# Ethernet Protocol Stack Chip CH394

Version: 1.0

https://wch-ic.com

## 1. Overview

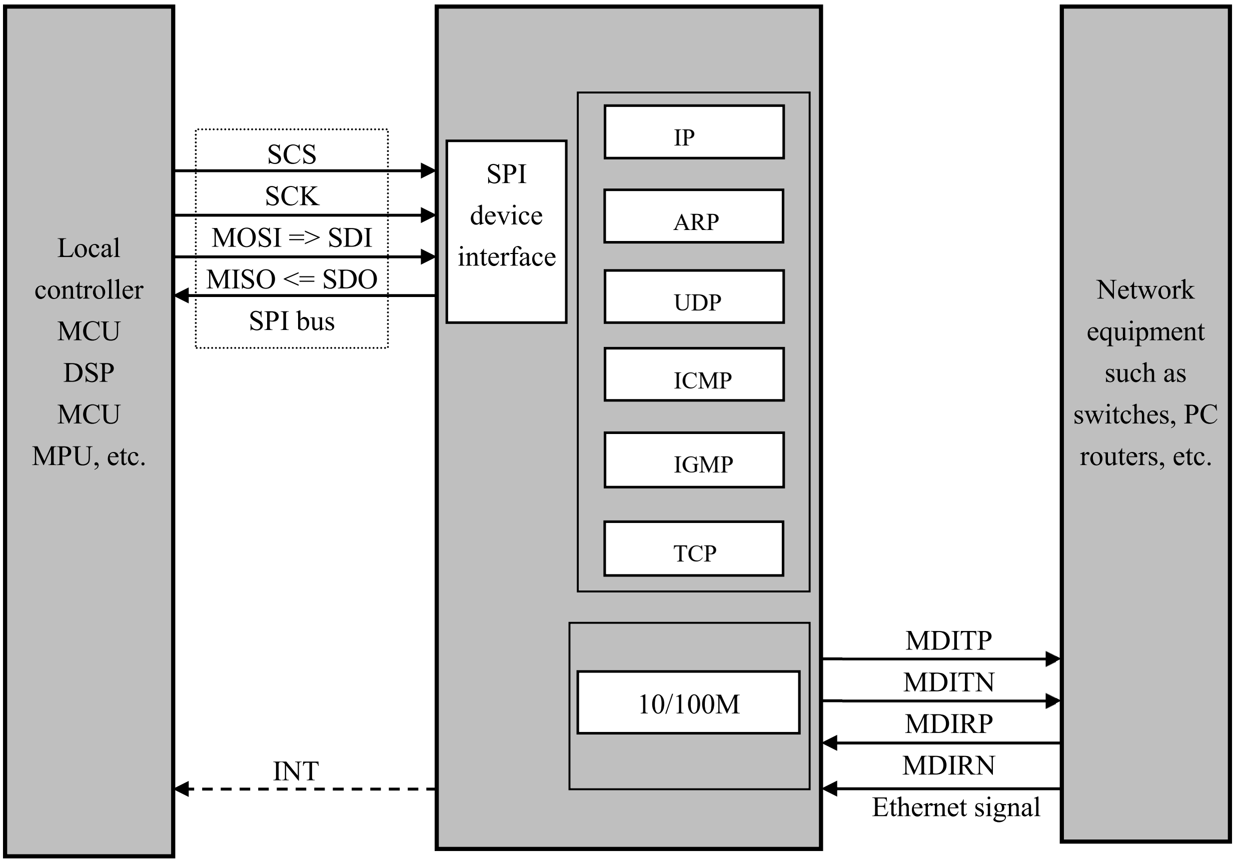
CH394 is an Ethernet protocol stack management chip, which is used for Ethernet communication in MCU system. CH394 chip comes with 10/100M Ethernet Media Access Control (MAC) and Physical layer (PHY), which is fully compatible with IEEE802.3 protocol, and has built-in Ethernet protocol stack firmware such as IP, ARP, ICMP, IGMP, UDP and TCP. MCU system can conveniently communicate with the network through CH394 chip. CH394 supports Wake-on-LAN (WOL) and power-down mode.

CH394Q provides SPI interface, CH394L provides SPI interface and 8-bit slave parallel interface. Controllers such as MCU/DSP/MCU/MPU can control CH394Q chip to communicate with Ethernet through SPI interface. Or through SPI interface or 8-bit parallel port to control CH394L chip for Ethernet communication.

The contents of this manual are mainly applicable to CH394Q and CH394L. For details, please refer to the CH394DS2 manual.

The following figure shows the application block diagram of CH394Q.

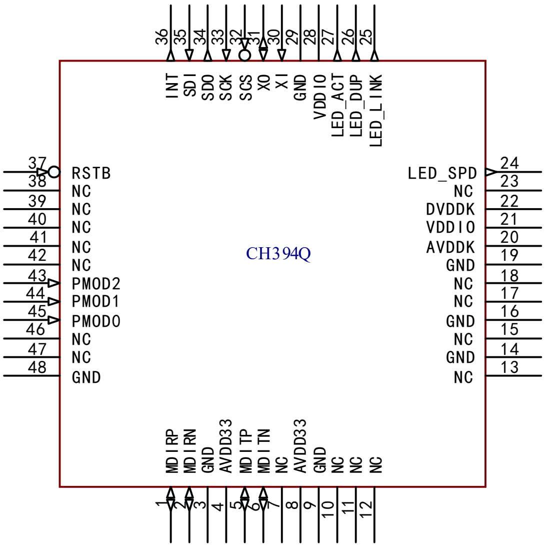
Figure 1-1 CH394Q application block diagram



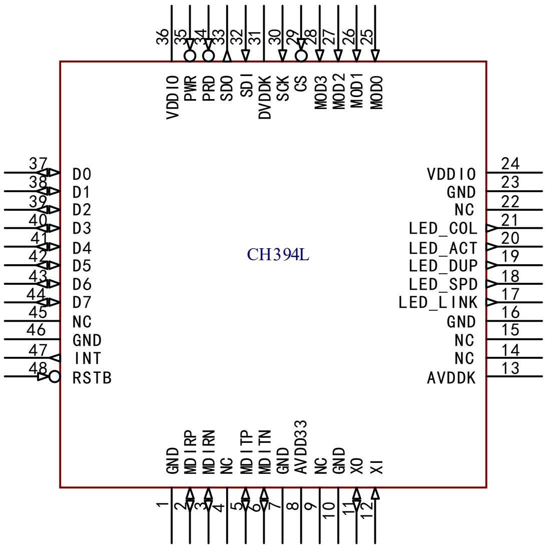
## 2. Features

- Built-in Ethernet Media Access Control (MAC) and Physical layer (PHY)
- Support 10/100M, full-duplex/half-duplex adaptation, and is compatible with 802.3 protocol
- Support automatic conversion of MDI/MDIX lines
- Built-in TCP/IP protocol suite. Support IPv4, ARP, ICMP, IGMP, UDP and TCP protocols
- CH394Q supports 8 sockets and CH394L supports 4 sockets. Can transmit and receive data at the same time
- Support MAC RAW mode and IP RAW mode (IP RAW mode is only supported by CH394L)
- Provide a SPI slave interface (SPI mode 0 or 3) up to 40MHz, with the high bit coming first
- The CH394L provides a high-speed 8-bit slave parallel interface that supports a parallel data bus connected to the microcontroller.
- Support network Wake-on-LAN (WOL) and power-down mode
- LED status display (Link, ACT, 10/100M, full-duplex/half-duplex, etc.)
- Built-in 32KRAM can be used for Ethernet data transceiver, and each Socket transceiver buffer can be configured flexibly
- LQFP48 lead-free package is provided.

# 3. Package



CH394Q



CH394L

Table 3-1 Package description

| Package form | Body size | Pin pitch | | Package description | Order model |
|---|---|---|---|---|---|
| LQFP48 | 7*7mm | 0.5mm | 19.7mil | Low Profile Quad Flat Pack | CH394Q |
| LQFP48 | 7*7mm | 0.5mm | 19.7mil | Low Profile Quad Flat Pack | CH394L |

# 4. CH394Q Pins

Table 4-1 CH394Q pin definition

| CH394Q Pin No. | Pin name | Type | Pin description |
|---|---|---|---|
| 1 | MDIRP | I/O | Differential input in 10BASE-T/100BASE-TX MDI mode; Differential output in 10BASE-T/100BASE-TX MDIX mode. |
| 2 | MDIRN | | |
| 5 | MDITP | I/O | Differential output in 10BASE-T/100BASE-TX MDI mode; Differential input in 10BASE-T/100BASE-TX MDIX mode. |
| 6 | MDITN | | |
| 4 | AVDD33 | P | For 3.3V main power input, it is recommended to place 0.1uF in parallel with 10uF or 4.7uF ground capacitance close to the chip, or a single 1uF ~ 4.7uF. |
| 8 | AVDD33 | P | 3.3V power input, it is recommended to connect 0.1uF or 1uF capacitor to ground. |
| 20 | AVDDK | P | The external 1uF capacitor to ground is placed close to the chip. |
| 21、28 | VDDIO | P | For the power input of I/O interface, it is recommended to place 0.1uF or 1uF capacitance to ground close to the chip. |
| 22 | DVDDK | P | The external 0.1uF or 1uF capacitor to ground is placed close to the chip. |
| 3、9、14、16、19、29、48 | GND | P | Analog ground. |
| 7、10、11、12、13、15、17、18、23、38、39、40、41、42、46、47 | NC | - | Reserved, it is recommended to float. |
| 24 | LED_SPD | O | Network speed indicator LED:<br>Low level indicates 100Mbps,<br>high level indicates 10Mbps. |
| 25 | LED_LINK | O | Network connection indication LED:<br>Low level indicates connected,<br>high level indicates not connected. |
| 26 | LED_DUP | O | Duplex indicator LED:<br>Low level indicates full-duplex,<br>high level indicates half-duplex. |
| 27 | LED_ACT | O | Carrier induction indicator LED:<br>Low level indicates that there is a carrier induction signal,<br>high level indicates that there is no carrier induction signal. |
| 30 | XI | I | Crystal oscillator input requires an external 25MHz crystal end or an external clock input. |
| 31 | XO | O | The inverted output of crystal oscillator needs to be externally connected to the other end of 25MHz crystal. |
| 32 | SCS | I, PU | SPI chip select input, active low. |
| 33 | SCK | I | SPI clock input, supporting mode 0 or 3. |

<table>
<tr><td>34</td><td>SDO</td><td>O</td><td>SPI serial data output, connected to MISO of SPI host of processor.</td></tr>
<tr><td>35</td><td>SDI</td><td>I</td><td>SPI serial data input, connected to MOSI of SPI host of processor.</td></tr>
<tr><td>36</td><td>INT</td><td>O</td><td>Interrupt request output, active low.</td></tr>
<tr><td>37</td><td>RSTB</td><td>I, PU</td><td>Reset input, active low.</td></tr>
<tr><td>43</td><td>PMOD2</td><td>I, PU</td><td rowspan="3">PHY operation mode selection
<table>
<tr><th>PMOD[2:0]</th><th>Description</th></tr>
<tr><td>000</td><td>10M half-duplex, auto-negotiation disabled.</td></tr>
<tr><td>001</td><td>10M full-duplex, auto-negotiation disabled.</td></tr>
<tr><td>010</td><td>100M half-duplex, auto-negotiation disabled.</td></tr>
<tr><td>011</td><td>100M full-duplex, auto-negotiation closed.</td></tr>
<tr><td>100</td><td>100M half-duplex, auto-negotiation enabled.</td></tr>
<tr><td>101</td><td>Reserved</td></tr>
<tr><td>110</td><td>Reserved</td></tr>
<tr><td>111</td><td>Start auto-negotiation (Recommended default mode)</td></tr>
</table></td></tr>
<tr><td>44</td><td>PMOD1</td><td>I, PU</td></tr>
<tr><td>45</td><td>PMOD0</td><td>I, PU</td></tr>
</table>

*Note 1: I = Input; O = Output; I/O = Input/Output;*

*P = Power supply; PU = Built-in pull-up resistor.*

# 5. CH394Q Data Format

The data in this manual have a suffix of B for binary numbers and H for hexadecimal numbers, otherwise they are decimal numbers.

