高性能以太网芯片 W5500 数据手册

W5500 是一款全硬件 TCP/IP 嵌入式以太网控制器，为嵌入式系统提供了更加简易的互联网连接方案。W5500 集成了 TCP/IP 协议栈，10/100M 以太网数据链路层（MAC） 及物理层（PHY），使得用户使用单芯片就能够在他们的应用中拓展网络连接。

久经市场考验的 WIZnet 全硬件 TCP/IP 协议栈支持 TCP,UDP,IPv4,ICMP,ARP,IGMP 以及 PPPoE 协议。W5500 内嵌 32K 字节片上缓存以供以太网包处理。如果你使用 W5500， 你只需要一些简单的 Socket 编程就能实现以太网应用。这将会比其他嵌入式以太网方案 更加快捷、简便。用户可以同时使用 8 个硬件 Socket 独立通讯。

W5500 提供了 SPI（外设串行接口）从而能够更加容易与外设 MCU 整合。而且， W5500 的使用了新的高效 SPI 协议支持 80MHz 速率，从而能够更好的实现高速网络通讯。 为了减少系统能耗，W5500 提供了网络唤醒模式（WOL）及掉电模式供客户选择使用。

特点

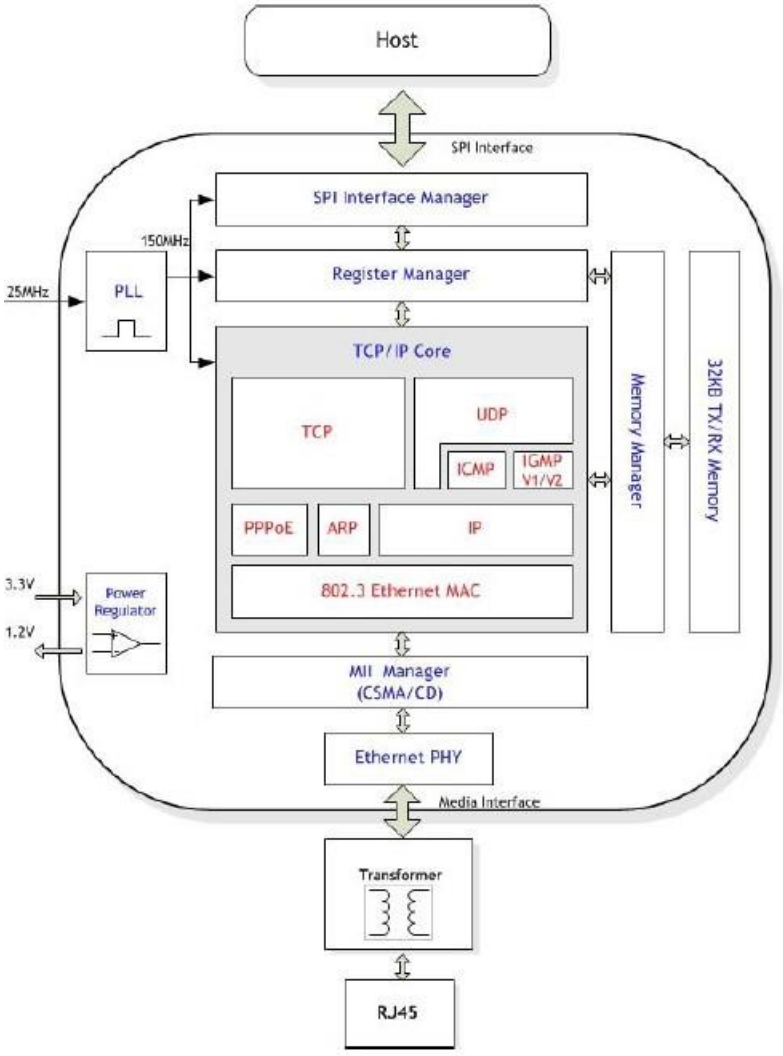
- 支持硬件 TCP/IP 协议：TCP, UDP, ICMP, IPv4, ARP, IGMP, PPPoE
- 支持 8 个独立端口（Socket）同时通讯
- 支持掉电模式
- 支持网络唤醒
- 支持高速串行外设接口（SPI 模式 0，3）
- 内部 32K 字节收发缓存
- 内嵌 10BaseT/100BaseTX 以太网物理层（PHY）
- 支持自动协商（10/100-Based 全双工/半双工）
- 不支持 IP 分片
- 3.3V 工作电压，I/O 信号口 5V 耐压；
- LED 状态显示（全双工/半双工，网络连接，网络速度，活动状态）
- 48 引脚 LQFP 无铅封装（7x7mm, 0.5mm 间距）

目标应用

W5500 适合于以下嵌入式应用：

- 家庭网络设备: 机顶盒、个人录像机、数码媒体适配器
- 串行转以太网: 门禁控制、LED 显示屏、无线 AP 继电器等
- 并行转以太网: POS/微型打印机、复印机

- USB 转以太网: 存储设备、网络打印机
- GPIO 转以太网: 家庭网络传感器
- 安全系统: 数字录像机、网络摄像机、信息亭
- 工厂和楼宇自动化控制系统
- 医疗监测设备
- 嵌入式服务器



1 引脚分配

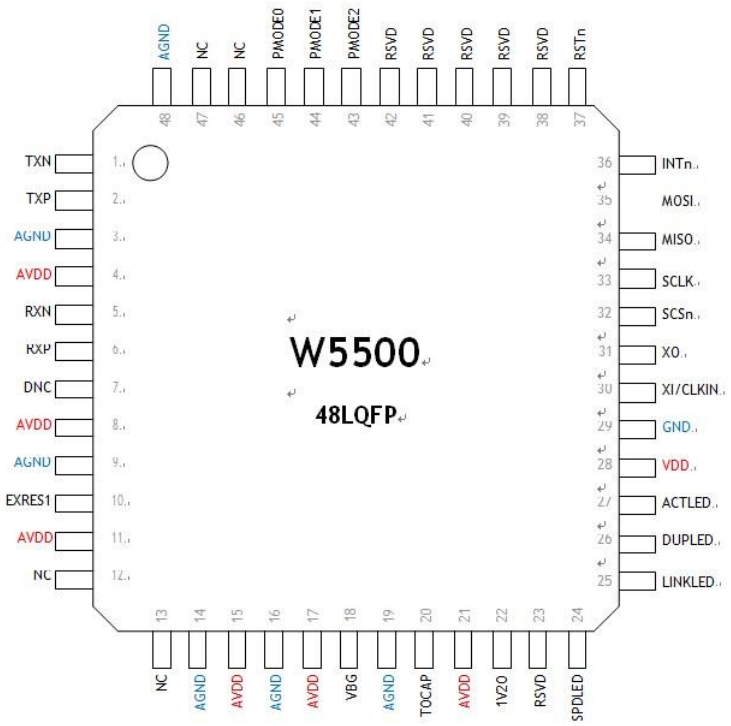


图 1 W5500 引脚分布

1.1 引脚描述

表格 1 引脚类型标记

类型	说明
I	输入（Input）
O	输出（Output）
I/O	输入/输出（Input / Output）
A	模拟（Analog）
PWR	3.3V电源
GND	地

