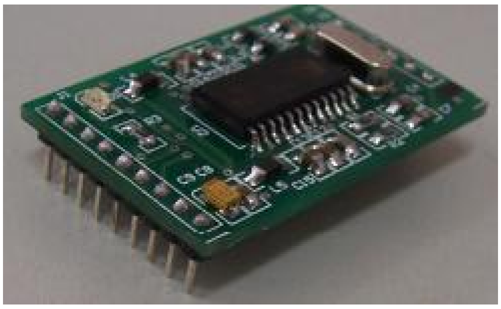
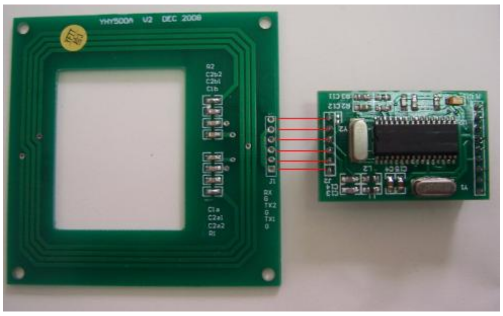
# YHY502CG

## 13.56MHz RFID Mifare® Read/Write Module

EHUOYAN

# DATASHEET





- ▲ Complete Read/Write module
- ▲ Auto checks for presence of a tag
- ▲ Contactless operating frequency 13.56 MHz
- ▲ Supports ISO14443A /MIFARE® ,Mifare® Classic1K,Mifare® Classic 4K
- ▲ TTL RS232 Interface, baud rate19200bps
- ▲ Fast data transfer Contactless communication up to 106KHz
- ▲ Secure Encrypted contactless communication
- ▲ Ideal for emoney,secure access and fast data collection applications
- ▲ Typical Operating Distance: 0 – 60 mm
- ▲ Operating Voltage ： DC 3.0-5.5V
- ▲ Watchdog timer
- ▲ 1 LED indicator
- ▲ Unique serial number on each device
- ▲ Size: 40mm × 25mm × 6mm

# Scope

This document describes the basic functionality and the electric specifications of the YHY502CG read/write module.

This contactless module is designed for an easy reader adaptation to a host to use this device for test and application purpose.

There is need only one command to finish one action, such as read or write card data. It needs no request, anticoll or selection.The module will do it for you automatically. What you need is just send one command to the module. Then it will send back what you want. Anything is just so **simple** and so **easy**.

Also, if there is any card go into the rf field, the red led on the module will light and the SIG pin will change from “1” to “0” to indicate the event.

# 1. Pin Information

**YHY502CG** is a module that is integrated with 8bit microcontroller, analog&digital signal processor and necessary passive components on both top layer to complete a fully functional Mifare® read/write module.

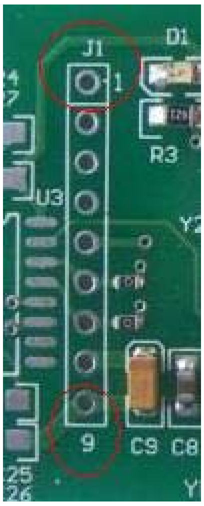


***Figure 1*** *– YHY502CG J1 PinOut TOPView*

J1 Interface:

| Pin | Symbol | IO Type | Description |
|---|---|---|---|
| J1-1 | RXD | I/O | Uart Receiver |
| J1-2 | TXD | I/O | Uart Transmitter |
| J1-3 | OUT1 | I/O | Output 1 |
| J1-4 | OUT2 | I | Output 2 |
| J1-5 | RST | I | Reset, active-low, floating for power-on reset by default |
| J1-6 | BUZ | I | Buzzer output, high level drive |
| J1-7 | SIG | O | Interrupt output, LOW level indicates card in the field |
| J1-8 | VCC | Power | Power positive |
| J1-9 | GND | GND | Power Negative |

***Table1*** *– J1 Pin information*

# 2. Introduction

YHY502CG is a compact 13.56MHz RFID Read / Write module designed for ISO14443A standard and supports Mifare® Classic 1K, Mifare® Classic 4K transponders. It is controlled by external device over UART with simple protocols defined in this sheet.

YHY502CG can be easily and quickly integrated into RFID applications with very less effort. Mifare® Classic is a secure memory (1Kbyte, 4KByte) chip/card often called contactless smart card. The reason it is called smartcard is because it has increment and decrement functions designed for especially payment systems. Mifare® Classic family of tags is being used in RFID applications where very high security and fast data collection systems are required. This family of tags has contactless communication speed up to 106 KHz and uses very strong encryption techniques. If the user want to copy or modify the content of the Mifare®Classic family of tags then he needs the correct key(s) when it is protected. As a result Mifare® become ideal for e-money applications, secure access, data storage and fast data collection systems. Not only limited with these applications but printed antenna technology makes possible to find very thin and low cost Mifare® tags (e.g. labels,stickers) so that extending the field of RFID applications.

# 3. Mifare®Brief Technical Information

For Mifare® tag memory organization and communication principles please refer to Mifare® S50 en.pdf document (Standard Card IC MF1 IC S50) of NXP. Mentioned document gives functional specification of the IC used in Mifare® 1K tags. Same communication principles are valid for Mifare® 4K (MF1 IC S70) tags. Documents can be downloaded at http://www.nxp.com. Communication principles are greatly simplified by YHY502CG module. When read or write the card, it just need to send one read/write command with keyA/B for authenticate, the module would perform request, anticoll and select card itself.

# 4. Communication Protocols

## 4.1 Command lists:

This chapter describes the protocol and commands which is used by the YHY502CG to communication with host.

| Code | Command | Description |
|---|---|---|
| 0x01 | Module Type | Read Module Type |
| 0x02 | Module Serial Number | Read Module Serial Number |
| 0x03 | Power Down | Set Module Power Down |
| 0x10 | Firmware Version | Read Module Firmware Version |
| 0x11 | Antenna control | Set Module Antenna on or off |
| 0x12 | Card IDLE | Set Card IDLE |
| 0x13 | Seek | Set Auto-Search Card |
| 0x14 | Beep | Set Buzzer ON/OFF |
| 0x15 | Beep interval | Set buzzer beep interval time |
| 0x16 | Output1 | Set Output 1 |
| 0x17 | Output2 | Set Output 2 |
| 0x19 | Card type | Read Card Type |
| 0x20 | Card serial number | Read card serial number |
| 0x21 | Block Read | Read Card Block data, 16 bytes |
| 0x22 | Block Write | Write Card Block data, 16 bytes |
| 0x23 | Initialize epurse | Initialize one block into epurse value |
| 0x24 | Value read | Read ePurse Value, 4 bytes |
| 0x25 | Increment | Increase ePurse Value, 4 bytes |
| 0x26 | Decrement | Decrease Purse Value, 4 bytes |
| | | |
| | | |

**Table2** – Command list

## 4.2 Protocol

**UART:（default：19200bps,N,8,1 ）**

The communication between the host and the module communicates at 19200bps, N, 8, 1.