## 5.1 Operation Mode

CH394Q can share the SPI interface with other SPI devices. The peripheral host only needs to pull down the SCS pin to communicate with CH394Q. The interface can be used by other devices during non-communication period.

## 5.2 Data Frame

### 5.2.1 Data Frame Format

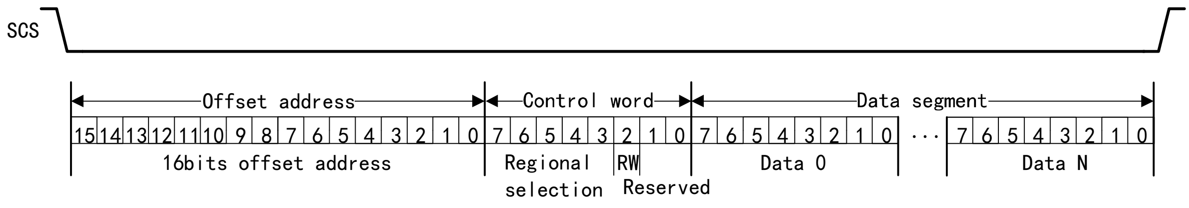
The CH394Q data frame consists of three parts: A 16-bit offset address, an 8-bit control word, and an N-byte data segment.

The offset address is the CH394Q register address or the offset address of the RX/TX buffer.

The control word is used to define the ownership of the offset area set in the address segment and determine the read/write mode.

Pulling SCS low indicates the start of a new data frame, and pulling SCS high indicates the end of a data frame.

Figure 5-1 CH394Q data frame



CH394Q supports continuous reading or writing of data, starting from the starting address. After each transmission of data at an offset address, the offset address will automatically increase by 1 to transmit the next data.

### 5.2.2 Control Phase

The CH394Q control phase specifies:

1. Ownership of the offset area set in the address segment;
2. SPI read/write mode;

Table 5-1 Control word

<table>
<tr><th>Bit</th><th>Name</th><th>Description</th></tr>
<tr><td>[7:3]</td><td>BS</td><td>Block selection bits:<br>Used to define the ownership of the offset area set in the address segment.<br>CH394Q contains 1 common register, 8 socket registers, and the corresponding socket read and write buffer area.
<table>
<tr><th colspan="2">BS [7:3]</th><th rowspan="2">Description</th></tr>
<tr><th>Segment number selection[7:5]</th><th>Function[4:3]</th></tr>
<tr><td>000</td><td>00</td><td>Common register</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><td></td><td></td><td>
<table>
<tr><td>000</td><td>01</td><td>Socket 0 register</td></tr>
<tr><td>000</td><td>10</td><td>Socket 0 TX buffer</td></tr>
<tr><td>000</td><td>11</td><td>Socket 0 RX buffer</td></tr>
<tr><td>001</td><td>00</td><td>Meaningless</td></tr>
<tr><td>001</td><td>01</td><td>Socket 1 register</td></tr>
<tr><td>001</td><td>10</td><td>Socket 1 TX buffer</td></tr>
<tr><td>001</td><td>11</td><td>Socket 1 RX buffer</td></tr>
<tr><td>010</td><td>00</td><td>Meaningless</td></tr>
<tr><td>010</td><td>01</td><td>Socket 2 register</td></tr>
<tr><td>010</td><td>10</td><td>Socket 2 TX buffer</td></tr>
<tr><td>010</td><td>11</td><td>Socket 2 RX buffer</td></tr>
<tr><td>011</td><td>00</td><td>Meaningless</td></tr>
<tr><td>011</td><td>01</td><td>Socket 3 register</td></tr>
<tr><td>011</td><td>10</td><td>Socket 3 TX buffer</td></tr>
<tr><td>011</td><td>11</td><td>Socket 3 RX buffer</td></tr>
<tr><td>100</td><td>00</td><td>Meaningless</td></tr>
<tr><td>100</td><td>01</td><td>Socket 4 register</td></tr>
<tr><td>100</td><td>10</td><td>Socket 4 TX buffer</td></tr>
<tr><td>100</td><td>11</td><td>Socket 4 RX buffer</td></tr>
<tr><td>101</td><td>00</td><td>Meaningless</td></tr>
<tr><td>101</td><td>01</td><td>Socket 5 register</td></tr>
<tr><td>101</td><td>10</td><td>Socket 5 TX buffer</td></tr>
<tr><td>101</td><td>11</td><td>Socket 5 RX buffer</td></tr>
<tr><td>110</td><td>00</td><td>Meaningless</td></tr>
<tr><td>110</td><td>01</td><td>Socket 6 register</td></tr>
<tr><td>110</td><td>10</td><td>Socket 6 TX buffer</td></tr>
<tr><td>110</td><td>11</td><td>Socket 6 RX buffer</td></tr>
<tr><td>111</td><td>00</td><td>Meaningless</td></tr>
<tr><td>111</td><td>01</td><td>Socket 7 register</td></tr>
<tr><td>111</td><td>10</td><td>Socket 7 TX buffer</td></tr>
<tr><td>111</td><td>11</td><td>Socket 7 RX buffer</td></tr>
</table>
</td></tr>
<tr><td>2</td><td>RW</td><td>Read/write selection bit:<br>0: SPI read;<br>1: SPI write.</td></tr>
<tr><td>[1:0]</td><td>Reserved</td><td>Reserved, default is 0.</td></tr>
</table>

## 5.3 Data Frame Example

The CH394Q registers use big-end mode for data storage. In memory, the low byte is placed in the high address bit while the high byte is located in the low address bit.

### 5.3.1 Data Frame Example

(1) Read 2-byte data

Read 2-byte of unreachable port information from the common register (Assuming the unreachable port is 0x03e8):

MOSI: 0x00,0x2c,0x00,0x00,0x00

MISO: - , - , - ,0x03,0xe8

(2) Write 6-byte data

Configure the Socket5 destination MAC as 0x11:0x22:0x33:0x44:0x55:0x66:

MOSI：0x00,0x06,0xac,0x11,0x22,0x33,0x44,0x55,0x66

MISO： - , - , - , - , - , - , - , - , -

# 6. CH394Q Register and Buffer

CH394Q has 1 common register area, 8 socket register areas, 8 receive buffer areas and 8 transmit buffer areas. There are multiple configurable registers in each register area. Each register in the same area has a different offset address. The register to be read or written is determined by the different offset addresses in the address segment. The BS bit of the control word is used to define which register area the offset address set in the address segment belongs to.

CH394Q has a total of 16K transmit buffer, 8 sockets each default 2K; 16K receive buffer, 8 sockets each default 2K.

When reading/writing the socket receive/transmit buffer, the offset address in the address segment (Obtained by querying the RD/WR register) represents the starting address of the data storage, and the BS bit in the control word controls the specific receive/transmit buffer to be read/written.

## 6.1 Register Table

### 6.1.1 Common Register Table

The CH394Q common register configures the CH394Q mode, IP, MAC, etc. To read/write the common register, you need to fill in the offset address corresponding to the register in the offset address part of the data frame, and set the BS part of the control word to 00000B.

Table 6-1 Common register table

| Offset address | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 0x0000 | MODE | RW | Mode register | 0x00 |
| 0x0001<br>…<br>0x0004 | GWIP0<br>…<br>GWIP3 | RW | Gateway address register | 0x00<br>…<br>0x00 |
| 0x0005<br>…<br>0x0008 | SMIP0<br>…<br>SMIP3 | RW | Subnet mask register | 0x00<br>…<br>0x00 |
| 0x0009<br>…<br>0x000E | MAC0<br>…<br>MAC5 | RW | MAC address register | 0xXX<br>…<br>0xXX |
| 0x000F<br>…<br>0x0012 | IP0<br>…<br>IP3 | RW | Source IP address register | 0x00<br>…<br>0x00 |
| 0x0013<br>0x0014 | IIT0<br>IIT1 | RW | Interrupt interval time register | 0x00<br>0x00 |
| 0x0015 | GINT | RW | Global interrupt register | 0x00 |
| 0x0016 | GINTE | RW | Global interrupt enable register | 0x00 |
| 0x0017 | SINT | RO | Socket interrupt register | 0x00 |
| 0x0018 | SINTE | RW | Socket interrupt enable register | 0x00 |
| 0x0019<br>0x001A | RTIME0<br>RTIME1 | RW | Retransmission time register | 0x07<br>0xD0 |
| 0x001B | RCNT | RW | Retransmission count register | 0x08 |
| 0x001C<br>… | - | - | Reserved | - |

| | | | | |
|---|---|---|---|---|
| 0x0027 | | | | |
| 0x0028<br>…<br>0x002B | UNIP0<br>…<br>UNIP3 | RO | Unreachable IP register | 0x00<br>…<br>0x00 |
| 0x002C<br>0x002D | UNPORT0<br>UNPORT1 | RO | Unreachable port register | 0x00<br>0x00 |
| 0x002E | PHY_CFG | RW | PHY configuration register | 0xB8 |
| 0x002F<br>…<br>0x0038 | - | - | Reserved | - |
| 0x0039 | CHIPV | RO | Chip version register | 0xXX |

### 6.1.2 Socket Register Table

CH394Q provides 8 Sockets, each Socket corresponds to a Socket register area, the register area corresponding to Socket n (0 ≤ n ≤ 7) is selected by the BS bit of the control word (Refer to 5.2.2 for details), and the register of Socket n to be read/written is determined by the offset address.