表格 2 W5500 引脚描述

引脚编号	符号	内部偏移[1]	类型	说明
1	TXN	-	AO	**TXP/TXN 信号对（TXP/TXN Signal Pair）** 差分信号传输
2	TXP	-	AO	
3	AGND	-	GND	模拟地（**Analog ground**）
4	AVDD	-	PWR	模拟 **3.3V** 电源（**Analog 3.3V power**）
5	RXN	-	AI	**RXP/RXN 信号对（RXP/RXN Signal Pair）** 差分信号接收
6	RXP	-	AI	
7	DNC	-	AI/O	未连接引脚（**Do Not connect Pin**）
8	AVDD	-	PWR	模拟 **3.3V** 电源（**Analog 3.3V power**）
9	AGND	-	GND	模拟地（**Analog ground**）
10	EXRES1	-	AI/O	外部参考电阻（**External Reference Resistor**） 该引脚需要连接一个精度为 1%的 12.4KΩ外部参考电阻，为内部模拟电路提供偏压； 详细内容请参考‘外部参考电阻’（图 2）；
11	AVDD	-	PWR	模拟 **3.3V**（**Analog 3.3V power**）
12	-		-	**NC**
13	-		-	**NC**
14	AGND	-	GND	模拟地（**Analog ground**）
15	AVDD	-	PWR	模拟 **3.3V**（**Analog 3.3V power**）
16	AGND	-	GND	模拟地（**Analog ground**）
17	AVDD	-	PWR	模拟 **3.3V**（**Analog 3.3V power**）
18	VBG	-	AO	带隙输出电压（**Band Gap Output Voltage**） 该引脚在25℃环境中测量为 1.2V，必须悬空；
19	AGND	-	GND	模拟地（**Analog ground**）
20	TOCAP	-	AO	外部参考电容（**External Reference Capacitor**） 该引脚必须连接一个 4.7uF 电容；而且至该电容的走线要尽量的短一些，从而保证内部信号的稳定；
21	AVDD	-	PWR	模拟 **3.3V**（**Analog 3.3V power**）
22	1V2O	-	AO	**1.2V** 输出稳压（**1.2V Regulator output voltage**） 该引脚必须连接一个 10nF 电容； 这是内部稳压器的输出电压；
23	RSVD	Pull-down	I	该引脚必须接地
24	SPDLED	-	O	网络速度指示灯（**SpeedLED**） 显示当前连接的网络速度状态： 低电平：100Mbps； 高电平：10Mbps；
25	LINKLED	-	O	网络连接指示灯（**LinkLED**） 显示当前连接状态： 低电平：连接建立； 高电平：未连接；
26	DUPLED	-	O	全/半双工指示灯（**Duplex LED**） 显示当前连接的双工状态： 低电平：全双工状态； 高电平：半双工状态；

27	ACTLED	-	O	活动状态指示灯（**ActiveLED**） 显示数据收/发活动时，物理介质子层的载波侦听活动情况： 低电平：有物理介质子层的载波侦听信号； 高电平：无物理介质子层的载波侦听信号；
28	VDD	-	PWR	数字 **3.3V**（**Digital 3.3V Power**）
29	GND	-	GND	数字地（**Digital Ground**）
30	XI/CLKIN	-	AI	外部时钟输入晶振（**Crystal input / External Clock input**） 外部25MHz 晶振输入。 这个引脚也可以连接单向 TTL 晶振。3.3V 时钟须采用外部时钟输入。如果采用该方式，XO 引脚需要悬空；详情参考‘晶振参考电路’（图 3）
31	XO	-	AO	外部时钟输入晶振输出（**Crystal output**） 外部25MHz 晶振输出； 注意：如果通过 XI/CLKIN 驱动使用外部时钟，该引脚悬空；

32	SCSn	Pull-up	I	片选（**Chip Select for SPI bus**） 选用W5500 的SPI 接口，该引脚低电平有效： 低电平：选用； 高电平：不选用；
33	SCLK	-	I	**SPI** 时钟输入（**SPI clock input**） 该引脚用于接收 SPI 主机的 SPI 时钟信号
34	MISO	-	O	**SPI** 主机输入从机（**W5500**）输出
35	MOSI	-	I	**SPI** 主机输出从机（**W5500**）输入
36	INTn	-	O	中断输出（**Interrupt output**） 低电平有效； 低电平：W5500 的中断生效； 高电平：无中断；

<table>
<tr><td>37</td><td>RSTn</td><td>Pull-up</td><td>I</td><td>重置（Reset）
低电平有效；
该引脚需要保持低电平至少 500 us，才能重置 W5500；</td></tr>
<tr><td>38</td><td>RSVD</td><td>Pull-down</td><td>I</td><td>必须接地；</td></tr>
<tr><td>39</td><td>RSVD</td><td>Pull-down</td><td>I</td><td>必须接地；</td></tr>
<tr><td>40</td><td>RSVD</td><td>Pull-down</td><td>I</td><td>必须接地；</td></tr>
<tr><td>41</td><td>RSVD</td><td>Pull-down</td><td>I</td><td>必须接地；</td></tr>
<tr><td>42</td><td>RSVD</td><td>Pull-down</td><td>I</td><td>必须接地；</td></tr>
<tr><td>43</td><td>PMODE2</td><td>Pull-up</td><td>I</td><td rowspan="3">PHY 工作模式选择引脚
这个引脚决定了网络工作模式。具体请参考以下表格：
<table>
<tr><th colspan="3">PMODE [2:0]</th><th rowspan="2">说明</th></tr>
<tr><th>2</th><th>1</th><th>0</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>10BT 半双工，关闭自动协商；</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>10BT 全双工，关闭自动协商；</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>100BT 半双工，关闭自动协商；</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>100BT 全双工，关闭自动协商；</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>100BT 半双工，启用自动协商；</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>未启用</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>未启用</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>所有功能，启动自动协商</td></tr>
</table></td></tr>
<tr><td>44</td><td>PMODE1</td><td>Pull-up</td><td>I</td></tr>
<tr><td>45</td><td>PMODE0</td><td>Pull-up</td><td>I</td></tr>
<tr><td>46</td><td>-</td><td>-</td><td>-</td><td>NC</td></tr>
<tr><td>47</td><td>-</td><td>-</td><td>-</td><td>NC</td></tr>
<tr><td>48</td><td>AGND</td><td>-</td><td>GND</td><td>模拟地（Analog ground）</td></tr>
</table>

在 EXRES1 引脚和模拟地之间需要接一个 12. 4KΩ，精度 1 %的电阻。如下图所示:

