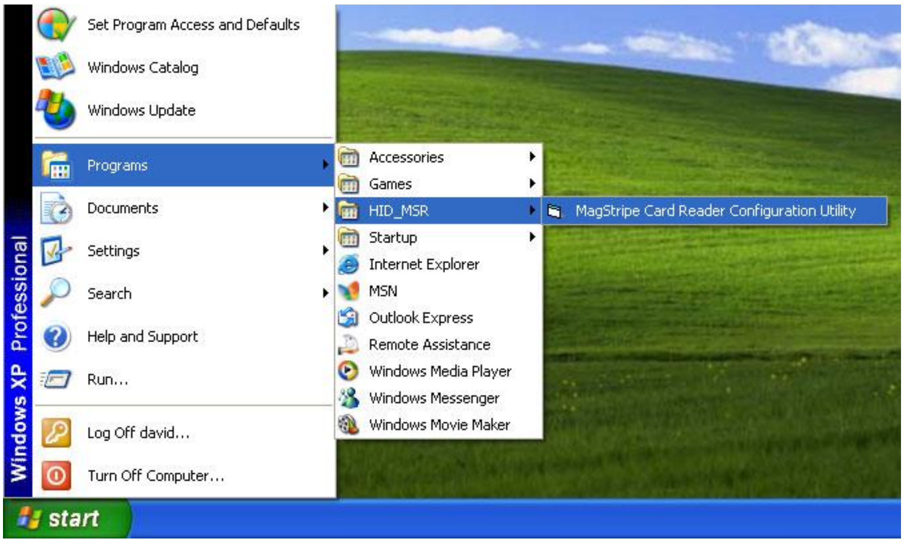
Below steps guide you how to install the MCRCU program.

- Insert the setup CD
- Run the **HID_MSR_PSW00003.exe** setup file that is located in the Software folder of CD.
- Follow the wizard to complete the installation.

Launching Program

Below steps guide you how to load the **MCRCU** program.

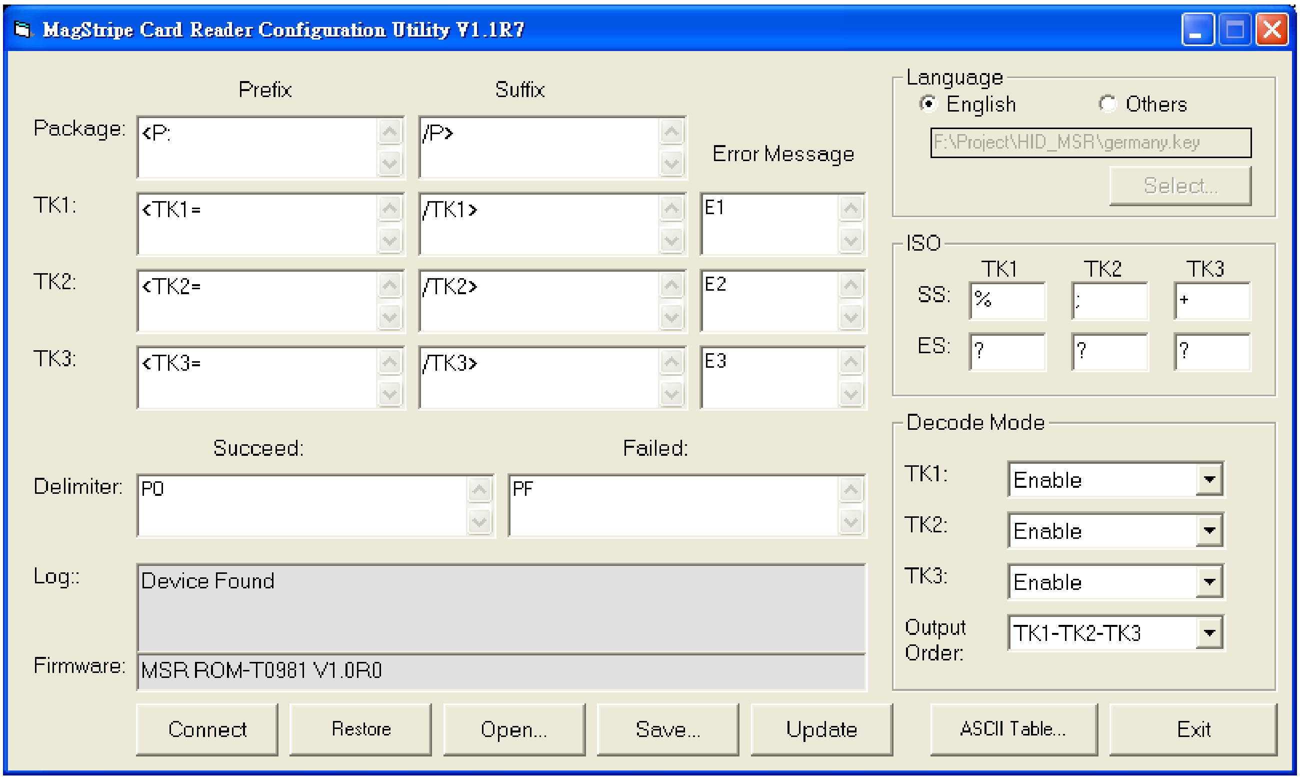
- From **Start**/**Programs**, click **HID_MSR** folder
- Click **MagStripe Card Reader Configuration Utility** to launch the program.