The host first sends the command and the module executes the operation and replies with a response to the command. The host can analyze the reply to check if the operation was successful or if any error occurred during the operation.

Following is the UART frame for the commands sent by the host:

| Header | Length | Command | Data | CSUM |
|---|---|---|---|---|
| 2 Byte | 1 Byte | 1 Byte | N Bytes | 1 Byte |

***Table 4.2-1** – UART frame send by Host*

**1. Header:** This header has 2 bytes that indicates the beginning of a frame. These 2 bytes should be always 0xAA 0xBB.

**2. Length:** This byte is used to indicate the length of the payload data. This includes the Length, Command and the Data bytes

**3. Command:** This byte is used to instruct the module on what operation to perform

**4. Data:** These are parameters for the module to execute the command. For example, for a Read command, the data will be the block number to be read and the authenticated key. For a Write command, this will be the block number and the authenticated key and 16 bytes data to write into the block. For other command,it maybe empty.

**5. CSUM:** This is the checksum byte. This byte is used on the host as well as the module to check the validity of the packet and to trap any data corruption. This is calculated by **XOR** all the bytes in the packet except the Header and CSUM byte.

**CSUM=Length ⊕ Command ⊕ Data[0] ⊕ Data[1]… ⊕ Data[n-1]**

Note: If there is one byte “0xAA” in the packet data from Length to CSUM，please insert one byte “0x00” after “0xAA”,but the Length need not change.

Code example:

```
//------------------------------------------------
if (cSendBuffer[i] == 0xAA)
{
        TI = 0;
        SBUF = 0;
        while (!TI);
}
//------------------------------------------------
```

Following is the UART frame for the response packets sent by YHY502CG module in response to the commands:

| Header | Length | Status | Response | CSUM |
|---|---|---|---|---|
| 2 Byte | 1 Byte | 1 Byte | N Bytes | 1 Byte |

**Table 4** – UART frame send by YHY502CG module

**1. Header:** This header has 2 bytes that indicates the beginning of a frame. These 2 bytes should be always 0xAA 0xBB.

**2. Length:** This byte is used to indicate the length of the payload data. This includes the Length, Command and the Data bytes

**3. Status:** This is the status for which the response is being sent back. If ok then the module return the command which host has sent, else it return the ones-complement code. For example, the command is 0x19, then the ones-complement code is 0xe6.

4. **Response**: This contains the result data if an operation was successful. It may be empty.

**5. CSUM:** This is the checksum byte. This byte is used on the host as well as the module to check the validity of the packet and to trap any data corruption. This is calculated by **XOR** all the bytes in the packet except the Header and CSUM byte.

**CSUM=Length ⊕ Command ⊕ Response[0] ⊕ Response[1] ⊕ … ⊕ Response[n-1]**

## *5. Commands & Respones*

In this chapter detailed information and UART frame examples are given for command and responses.

After power on the module, the red led will flash one time.

If a Mifare® tag detected by the module, the red led will light and it will set SIG pin to “0” till the tag moves out of field.

（The following data are not specified is in hexadecimal）

### 5.1 Module Type

**Command description**: Read module type

Data Frame Format:

| Send | Head | Length | Command | XOR Checksum |
|---|---|---|---|---|
| | AA BB | 02 | 01 | 03 |

| Receive | Head | Length | Command | Module Type | XOR Checksum |
|---|---|---|---|---|---|
| Success | AA BB | 0A | 01 | 8Bytes | XOR Checksum |
| Failure | AA BB | 02 | FE | - | FC |

Example:

| | | |
|---|---|---|
| Send | AA BB 02 01 *03* | |
| Description | AA BB<br>02<br>01<br>*03* | *Head of this COMMAND*<br>*Length of this COMMAND*<br>*COMMAND*<br>*02 ⊕01* |
| Receive(Success) | AA BB 0A 01 *48 59 35 30 32 43 20 20* *6E* | |
| Description | AA BB<br>0A<br>01<br>*48 59 35 30 32 43 20 20*<br>*6E* | *Head of this DATA*<br>*Length of this DATA*<br>*COMMAND*<br>*Module TYPE*<br>*0A ⊕01 ⊕48 ⊕59 ⊕35 ⊕30 ⊕32 ⊕43 ⊕20 ⊕20* |
| Receive(Failure) | AA BB 02 FE *FC* | |
| Description | AA BB<br>02<br>FE<br>*FC* | *Head of this DATA*<br>*Length of this DATA*<br>*One's complement of COMMAND*<br>*02 ⊕FE* |

### 5.2 Module Serial Number

Command description: Read Module Serial Number

Note: Each module has it’s unique serial number.(NOT card serial number)

Data Frame Format:

| Send | Head | Length | Command | XOR Checksum |
|---|---|---|---|---|
| | AA BB | 02 | 02 | 00 |

| Receive | Head | Length | Command | Module SN | XOR Checksum |
|---|---|---|---|---|---|
| Success | AA BB | 06 | 02 | 4Bytes | XOR Checksum |
| Failure | AA BB | 02 | FD | - | FF |

Example:

| | | |
|---|---|---|
| Send | AA BB 02 02 *00* | |
| Description | AA BB<br>02<br>02 | *Head of this COMMAND*<br>*Length of this COMMAND*<br>*COMMAND* |

| | | |
|---|---|---|
| | 00 | 02 ⊕02 |
| Receive(Success) | **AA BB 06 02 00 00 00 01 05** | |
| Description | AA BB<br>06<br>02<br>00 00 00 01<br>05 | Head of this DATA<br>Length of this DATA<br>COMMAND<br>Module SN<br>06 ⊕02 ⊕00 ⊕00 ⊕00 ⊕01 |
| Receive(Failure) | **AA BB 02 FD FF** | |
| Description | AA BB<br>02<br>FD<br>FF | Head of this DATA<br>Length of this DATA<br>One's complement of COMMAND<br>02 ⊕FD |

## 5.3 Power Down

Command description: After execute this Command the module will power down , To wake up the module need to give the RST pin a low-level pulse or Re-power on.