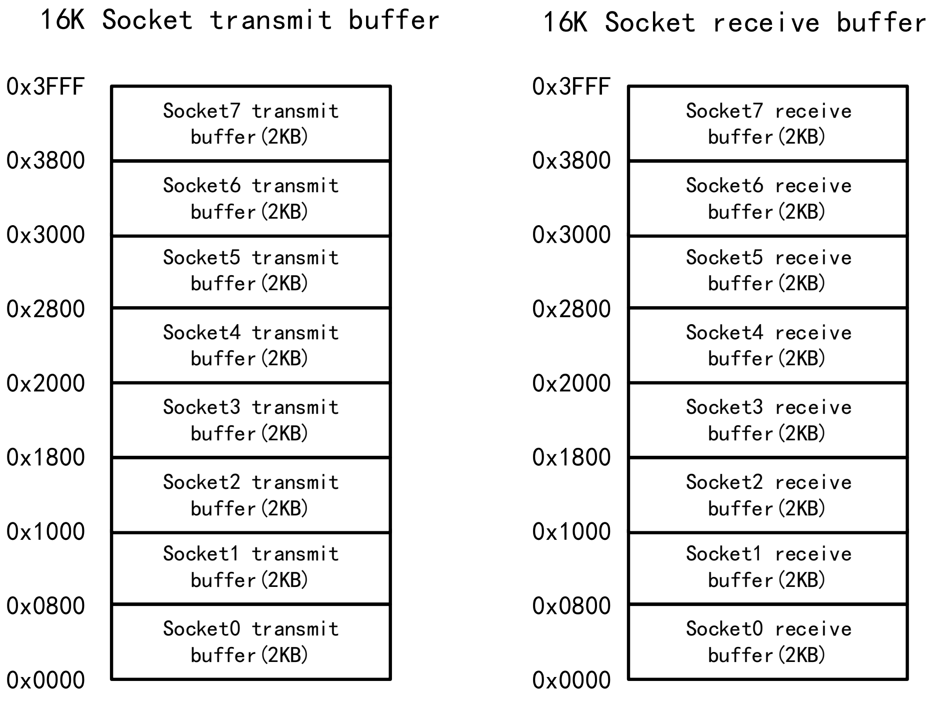
Table 6-2 Socket register table

| Offset address | Name | Access | Description | Reset value |
|---|---|---|---|---|
| 0x0000 | Sn_MODE | RW | Socket n mode register | 0x00 |
| 0x0001 | Sn_CTRL | RW | Socket n control register | 0x00 |
| 0x0002 | Sn_INT | RW | Socket n interrupt register | 0x00 |
| 0x0003 | Sn_STA | RO | Socket n status register | 0x00 |
| 0x0004<br>0x0005 | Sn_PORT0<br>Sn_PORT1 | RW | Socket n source port register | 0x00<br>0x00 |
| 0x0006<br>…<br>0x000B | Sn_DMAC0<br>…<br>Sn_DMAC5 | RW | Socket n destination MAC register | 0x00<br>…<br>0x00 |
| 0x000C<br>…<br>0x000F | Sn_DIP0<br>…<br>Sn_DIP3 | RW | Socket n destination IP register | 0x00<br>…<br>0x00 |
| 0x0010<br>0x0011 | Sn_DPORT0<br>Sn_DPORT1 | RW | Socket n destination port register | 0x00<br>0x00 |
| 0x0012<br>0x0013 | Sn_MTU0<br>Sn_MTU1 | RW | Socket n maximum transfer unit register | 0x00<br>0x00 |
| 0x0014 | - | - | Reserved | - |
| 0x0015 | Sn_TOS | RW | Socket n IP service type register | 0x00 |
| 0x0016 | Sn_TTL | RW | Socket IP time-to-live register | 0x80 |
| 0x0017<br>…<br>0x001D | - | - | Reserved | - |
| 0x001E | Sn_RXBUF_SIZE | RW | Socket n receive buffer size register | 0x02 |
| 0x001F | Sn_TXBUF_SIZE | RW | Socket n transmit buffer size register | 0x02 |
| 0x0020 | Sn_TX_FS0 | RO | Socket n idle transmit buffer length | 0x08 |

| | | | | |
|---|---|---|---|---|
| 0x0021 | Sn_TX_FS1 | | register | 0x00 |
| 0x0022<br>0x0023 | Sn_TX_RD0<br>Sn_TX_RD1 | RO | Socket n transmit buffer read pointer register | 0xXX |
| 0x0024<br>0x0025 | Sn_TX_WR0<br>Sn_TX_WR1 | RW | Socket n transmit buffer write pointer register | 0xXX |
| 0x0026<br>0x0027 | Sn_RX_RS0<br>Sn_RX_RS1 | RO | Socket n receive data length register | 0x00<br>0x00 |
| 0x0028<br>0x0029 | Sn_RX_RD0<br>Sn_RX_RD1 | RW | Socket n receive buffer read pointer register | 0xXX |
| 0x002A<br>0x002B | Sn_RX_WR0<br>Sn_RX_WR1 | RO | Socket n receive buffer write pointer register | 0xXX |
| 0x002C | Sn_INTE | RW | Socket n interrupt enable register | 0xFF |
| 0x002D<br>0x002E | Sn_IPF0<br>Sn_IPF1 | RW | Socket n IP fragment register | 0x40<br>0x00 |
| 0x002F | Sn_KEEPALIVE | RW | Socket n KeepAlive time register | 0x00 |

## 6.2 RX/TX Buffer

Figure 6-1 RX/TX buffer


16K Socket transmit buffer
16K Socket receive buffer
0x3FFF
0x3800
0x3000
0x2800
0x2000
0x1800
0x1000
0x0800
0x0000
Socket7 transmit buffer(2KB)
Socket6 transmit buffer(2KB)
Socket5 transmit buffer(2KB)
Socket4 transmit buffer(2KB)
Socket3 transmit buffer(2KB)
Socket2 transmit buffer(2KB)
Socket1 transmit buffer(2KB)
Socket0 transmit buffer(2KB)
0x3FFF
0x3800
0x3000
0x2800
0x2000
0x1800
0x1000
0x0800
0x0000
Socket7 receive buffer(2KB)
Socket6 receive buffer(2KB)
Socket5 receive buffer(2KB)
Socket4 receive buffer(2KB)
Socket3 receive buffer(2KB)
Socket2 receive buffer(2KB)
Socket1 receive buffer(2KB)
Socket0 receive buffer(2KB)

Each socket (n = 0 ~ 7) of bit CH394Q provides an independent transmitting and receiving memory buffer, with a total of 16K receiving buffer and 16K transmitting buffer.

In the initial state, the transmitting and receiving buffers of each Socket are allocated to 2KB (The receiving buffer is 16KB in total and the transmitting buffer is 16KB in total). Users can reallocate the 16KB memory resources to each Socket by configuring the Socket transmit buffer size register (Sn_TXBUF_SIZE) and the Socket receive buffer size register (Sn_RXBUF_SIZE) as required, but it is necessary to ensure that the sum of the receive buffer and transmit buffer sizes of all sockets does not exceed 16KB to prevent data transmission errors.

# 7. CH394Q Register Description

Socket Pair contains a quadruple of source IP, source port, destination IP and destination port, which can uniquely identify the two connected parties in the Internet. This manual is referred to as Socket for short. CH394Q can provide 8 Socket at the same time, and their index values are 0, 1, 2, 3 ....

The high and low bytes of IP and MAC addresses agreed in this manual may be different from some documents, just for convenience:

For example, the IP address is 192.168.1.2, where 192 is the highest byte and 2 is the lowest byte.

For example, the MAC address is 0x11:0x22:0x33:0x44:0x55:0x66, where 0x11 is the highest byte and 0x66 is the lowest byte.

## 7.1 Common Register Description

CH394Q register uses big terminal mode for data storage. In memory, the low byte is placed in the high address bit, while the high byte is placed in the low address bit.

### 7.1.1 Mode Register (MODE) [0x0000]

This register is used to control the operation mode.

Table 7-1 Mode Register

| Bit | Name | Description | Access | Default value |
|---|---|---|---|---|
| 7 | RST | Set 1 for software reset, and it will be cleared automatically after the reset is completed. | RW, SC | 0 |
| 6 | Reserved | Reserved | RO | 0 |
| 5 | WOL | Wake on LAN enabled:<br>0: Disable WOL mode;<br>1. Enable WOL mode.<br>CH394Q integrates the magic packet WOL mode. After enabling the magic packet wake-up, CH394Q will pull down the INT pin after receiving the magic packet (0xFFFFFFFFFFFF+16 CH394Q MAC addresses) that matches its own MAC address from UDP. | RW | 0 |
| 4 | PB | PING packet block enabled:<br>0: Disable PING block;<br>1: Enable PING block. | RW | 0 |
| 3 | Reserved | Reserved | RO | 0 |
| 2 | Reserved | Reserved | RO | 0 |
| 1 | FARP | Forced ARP enable:<br>0: Disable forced ARP mode;<br>1: Enable forced ARP mode.<br>Turn on forced ARP mode. When forced ARP is turned on, an ARP request will be forced to be sent before each UDP packet is sent. | RW | 0 |
| 0 | Reserved | Reserved | RO | 0 |

### 7.1.2 Gateway IP Address Register (GWIP) [0x0001-0x0004]

This register is used to set the gateway address, and the low byte address stores the high byte of the gateway address. For example: The gateway IP is 192.168.1.1.

| Address | 0x0001 | 0x0002 | 0x0003 | 0x0004 |
|---|---|---|---|---|
| Data | 0xC0 | 0xA8 | 0x01 | 0x01 |

### 7.1.3 Subnet Mask IP Address Register (SMIP) [0x0005-0x0008]

This register is used to set the subnet mask address, and the low byte address stores the high byte of the subnet mask address.

For example: Subnet mask IP 255.255.255.0

| Address | 0x0005 | 0x0006 | 0x0007 | 0x0008 |
|---|---|---|---|---|
| Data | 0xFF | 0xFF | 0xFF | 0x00 |

### 7.1.4 MAC Address Register (MAC) [0x0009-0x000E]

This register is used to set the MAC address of CH394Q, and the low byte address stores the high byte of the MAC address.

The CH394Q chip has burned the MAC address assigned by IEEE when it leaves the factory. Please do not set the MAC address unless it is necessary.

For example: The MAC address is 0x38:0x3B:0x26:0x11:0x22:0x33

| Address | 0x0009 | 0x000A | 0x000B | 0x000C | 0x000D | 0x000E |
|---|---|---|---|---|---|---|
| Data | 0x38 | 0x3B | 0x26 | 0x11 | 0x22 | 0x33 |

### 7.1.5 Source IP Address Register (IP) [0x000F-0x0012]

This register is used to set the IP address of CH394Q, and the low byte address stores the high byte of the IP address. For example: The source address is 192.168.1.100

| Address | 0x000F | 0x0010 | 0x0011 | 0x0012 |
|---|---|---|---|---|
| Data | 0xC0 | 0xA8 | 0x01 | 0x64 |

### 7.1.6 Interrupt Interval Register (IIT) [0x0013-0x0014]

This register is used to set the waiting time for the interrupt to take effect, and the default value is 0. When an interrupt event is handled, the INT signal line will return to high level, and it will take a set IIT interval time before CH394Q will notify the next interrupt and pull the INT signal line low.

The formula for calculating the interval time is: Time = 1/120000000*4*(IIT+1).

For example, if the register value is set to 1000, then the interval time is about 33.3us

| Address | 0x0013 | 0x0014 |
|---|---|---|
| Data | 0x03 | 0xE8 |

### 7.1.7 Global Interrupt Register (GINT) [0x0015]

This register is used to obtain the global interrupt status. When an interrupt event occurs, the corresponding bit in the GINT register will be set to 1. Writing a 1 to this bit can clear the interrupt flag bit. If the enable bit of this interrupt in GINTE is also 1, the level of the INT pin becomes low. Thereafter, once the corresponding bit of the event is cleared or the corresponding interrupt enable bit is set to 0, and the SINT is 0, the INT pin is restored to high level.