- **MCRCU** program will detect the connected reader. If detected, all the input text boxes will be enabled.
- If the reader has not been connected to PC yet, please connect the reader and then click **Refresh** to get connected.

Configuration

Below is the main window of **MCRCU** program.



	Prefix	Suffix	Error Message
Package:	<P:	/P>	
TK1:	<TK1=	/TK1>	E1
TK2:	<TK2=	/TK2>	E2
TK3:	<TK3=	/TK3>	E3

	Succeed:	Failed:
Delimiter:	P0	PF

Log:: Device Found

Firmware: MSR ROM-T0981 V1.0R0

Language
● English ○ Others
F:\Project\HID_MSR\germany.key

ISO

	TK1	TK2	TK3
SS:	%	;	+
ES:	?	?	?

Decode Mode
TK1: Enable
TK2: Enable
TK3: Enable
Output Order: TK1-TK2-TK3

For the settings, there are:

- **Language**: The language defines the code positions of the keyboard. Each language should use its own settings. Wrong language selected will cause the wrong character displayed.
- **Prefix/Suffix:** Defines the data string which you would like to append in front or end of the MSR data string.
- **Error Message:** Indicates which track number cause the error.
- **Delimiter**: Indicates the swipe result.
- **ISO:** Define start and end sentinel character.
- **Decode Mode:** Determines the way of outputting the three tracks data.

Shown below is the data structure of the output string for MSR.

PP	PR1	SS1	**TK1**	ES1	SU1	PR2	SS2	**TK2**	ES2	SU2	PR3	SS3	**TK3**	ES3	SU3	SU	DM

- **PP:** Prefix for package.
- **PR1:** Prefix for track 1.
- **SS1:** Start sentinel for track 1.
- **TK1:** Data for track 1, if error happens, using Error Message instead.
- **ES1:** End sentinel for track 1.
- **SU1:** Suffix for track 1.
- **PR2:** Prefix for track 2.
- **SS2:** Start sentinel for track 2.
- **TK2:** Data for track 2, if error happens, using Error Message instead..
- **ES2:** End sentinel for track 2.

- **SU2:** Suffix for track 2.
- **PR3:** Prefix for track 3.
- **SS3:** Start sentinel for track 3.
- **TK3:** Data for track 3, if error happens, using Error Message instead..
- **ES3:** End sentinel for track 3.
- **SU3:** Suffix for track 3.
- **SU:** Suffix for package.
- **DM:** Delimiter for the swipe result.