Data Frame Format:

| **Send** | Head | Length | Command | XOR Checksum |
|---|---|---|---|---|
| | AA BB | 02 | 03 | 01 |

| **Receive** | Head | Length | Command | XOR Checksum |
|---|---|---|---|---|
| Success | AA BB | 02 | 03 | 01 |
| Failure | AA BB | 02 | FC | FE |

Example:

| | | |
|---|---|---|
| Send | **AA BB 02 03 01** | |
| Description | AA BB<br>02<br>03<br>01 | Head of this COMMAND<br>Length of this COMMAND<br>COMMAND<br>02 ⊕03 |
| Receive(Success) | **AA BB 02 03 01** | |
| Description | AA BB<br>02<br>03<br>01 | Head of this DATA<br>Length of this DATA<br>COMMAND<br>02 ⊕03 |
| Receive(Failure) | **AA BB 02 FC FE** | |
| Description | AA BB<br>02<br>FC<br>FE | Head of this DATA<br>Length of this DATA<br>One's complement of COMMAND<br>02 ⊕FC |

## 5.4 Module Firmware Version

Command description: Read Module Firmware Version

Data Frame Format:

| **Send** | Head | Length | Command | XOR Checksum |
|---|---|---|---|---|
| | AA BB | 02 | 10 | 12 |

| **Receive** | Head | Length | Command | Module Firmware Version | XOR Checksum |
|---|---|---|---|---|---|
| Success | AA BB | 06 | 10 | 4Bytes | XOR Checksum |
| Failure | AA BB | 02 | EF | - | ED |

Example:

| Send | AA BB 02 10 *12* | |
|---|---|---|
| Description | AA BB | *Head of this COMMAND* |
| | 02 | *Length of this COMMAND* |
| | 10 | *COMMAND* |
| | *12* | *02 ⊕10* |
| Receive(Success) | AA BB 06 10 ***00 00 02 01*** *15* | |
| Description | AA BB | *Head of this DATA* |
| | 06 | *Length of this DATA* |
| | 10 | *COMMAND* |
| | *00 00 02 01* | *Module SN* |
| | *15* | *06 ⊕10 ⊕00 ⊕00 ⊕02 ⊕01* |
| Receive(Failure) | AA BB 02 EF *ED* | |
| Description | AA BB | *Head of this DATA* |
| | 02 | *Length of this DATA* |
| | EF | *One's complement of COMMAND* |
| | *ED* | *02 ⊕EF* |

## 5.5 Antenna control

Command description: Set the Module antenna power on or off .This command will switch RF field.

Data Frame Format:

| **Send** | Head | Length | Command | Data | XOR Checksum |
|---|---|---|---|---|---|
| | AA BB | 03 | 11 | 1Byte<br>'*00*': antenna off<br>'*01*': antenna on | XOR Checksum |

| **Receive** | Head | Length | Command | XOR Checksum |
|---|---|---|---|---|
| Success | AA BB | 02 | 11 | 13 |
| Failure | AA BB | 02 | EE | EC |

Example:

| Send | AA BB 03 11 00 *12* | |
|---|---|---|
| Description | AA BB | *Head of this COMMAND* |
| | 03 | *Length of this COMMAND* |
| | 11 | *COMMAND* |
| | 00 | *00:* antenna off |
| | *12* | *03 ⊕11 ⊕00* |
| Receive(Success) | AA BB 02 11 *13* | |
| Description | AA BB | *Head of this DATA* |
| | 02 | *Length of this DATA* |
| | 11 | *COMMAND* |
| | *13* | *02 ⊕11* |
| Receive(Failure) | AA BB 02 EE *EC* | |
| Description | AA BB | *Head of this DATA* |
| | 02 | *Length of this DATA* |
| | EE | *One's complement of COMMAND* |
| | *EC* | *02 ⊕EE* |

## 5.6 Card IDLE

Command description: Set the Card into IDLE . After successfully operation the card will be idle. Reactivate the card need to remove the card from antenna area and put the card into antenna area again.

Data Frame Format:

| **Send** | Head | Length | Command | XOR Checksum |
|---|---|---|---|---|
| | AA BB | 02 | 12 | 10 |

| **Receive** | Head | Length | Command | XOR Checksum |
|---|---|---|---|---|
| Success | AA BB | 02 | 12 | 10 |
| Failure | AA BB | 02 | ED | EF |

Example:

| Send | **AA BB 02 12 *10*** | |
|---|---|---|
| Description | AA BB<br>02<br>12<br>*10* | *Head of this COMMAND*<br>*Length of this COMMAND*<br>*COMMAND*<br>*02 ⊕12* |
| Receive(Success) | **AA BB 02 12 *10*** | |
| Description | AA BB<br>02<br>12<br>*10* | *Head of this DATA*<br>*Length of this DATA*<br>*COMMAND*<br>*02 ⊕12* |
| Receive(Failure) | **AA BB 02 ED *EF*** | |
| Description | AA BB<br>02<br>ED<br>*EF* | *Head of this DATA*<br>*Length of this DATA*<br>*One's complement of COMMAND*<br>*02 ⊕ED* |

### 5.7 Seek

Command description: Set the module automatic search cards, 1 byte of data, 0x01 open automatic search cards, 0x00 closed. SIG pin active low when find a card untill remove the card or card idle.