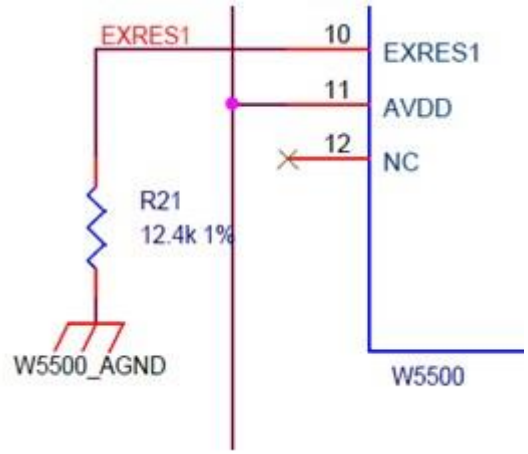


图 2 外部参考电阻

晶振参考周边电路如下图所示：

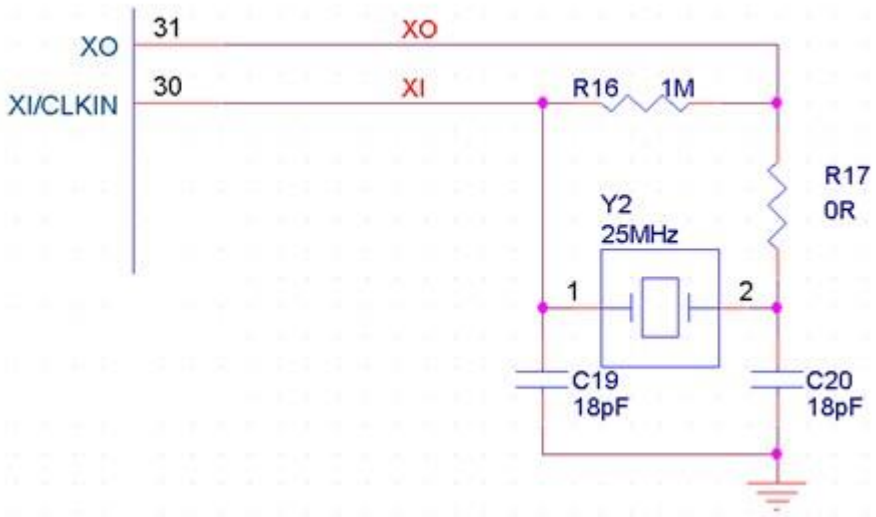


图 3 晶振参考电路

2 主机接口

W5500 提供了 SPI（串行外部接口）作为外设主机接口，共有 SCSn, SCLK, MOSI, MISO 4 路信号，且作为 SPI 从机工作。

W5500 与 MCU 的连接方式如图 4 和图 5 所示。根据其工作模式（可变数据长度模 式/固定数据长度模式）将分别在第 2.3 章节和 2.4 章节做解释说明。

如图 4 所示，可以与其他 SPI 设备共用 SPI 接口

在可变数据长度模式中（如图 4 所示），W5500 可以与其他 SPI 设备共用 SPI 接口。 但是一旦将 SPI 接口指定给 W5500 之后，则不能再与其他 SPI 设备共用，如图 5 所示。

在可变数据长度模式（如图 4 所示），W5500 可以与其他 SPI 设备共用 SPI 接口。 然而，在固定数据长度模式（如图 5 所示），SPI 将指定给 W5500，不能与其他 SPI 设 备共享。

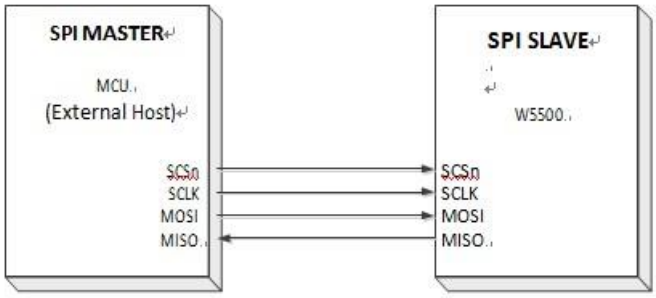


图 4 可变数据长度模式（SCSn 受主机控制）

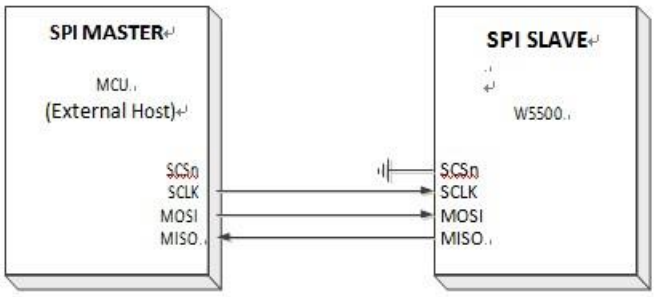


图 5 固定数据长度模式（SCSn 保持接地）

SPI 协议定义了四种工作模式（模式 0，1，2，3）。每种模式的区别是根据 SCLK 的极性及相位不同定义的。SPI 的模式 0 和模式 3 唯一不同的就是在非活动状态下， SCLK 信号的极性。SPI 的模式 0 和 3，数据都是在 SCLK 的上升沿锁存，在下降沿输出。

W5500 支持 SPI 模式 0 及模式 3.MOSI 和 MISO 信号无论是接收或发送，均遵从从最 高标志位（MSB）到最低标志位（LSB）的传输序列。

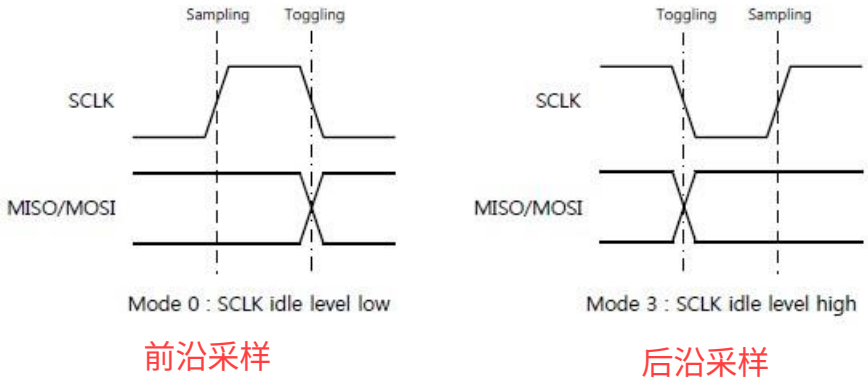


图 6 SPI 模式 0&3

2.1 SPI 工作模式

W5500 与外设主机的通讯受 SPI 数据帧控制（参考第 2.2 章节 SPI 数据帧） 。

W5500 的帧分为 3 段：地址段，控制段，数据段。

地址段为 W5500 寄存器或 TX/RX 内存指定了 16 位的偏移地址。 控制段指定了地址段设定的偏移区域的归属，读/写访问模式以及 SPI 工作模式（可变长度模式/固定长度模式）。

数据段可以设定为任意长度（N-字节，1≤N）或者是固定的长度：1 字节，2 字节 或 4 字节；如果 SPI 工作模式设置为可变数据长度模式（VDM），SPI 的 SCSn 信号需要由外部 主机通过 SPI 帧控制。

在可变数据长度模式下，SCSn 控制 SPI 帧的开始和停止：

SCSn 信号拉低（高电平到低电平），即代表 W5500 的 SPI 帧开始（地址段）；

SCSn 信号拉高（低电平到高电平），即代表 W5500 的 SPI 帧结束（数据段的随机 N 字节数据结尾）；

2.2 SPI 数据帧

W5500 的 SPI 数据帧包括了 16 位地址段的偏移地址，8 位控制段和 N 字节数据段。 如图 7 所示。

8 位控制段可以通过修改区域选择位 (BSB[4:0])，读/写访问模式位(RWB)以及 SPI 工作模式位(OM[1:0])来重新定义。 区域选择位选择了归属于偏移地址的区域。