Table 7-2 Global interrupt register

| Bit | Name | Description | Access | Default value |
|---|---|---|---|---|
| 7 | IP_CONFLI | IP conflict interrupt:<br>This interrupt occurs when CH394Q detects that its own IP address is the same as that of other network devices in the same network segment. | RW | 0 |
| 6 | UNREACH | Unreachable interrupt:<br>When CH394Q receives the ICMP unreachable interrupt message, it saves the IP address, port and protocol type of the unreachable IP packet in the unreachable information table, and then generates this interrupt.<br>After receiving this interrupt, MCU can query UNIP and UNPORT registers to obtain unreachable information. | RW | 0 |
| 5 | Reserved | Reserved | RO | 0 |
| 4 | MP | Magic packet wake-up interrupt:<br>This interrupt occurs when the Magic Packet function is enabled and the Magic Packet WOL mode is received through UDP. | RW | 0 |
| [3:0] | Reserved | Reserved | RO | 0 |

### 7.1.8 Global Interrupt Enable Register (GINTE) [0x0016]

This register is used to control which interrupt sources can trigger interrupts, and each interrupt event enable bit corresponds to one bit of the global interrupt register (GINT).

When the interrupt event is generated, only when the corresponding bit in GINTE is 1, CH394Q will pull down the INT pin and generate an interrupt.

Table 7-3 Global interrupt enable register

| Bit | Name | Description | Access | Default value |
|---|---|---|---|---|
| 7 | IP_CONFLI | IP conflict interrupt enable bit:<br>Enable this bit allows IP collision interrupts to be generated. | RW | 0 |
| 6 | UNREACH | Unreachable interrupt enable bit:<br>Enable this bit allows unreachable interrupts to be generated. | RW | 0 |
| 5 | Reserved | Reserved | RO | 0 |
| 4 | MP | Magic Packet Wake-up Interrupt Enable Bit:<br>Enable this bit allows the generation of Magic Packet Wake-up Interrupt. | RW | 0 |
| [3:0] | Reserved | Reserved | RO | 0 |

### 7.1.9 Socket Interrupt Register (SINT) [0x0017]

This register is used to obtain the socket interrupt status. When an interrupt event occurs on socket n, the corresponding bit of the Sn_INT register will be 1 and the nth bit of SINT will be 1, and the INT pin will be pulled low. When Sn_INT is 0, the corresponding bit of SINT will also be cleared. The INT pin is pulled high when both SINT and GINT are '0x00'.

Table 7-4 Socket interrupt register

| Bit | Name | Description | Access | Default value |
|---|---|---|---|---|
| 7 | S7_INT | Socket7 interrupt | RO | 0 |
| 6 | S6_INT | Socket6 interrupt | RO | 0 |
| 5 | S5_INT | Socket5 interrupt | RO | 0 |
| 4 | S4_INT | Socket4 interrupt | RO | 0 |
| 3 | S3_INT | Socket3 interrupt | RO | 0 |
| 2 | S2_INT | Socket2 interrupt | RO | 0 |
| 1 | S1_INT | Socket1 interrupt | RO | 0 |
| 0 | S0_INT | Socket0 interrupt | RO | 0 |

### 7.1.10 Socket Interrupt Enable Register (SINTE) [0x0018]

This register is used to control which Sockets can trigger interrupts, and each enable bit corresponds to one bit of the socket interrupt register (SINT).

When the Socket interrupt event is generated, only when the corresponding bit in SINTE is 1, CH394Q will pull down the INT pin and generate an interrupt.

Table 7-5 Socket interrupt enable register

| Bit | Name | Description | Access | Default value |
|---|---|---|---|---|
| 7 | S7_INT | Socket7 interrupt enable bit:<br>Enabling this bit allows the generation of Socket 7 interrupts. | RW | 0 |
| 6 | S6_INT | Socket6 interrupt enable bit:<br>Enabling this bit allows the generation of Socket 6 interrupts. | RW | 0 |
| 5 | S5_INT | Socket5 interrupt enable bit:<br>Enabling this bit allows the generation of Socket 5 interrupts. | RW | 0 |
| 4 | S4_INT | Socket4 interrupt enable bit:<br>Enabling this bit allows the generation of Socket 4 interrupts. | RW | 0 |
| 3 | S3_INT | Socket3 interrupt enable bit:<br>Enabling this bit allows the generation of Socket 3 interrupts. | RW | 0 |
| 2 | S2_INT | Socket2 interrupt enable bit:<br>Enabling this bit allows the generation of Socket 2 interrupts. | RW | 0 |
| 1 | S1_INT | Socket1 interrupt enable bit:<br>Enabling this bit allows the generation of Socket 1 interrupts. | RW | 0 |
| 0 | S0_INT | Socket0 interrupt enable bit:<br>Enabling this bit allows the generation of Socket 0 | RW | 0 |

| | | | | |
|---|---|---|---|---|
| | | interrupts. | | |

### 7.1.11 Retransmission Time Register (RTIME) [0x0019-0x001A]

This register is used to set the retransmission time in TCP communication and the retransmission time of ARP. If TCP transmission or ARP request does not receive a response from the other party within the RTIME time, a retransmission operation will be performed or a timeout interrupt will be triggered.

RTIME requires a two-byte time value with a unit of 0.1ms. The default value is 2000 (0x07D0), i.e. 200ms (2000*0.1ms). Note that the RTIME value must be a multiple of 1000.

For example: Set retransmission time as 5000 (0x1388), i.e. 500ms.

| Address | 0x0019 | 0x001A |
|---|---|---|
| Data | 0x13 | 0x88 |

### 7.1.12 Retransmission Count Register (RCN) [0x001B]

This register is used to set the retransmission timeout count. The retransmission timeout count refers to the maximum number of times a packet is retransmitted without a response, and the default value is 8. When the retransmission exceeds the set number of times, the corresponding socket timeout interrupt bit will be set to 1.

### 7.1.13 Unreachable IP Register (UNIP) [0x0028-0x002B]

This register is used to obtain an unreachable IP address. When CH394Q transmits data to an unreachable port, the other party will reply with an ICMP packet (Destination port unreachable). After receiving the unreachable message, CH394Q will generate an unreachable interrupt. The microcontroller can use this command to obtain the unreachable address.

For example: Unreachable IP is 192.168.1.200

| Address | 0x0028 | 0x0029 | 0x002A | 0x002B |
|---|---|---|---|---|
| Data | 0xC0 | 0xA8 | 0x01 | 0xC8 |

### 7.1.14 Unreachable Port Register (UNPORT) [0x002C-0x002D]

This register is used to obtain an unreachable port. When CH394Q transmits data to an unreachable port, the other party will reply with an ICMP packet (Destination port unreachable). After receiving the unreachable message, CH394Q will generate an unreachable interrupt. The microcontroller can use this command to obtain the unreachable port.

For example: Unreachable Port is 2000

| Address | 0x002C | 0x002D |
|---|---|---|
| Data | 0x07 | 0xD0 |

### 7.1.15 PHY Configuration Register (PHY_CFG) [0x002E]

This register configures status of PHY, and it indicates the current PHY connection status.

Table 7-6 PHY configuration register

| Bit | Name | Description | Access | Default value |
|---|---|---|---|---|
| 7 | RST | Reset:<br>0: PHY reset;<br>1: Normal work.<br>*Note: This bit needs to be set to 1 manually after the PHY* | RW | 1 |

<table>
<tr><td></td><td></td><td><i>reset is completed.</i></td><td></td><td></td></tr>
<tr><td>6</td><td>OPWD</td><td>Operation mode selection bit:<br>0: Configure the PHY mode through hardware pins (PMOD[2:0]);<br>1: Configure the PHY mode via the PHY Configuration Register [5:3] bits;<br>When CH394Q is powered on or reset by RSTB pin, it detects the level status of PMOD[2:0] pin to set the PHY operation mode.<br>When the user wants to control the PHY mode through OPMDC, it is necessary to:<br>1. Set OPWD as 1;<br>2. Set OPMDC to the desired mode value.<br>When the user wants to use the level state of the PMOD[2:0] pin to set the PHY operation mode again, he only needs to set the OPWD bit from 1 to 0 without powering on or resetting it again.</td><td>RW</td><td>0</td></tr>
<tr><td>[5:3]</td><td>OPMDC</td><td><table>
<tr><th>[5:3]</th><th>Description</th></tr>
<tr><td>000</td><td>10M half-duplex, auto-negotiation disabled</td></tr>
<tr><td>001</td><td>10M full-duplex, auto-negotiation disabled</td></tr>
<tr><td>010</td><td>100M half-duplex, auto-negotiation disabled</td></tr>
<tr><td>011</td><td>100M full-duplex, auto-negotiation disabled</td></tr>
<tr><td>100</td><td>100M half-duplex, auto-negotiation enabled</td></tr>
<tr><td>101</td><td>Reserved</td></tr>
<tr><td>110</td><td>Power-down mode</td></tr>
<tr><td>111</td><td>Enable auto-negotiation</td></tr>
</table>PHY mode configuration bit</td><td>RW</td><td>111b</td></tr>
<tr><td>2</td><td>DUPS</td><td>Duplex status<br>0: half-duplex;<br>1: full-duplex.</td><td>RO</td><td>0</td></tr>
<tr><td>1</td><td>SPDS</td><td>Speed status<br>0: 10Mb/s;<br>1: 100Mb/s.</td><td>RO</td><td>0</td></tr>
<tr><td>0</td><td>LINKS</td><td>Link status<br>0: The physical layer does not establish a link;<br>1: The physical layer establishes the link.</td><td>RO</td><td>0</td></tr>
</table>

### 7.1.16 Chip Version Register (CHIPV) [0x0039]

This register is used to obtain the chip version.

## 7.2 Socket Register Description

### 7.2.1 Socket n Mode Register (Sn_MODE) [0x0000]

This register is used to configure the operation mode of Socket n.

Table 7-7 Socket n mode register

| Bit | Name | Description | Access | Default value |
|---|---|---|---|---|
| 7 | MUL_MFEN | When Socket n works in UDP mode:<br>0: Disable multicast mode;<br>1: Enable multicast mode.<br>When using multicast mode, you need to configure the multicast IP, multicast MAC, and port number through Sn_DIP and Sn_DPORT in advance before the socket is opened.<br>When Socket n is operating in MAC RAW mode:<br>0: Disables MAC address filtering;<br>1: Enables MAC address filtering.<br>When MAC RAW mode is used, with MAC address filtering turned on, Socket n will only receive packets on the network whose destination MAC is the CH394Q's own MAC address as well as the broadcast address. | RW | 0 |
| 6 | BCASTB | Broadcast blocking shielding bit:<br>0: Disable broadcast blocking;<br>1: Enable broadcast blocking.<br>After this position 1, in UDP and MAC RAW mode, Socket n will not receive broadcast blocking on the network. | RW | 0 |
| 5 | NA_MV_MM | When Socket n is operating in TCP mode:<br>0: Disable no-delay ACK;<br>1: Enable no-delay ACK.<br>After enabling no-delay ACK, Socket n will reply to ACK as soon as possible after receiving the packet from the opposite end without delay; otherwise, it will wait for 10ms before replying to ACK.<br>When Socket n is operating in UDP multicast mode:<br>0: Use IGMP version 2;<br>1: Use IGMP version 1.<br>When Socket n is operating in MAC RAW mode:<br>0: Disable multicast blocking;<br>1: Enable multicast blocking.<br>When this position is set to 1, in MACRAW mode, Socket n will not receive multicast packets on the network. | RW | 0 |

<table>
<tr><td>4</td><td>UCASTB_<br>MIP6B</td><td>When Socket n is operating in UDP multicast mode:<br>0: Disable unicast blocking;<br>1: Enable unicast blocking.<br>When this position is set to 1, in UDP multicast mode, Socket n will not receive unicast packets on the network.<br>When Socket n is working in MAC RAW mode:<br>0: Disable IPv6 blocking;<br>1: Enable IPv6 blocking.<br>After this position 1, in MAC RAW mode, Socket n will not receive IPv6 packets on the network.</td><td>RW</td><td>0</td></tr>
<tr><td>[3:0]</td><td>PMD</td><td>Set Socket operation mode:<table><tr><th>[3:0]</th><th>Description</th></tr><tr><td>0000</td><td>OFF</td></tr><tr><td>0001</td><td>TCP</td></tr><tr><td>0010</td><td>UDP</td></tr><tr><td>0100</td><td>MAC RAW</td></tr></table><i>Note: Only Socket 0 can use MAC RAW mode.</i></td><td>RW</td><td>0</td></tr>
</table>

### 7.2.2 Socket n Control Register (Sn_CTRL) [0x0001]

This register is used to set the control command of Socket n. After configuring this register, the register will be automatically cleared. After setting the control command, the control command usually takes a certain amount of time to execute. Users can judge the command execution status through the Sn_INT or Sn_STA register.