Prefix/Suffix

In default, the prefix and suffix settings are all keep blank. There are 4 kinds of prefix and suffix to be defined, which are:

- **Package:** For the prefix string, it is appended in the front of the whole MSR data. For the suffix, it is appended in the end of the whole MSR data. In most case, the suffix for package is always to be the "Enter" or "Tab" character. The max data length of the prefix and suffix for the package can be up to 127.
- **TK1:** For the prefix string, it is appended in the front of the start sentinel of track 2. For the suffix, it is appended in the end of the end sentinel of track 2. The max data length of the prefix and suffix for the TK1 can be up to 127.
- **TK2:** For the prefix string, it is appended in the front of the start sentinel of track 2. For the suffix, it is appended in the end of the end sentinel of track 2. The max data length of the prefix and suffix for the TK1 can be up to 127.
- **TK3:** For the prefix string, it is appended in the front of the start sentinel of track 3. For the suffix, it is appended in the end of the end sentinel of track 3. The max data length of the prefix and suffix for the TK1 can be up to 127.

ISO

This group defines the start and end sentinel for each track. The sentinel is always used to extract the track data from the whole MSR data string. The data length for each sentinel is fixed to one character. Because there is ISO standard that defining the start and end sentinel for the three tracks. For the compatible reason, please do not modify the default value if possible.

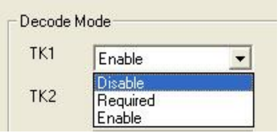
Decode Mode

For this group, it contains two kinds of settings, which are:

- Track Data Filtering: Determine which track to be, not to be output or needed to be output.
- Switch Output Order: Change the output order of track 1 ~ 3.

Track Data Filtering

Shown below is the filter setting for track 1. This provides a fool-proofing method in case of receiving unwanted or uncompleted track data.

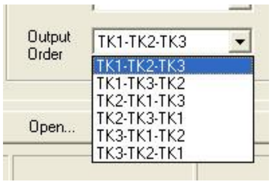


These three filter settings are:

- **Enable:** If selected, the data of specified track will be packaged in the MSR data string. If the specified track data is not decoded, it will leave blank in the MSR data string.
- **Required:** If selected, which means the output MSR data string must contain the specified track data. If the specified track data is not decoded, even MSR data string contains other track data, it will still not to be output.
- **Disable:** If selected, the data of specified track will not be packaged in the MSR data string. No matter it is decoded or not.

Switch Output Order

Show below is the selection of the three track data output order (sequence). The default order is Track 1–Track 2–Track 3.



There 6 orders allow to be selected. Please select one to fit your application needs.

Update Settings

Once complete the settings, click **Update** to update the settings to connected HID MSR reader.

Save Settings

To save the settings to a file, click **Save**; specify the file name and location to be saved.

Open Settings

To load pre-saved settings, click **Open**, specify the settings file, and then click OK to load into program.

Restore MSR Reader Settings

To load restore settings of connected MSR reader, click **Restore**.

ASCII Table

The ASCII table is used to enter the un-typed (by keyboard) characters.

OPOS MSR Register

The **OPOS MSR Register** program is used to set up the registry information of MSRHK reader for OPOS program uses.

Installation

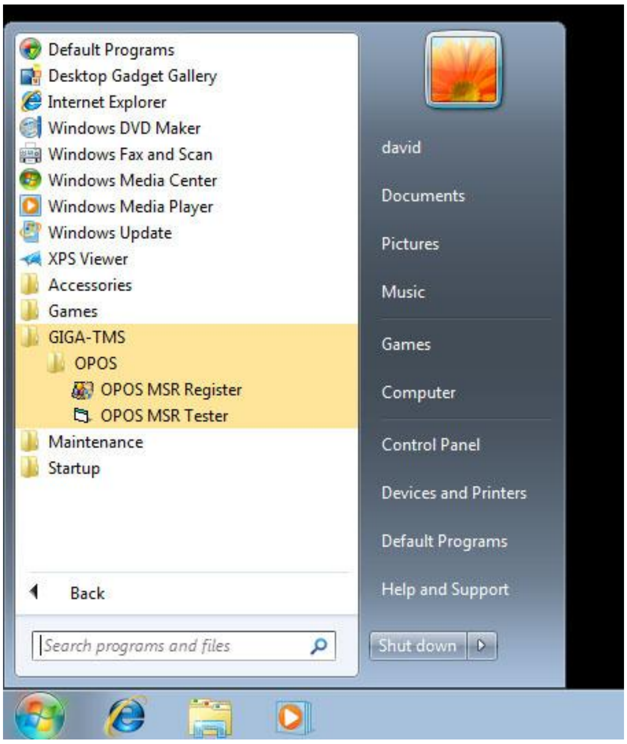
Below steps guide you how to install the **OPOS MSR Register** program.