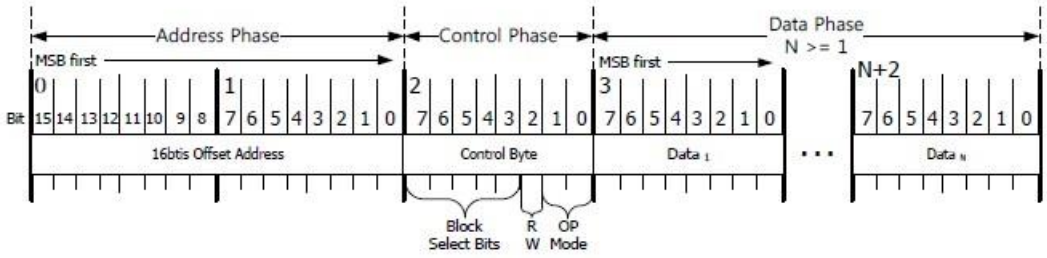


图 7 SPI 数据帧格式

W5500 支持数据的连续读/写。其流程为数据从（2/4/N 字节连续数据的）偏移地址的基址开始传输，偏移地址会（自增寻址）加 1 传输接下来 的数据。

2.2.1 地址段

地址段为 W5500 的寄存器或 TX/RX 缓存区指定了 16 位的偏移地址。 这 16 位偏移地址的值来自于从最高标志位到最低标志位的顺序传输。

SPI 数据帧的数据段（2/4/N 字节）通过偏移地址自增（每传输 1 字节偏移地址加 1） 支持连续数据读/写。

2.2.2 控制段

控制段指定了地址段设定的偏移区域的归属，读/写访问模式以及 SPI 工作模式。

7	6	5	4	3	2	1	0
BSB4	BSB3	BSB2	BSB1	BSB0	RWB	OM1	OM0

表格 3 SPI 数据帧控制段对应位的说明

位	符号	说明
7~3	BSB [4:0]	区域选择位-Block Select Bits W5500有1个通用寄存器，8个Socket寄存器，以及对应每个 Socket的写缓存。 接下来的表格中显示了区域选择位（BSB[4:0]）的区域选择：

BSB [4:0]	含义Meaning
00000	选择通用寄存器
00001	选择Socket 0寄存器
00010	选择Socket 0发送缓存
00011	选择Socket 0接收缓存
00100	保留位
00101	选择Socket 1寄存器
00110	选择Socket 1发送缓存
00111	选择Socket 1接收缓存
01000	保留位
01001	选择Socket 2寄存器
01010	选择Socket 2发送缓存
01011	选择Socket 2接收缓存
01100	保留位
01101	选择Socket 3寄存器
01110	选择Socket 3发送缓存
01111	选择Socket 3接收缓存
10000	保留位
10001	选择Socket 4寄存器
10010	选择Socket 4发送缓存
10011	选择Socket 4接收缓存
10100	保留位
10101	选择Socket 5寄存器
10110	选择Socket 5发送缓存
10111	选择Socket 5接收缓存

		11000	保留位
		11001	选择Socket 6寄存器
		11010	选择Socket 6发送缓存
		11011	选择Socket 6接收缓存
		11100	保留位
		11101	选择Socket 7寄存器
		11110	选择Socket 7发送缓存
		11111	选择Socket 7接收缓存
		如果选择了保留位，将会导致 W5500 故障。	
2	RWB	读/写访问模式位 - **Read/Write Access Mode Bit** 该位设置读/写访问模式： '0' ：读 '1' ：写	
1~0	OM [1:0]	**SPI 工作模式位 - SPI Operation Mode Bits** 该位设置SPI 工作模式。 SPI 模式支持2 种模式：可变数据长度模式和固定长度模式 - 可变数据长度模式（**VDM**）：数据长度通过 SCSn 控制； 外设主机使 SCSn 信号拉低（高电平到低电平），并通知W5500 SI 数据帧地址段的起始地址。 然后外设主机传输控制段。此时， OM[1:0]='00'。 在 N 字节数据段传输完毕后，SCSn 信号拉高（低电平到高电平） 通知W5500SPI 数据帧数据段的结束地址。 在可变数据长度模式下 SCSn 必须通过外设主机通过 SPI 数据帧单元控制。（参见图 4） - 固定数据长度模式（**FDM**） ：在固定数据长度模式下，数据的长度通过 OM[1:0]位来设定，但 不能为 '00' 。 所以，SCSn 信号应该保持低电平状态，然后根据 OM[1:0]的值确定 种长度类型（介于 1 字节，2 字节，4 字节）。（参见图 5）下图显示 的是根据 OM[1:0]位不同值下的 SPI 工作状态：	

		OM[1:0]	含义Meaning
		00	可变数据长度模式，N字节数据段（1 ≤ N）
		01	固定数据长度模式，1字节数据长度（N = 1）
		10	固定数据长度模式，1字节数据长度（N = 2）
		11	固定数据长度模式，1字节数据长度（N = 4）

2.2.3 数据段

在 SPI 工作模式位 OM[1:0]设定了控制端之后，数据段被设定为 2 种长度类型：1 种为可变的 N 字节长度（可变数据长度模式），另以一种为确定的 1/2/4 字节长度（固定数 据长度模式）。

此时，1 字节数据从最大标志位到最小标志位，通过 MOSI 或者 MISO 信号传输。

2.3 可变数据长度模式

在 VDM 模式下，SPI 数据帧的长度被外设主机控制的 SCSn 所定义。这就意味着数据 段长度根据 SCSn 的控制，可以是一个随机值（从 1 字节到 N 字节任何长度均可）。

在 VDM 模式下，M[1:0]位必须为‘00’。

2.3.1 写访问——VDM 模式

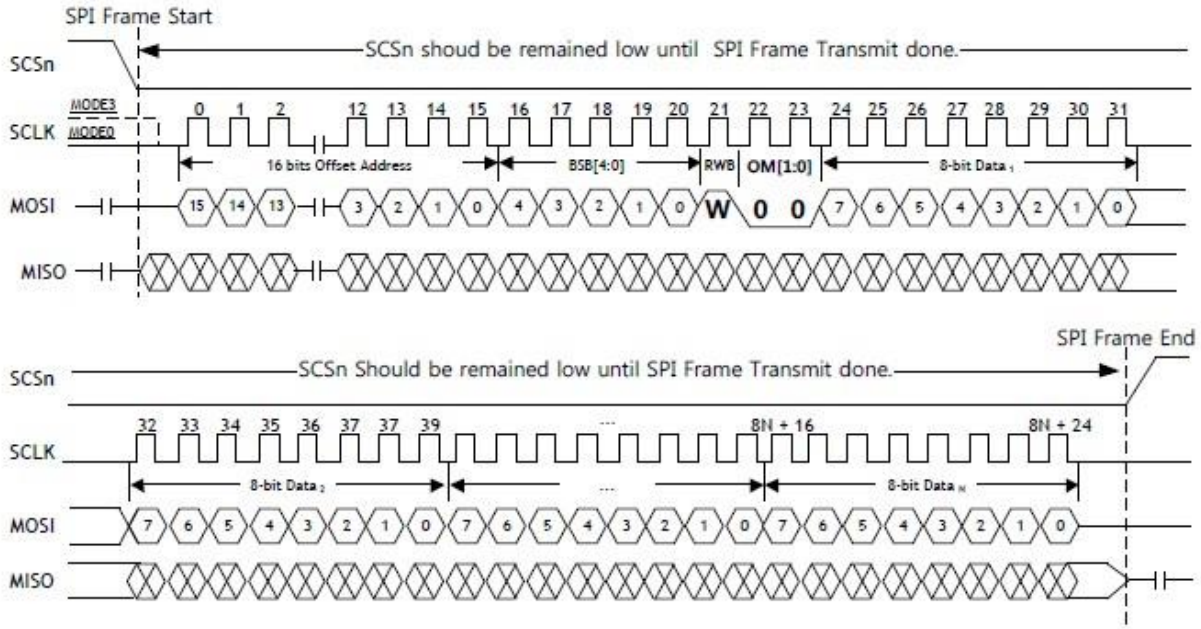


图 8 在 VDM 模式下读 SPI 数据帧

图 8 显示的是在外部主机控制 W5500 读操作时的 SPI 数据帧。

在 VDM 模式下，SPI 数据帧的控制段：读写控制位（RWB）为‘1’，工作模式位为’00’。

此时外设主机在传输 SPI 数据帧之前，须拉低 SCSn 信号引脚。

然后主机通过 MOSI 将 SPI 数据帧的所有位传输给 W5500 ，并在 SCLK 的下降沿同 步。

在完成 SPI 数据帧的传输后，主机拉高 SCSn 信号（低电平到高电平）。 当 SCSn 保持低电平且数据段持续传输，即可实现连续数据写入。

1 字节数据写访问示例

当主机在 VDM 模式下，向通用寄存器区域中的 Socket 中断屏蔽寄存器写入数据‘0xA

A’时，SPI 数据帧的写操作如下所示：

Offset Address = 0×0018

BSB[4:0] = ‘00000’ RWB = ‘1’ OM[1:0] = ‘00’