Data Frame Format:

| **Send** | Head | Length | Command | Data | XOR Checksum |
|---|---|---|---|---|---|
| | AA BB | 03 | 13 | 1Byte<br>‘*01*’: seek on<br>‘*00*’: seek off | XOR Checksum |

| **Receive** | Head | Length | Command | XOR Checksum |
|---|---|---|---|---|
| Success | AA BB | 02 | 13 | 11 |
| Failure | AA BB | 02 | EC | EE |

Example:

| Send | **AA BB 03 13 00 *10*** | |
|---|---|---|
| Description | AA BB<br>03<br>13<br>00<br>*10* | *Head of this COMMAND*<br>*Length of this COMMAND*<br>*COMMAND*<br>*00: auto off*<br>*03 ⊕13 ⊕00* |
| Receive(Success) | **AA BB 02 13 *11*** | |

| | | |
|---|---|---|
| Description | AA BB<br>02<br>13<br>*11* | *Head of this DATA*<br>*Length of this DATA*<br>*COMMAND*<br>*02 ⊕13* |
| Receive(Failure) | **AA BB 02 EC *EE*** | |
| Description | AA BB<br>02<br>EC<br>*EE* | *Head of this DATA*<br>*Length of this DATA*<br>*One's complement of COMMAND*<br>*02 ⊕EC* |

## 5.8 Set Buzzer ON/OFF

Command description: Set the buzzer ON or OFF, and control the buzzer beep times.

Data Frame Format:

| **Send** | Head | Length | Command | Data | XOR Checksum |
|---|---|---|---|---|---|
| | AA BB | 03 | 14 | 1Byte<br>‘*1y*’: Buzzer ON and sound *y* times<br>‘*0F*’: Buzzer OFF | XOR Checksum |

| **Receive** | Head | Length | Command | XOR Checksum |
|---|---|---|---|---|
| Success | AA BB | 02 | 14 | 16 |
| Failure | AA BB | 02 | EB | E9 |

Example:

| | | |
|---|---|---|
| Send | **AA BB 03 14 13 *04*** | |
| Description | AA BB<br>03<br>14<br>13<br>*04* | *Head of this COMMAND*<br>*Length of this COMMAND*<br>*COMMAND*<br>*beep 3 times*<br>*03 ⊕14 ⊕13* |
| Receive(Success) | **AA BB 02 14 *16*** | |
| Description | AA BB<br>02<br>14<br>*16* | *Head of this DATA*<br>*Length of this DATA*<br>*COMMAND*<br>*02 ⊕14* |
| Receive(Failure) | **AA BB 02 EB *E9*** | |
| Description | AA BB<br>02<br>EB<br>*E9* | *Head of this DATA*<br>*Length of this DATA*<br>*One's complement of COMMAND*<br>*02 ⊕EB* |

## 5.9 Set buzzer beep time interval

Command description: Set buzzer beep time interval .

Data Frame Format:

| **Send** | Head | Length | Command | Ringing Interval | XOR Checksum |
|---|---|---|---|---|---|
| | AA BB | 03 | 15 | 1Byte | XOR Checksum |

| **Receive** | Head | Length | Command | XOR Checksum |
|---|---|---|---|---|
| Success | AA BB | 02 | 15 | 17 |
| Failure | AA BB | 02 | EA | E8 |

Example:

| | | |
|---|---|---|
| Send | **AA BB 03 15 10** ***06*** | |
| Description | AA BB | *Head of this COMMAND* |
| | 03 | *Length of this COMMAND* |
| | 15 | *COMMAND* |
| | 10 | *Beep time Interval* |
| | *06* | *03 ⊕15 ⊕10* |
| Receive(Success) | **AA BB 02 15** ***17*** | |
| Description | AA BB | *Head of this DATA* |
| | 02 | *Length of this DATA* |
| | 15 | *COMMAND* |
| | *17* | *02 ⊕17* |
| Receive(Failure) | **AA BB 02 EA** ***E8*** | |
| Description | AA BB | *Head of this DATA* |
| | 02 | *Length of this DATA* |
| | EA | *One's complement of COMMAND* |
| | *E8* | *02 ⊕EA* |

## 5.10 Output 1

Command description: Set Output1

Data Frame Format:

| **Send** | Head | Length | Command | Data | XOR Checksum |
|---|---|---|---|---|---|
| | AA BB | 03 | 16 | 1Byte<br>*'00'*: Output *'1'*<br>*'01'*: Output *'0'* | XOR Checksum |

| **Receive** | Head | Length | Command | XOR Checksum |
|---|---|---|---|---|
| Success | AA BB | 02 | 16 | 14 |
| Failure | AA BB | 02 | E9 | EB |

Example:

| | | |
|---|---|---|
| Send | **AA BB 03 16 01** ***04*** | |
| Description | AA BB | *Head of this COMMAND* |
| | 03 | *Length of this COMMAND* |
| | 16 | *COMMAND* |
| | 01 | *Output 0* |
| | *04* | *03 ⊕16 ⊕01* |
| Receive(Success) | **AA BB 02 16** ***14*** | |
| Description | AA BB | *Head of this DATA* |
| | 02 | *Length of this DATA* |
| | 15 | *COMMAND* |
| | *17* | *02 ⊕17* |
| Receive(Failure) | **AA BB 02 E9** ***EB*** | |
| Description | AA BB | *Head of this DATA* |
| | 02 | *Length of this DATA* |
| | E9 | *One's complement of COMMAND* |
| | *EB* | *02 ⊕E9* |

## 5.11 Output 2

Command description: Set Output2

Data Frame Format:

| **Send** | Head | Length | Command | Data | XOR Checksum |
|---|---|---|---|---|---|

| | | | | | |
|---|---|---|---|---|---|
| | AA BB | 03 | 17 | 1Byte<br>*‘00’*: Output *‘1’*<br>*‘01’*: Output *‘01’* | XOR Checksum |

| **Receive** | Head | Length | Command | XOR Checksum |
|---|---|---|---|---|
| Success | AA BB | 02 | 17 | 15 |
| Failure | AA BB | 02 | E8 | EA |

Example:

| Send | **AA BB 03 17 01 *05*** | |
|---|---|---|
| Description | AA BB<br>03<br>17<br>01<br>*05* | *Head of this COMMAND*<br>*Length of this COMMAND*<br>*COMMAND*<br>*Output 0*<br>*03 ⊕17 ⊕01* |
| Receive(Success) | **AA BB 02 17 *15*** | |
| Description | AA BB<br>02<br>17<br>*15* | *Head of this DATA*<br>*Length of this DATA*<br>*COMMAND*<br>*02 ⊕17* |
| Receive(Failure) | **AA BB 02 E8 *EA*** | |
| Description | AA BB<br>02<br>E8<br>*EA* | *Head of this DATA*<br>*Length of this DATA*<br>*One's complement of COMMAND*<br>*02 ⊕E8* |

## 5.12 Card Type

Command description: Read card type. S50 card is *‘0x0400’*， S70 card is *‘0x0200’*， the others can refer to card datasheet.