- Insert the setup CD
- Run the **MSROPOSRegister_Tester_PSW00098.exe** setup file that is located in the Software folder of CD.
  * This setup also installs the OPOS MSR Tester program.
- Follow the wizard to complete the installation.

Launching Program

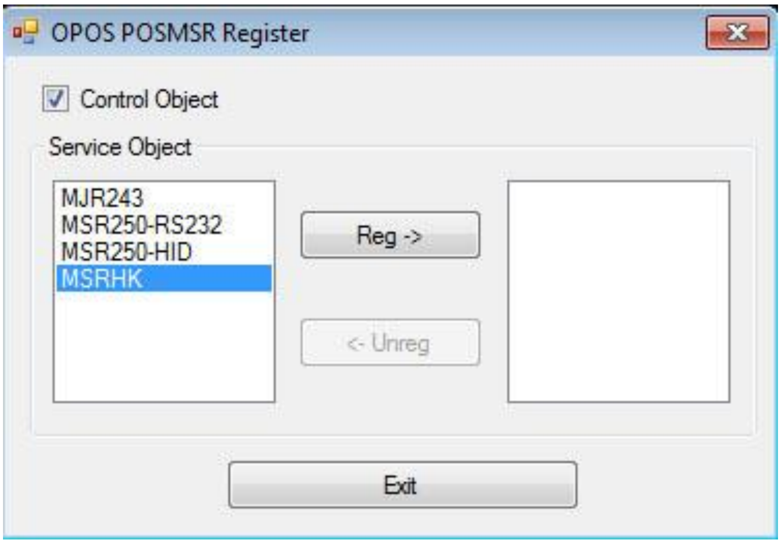
Below steps guide you how to load the **OPOS MSR Register** program.

- From **Start/Programs/GIGA-TMS**, click **OPOS** folder
- Click **OPOS MSR Register** to launch the program.



Configuration

Below is the main window of **OPOS MSR Register** program.