1st Data = 0xAA

在传输 SPI 数据帧之前，外设主机须拉低 SCSn，然后主机在时钟（SCLK）跳变时同步 传输 1 位数据。在 SPI 数据帧传输完毕后，外设主机拉高 SCSn。（参考图 9）

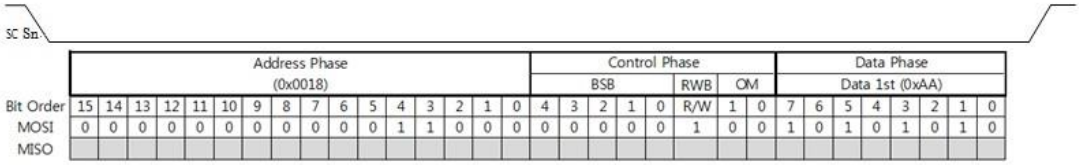


图 9 VDM 模式下，SIMR 寄存器写操作

N 字节写访问示例

当主机在 VDM 模式下，向通用寄存器区域中的 Socket 中断屏蔽寄存器写入 5 字节数 据时（0×11, 0×22, 0×33, 0×44, 0×55），SPI 数据帧的写操作如下所示：

```
Offset Address = 0x0040
BSB[4:0]       = ‘00110’
RWB            =  ‘1’
OM[1:0]        = ‘00’
1st Data       = 0x11
2nd Data       = 0x22
3rd Data       = 0x33
4th Data       = 0x44
5th Data       = 0x55
```

N 字节的写访问如图 10 所示。

5 字节的数据被连续地写入 Socket 1 的写缓存地址：0×0040 – 0×0044。 在 SPI 数据帧传输时，外设主机拉低 SCSn（高电平到低电平）。

在 SPI 数据帧传输完毕时，外设主机拉高 SCSn（低电平到高电平）。

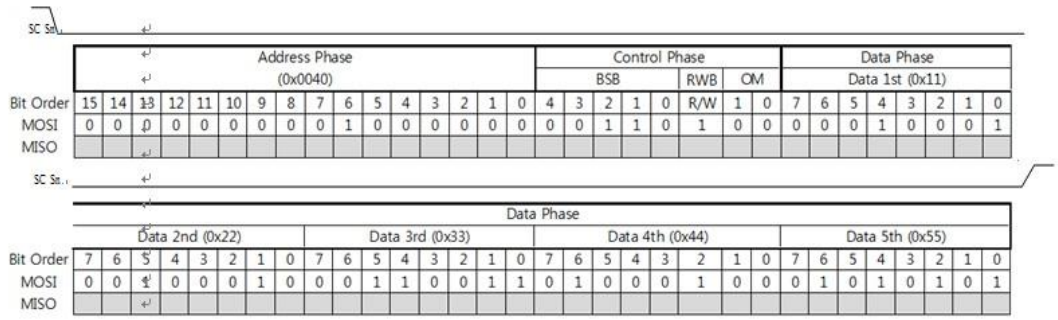


图 10 在 VDM 模式下，向 Socket1 的发送缓存区 0×0040 中写入 5 字节数据

2.3.2 读访问——VDM 模式

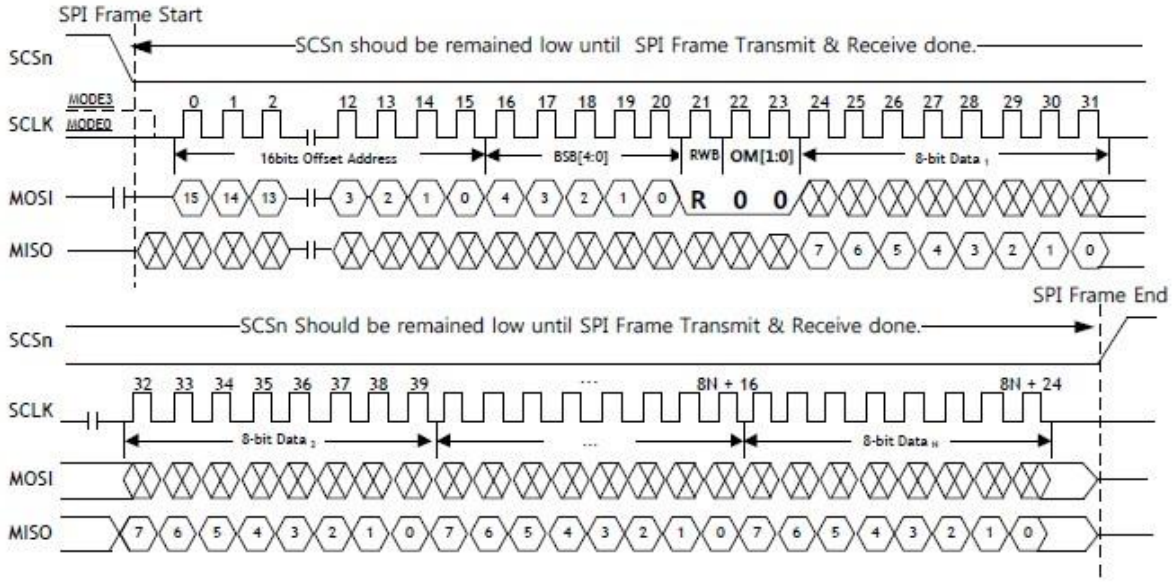


图 11 在 VDM 模式下读 SPI 数据帧

图 11 显示的是当外设主机访问 W5500 做读访问时，SPI 的数据帧格式。

在 VDM 模式下，读/写访问位（RWB）为‘0’（读模式），SPI 数据帧控制段的工作 模式位（OM[1:0]）为‘00’。

与此同时，在 SPI 数据帧传输之前，外设主机拉低 SCSn（高电平到低电平）。 然后主机通过 MOSI 将地址及控制段的所有位传输给 W5500.所有为将在 SCLK 的下降 沿同步。

之后在同步采样时钟（SCLK）的上升沿，主机通过 MISO 接收到所有数据位。 在接收完所有数据后，主机拉高 SCSn（低电平到高电平）。

当 SCSn 保持低电平且数据段持续传输，即可实现连续数据读取。

1 字节数据读访问示例

在 VDM 模式下，当主机读取 Socket 7 寄存器区的 Socket 状态寄存器 (S7_SR),SPI 数据帧的数据读取如下所示。我们让 S7_SR 设置为 Socket 建立模式下（0×17）。

```
Offset Address = 0x0003
BSB[4:0]       = '11101'
RWB            = '0'
OM[1:0]        = '00'
1st Data       = 0x17
```

在 SPI 数据帧传输之前，外设主机拉低 SCSn（高电平到低电平）。然后外设主机通过 M OSI 传输地址段和控制段给 W5500.