Table 7-8 Socket n mode register

<table>
<tr><th>Code</th><th>Name</th><th>Description</th></tr>
<tr><td>01H</td><td>OPEN</td><td>Open Socket n:<br>Initialize and open Socket n according to the mode set by Sn_MODE[3:0]. You can query the command execution status through Sn_STA, and when the command is executed successfully, the corresponding Sn_STA value should be:<table><tr><th>Sn_MODE[3:0]</th><th>Sn_STA</th></tr><tr><td>0000 (OFF)</td><td>-</td></tr><tr><td>0001 (TCP)</td><td>0x13 (INIT)</td></tr><tr><td>0010 (UDP)</td><td>0x22 (UDP)</td></tr><tr><td>0100 (MAC RAW)</td><td>0x42 (MAC RAW)</td></tr></table></td></tr>
<tr><td>02H</td><td>LISTEN</td><td>Set Socket n to the listening state:<br>This command takes effect only when Socket n is in TCP mode and is in the INIT state.<br>When Socket n is set to LISTEN state, Socket n is in TCP Sever mode and waits for the client to connect, and the Sn_STA register becomes 0x14 (LISTEN). After the client connects successfully, the Sn_STA register value becomes 0x17 (ESTABLISHED) and Sn_INT[0] is set to 1.<br>If the connection fails the Sn_STA register value becomes 0x00 (CLOSE) and Sn_INT[3] is set to 1.</td></tr>
</table>

| | | |
|---|---|---|
| 04H | CONNECT | Enable Socket n to enter connection mode:<br>This command takes effect only when Socket n is in TCP mode and is in INIT state.<br>Enabling Socket to enter connection mode means TCP Client mode. When this command is executed, Socket n connects to the server according to the values set by Sn_DIP and Sn_DPORT, and the value of Sn_STA register changes to 0x17 (ESTABLISHED) and Sn_INT[0] is set to 1 when the connection is successful.<br>When the socket connection fails:<br>1: ARP failure, cannot get the destination MAC address (Sn_INT[3] set to 1);<br>2: Connection timeout, cannot receive SYN/ACK packet from the other party after transmitting SYN packet, after retransmission timeout (Sn_INT[3] set to 1);<br>3: Receiving RST packet, being disconnected by the other party on its own initiative.<br>The Sn_STA register value becomes 0x00 (CLOSE) when the above situation occurs. |
| 08H | DISCONNECT | Socket n disconnect:<br>This command is effective only when Socket n is in TCP mode.<br>Active close: Actively transmit a FIN packet to the other end device;<br>Passive close: After receiving a FIN packet from the other end, execute this command to reply a FIN packet to the other end.<br>If the FIN packet does not receive the ACK response from the other side, after the retransmission timeout, Sn_INT[3] is set to 1, and the value of Sn_STA register becomes 0x00(CLOSE). |
| 10H | CLOSE | Close Socket n:<br>If Socket n is in UDP or MAC RAW mode, this command will directly close Socket.<br>If Socket n is in TCP mode, this command will transmit a RST packet to close the Socket. Unlike the standard TCP closing process, this command will not transmit a FIN packet when it is closed. |
| 20H | SEND | Socket n transmitting data:<br>Transmitting the data in the Socket n transmitting buffer. See section 11.2.2 Data transmitting process for details. |
| 21H | SNED_MAC | Socket n transmits data without ARP:<br>This command only takes effect when Socket n is in UDP mode.<br>The SEND process is the same as the transmit command, but the MAC address of the opposite end is not obtained through ARP before the contract is sent, and the MAC address set by Sn_DMAC is used directly. |
| 22H | SEND_KEEP | Socket n transmits KeepAlive package:<br>This command only takes effect when Socket n is in TCP mode.<br>Actively transmit a heartbeat packet to the opposite device. If no reply is obtained after the timeout, Sn_INT[3] is set to 1, and the value of Sn_STA register becomes 0x00(CLOSE). |

| | | |
|---|---|---|
| 40H | RECV | Socket n receiving data:<br>Complete the process of Socket n receiving data. For detailed steps, please refer to Chapter 11.2.1 Data Receiving Process. |

### 7.2.3 Socket n Interrupt Register (Sn_INT) [0x0002]

This register is used to obtain the interrupt status of Socket n. When the Socket n interrupt event occurs, the corresponding bit of the Sn_INT register will be set to 1, and writing a 1 to this bit can clear the interrupt flag bit.

Table 7-9 Socket n interrupt register

| Bit | Name | Description | Access | Default value |
|---|---|---|---|---|
| [7:5] | Reserved | Reserved | RO | 0 |
| 4 | INT_SEND_SUC | Transmit completion interrupt:<br>This interrupt is generated when the Socket n SEND command is completed. | RW | 0 |
| 3 | INT_TIMEOUT | Timeout interrupt:<br>This interrupt is generated when Socket n ARP times out or TCP retransmission times out. | RW | 0 |
| 2 | INT_RECV | Receive interrupt:<br>This interrupt is generated when Socket n receives a packet. | RW | 0 |
| 1 | INT_DISCONNECT | Disconnect interrupt:<br>This interrupt is generated when Socket n receives the other party's FIN or FIN/ACK or RST packet. | RW | 0 |
| 0 | INT_CONNECT | Connection interrupt:<br>This interrupt is generated when the Socket n is successfully connected with the opposite end. | RW | 0 |

### 7.2.4 Socket n Status Register (Sn_STA) [0x0003]

This register is used to obtain the current state of Socket n and the temporary state in operation.

Table 7-10 Socket n status values

| Code | Name | Description |
|---|---|---|
| 00H | CLOSE | Close:<br>Socket n is in the closed state. |
| 13H | INIT | TCP Ready:<br>When Socket n is in TCP mode and open is successful, Socket n is in INIT state. Only when Socket n is in INIT state, users can use LISTEN or CONNECT command to make the next connection. |
| 14H | LISTEN | TCP Listening:<br>This status indicates that Socket n is in listening state as a TCP server, waiting for TCP client connection.<br>Sn_STA will become 0x17 (ESTABLISHED) after successful connection. |
| 17H | ESTABLISHED | Socket establishes connection:<br>When Socket is in the LISTEN state, it is connected successfully by the opposite TCP client as a TCP server mode; Or use the CONNECT command |

| | | |
|---|---|---|
| | | as a TCP client to connect successfully, and Sn_STA will become 0x17(ESTABLISHED).<br>When Sn _ STA is 0x17 (ESTABLISHED), the connection is successful, and data transmission can be carried out normally at this time. |
| 1CH | CLOSE_WAIT | Close Waiting:<br>When Socket n is in TCP mode and receives a disconnect request packet (FIN packet) from the other party, Sn_STA will become 0x10(CLOSE_WAIT). At this time, the TCP connection is in a semi-disconnected state. If you want to completely disconnect, you need to use DISCONNT command, and if you want to CLOSE it directly, you can use Close command. |
| 22H | UDP | UDP mode:<br>Indicates that Socket n is in UDP mode. |
| 42H | MAC_RAW | MAC RAW mode:<br>Indicates that Socket n is in MAC RAW mode.<br>*Note: Only Socket 0 can use MAC RAW mode.* |

Temporary state during TCP connection:

Table 7-11 Socket n TCP temporary status value

| Code | Name | Description |
|---|---|---|
| 15H | SYN_SENT | SYN request sent:<br>When the CONNECT command is used, Socket n will transmit a packet of SYN request to the opposite end actively, and at this time, Sn_STA will become SYN_SENT. When the opposite end replies to the SYN/ACK packet, the connection is successfully established, and Sn_STA will become 0x17(ESTABLISHED).<br>If there is no reply to the SYN/ACK packet at the opposite end, after the SYN request retransmission times out, Sn_INT[3] is set to 1, and the value of Sn_STA register becomes 0x00(CLOSE). |
| 16H | SYN_RECV | SYN request received:<br>When Socket n is in TCP Sever mode in listening state, Sn_STA becomes SYN_RECV after receiving the connection request packet from the opposite end. At this time, Socket n will reply the SYN/ACK packet to the opposite end. After success, Sn_STA will become 0x17(ESTABLISHED).<br>If the other party fails to reply to the ACK in time after transmitting the SYN/ACK packet, resulting in the retransmission timeout, then Sn_INT[3] is set to 1, and the value of Sn_STA register becomes 0x00(CLOSE). |
| 18H | FIN_WAIT | These states all indicate that Socket n is closing.<br>Generally, there is no need to care about these States in application, and the chip will automatically process and update.<br>When the Socket n is closed successfully, or after the timeout (Sn_INT[3] is set to 1), the value of Sn_STA register becomes 0x00(CLOSE). |
| 1AH | CLOSING | |
| 1BH | TIME_WAIT | |
| 1DH | LAST_ACK | |

### 7.2.5 Socket n Source Port Register (Sn_PORT) [0x0004-0x0005]

This register is used to set the source port number of Socket n. If two or more Socket use the same mode, the source port numbers must not be the same. For example, Socket0 is in UDP mode, the source port is 600, and Socket1 is also in UDP mode. You can't use the source port 600 again, otherwise it may lead to opening failure.

For example: The source port number is 2000.

| Address | 0x0004 | 0x0005 |
|---|---|---|
| Data | 0x07 | 0xD0 |

### 7.2.6 Socket n Destination MAC Register (Sn_DMAC) [0x0006-0x000B]

This register is used to get or set the destination MAC address of Socket n. This register can be used in two ways:

1: Get the target MAC address. After ARP is successful, this register stores the MAC address obtained in the process of Socket n ARP.

2: Set the target MAC address. SNED_MAC command in UDP mode will be sent directly according to the set MAC address.

For example: The MAC address is 0x38:0x3B:0x26:0x44:0x55:0x66.

| Address | 0x0006 | 0x0007 | 0x0008 | 0x0009 | 0x000A | 0x000B |
|---|---|---|---|---|---|---|
| Data | 0x38 | 0x3B | 0x26 | 0x44 | 0x55 | 0x66 |

### 7.2.7 Socket n Destination IP Register (Sn_DIP) [0x000C-0x000F]

This register is used to get or set the destination IP address of Socket n. This register can be used in two ways:

1. Get the target IP address. In TCP Sever mode, after the connection is successful, this register stores the IP address of the TCP client.

2: Set the destination IP address, which must be set in UDP and TCP Client modes.