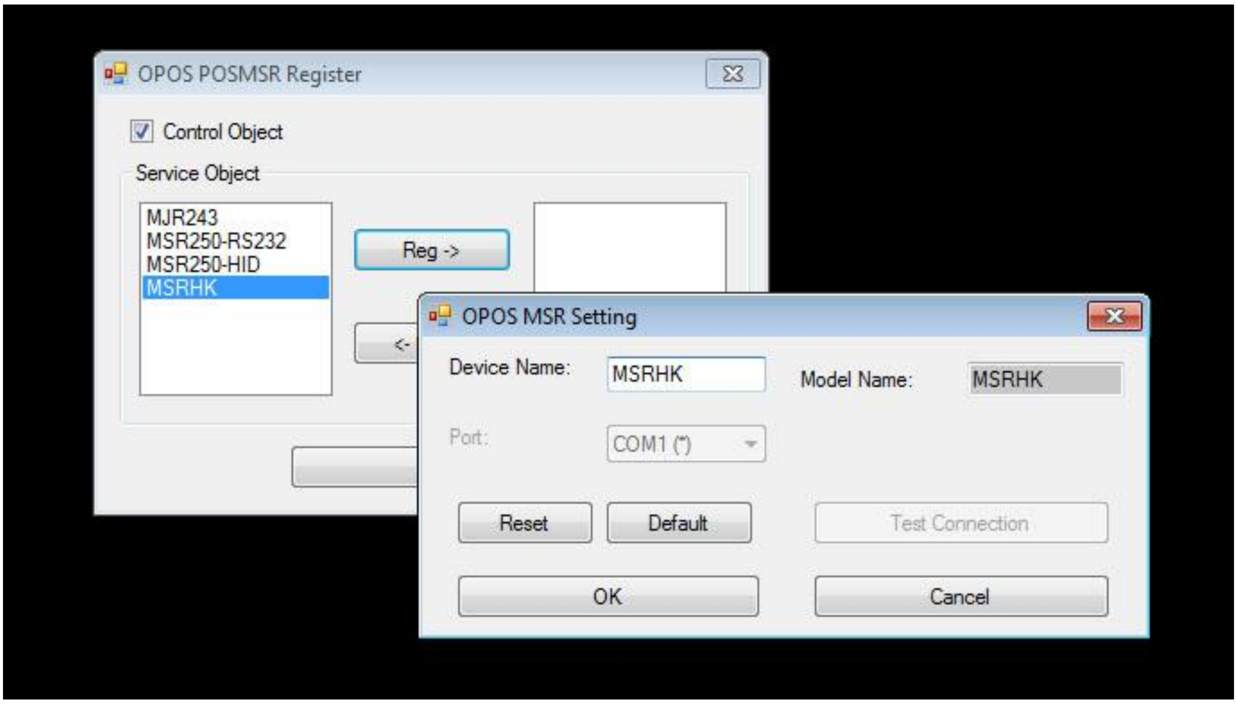


The main window has following components:

- **Control Object** check box: To register the OPOSMSR.ocx common control object driver. This needs to be checked to run the OPOS MSR Tester program.
- **Service Object** List box (left side): The Service Object driver types. So far only four types are supported. Each type support specific MSR readers. For more details, please refer to Types of Service Object for MSR Reader.
- **Service Object** List box (Right side): The registered MSR with specified device name.
- **Reg->** button: Create a new device name for selected MSR.
- **<-Unreg** button**:** Remove selected device name from registry.
- **Exit** button: End the program.

To register the MSRHK OPOS information, please do followings:

- Select an item in **Service Object** List box (left side), please refer to the table in Types of Service Object for MSR Reader to ensure the correct item is selected.
- Click **Reg->**
- In the **OPOS MSR Setting** Window, enter the *device name* and Click **OK**



Device Name: MSRHK

Model Name: MSRHK

Port: COM1 (*)

- If your system doesn’t have other Common Control driver, then click **Control Object** check box.
  Note: If you want to run the OPOPS MSR Tester program, then the **Control Object** must be checked.

Types of Service Object for MSR Reader

Service Object type	MSR Reader
MSR250-RS232	MAGTEK Port-power swipe reader
MSR250-HID	MAGTEK USB HID swipe reader
MJR243	PROMAG MSR 170/220/250/350/400 R/U(VC) PROMAG MJR 243/350/400 R/U(VC)
MSRHK	PROMAG MSR 170/220/250/350/400 K PROMAG MJR 243/350/400 K PROMAG MSR170/220/250/350/400 HK PROMAG MJR 243/350/400 HK

Registry Information

Below are the default registry settings for each Service Object type.

MSR250-RS232 type

	Key Name	**Type**	**Default Value**	**Note**
	CapISO	string	1	Capability for reading ISO track data
	CapJISOne	string	0	(reserved)
	CapJISTwo	string	0	(reserved)
	CapTransmitSentinels	string	1	Capability for reading Transmit Sentinels
	Debug	string	0	Enable the tracing, and create a log file
	Description	string	GIGATMS MSR POS	Description for SO driver
	DeviceName	string	MSR250-RS232	Devive Name for CO open
	FileName	string	(NULL)	(reserved)
	HardwareProvider	string	1	(reserved)
	Model	string	MSR250-RS232	Device model name
	Parity	string	None	Parity for the communication port
	Port	string	COM1	Comport Number
	Protocol	string	Hardware	Communication Control
	Baudrate	string	9600	RS232 baudrate

MSR250-HID type

	Key Name	**Type**	**Default Value**	**Note**
	CapISO	string	1	Capability for reading ISO track data