然后主机通过 MISO 接收到接收完的数据。

在完成数据段的接收后，主机拉高 SCSn（低电平到高电平）。（参考图 12）

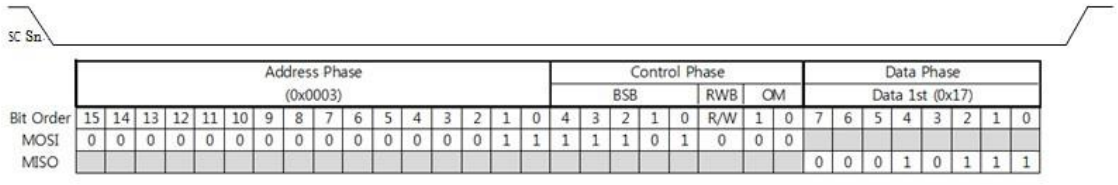


图 12 在 VDM 模式下读 S7_SR

N 字节读访问示例

在 VDM 模式下，当从 Socket3 的地址为 0×0100 的读取缓存中读取 5 字节的数据（0xAA, 0xBB, 0xCC, 0xDD,0xEE）。这 5 个字节数据的读访问 SPI 数据帧如下所示。

```
Offset Address  = 0x0100
BSB[4:0]        = '01111'
RWB             = '0'
OM[1:0]         = '00'
1st Data        = 0xAA
2nd Data        = 0xBB
3rd Data        = 0xCC
4th Data        = 0xDD
5th Data        = 0xEE
```

N 字节读访问如图 13 所示。

从 Socket 3 的接收缓存（地址 0×0100 – 0×0104），连续地读取这 5 字节的数据（0xAA,

0xBB, 0xCC, 0xDD, 0xEE）。

在 SPI 传输数据帧之前，外设主机将 SCSn 拉低。（高电平到低电平） 在 SPI 数据段结束时，外设主机将 SCSn 拉高。（低电平到高电平）

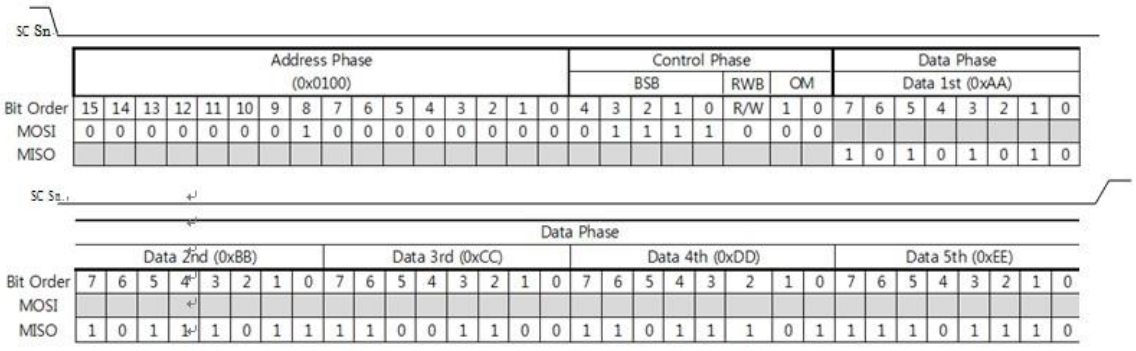


图 13 在 VDM 模式下，读取 Socket 3 接收缓存 0×0100 中的 5 字节数据

2.4 固定数据长度模式（FDM）

在外设主机不能控制 SCSn 时，可以使用固定数据长度模式。

此时，SCSn 必须连接到低电平（保持接地）。与此同时，SPI 接口不能与其他 SPI 设备 共享。（如图 5 所示）

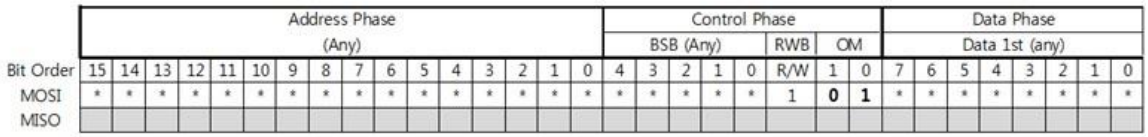
在可变数据长度模式（VDM）中，数据段长度右 SCSn 控制。 但是在固定长度模式（FDM）中，数据长度由 SPI 工作模式位的控制段的值控制（（OM [1:0]）='01'/'10'/'11'）。

由于除了 SCSn 信号和工作模式位（OM[1:0]）设置之外，FDM 模式下 SPI 数据帧与 VD M 模式下的相同，所以此时具体的描述就省略了。

除非特殊情况，一般不提倡使用 FDM 模式。此外，如‘2.4.1 章节’及‘2.4.2 章节’ 所述，我们只能使用 1/2/4 字节 SPI 数据帧。使用其他长度数据帧会导致 W5500 功能问 题。

2.4.1 写访问——FDM 模式

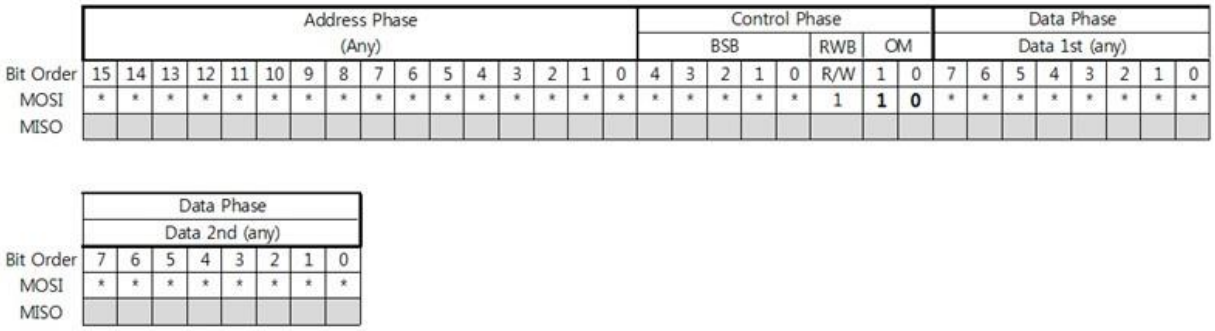
1 字节写访问



	Address Phase (Any)																Control Phase								Data Phase							
																	BSB (Any)					RWB	OM		Data 1st (any)							
Bit Order	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	4	3	2	1	0	R/W	1	0	7	6	5	4	3	2	1	0
MOSI	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	1	**0**	**1**	*	*	*	*	*	*	*	*
MISO																																

图 14 在 FDM 模式下，1 字节写访问 SPI 数据帧

2 字节写访问



	Address Phase (Any)																Control Phase								Data Phase							
																	BSB					RWB	OM		Data 1st (any)							
Bit Order	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	4	3	2	1	0	R/W	1	0	7	6	5	4	3	2	1	0
MOSI	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	1	**1**	**0**	*	*	*	*	*	*	*	*
MISO																																

	Data Phase							
	Data 2nd (any)							
Bit Order	7	6	5	4	3	2	1	0
MOSI	*	*	*	*	*	*	*	*
MISO								

图 15 在 FDM 模式下，2 字节写访问 SPI 数据帧

4 字节写访问

	Address Phase (Any)																Control Phase								Data Phase							
																	BSB					RWB	OM		Data 1st (any)							
Bit Order	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	4	3	2	1	0	R/W	1	0	7	6	5	4	3	2	1	0
MOSI	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	1	**1**	**1**	*	*	*	*	*	*	*	*
MISO																																

	Data Phase								Data Phase								Data Phase							
	Data 2nd (any)								Data 3rd (any)								Data 4th (any)							
Bit Order	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
MOSI	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*
MISO																								