For example: The destination IP address is 192.168.1.200.

| Address | 0x000C | 0x000D | 0x000E | 0x000F |
|---|---|---|---|---|
| Data | 0xC0 | 0xA8 | 0x01 | 0xC8 |

### 7.2.8 Socket n Destination Port Register (Sn_DPORT) [0x0010-0x0011]

This register is used to get or set the destination IP port of Socket n. There are two ways to use this register:

1. Get the destination port. In TCP Sever mode, after the connection is successful, this register stores the port number of the TCP client.

2: Set the destination port, which must be set in UDP and TCP Client modes.

For example: The target port number is 3000.

| Address | 0x0010 | 0x0011 |
|---|---|---|
| Data | 0x0B | 0xB8 |

### 7.2.9 Socket n Maximum Transfer Unit Register (Sn_MTU) [0x0012-0x0013]

This register is used to set the maximum transmission unit (MTU) of Socket n, which takes effect in TCP mode and UDP mode. When the transmission length of user data exceeds the preset MTU size, the CH394Q built-in protocol stack will automatically perform data segmentation to ensure that each packet of data does not exceed the set value of MTU.

For example: Set MTU as 1000.

| Address | 0x0012 | 0x0013 |
|---|---|---|
| Data | 0x03 | 0xE8 |

### 7.2.10 Socket n IP Service Type Register (Sn_TOS) [0x0015]

This register is used to set the TOS service field in the IP header of the IP layer, which should be set before the Socket n OPEN. The default value is 0, and it does not need to be set by default.

### 7.2.11 Socket n IP Time-to-Life Register (Sn_TTL) [0x0016]

This register is used to set the TTL service field in the IP header of the IP layer and should be set before Socket n OPEN. The default value is 0x80, the default value is not required, and the maximum value is 128.

### 7.2.12 Socket n RX Buffer Size Register (Sn_RXBUF_SIZE) [0x001E]

This register is used to set the receive buffer size of Socket n, and the default value is 2(2KB). The register value can be set from 0 to 16, corresponding to 0 to 16KB. CH394Q allocates 2K space for each Socket by default.

<table>
<tr><td colspan="2">Socket 0 receive buffer</td><td colspan="2">Socket 1 receive buffer</td><td></td><td colspan="2">Socket 7 receive buffer</td></tr>
<tr><td>Block 0</td><td>Block 1</td><td>Block 2</td><td>Block 3</td><td>……</td><td>Block 14</td><td>Block 15</td></tr>
<tr><td colspan="2">2K</td><td colspan="2">2K</td><td></td><td colspan="2">2K</td></tr>
</table>

The default allocation of the internal receiving buffer of CH394Q is as shown in the figure above, which consists of 16 blocks in total, and the length of each block is 1024 bytes. MCU can freely allocate the size of each Socket receiving buffer. After the configuration of Sn_RXBUF_SIZE is completed, the receive buffer will be reallocated in the order of Socket0 0 to 7. When setting the receiving buffer, it should be noted that the sum of all Socket receiving buffers cannot exceed 16KB.

### 7.2.13 Socket n TX Buffer Size Register (Sn_TXBUF_SIZE) [0x001F]

This register is used to set the transmit buffer size of Socket n, and the default value is 2(2KB). The register value can be set from 0 to 16, corresponding to 0 to 16KB. CH394Q allocates 2K space for each Socket by default.

<table>
<tr><td colspan="2">Socket 0 receive buffer</td><td colspan="2">Socket 1 receive buffer</td><td></td><td colspan="2">Socket 7 receive buffer</td></tr>
<tr><td>Block 0</td><td>Block 1</td><td>Block 2</td><td>Block 3</td><td>……</td><td>Block 14</td><td>Block 15</td></tr>
<tr><td colspan="2">2K</td><td colspan="2">2K</td><td></td><td colspan="2">2K</td></tr>
</table>

The default allocation of CH394Q's internal transmit buffer is as shown above, which consists of 16 blocks, each of which is 1024 bytes long. MCU can freely allocate the size of each Socket transmitting buffer. After the configuration of Sn_TXBUF_SIZE is completed, the TX buffer will be reallocated in the order of Socket0 0 to 7. When setting the transmit buffer, it should be noted that the sum of all Socket transmit buffers cannot exceed 16KB.

### 7.2.14 Socket n TX Free Size Register (Sn_TX_FS) [0x0020-0x0021]

This register is used to obtain the size of the free space in the transmit buffer of Socket n. When no packet is sent, the free length is the size of the transmitted buffer of Socket n.

Before transmitting data, the user must query this register to obtain the size of the free space (See section 11.2.2 for details), and the length of the transmitted packet shall not be greater than the size of the free space.

For example: The length of the free buffer is 1000.

| Address | 0x0020 | 0x0021 |
|---|---|---|
| Data | 0x03 | 0xE8 |

### 7.2.15 Socket n TX Buffer Read Pointer Register (Sn_TX_RD) [0x0022-0x0023]

This register is used to obtain the transmitting and reading pointer of Socket n, which indicates the current position of the transmitting and reading pointer of CH394Q internal protocol stack. Users generally don't need to care about the value of this register, and CH394Q automatically manages the update.

After successful initialization of Socket n, this register points to the initial address of Socket n transmit buffer. After the user finishes writing the data into the SEND buffer and updates the pointer of Sn_TX_WR, CH394Q will automatically process the data transmission from Sn_TX_RD to Sn_TX_WR by transmitting the SEND command. See section 11.2.2 for the detailed operation.

### 7.2.16 Socket n TX Buffer Write Pointer Register (Sn_TX_WR) [0x0024-0x0025]

This register is used to get or set the transmitting and writing pointer of Socket n. It is necessary to get the transmitting and writing pointer of the current Socket n before each transmitting data process, and update the register value after writing the data to be transmitted. For details, please refer to Section 11.2.2 data transmitting process.

For example: The Socket n TX buffer write pointer is 2048.

| Address | 0x0024 | 0x0025 |
|---|---|---|
| Data | 0x08 | 0x00 |

### 7.2.17 Socket n RX Size Register (Sn_RX_RS) [0x0026-0x0027]

This register is used to obtain the size of data received by Socket n. Refer to Section 11.2.1 receiving data flow for details.

For example: Socket n receives data with a length of 1024.

| Address | 0x0026 | 0x0027 |
|---|---|---|
| Data | 0x04 | 0x00 |

### 7.2.18 Socket n RX Buffer Read Pointer Register (Sn_RX_RD) [0x0028-0x0029]

This register is used to get or set the Socket n receiving buffer reading pointer, indicating the current position of the user receiving the buffer reading pointer.

It is necessary to obtain the reading pointer of the current Socket n before receiving the data flow every time, and update the register value after reading the data. Refer to Section 11.2.1 receiving data flow for details.

For example: The read pointer of Socket n receiving buffer is 2048.

| Address | 0x0028 | 0x0029 |
|---|---|---|
| Data | 0x08 | 0x00 |

### 7.2.19 Socket n RX Buffer Write Pointer Register (Sn_RX_WR) [0x002A-0x002B]

This register is used to obtain the receiving buffer write pointer of Socket n, which indicates the current position of the receiving data write pointer of CH394Q internal protocol stack. Users generally don't need to care about the value of this register, and CH394Q automatically manages the update.

After successful initialization of Socket n, this register points to the initial address of Socket n receiving buffer. When CH394Q receives the required data from Ethernet and loads it into the receiving buffer, it will update the Sn_RX_WR pointer accordingly. Refer to Section 11.2.1 Receiving Data Flow for details.

### 7.2.20 Socket n Interrupt Enable Register (Sn_INTE) [0x002C]

This register is used to control which interrupt sources in Socket n can trigger interrupts. The default value is 0xFF. Each interrupt event enable bit corresponds to one bit of the Socket n interrupt register (Sn_INT).

When the interrupt event is generated, only when the corresponding bit in Sn_INTE is 1 and the nth bit of SINTE is 1 (Socket n interrupt is enabled) will CH394Q pull down the INT pin and generate an interrupt.

Table 7-12 Socket n interrupt enable register

| Bit | Name | Description | Access | Default value |
|---|---|---|---|---|
| [7:5] | Reserved | Reserved | RO | - |
| 4 | INT_SEND_SUC | Transmit complete interrupt:<br>Enabling this bit allows the generation of transmit complete interrupt. | RW | 1 |
| 3 | INT_TIMEOUT | Timeout interrupt:<br>Enabling this bit allows a timeout interrupt to be generated. | RW | 1 |
| 2 | INT_RECV | Receive Interrupt:<br>Enabling this bit allows a receive interrupt to be generated. | RW | 1 |
| 1 | INT_DISCONNECT | Disconnect Interrupt:<br>Enabling this bit allows disconnection interrupt to be generated. | RW | 1 |
| 0 | INT_CONNECT | Connection Interrupt:<br>Enabling this bit allows connection interruption. | RW | 1 |

### 7.2.21 Socket n IP Fragment Register (Sn_IPF) [0x002D-0x002E]

This register is used to set the Socket n IP fragmentation field, and the default value is 0x4000.
For example: The Socket n IP fragmentation field is 0x4000 (No fragmentation).

| Address | 0x002D | 0x002E |
|---|---|---|
| Data | 0x40 | 0x00 |

### 7.2.22 Socket n Keep Alive Time Register (Sn_KEEPALIVE) [0x002F]

This register is used to set the Socket n Keep Alive time, which only takes effect in TCP mode. The default value is 0, indicating that the Keep Alive function is not activated. To enable auto Keep Alive, you need to set the value of this register to be greater than zero. The unit time of the register is 5 seconds, that is, when the register value is 1, the Keep Alive interval is 5 seconds, when it is 2, it is 10 seconds, and so on.

When Socket n is in TCP mode and enters the idle (No data circulation) state after successfully establishing a connection with the opposite end (Sn_STA state is 0x17), the system will automatically transmit Keep Alive packets at the time interval specified by Sn_KEEPALIVE.

If the Keepalive packet is continuously sent without acknowledgement from the other party, the system will trigger a timeout interrupt after the retransmission time limit is exceeded.

When Sn_KEEPALIVE is 0, users can manually transmit Keep Alive packets through the SEND_KEEP command. When Sn_KEEPALIVE is not 0, the SEND_KEEP command does not take effect.

For example: Socket n Keep Alive time registration is 4 (a Keep Alive packet is automatically sent every 20 seconds when it is idle).

| Address | 0x002F |
|---|---|
| Data | 0x04 |

# 8. CH394Q Function Description

## 8.1 SPI Serial Interface

SPI synchronous serial interface signal line includes SPI chip selection input pin SCS, serial clock input pin SCK, serial data input pin SDI and serial data output pin SDO. Through SPI interface, CH394Q can be connected to the SPI serial bus of various MCUs, DSP and MCU with less wires, or make long-distance point-to-point connection.