Data Frame Format:

| **Send** | Head | Length | Command | XOR Checksum |
|---|---|---|---|---|
| | AA BB | 02 | 19 | 1B |

| **Receive** | Head | Length | Command | Card Type | XOR Checksum |
|---|---|---|---|---|---|
| Success | AA BB | 04 | 19 | 2Bytes | XOR Checksum |
| Failure | AA BB | 02 | E6 | - | E4 |

Example:

| Send | **AA BB 02 19 *1B*** | |
|---|---|---|
| Description | AA BB<br>02<br>19<br>*1B* | *Head of this COMMAND*<br>*Length of this COMMAND*<br>*COMMAND*<br>*02 ⊕19* |
| Receive(Success) | **AA BB 04 19 04 00 *19*** | |
| Description | AA BB<br>04<br>19<br>04 00<br>*19* | *Head of this DATA*<br>*Length of this DATA*<br>*COMMAND*<br>*Card TYPE 04 00: S50 Card; 02 00: S70 Card*<br>*02 ⊕19 ⊕04 ⊕00* |
| Receive(Failure) | **AA BB 02 E6 *E4*** | |
| Description | AA BB<br>02 | *Head of this DATA*<br>*Length of this DATA* |

| | | |
|---|---|---|
| | E6<br>*E4* | *One's complement of COMMAND*<br>*02 ⊕E6* |

## 5.13 Card serial number

Command description: This command reads card serial number

Data Frame Format:

| **Send** | Head | Length | Command | XOR Checksum |
|---|---|---|---|---|
| | AA BB | 02 | 20 | 22 |

| **Receive** | Head | Length | Command | Card SN | XOR Checksum |
|---|---|---|---|---|---|
| Success | AA BB | 06 | 20 | 4Bytes | XOR Checksum |
| Failure | AA BB | 02 | DF | - | DD |

Example:

| Send | **AA BB 02 20 *22*** | |
|---|---|---|
| Description | AA BB<br>02<br>20<br>*22* | *Head of this COMMAND*<br>*Length of this COMMAND*<br>*COMMAND*<br>*02 ⊕20* |
| Receive(Success) | **AA BB 06 20 *92 BF 72 59* *20*** | |
| Description | AA BB<br>06<br>20<br>*92 BF 72 59*<br>*20* | *Head of this DATA*<br>*Length of this DATA*<br>*COMMAND*<br>*Card SN*<br>*06 ⊕20 ⊕92 ⊕BF ⊕72 ⊕59* |
| Receive(Failure) | **AA BB 02 DF *DD*** | |
| Description | AA BB<br>02<br>DF<br>*DD* | *Head of this DATA*<br>*Length of this DATA*<br>*One's complement of COMMAND*<br>*02 ⊕DF* |

## 5.14 Block read

Command description: Read data from appointed card's block.

Data Frame Format:

| **Send** | Head | Length | Command | Key A or Key B | Block Number | Key | XOR Checksum |
|---|---|---|---|---|---|---|---|
| | AA BB | 0A | 21 | 1Byte<br>'*00*': Key A<br>'*01*': Key B | 1Byte | 6 Bytes | XOR Checksum |

| **Receive** | Head | Length | Command | Block Data | XOR Checksum |
|---|---|---|---|---|---|
| Success | AA BB | 12 | 21 | 16Bytes | XOR Checksum |
| Failure | AA BB | 02 | DE | - | DC |

Example:

| Send | **AA BB 0A 21 00 08 *FF FF FF FF FF FF* *23*** | |
|---|---|---|
| Description | AA BB<br>0A<br>21<br>00<br>08<br>*FF FF FF FF FF FF* | *Head of this COMMAND*<br>*Length of this COMMAND*<br>*COMMAND*<br>*Authenticate with A Key*<br>*Read Block 08*<br>*Keys* |

| | | |
|---|---|---|
| | 23 | 0A ⊕21 ⊕00 ⊕08 ⊕FF ⊕FF ⊕FF ⊕FF ⊕FF ⊕FF |
| Receive(Success) ( *) | AA BB 12 21 00 11 22 33 44 55 66 77 88 99 AA 00 BB CC DD EE FF 23 | |
| Description | AA BB<br>12<br>21<br>00 11 22 33 44 55 66 77<br>88 99 AA 00 BB CC DD<br>EE FF<br>23 | Head of this DATA<br>Length of this DATA<br>COMMAND<br>16 Bytes Data of Block 08<br><br><br>12 ⊕21 ⊕00 ⊕11 ⊕22 ⊕33 ⊕44 ⊕55 ⊕66 ⊕77 ⊕88 ⊕99 ⊕AA ⊕BB ⊕CC ⊕DD ⊕EE ⊕FF |
| Receive(Failure) | AA BB 02 DE DC | |
| Description | AA BB<br>02<br>DE<br>DC | Head of this DATA<br>Length of this DATA<br>One's complement of COMMAND<br>02 ⊕DE |

( *): 00 is added but the length does not change.

## 5.15 Block Write

Command description: Write data to appointed card's block.

Data Frame Format:

| **Send** | Head | Length | Command | Key A or Key B | Block Number | Key | Data want to write | XOR Checksum |
|---|---|---|---|---|---|---|---|---|
| | AA BB | 1A | 22 | 1Byte<br>'00': Key A<br>'01': Key B | 1Byte | 6 Bytes | 16Bytes | XOR Checksum |

| **Receive** | Head | Length | Command | XOR Checksum |
|---|---|---|---|---|
| Success | AA BB | 02 | 22 | 20 |
| Failure | AA BB | 02 | DD | DF |

Example:

| | | |
|---|---|---|
| Send (*) | AA BB 1A 22 00 08 FF FF FF FF FF FF 00 11 22 33 44 55 66 77 88 99 AA 00 BB CC DD EE FF 30 | |
| Description | AA BB<br>1A<br>22<br>00<br>08<br>FF FF FF FF FF FF<br>00 11 22 33 44 55 66 77<br>88 99 AA BB CC DD EE<br>FF<br>30 | Head of this COMMAND<br>Length of this COMMAND<br>COMMAND<br>Authenticate with A Key<br>Read Block 08<br>Keys<br>16 Bytes Data want to Write<br><br><br>1A ⊕21 ⊕00 ⊕08 ⊕FF ⊕FF ⊕FF ⊕FF ⊕FF ⊕FF ⊕00 ⊕11 ⊕22 ⊕33 ⊕44 ⊕55 ⊕66 ⊕77 ⊕88 ⊕99 ⊕AA ⊕BB ⊕CC ⊕DD ⊕EE ⊕FF |
| Receive(Success) | AA BB 02 22 20 | |
| Description | AA BB<br>02<br>22<br>20 | Head of this DATA<br>Length of this DATA<br>COMMAND<br>02 ⊕22 |
| Receive(Failure) | AA BB 02 DD DF | |
| Description | AA BB<br>02<br>DD | Head of this DATA<br>Length of this DATA<br>One's complement of COMMAND |

| | | |
|---|---|---|
| | DF | 02 ⊕DD |

( *): 00 is added but the length does not change.