图 16 在 FDM 模式下，4 字节写访问 SPI 数据帧

2.4.2 读访问——FDM 模式

1 字节读访问

	Address Phase (Any)																Control Phase								Data Phase						
																	BSB (Any)					RWB	OM		Data 1st (Any)						
Bit Order	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	4	3	2	1	0	R/W	1	0	7	6	5	4	3	2	1
MOSI	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	0	**0**	**1**							
MISO																									*	*	*	*	*	*	*

图 17 在 FDM 模式下，1 字节读访问 SPI 数据帧

2 字节读访问

	Address Phase (Any)																Control Phase								Data Phase							
																	BSB (Any)					RWB	OM		Data 1st (Any)							
Bit Order	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	4	3	2	1	0	R/W	1	0	7	6	5	4	3	2	1	0
MOSI	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	0	**1**	**0**								
MISO																									*	*	*	*	*	*	*	*

	Data Phase							
	Data 2nd (Any)							
Bit Order	7	6	5	4	3	2	1	0
MOSI								
MISO	*	*	*	*	*	*	*	*

图 18 在 FDM 模式下，2 字节读访问 SPI 数据帧

4 字节读访问

	Address Phase (Any)																Control Phase								Data Phase							
																	BSB (Any)					RWB	OM		Data 1st (Any)							
Bit Order	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	4	3	2	1	0	R/W	1	0	7	6	5	4	3	2	1	0
MOSI	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	0	**1**	**1**								
MISO																									*	*	*	*	*	*	*	*

	Data Phase								Data Phase								Data Phase							
	Data 2nd (Any)								Data 3rd (Any)								Data 4th (Any)							
Bit Order	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
MOSI																								
MISO	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*

图 19 在 FDM 模式下，4 字节读访问 SPI 数据帧

3 寄存器和内存构成

W5500 有 1 个通用寄存器，8 个 Socket 寄存器区，以及对应每个 Socket 的收/发缓存区。

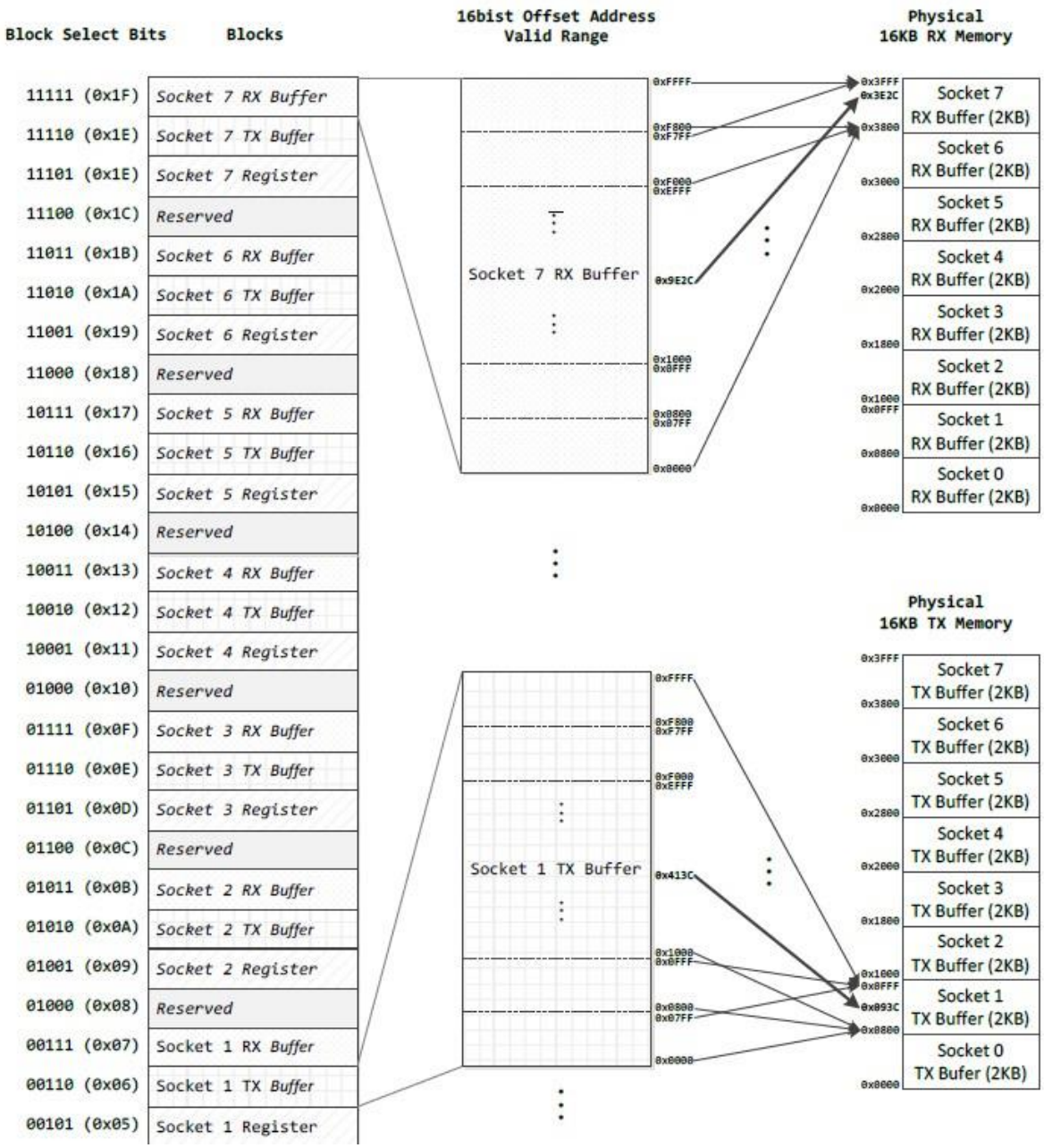
每个区域均通过 SPI 数据帧的区域选择位（BSB[4:0]）来选取。图 20 显示了区域选择位

（BSB[4:0]）选择的区域以及收/发缓存区的可用偏移地址范围。每一个 Socket 的发送 缓存区都在一个 16KB 的物理发送内存中，初始化分配为 2KB。每一个 Socket 的接收缓 存区都在一个 16KB 的物理接收内存中，初始化分配为 2KB。