	CapJISOne	string	0	(reserved)
	CapJISTwo	string	0	(reserved)
	CapTransmitSentinels	string	1	Capability for reading Transmit Sentinels
	Debug	string	0	Enable the tracing, and create a log file
	Description	string	GIGATMS MSR POS	Description for SO driver
	DeviceName	string	MSR250-HID	Devive Name for CO open
	FileName	string	(NULL)	(reserved)
	HardwareProvider	string	1	(reserved)
	Model	string	MSR250-HID	Device model name
	Parity	string	None	Parity for the communication port
	Port	string	HID;VID=2049;PID=2	USB HID interface
	Protocol	string	Hardware	Communication Control

MJR243 type

	Key Name	**Type**	**Default Value**	**Note**
	CapISO	string	1	Capability for reading ISO track data
	CapJISOne	string	1	(reserved)
	CapJISTwo	string	1	(reserved)
	CapTransmitSentinels	string	1	Capability for reading Transmit Sentinels
	Debug	string	0	Enable the tracing, and create a log file
	Description	string	GIGATMS MSR POS	Description for SO driver
	DeviceName	string	MJR243	Devive Name for CO open
	FileName	string	(NULL)	(reserved)
	HardwareProvider	string	0	(reserved)
	Model	string	MJR243	Device model name
	Parity	string	None	Parity for the communication port
	Port	string	COM1	Comport Number
	Protocol	string	Hardware	Communication Control
	Baudrate	string	19200	RS232 baudrate

MSRHK type

	Key Name	**Type**	**Default Value**	**Note**
	CapISO	string	1	Capability for reading ISO track data
	CapJISOne	string	1	(reserved)
	CapJISTwo	string	1	(reserved)
	CapTransmitSentinels	string	1	Capability for reading Transmit Sentinels
	Debug	string	0	Enable the tracing, and create a log file
	Description	string	GIGATMS MSR POS	Description for SO driver
	DeviceName	string	MSRHK	Devive Name for CO open

	FileName	string	(NULL)	(reserved)
	HardwareProvider	string	0	(reserved)
	Model	string	MSRHK	Device model name
	Parity	string	NONE	Parity for the communication port
	Port	string	HOK	Hook driver
	Protocol	string	Hardware	Communication Control

OPOS APIs Supported List

Below are the OPOS supported APIs:

Type	Category	Type	Name	Mutability	OPOS APG Ver.	MSR.SO
Properties	common	bool	AutoDisable	R/W	1.2	Not Applicable
	common	long	BinaryConversion	R/W	1.2	Not Applicable
	common	bool	CapCompareFirmwareVersion	Read Only	1.9	Not Applicable
	common	long	CapPowerReporting	Read Only	1.3	Not Applicable
	common	bool	CapStatisticsReporting	Read Only	1.8	Not Applicable
	common	bool	CapUpdateFirmware	Read Only	1.9	Not Applicable
	common	bool	CapUpdateStatistics	Read Only	1.8	Not Applicable
	common	string	CheckHealthText	Read Only	1.0	Not Applicable
	common	bool	Claimed	Read Only	1.0	Supported
	common	long	DataCount	Read Only	1.2	Supported
	common	bool	DataEventEnabled	R/W	1.0	Supported
	common	bool	DeviceEnabled	R/W	1.0	Supported
	common	bool	FreezeEvents	R/W	1.0	Supported
	common	long	OpenResult	Read Only	1.5	Supported
	common	long	OutputID	Read Only	1.0	Not Applicable
	common	long	PowerNotify	R/W	1.3	Not Applicable
	common	long	PowerState	Read Only	1.3	Not Applicable
	common	long	ResultCode	Read Only	1.0	Supported
	common	long	ResultCodeExtended	Read Only	1.0	Supported
	common	long	State	Read Only	1.0	Supported
	common	string	ControlObjectDescription	Read Only	1.0	Supported
	common	long	ControlObjectVersion	Read Only	1.0	Supported
	common	string	ServiceObjectDescription	Read Only	1.0	Supported
	common	long	ServiceObjectVersion	Read Only	1.0	Supported
	common	string	DeviceDescription	Read Only	1.0	Supported
	common	string	DeviceName	Read Only	1.0	Supported
	Specific	bool	CapISO	Read Only	1.0	Supported
	Specific	bool	CapJISOne	Read Only	1.0	Not Applicable
	Specific	bool	CapJISTwo	Read Only	1.0	Not Applicable