### 5.16 Initialize ePurse

Command description: Initialize block as epurse value, the 4-byte purse value of command related to purse operation is low byte first, and the purse value is 4 bytes signed.

Data Frame Format:

| **Send** | Head | Length | Command | Key A or Key B | Block Number | Key | Purse Value | XOR Checksum |
|---|---|---|---|---|---|---|---|---|
| | AA BB | 0E | 23 | 1Byte<br>‘00’: Key A<br>‘01’: Key B | 1Byte | 6 Bytes | 4Bytes (LSB⋯MSB) | XOR Checksum |

| **Receive** | Head | Length | Command | XOR Checksum |
|---|---|---|---|---|
| Success | AA BB | 02 | 23 | 21 |
| Failure | AA BB | 02 | DC | DE |

Example:

| | | |
|---|---|---|
| Send | **AA BB 0E 23 00 09 *FF FF FF FF FF FF* *11 11 00 00* 24** | |
| Description | AA BB<br>0E<br>23<br>00<br>09<br>FF FF FF FF FF FF<br>11 11 00 00<br>*24* | *Head of this COMMAND*<br>*Length of this COMMAND*<br>*COMMAND*<br>*Authenticate with A Key*<br>*Initialize Block 09 as a Purse*<br>*Keys*<br>*4 Bytes Value of Purse*<br>*0E ⊕23 ⊕00 ⊕09 ⊕FF ⊕FF ⊕FF ⊕FF ⊕FF ⊕FF ⊕11 ⊕11 ⊕00 ⊕00* |
| Receive(Success) | **AA BB 02 23 *21*** | |
| Description | AA BB<br>02<br>23<br>*21* | *Head of this DATA*<br>*Length of this DATA*<br>*COMMAND*<br>*02 ⊕23* |
| Receive(Failure) | **AA BB 02 DC *DE*** | |
| Description | AA BB<br>02<br>DC<br>*DE* | *Head of this DATA*<br>*Length of this DATA*<br>*One's complement of COMMAND*<br>*02 ⊕DC* |

### 5.17 Read Purse Value

Command description: Read purse value.

Data Frame Format:

| **Send** | Head | Length | Command | Key A or Key B | Block Number | Key | XOR Checksum |
|---|---|---|---|---|---|---|---|
| | AA BB | 0A | 24 | 1Byte<br>‘00’: Key A<br>‘01’: Key B | 1Byte | 6Bytes | XOR Checksum |

| **Receive** | Head | Length | Command | Purse Value | XOR Checksum |
|---|---|---|---|---|---|
| Success | AA BB | 06 | 24 | 4Bytes (LSB⋯MSB) | XOR Checksum |
| Failure | AA BB | 02 | DB | - | D9 |

Example:

| | | |
|---|---|---|
| Send | **AA BB 0A 24 00 09 FF FF FF FF FF FF *27*** | |
| Description | AA BB | *Head of this COMMAND* |
| | 0A | *Length of this COMMAND* |
| | 24 | *COMMAND* |
| | 00 | *Authenticate with A Key* |
| | 09 | *Block 09 is a Purse* |
| | FF FF FF FF FF FF | *Keys* |
| | *27* | *0A ⊕24 ⊕00 ⊕09 ⊕FF ⊕FF ⊕FF ⊕FF ⊕FF ⊕FF* |
| Receive(Success) | **AA BB 06 24 *11 11 00 00* *22*** | |
| Description | AA BB | *Head of this DATA* |
| | 06 | *Length of this DATA* |
| | 24 | *COMMAND* |
| | *11 11 00 00* | *Value of Purse* |
| | *22* | *06 ⊕24 ⊕11 ⊕11 ⊕00 ⊕00* |
| Receive(Failure) | **AA BB 02 DB *D9*** | |
| Description | AA BB | *Head of this DATA* |
| | 02 | *Length of this DATA* |
| | DB | *One's complement of COMMAND* |
| | *D9* | *02 ⊕DB* |

## 5.18 Increase Purse Value

Command description: Increase purse value

Data Frame Format:

| **Send** | Head | Length | Command | Key A or Key B | Block Number | Key | Increase Value | XOR Checksum |
|---|---|---|---|---|---|---|---|---|
| | AA BB | 0E | 25 | 1Byte<br>*'00'*: Key A<br>*'01'*: Key B | 1Byte | 6 Bytes | 4Bytes (LSB⋯MSB) | XOR Checksum |

| **Receive** | Head | Length | Command | XOR Checksum |
|---|---|---|---|---|
| Success | AA BB | 02 | 25 | 27 |
| Failure | AA BB | 02 | DA | D8 |