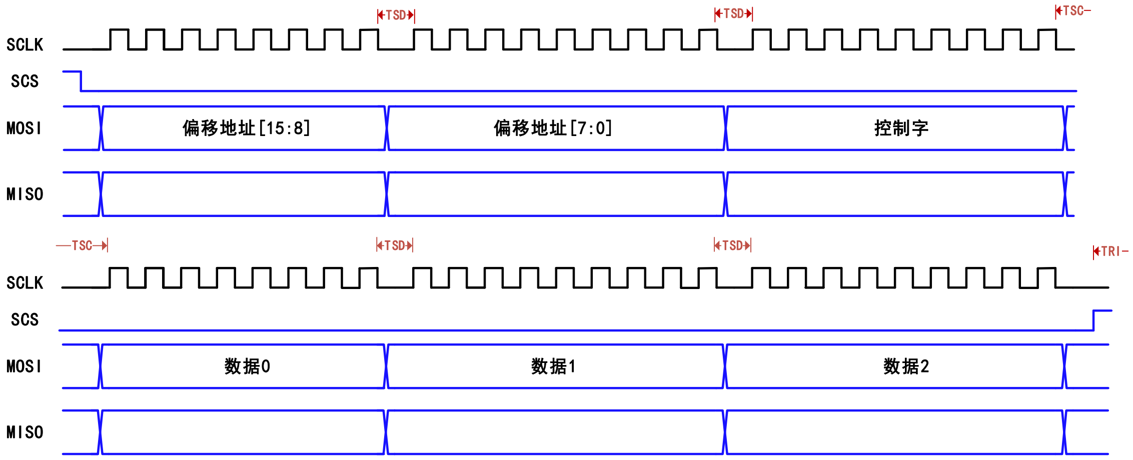
SCS pin of CH394Q chip is driven by SPI chip selection output pin or common output pin of MCU, SCK pin is driven by SPI clock output pin SCK of MCU, SDI pin is driven by SPI data output pin SDO or MOSI of MCU, and SDO pin is connected to SPI data input pin SDI or MISO of MCU. For the hardware SPI interface, it is suggested that the SPI setting is CPOL=CPHA=0 or CPOL=CPHA=1, and the order of data bits is MSB first. The SPI interface of CH394Q also supports the MCU to communicate with the ordinary I/O pin analog SPI interface.

The SPI interface of CH394Q supports SPI mode 0 and SPI mode 3. CH394Q always inputs data from the rising edge of SPI clock SCK, and outputs data from the falling edge of SCK when output is allowed. The order of data bits is high, and 8 bits count as one byte.

When reading and writing the same register, after writing the register value, you need to wait for a $t_{RI}$ processing time, which is about 0.7us. During the processing time, the host is forbidden to operate the register again.

The interval between each byte of SPI is at least one $t_{SD}$ time, about 50ns; The interval between the control word and the data is at least one $t_{SC}$ time, about 100ns, as shown in Figure 8-1 below.

Figure 8-1 SPI serial interface diagram



### 8.1.1 SPI Operation Steps

① SPI chip selection of CH394Q chip generated by MCU is effective at low level;

② MCU transmits data according to SPI output mode. CH394Q always takes the first two bytes received after SPI chip SCS is valid as offset address, the third byte as control word, and the subsequent bytes as data;

③ If it is a write operation, after the MCU transmits the offset address and control word, it continues to transmit several bytes of data to be written, and CH394Q receives them in turn until the MCU prohibits SPI chip selection;

④ If it is a read operation, after the MCU transmits the offset address and control word, it reads several bytes of data from CH394Q until the MCU prohibits SPI chip selection;

⑤ MCU prohibits SPI chip selection of CH394Q chip to end the current SPI operation.

## 8.2 Other Hardware

The CH394Q chip internally integrates 10M/100M Ethernet PHY and MAC, SPI-Slave controller, SRAM, high-speed MCU, firmware program, crystal oscillator and PLL multiplier, power supply power-on reset circuit, etc. The CH394Q chip supports MDI/MDIX lines.

The MDIRP, MDIRN, MDITP, and MDITN of CH394Q chip are the signal lines for Ethernet. PHY of CH394Q supports automatic conversion of MDI/MDIX lines.

The CH394Q chip has a built-in power-on reset circuit, which normally does not require external reset. RSTB pin is used to input an asynchronous reset signal from the outside; when the RSTB pin is low, the CH394Q chip is reset; when the RSTB pin returns to a high level, the CH394Q will continue to delay reset for about 10ms, and the host is prohibited from operating the CH394Q during this period of time. In order to reliably reset the CH394Q during power-up and reduce external interference, a capacitor with a capacity of about 0.1uF can be connected across the RSTB pin and ground.

# 9. CH394Q Parameters

## 9.1 Absolute Maximum Ratings

(Critical or exceeding the absolute maximum value will probably cause the chip to work improperly or even be damaged)

Table 9-1 Absolute maximum value

| Name | Parameter description | Min. | Max. | Unit |
|---|---|---|---|---|
| $T_A$ | Ambient operating temperature | -40 | 85 | ℃ |
| $T_J$ | Junction temperature range | -40 | 100 | ℃ |
| $T_S$ | Ambient temperature during storage | -55 | 150 | ℃ |
| $AV_{DD33}$ | Operating power supply voltage | -0.4 | 4.0 | V |
| $V_{DDIO}$ | I/O supply voltage | -0.4 | 4.0 | V |
| $AV_{DDK}$ | Power decoupling end of core analog circuit | -0.4 | 1.5 | V |
| $DV_{DDK}$ | Power decoupling end of core digital circuit | -0.4 | 1.5 | V |
| $V_{ETH}$ | Voltage on ETH physical signal pin | -0.4 | $AV_{DD33}$+0.4 | V |
| $V_{IN}$ | Input voltage on pin | -0.4 | $V_{DDIO}$+0.4 | V |
| $V_{ESD(HBM)}$ | ESD electrostatic discharge voltage (HBM) for common I/O pins | 4K | | V |
| $I_{IO}$ | Input current on I/O pin | | 20 | mA |
| | Output current on I/O pin | | 20 | |

## 9.2 Electrical Parameters

Table 9-2 Electrical Parameter (AVDD33 = 3.3V, VDDIO = 3.3V, TA = 25℃)

| Name | Parameter description | | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| $AV_{DD33}$ | Operating power supply voltage | | 3.2 | 3.3 | 3.4 | V |
| $V_{DDIO}$ | I/O supply voltage | | 1.7 | 3.3 | 3.6 | V |
| $V_{IL}$ | I/O pin, input low voltage. | $V_{DDIO}$ = 3.3V | 0 | | 0.8 | V |
| | | $V_{DDIO}$ = 1.8V | 0 | | 0.6 | |
| $V_{IH}$ | I/O pin, input high voltage. | $V_{DDIO}$ = 3.3V | 2.0 | | $V_{DDIO}$ | V |
| | | $V_{DDIO}$ = 1.8V | 1.2 | | $V_{DDIO}$ | |
| $V_{OL}$ | Low level output voltage | | | 0.4 | 0.6 | V |
| $V_{OH}$ | High level output voltage | | $V_{DDIO}$-0.6 | $V_{DDIO}$-0.4 | | V |
| $V_{hys}$ | Voltage hysteresis of I/O Schmitt trigger | | | 150 | | mV |
| $C_{IO}$ | I/O pin capacitance | | | 5 | | pF |
| $R_{PU}$ | Pull up equivalent resistance | | 30 | 40 | 55 | kΩ |
| $R_{PD}$ | Pull down equivalent resistance | | 30 | 40 | 55 | kΩ |
| $t_{f(IO)out}$ | Fall time of output high to low level | | | | 6.5 | ns |
| $t_{r(IO)out}$ | Rise time of output low to high level | | | | 6.5 | ns |

## 9.3 Power Consumption

Table 9-3 Power consumption (AVDD33 = 3.3V, VDDIO = 3.3V, TA = 25°C)

| Symbol | Parameter | Condition<br>(All current, with network regulator) | Typical | Unit |
|---|---|---|---|---|
| $I_{DD}$ | Supply current in transfer status | The link of 100BASE-TX path is successful and there are packets on the transceiver channel. | 77.4 | mA |
| | | The link of 10BASE-TX path is successful and there are packets on the transceiver channel. | 48.7 | |
| | Supply current in idle status | The link of 100BASE-TX path is successful and there are no data packets on the transceiver channel. | 76.9 | mA |
| | | The link of 10BASE-TX path is successful and there are no data packets on the transceiver channel. | 47.9 | |
| | Supply current in disconnect status | 100BASE-TX and 10BASE-TX paths are not linked successfully and PHY is in auto-negotiation state. | 58.1 | mA |
| | Supply current in PHY power-down status | | 18.6 | |