	Specific	bool	CapTransmitSentinels	Read Only	1.5	Supported
	Specific	long	CapWriteTracks	Read Only	1.10	Not Applicable
	Specific	string	AccountNumber	Read Only	1.0	Supported
	Specific	bool	DecodeData	R/W	1.0	Supported
	Specific	long	EncodingMaxLength	Read Only	1.10	Not Applicable
	Specific	long	ErrorReportType	R/W	1.2	Not Applicable
	Specific	string	ExpirationDate	Read Only	1.0	Supported
	Specific	string	FirstName	Read Only	1.0	Supported
	Specific	string	MiddleInitial	Read Only	1.0	Supported
	Specific	bool	ParseDecodeData	R/W	1.0	Supported
	Specific	string	ServiceCode	Read Only	1.0	Supported
	Specific	string	Suffix	Read Only	1.0	Supported
	Specific	string	Surname	Read Only	1.0	Supported
	Specific	string	Title	Read Only	1.0	Supported
	Specific	binary	Track1Data	Read Only	1.0	Supported
	Specific	binary	Track1DiscretionaryData	Read Only	1.0	Supported
	Specific	binary	Track2Data	Read Only	1.0	Supported
	Specific	binary	Track2DiscretionaryData	Read Only	1.0	Supported
	Specific	binary	Track3Data	Read Only	1.0	Supported
	Specific	binary	Track4Data	Read Only	1.5	Not Applicable
	Specific	long	TracksToRead	R/W	1.0	Supported
	Specific	long	TracksToWrite	R/W	1.10	Not Applicable
	Specific	bool	TransmitSentinels	R/W	1.5	Supported
Methods	common		Open		1.0	Supported
	common		Close		1.0	Supported
	common		Claim		1.0	Supported
	common		ClaimDevice		1.5	Supported
	common		Release		1.0	Supported
	common		ReleaseDevice		1.5	Supported
	common		CheckHealth		1.0	Not Applicable
	common		ClearInput		1.0	Supported
	common		ClearInputProperties		1.10	Supported
	common		ClearOutput		1.0	Not Applicable
	common		DirectIO		1.0	Not Applicable
	common		CompareFirmwareVersion		1.9	Not Applicable
	common		ResetStatistics		1.8	Not Applicable
	common		RetrieveStatistics		1.8	Not Applicable
	common		UpdateFirmware		1.9	Not Applicable
	common		UpdateStatistics		1.8	Not Applicable
Events	common		DataEvent		1.0	Supported
	common		DirectIOEvent		1.0	Not Applicable
	common		ErrorEvent		1.0	Not Applicable

	common	OutputCompleteEvent		1.0	Not Applicable
	common	StatusUpdateEvent		1.0	Not Applicable

OPOS MSR Tester

The **OPOS MSR Tester** program is used to get the track data of MSRHK reader via the OPOS driver. Before running this program, make sure the device name registry information for MSRHK reader is already created by OPOS MSR Register program.