Example:

| | | |
|---|---|---|
| Send | **AA BB 0E 25 00 09 FF FF FF FF FF FF *11 11 00 00* *22*** | |
| Description | AA BB | *Head of this COMMAND* |
| | 0E | *Length of this COMMAND* |
| | 25 | *COMMAND* |
| | 00 | *Authenticate with A Key* |
| | 09 | *Block 09 is a Purse* |
| | FF FF FF FF FF FF | *Keys* |
| | *11 11 00 00* | *Vlaue of Increase* |
| | *22* | *0E ⊕25 ⊕00 ⊕09 ⊕FF ⊕FF ⊕FF ⊕FF ⊕FF ⊕FF ⊕11 ⊕11 ⊕00 ⊕00* |
| Receive(Success) | **AA BB 02 25 *27*** | |
| Description | AA BB | *Head of this DATA* |
| | 02 | *Length of this DATA* |
| | 25 | *COMMAND* |
| | *27* | *02 ⊕25* |
| Receive(Failure) | **AA BB 02 DA *D8*** | |
| Description | AA BB | *Head of this DATA* |
| | 02 | *Length of this DATA* |
| | DA | *One's complement of COMMAND* |

| | | |
|---|---|---|
| | D8 | 02 ⊕DA |

## 5.19 Decrease Purse Value

Command description: Decrease purse value

Data Frame Format:

| **Send** | Head | Length | Command | Key A or Key B | Block Number | Key | Decrease Value | XOR Checksum |
|---|---|---|---|---|---|---|---|---|
| | AA BB | 0E | 26 | 1Byte<br>'*00*': Key A<br>'*01*': Key B | 1Byte | 6 Bytes | 4Bytes (LSB…MSB) | XOR Checksum |

| **Receive** | Head | Length | Command | XOR Checksum |
|---|---|---|---|---|
| Success | AA BB | 02 | 26 | 24 |
| Failure | AA BB | 02 | D9 | DB |

Example:

| Send | AA BB 0E 26 00 09 FF FF FF FF FF FF 11 11 00 00 21 | |
|---|---|---|
| Description | AA BB<br>0E<br>26<br>00<br>09<br>FF FF FF FF FF FF<br>11 11 00 00<br>21 | *Head of this COMMAND*<br>*Length of this COMMAND*<br>*COMMAND*<br>*Authenticate with A Key*<br>*Block 09 is a Purse*<br>*Keys*<br>*Vlaue of Decrease*<br>*0E ⊕26 ⊕00 ⊕09 ⊕FF ⊕FF ⊕FF ⊕FF ⊕FF ⊕FF ⊕11 ⊕11 ⊕00 ⊕00* |
| Receive(Success) | AA BB 02 26 24 | |
| Description | AA BB<br>02<br>26<br>24 | *Head of this DATA*<br>*Length of this DATA*<br>*COMMAND*<br>*02 ⊕26* |
| Receive(Failure) | AA BB 02 D9 DB | |
| Description | AA BB<br>02<br>D9<br>DB | *Head of this DATA*<br>*Length of this DATA*<br>*One's complement of COMMAND*<br>*02 ⊕D9* |

# 6. ELECTRICAL CHARACTERISTICS

## 6.1 ABSOLUTE MAXIMUM RATINGS

| SYMBOL | PARAMETER | MIN | MAX | UNIT |
|---|---|---|---|---|
| Tamb,abs | Ambient or Storage Temperature Range | -40 | +150 | °C |
| VDD | DC Supply Voltages | -0.5 | 6 | V |
| Vin,abs | Absolute voltage on any digital pin to GND | -0.5 | VDD +0.5 | V |

*Table 6-1: Absolute Maximum Ratings*

## 6.2 **Operating Condition Range**

| SYMBOL | PARAMETER | CONDITIONS | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|
| Tamb | Ambient Temperature | - | -25 | +25 | +85 | °C |
| VDD | DC Supply Voltages | GND = 0V | 3.0 | 3.3 | 3.6 | V |
| | | | 4.5 | 5.0 | 5.5 | V |
| RD | Reading Distance | VDD=5.0V | 0 | 50 | 60 | mm |
| | | VDD=3.3V | 0 | 35 | 50 | |
| WD | Writing Distance | VDD=5.0V | 0 | 45 | 55 | mm |
| | | VDD=3.3V | 0 | 30 | 45 | |

*Table 6-2: Operating Condition Range*

## 6.3 Current Consumption

| SYMBOL | PARAMETER | CONDITIONS | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|---|
| IVDD1 | Supply Current 1<br>VDD=4.5V-5.5V | Continuous read or write | | 70 | 150 | mA |
| | | Antenna Soft Power Down | | 11 | 20 | mA |
| | | Module Hard Power Down | | 70 | 160 | µA |
| IVDD2 | Supply Current 2<br>VDD=3.0V-3.6V | Continuous read or write | | 45 | 120 | mA |
| | | Antenna Soft Power Down | | 8 | 16 | mA |
| | | Module Hard Power Down | | 65 | 150 | µA |