## 9.4 AC Electrical Characteristic and Timing

### 9.4.1 SPI Timing

Figure 9-1 SPI mode 0 timing figure

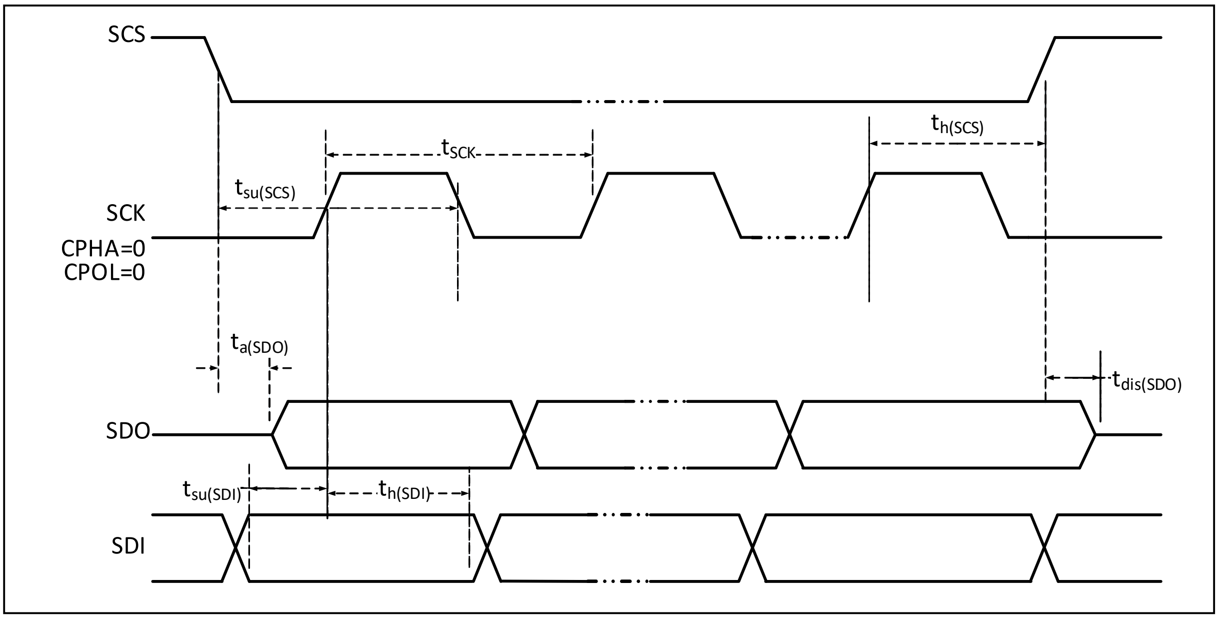


Figure 9-2 SPI mode 3 timing figure

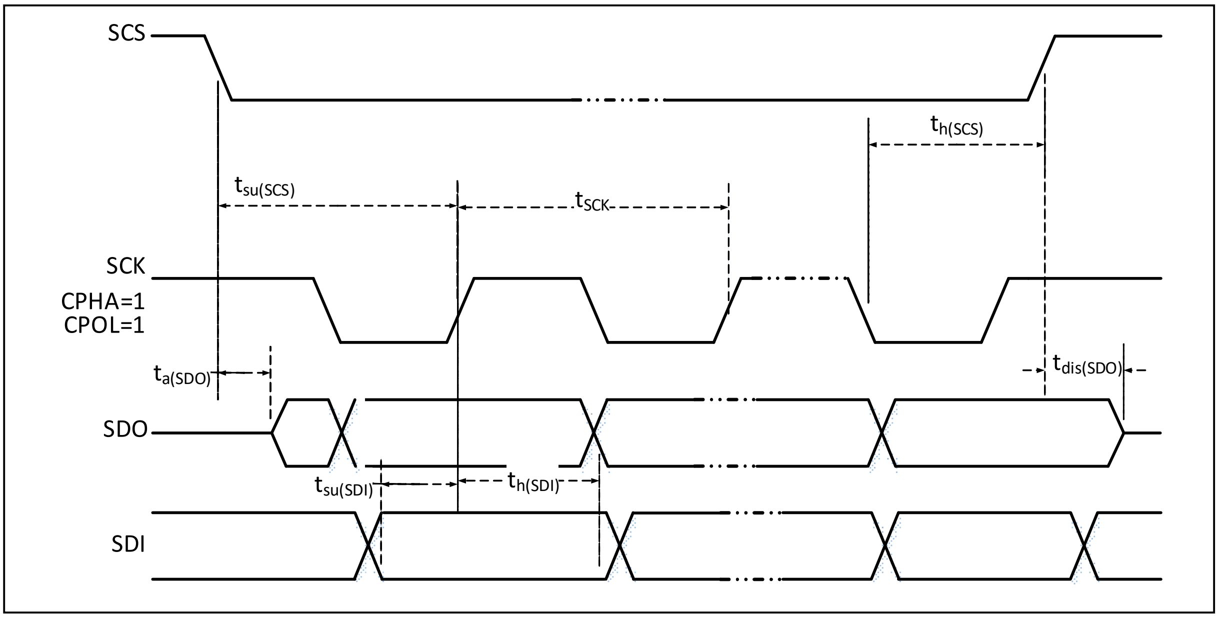


Table 9-4 SPI parameter table (AVDD33 = 3.3V, VDDIO = 3.3V, TA = 25℃)

| Symbol | Parameter | Condition | Min. | Max. | Unit |
|---|---|---|---|---|---|
| $f_{SCK}/t_{SCK}$ | SPI clock frequency | | | 40 | MHz |
| $t_{SU(SCS)}$ | SCS setup time | | 17 | | ns |
| $t_{h(SCS)}$ | SCS holding time | | 17 | | ns |
| $t_{SU(SDI)}$ | Data input setup time | | 4 | | ns |
| $t_{h(SDI)}$ | Data input holding time | | 2 | | ns |
| $t_{a(SDO)}$ | Data output access time | | 0 | 8 | ns |
| $t_{dis(SDO)}$ | Data output disable time | | 0 | 10 | ns |
| $t_{SC}$ | Required interval between SPI transmission control word and data | | 100 | | ns |
| $t_{SD}$ | Required interval between SPI transmission data | | 50 | | ns |
| $t_{RI}$ | Interval required for data frame operation | Write the same address first and then read it. | 0.7 | | us |
| | | Other cases | 0.4 | | |

### 9.4.2 Oscillator and Crystal Oscillator Timing

Table 9-5 Parameters table of oscillator and crystal oscillator timing

| Symbol | Parameter | Condition | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| TCKF | Crystal frequency | Recommended within 30ppm | 24.999 | 25 | 25.001 | MHz |
| TPWH | High clock pulse width | | 15 | 20 | 25 | ns |
| TPWL | Low clock pulse width | | 15 | 20 | 25 | ns |

*Note: The XI and XO pins have built-in two oscillation capacitors required by an external crystal with a load capacitor of 12pF, and only the crystal is needed externally; If an external crystal with a load capacitance of 20pF* *is selected, XI and XO need an additional oscillation capacitance of 15pF to the ground respectively.*

### 9.4.3 Reset Timing

Table 9-6 Reset timing parameter table

| Symbol | Parameter | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|
| $t_{RSTTEMPO}$ | RSTB low level width | 1 | | | us |
| $t_{RSTTEMP1}$ | RSTB high to host operable. | 7 | 9 | 13 | ms |
| | Power on and reset to the host for operation. | 27 | 30 | 35 | ms |

# 10. CH394Q Package

*Note: All dimensions are in millimeters.*

*Pin center spacing is nominal with no error, except for dimensional error of no more than ±0.2mm.*

## 10.1 LQFP48

# 11. CH394Q Application

## 11.1 Application Base

CH394Q integrates IPv4, ARP, ICMP, IGMP, UDP, TCP and other protocols.

TCP and UDP are two important transport layer protocols, both of which use IP as the network layer protocol.

TCP is a connection-oriented transmission, which can provide reliable byte stream transmission service.

UDP is a simple datagram-oriented transport layer protocol. Unlike TCP, UDP can't guarantee that the datagram can reach its destination accurately.

TCP provides highly reliable communication for network devices. Its work includes dividing the data handed to it by the application program into appropriate small pieces and handing it to the lower network layer, confirming the received packets, setting the timeout clock, etc. Because the transport layer provides highly reliable end-to-end communication, the application layer customers ignore all the details. UDP, on the other hand, provides a very simple service for the application layer, which is faster than TCP. It only transmits datagrams from one network terminal to another, but it does not guarantee that the datagrams can reach the other end. Any necessary reliability must be provided by the application layer.

IP is a protocol on the network layer, which is used by both TCP and UDP. Each set of data of TCP and UDP is transmitted in the network through the IP layer.

ICMP is an accessory protocol of IP protocol, which is used by IP layer to exchange error messages or other important information with other hosts or routers. For example, when CH394Q generates unreachable interrupt, it is through ICMP that error message switching is performed. PING also uses the ICMP protocol.

IGMP is an Internet group management protocol, which is mainly used to multicast a UDP datagram to multiple hosts.

ARP is an address resolution protocol, which is used to convert the addresses used by IP layer and network interface layer.

## 11.2 Application Reference Steps

This chapter introduces the operation process of transmitting and receiving data, and can refer to the routine for details.

### 11.2.1 Receive Data Process

When receiving data, we must:

1. Read the received data length register (Sn_RX_RS) of Socket n to obtain the length of the received data;
2. Read the read pointer register (Sn_RX_RD) of the Socket n receiving buffer, and determine the starting address of the received data in the buffer;
3. The host reads data from the address;
4. Write the updated address (Starting address of received data+read length) into the read pointer register (Sn_RX_RD) in the Socket n receiving buffer.
5. Set the RECV bit of Sn_CTRL register of Socket n to complete the receiving operation.

### 11.2.2 Transmit Data Process

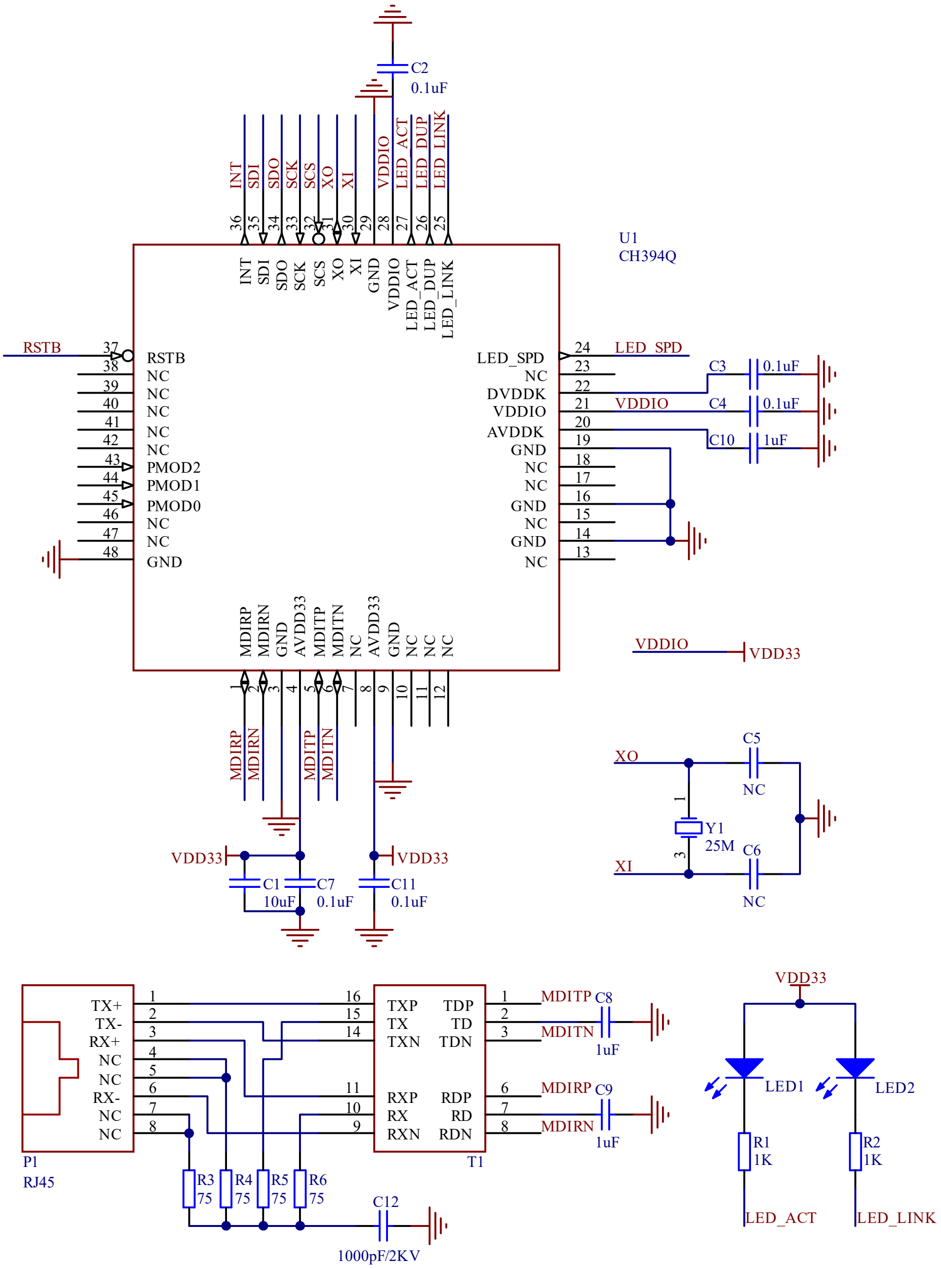
When transmitting data, it is necessary to:

1. Read the idle transmitting buffer length register (Sn_TX_FS) of Socket n to ensure that the buffer area has enough space;
2. Read the transmitting and writing pointer register (Sn_TX_WR) of Socket n, and determine the effective address of the buffer where the current data to be sent should be written;

3. The host writes the data to be transmitted to the address;
4. Write the updated address (the starting address of transmitting data+the length of transmitting data) into the transmitting and writing pointer register (sn _ tx _ wr) of Socket n;
5. Set the SEND bit of Sn_CTRL register of Socket n to transmit data.

## 11.3 CH394Q Hardware Circuit Design

Figure 11-1 Hardware circuit design diagram



*Note:*

*1.CH394Q has built-in partial oscillation capacitance of crystal Y1, and C5 and C6 can be adjusted according to crystal parameters. For Y1 with a load capacitance of 12pF, C5 and C6 are not needed; For Y1, C5 and C6 with a load capacitance of 20pF, 15pF each is recommended.*

*2.CH394Q has built-in Ethernet 50Ω impedance matching resistor, so don't connect 49.9Ω or 50Ω resistor externally, which is equivalent to voltage driving.*

*3.T1 is an ethernet network transformer, and its center tap is grounded through capacitors C8 and C9 respectively, so don't connect any power supply.*