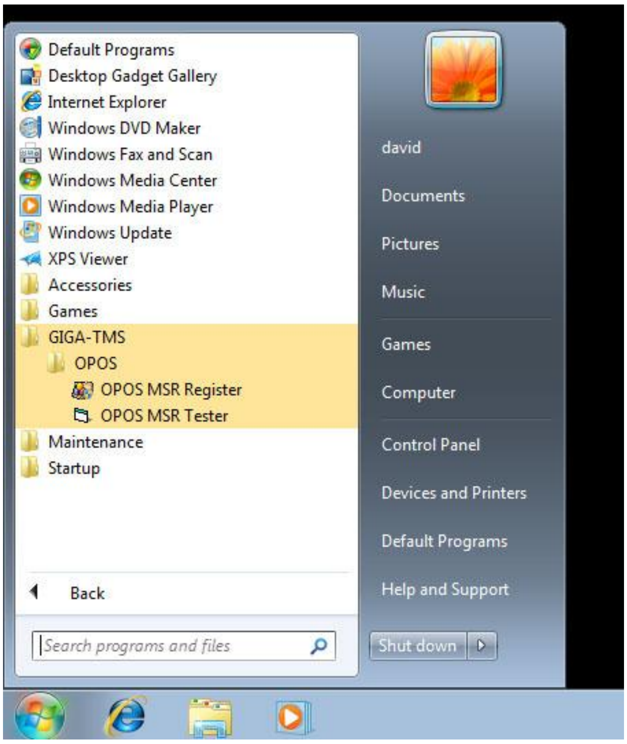
Installation

The installation of **OPOS MSR Tester** program is together with OPOS MSR Register program. Refer this for details.

Launching Program

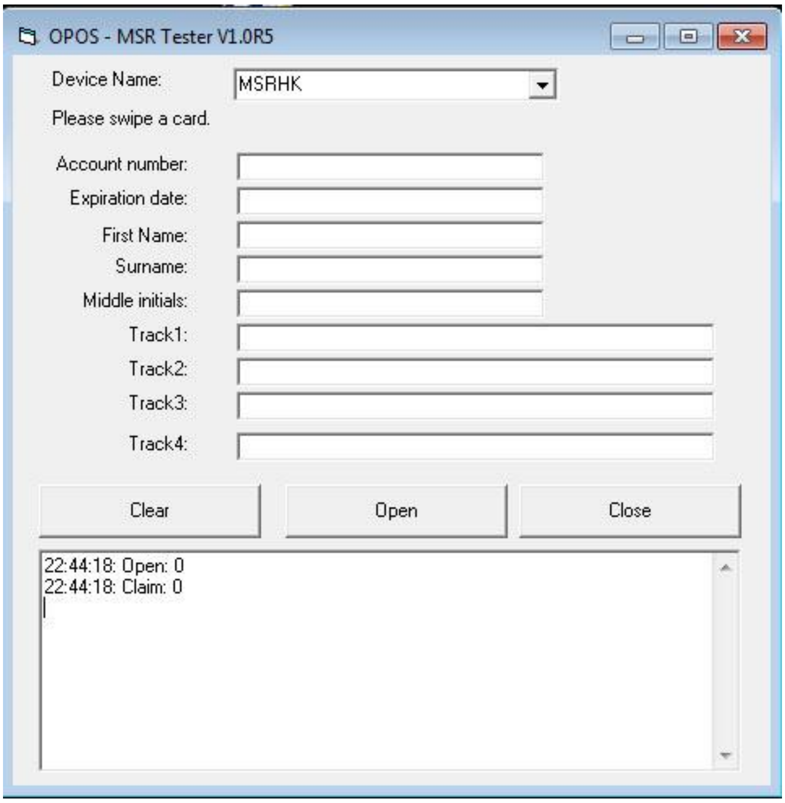
Below steps guide you how to load the **OPOS MSR Tester** program.

- From **Start/Programs/GIGA-TMS**, click **OPOS** folder
- Click **OPOS MSR Tester** to launch the program.



Configuration

Below is the main window of **OPOS MSR Tester** program.



Device Name: MSRHK

Please swipe a card.

```
22:44:18: Open: 0
22:44:18: Claim: 0
```

The main window has following components:

- **Device Name** combo box: Enter the device name that to be loaded to the program.
- **Track Data** Text boxes: Show the raw and parsed track data.
- **Clear** button: Clear all the track data in the text boxes.
- **Open** button: Open the OPOS driver and ready to get track data.
- **Close** button: Close the OPOS driver.
- **Message** text box: Display the result message of running the OPOS driver.

To start using OPOS driver to get track data, please do the following steps:

- Entering the **Device Name**.
- Clicking **Open** button.
- Swiping card to get track data.