*Table 6-3: Current Consumption*

# 7. Packaging Information

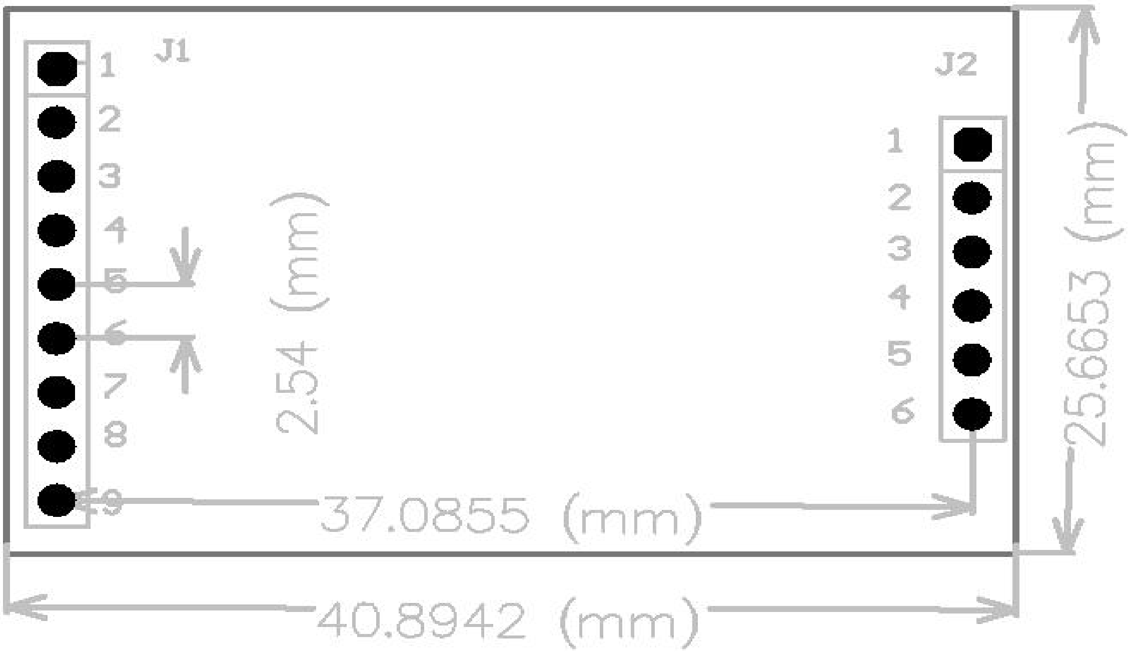


**Figure 7-1 –** Top View YHY502XG module size

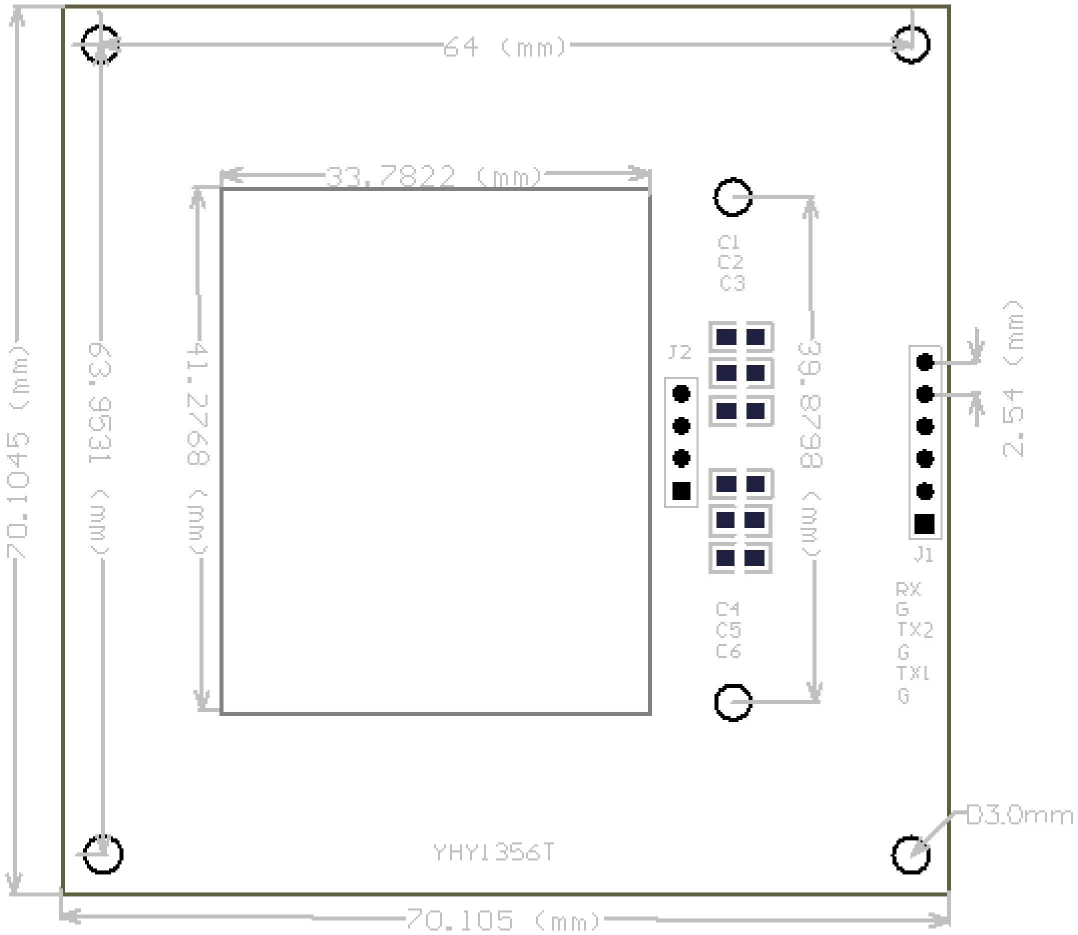


**Figure 7-2 –** Top View YHY1356T Antenna size

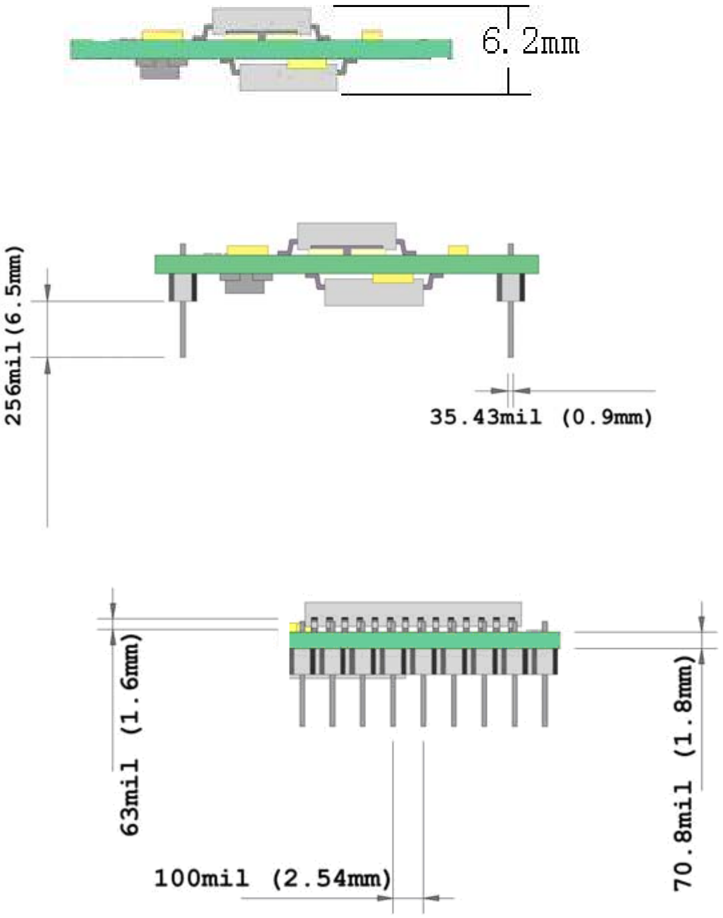


**Figure 7-3 –**Side View YHY502XG module

# 8. Sales And Service Information

To obtain information about EHUOYAN Tech products and technical support, reference the following information.

## Contact Information:

Ehuoyan Technology Co.,Ltd.
Rm 605 Tower 6 Qingyuan-Xili Long Gang
Lu Haidian District Beijing, China 100192
Tel: +86-010-59870151
Fax: +86-010-59754725
Email: info@ehuoyan.com
WebSite: www.ehuoyan.com