无论给每个 Socket 分配多大的收/ 发缓存，都必须在 16 位的偏移地址范围内（从

0×0000 到 0xFFFF）。

关于 16KB 收/发内存的构成及访问方式的更多信息，请参考‘3.3 章节’。



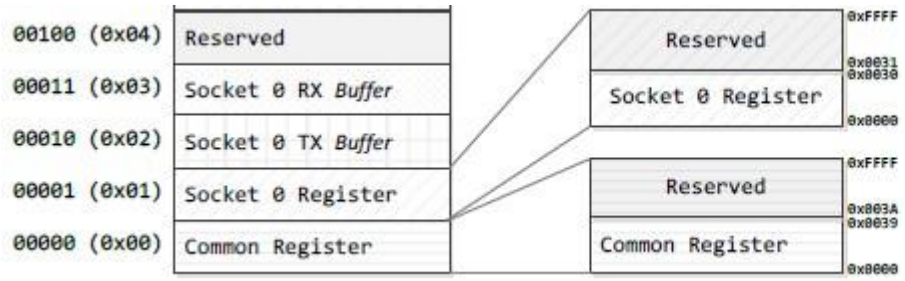


图 20 寄存器及内存构成

3.1 通用寄存器区

通用寄存器区配置了 W5500 的基本信息，例如：IP 及 MAC 地址。该区域可以通过 SPI

数据帧的区域选择位（BSB[4:0]）的值选定。表 3 描述了该区域寄存器的偏移地址。对 于每个寄存器的详细信息，请参考‘4.1 章节’。

Offset	Register	Offset	Register	Offset	Register
	Mode		Interrupt Low Level Timer	0x0021	(PHAR3)
0x0000	(MR)	0x0013	(INTLEVEL0)	0x0022	(PHAR4)
	Gateway Address	0x0014	(INTLEVEL1)	0x0023	(PHAR5)
0x0001	(GAR0)		Interrupt		PPP Session Identification
0x0002	(GAR1)	0x0015	(IR)	0x0024	(PSID0)
0x0003	(GAR2)		Interrupt Mask	0x0025	(PSID1)
0x0004	(GAR3)	0x0016	(IMR)		PPP Maximum Segment Size
	Subnet Mask Address		Socket Interrupt	0x0026	(PMRU0)
0x0005	(SUBR0)	0x0017	(SIR)	0x0027	(PMRU1)
0x0006	(SUBR1)		Socket Interrupt Mask		Unreachable IP address
0x0007	(SUBR2)	0x0018	(SIMR)	0x0028	(UIPR0)
0x0008	(SUBR3)		Retry Time	0x0029	(UIPR1)
	Source Hardware Address	0x0019	(RTR0)	0x002A	(UIPR2)
0x0009	(SHAR0)	0x001A	(RTR1)	0x002B	(UIPR3)
0x000A	(SHAR1)		Retry Count		Unreachable Port
0x000B	(SHAR2)	0x001B	(RCR)	0x002C	(UPORTR0)
0x000C	(SHAR3)		PPP LCP Request Timer	0x002D	(UPORTR1)
0x000D	(SHAR4)	0x001C	(PTIMER)		PHY Configuration
0x000E	(SHAR5)		PPP LCP Magic number	0x002E	(PHYCFGR)
	Source IP Address	0x001D	(PMAGIC)	0x002F	
0x000F	(SIPR0)		PPP Destination MAC Address	~	Reserved
0x0010	(SIPR1)	0x001E	(PHAR0)	0x0038	
0x0011	(SIPR2)	0x001F	(PHAR1)		Chip version
0x0012	(SIPR3)	0x0020	(PHAR2)	0x0039	(VERSIONR)
0x003A ~ 0xFFFF		Reserved			

3.2 Socket 寄存器区

W5500 支持 8 个 Socket 作为通讯信道。每一个 Socket 通过 Socket n 寄存器区控制（0≤n≤7）。Socket n 寄存器可以通过 SPI 数据帧中的区域选择寄存器(BSB[4:0])来选定对应的寄存器 n。<表 5> 定义 了 Socket n 寄存器区对应的 16 位偏移地址。

关于每个寄存器，详情参考‘4.2 章节’

表格 5 Socket n 寄存器区中的偏移地址(0≤n≤7)

Offset	Register
0x0000	Socket n Mode (Sn_MR)
0x0001	Socket n Command (Sn_CR)
0x0002	Socket n Interrupt (Sn_IR)
0x0003	Socket n Status (Sn_SR)
0x0004 0x0005	Socket n Source Port (Sn_PORT0) (Sn_PORT1)
0x0006 0x0007 0x0008 0x0009 0x000A 0x000B	Socket n Destination Hardware Address (Sn_DHAR0) (Sn_DHAR1) (Sn_DHAR2) (Sn_DHAR3) (Sn_DHAR4) (Sn_DHAR5)
0x000C 0x000D 0x000E 0x000F	Socket n Destination IP Address (Sn_DIPR0) (Sn_DIPR1) (Sn_DIPR2) (Sn_DIPR3)
0x0010 0x0011	Socket n Destination Port (Sn_DPORT0) (Sn_DPORT1)
0x0012 0x0013	Socket n Maximum Segment Size (Sn_MSSR0) (Sn_MSSR1)
0x0014	Reserved
0x0015	Socket n IP TOS (Sn_TOS)
0x0016	Socket n IP TTL (Sn_TTL)
0x0017 ~ 0x001D	Reserved
0x001E	Socket n Receive Buffer Size (Sn_RXBUF_SIZE)
0x001F	Socket n Transmit Buffer Size (Sn_TXBUF_SIZE)
0x0020 0x0021	Socket n TX Free Size (Sn_TX_FSR0) (Sn_TX_FSR1)
0x0022 0x0023	Socket n TX Read Pointer (Sn_TX_RD0) (Sn_TX_RD1)
0x0024 0x0025	Socket n TX Write Pointer (Sn_TX_WR0) (Sn_TX_WR1)
0x0026 0x0027	Socket n RX Received Size (Sn_RX_RSR0) (Sn_RX_RSR1)
0x0028 0x0029	Socket n RX Read Pointer (Sn_RX_RD0) (Sn_RX_RD1)
0x002A 0x002B	Socket n RX Write Pointer (Sn_RX_WR0) (Sn_RX_WR1)
0x002C	Socket n Interrupt Mask (Sn_IMR)
0x002D 0x002E	Socket n Fragment Offset in IP header (Sn_FRAG0) (Sn_FRAG1)
0x002F	Keep alive timer (Sn_KPALVTR)
0x0030 ~ 0xFFFF	Reserved

3.3 内存 Memory

W5500 有一个 16KB 的发送内存用于 Socket n 的发送缓存区，以及一个 16KB 的接收内存用于 Socket n 的接收缓存区。

16KB 的发送内存初始化被分配为每个 Socket 2KB 发送缓存区（2KB X 8 = 16KB）。初始化分配的 2KB Socket 发送缓存，可以通过使用 Socket 发送缓存大小寄存器（Sn_TXBUF_SIZE）重新分配。

一旦所有的 Socket 发送缓存大小寄存器（Sn_TXBUF_SIZE）配置完成，16KB 的发送内 存就会按照配置分配给每个 Socket 的发送缓存，并按照从 Socket 0 到 7 顺序分配。16KB 物 理内存的地址是可以自增的。但是 ， 为了避免数据传输错误 ， 需要避免发送缓存大小寄存器

（Sn_TXBUF_SIZE）的和超过 16。

16KB 的读取内存的分派方式与 16KB 的发送内存一样。16KB 的接收内存初始化被分配 为每个 Socket 2KB 接收缓存区（2KB X 8 = 16KB）。初始化分配的 2KB Socket 接收缓存，可 以通过使用 Socket 接收缓存大小寄存器（Sn_XBUF_SIZE）重新分配。

一旦所有的 Socket 发缓存大小寄存器（Sn_TXBUF_SIZE）配置完成，16KB 的发送内存 就会按照配置分配给每个 Socket 的发送缓存，并按照从 Socket 0 到 7 顺序分配。16KB 物理 内存的地址是可以自增的。但是 ， 为了避免数据传输错误 ， 需要避免发送缓存大小寄存器 （ Sn_TXBUF_SIZE）的和超过 16。

对于 16 字节收 / 发内 存的分 配 , 请参 考‘ 第 4 .2 章节’ Sn_TXBUF_SIZE 和

Sn_RXBUF_SIZE 的相关描述。

16KB 的发送内存中分配了对应 Socket n 的发送缓存区，用于为来自主机传输的数据做 缓存。Socket n 的发送缓存区。Socket n 发送缓存区的 16 位偏移地址支持 64KB 的寻址范 围（从 0×000 到 0xFFFF），关于他的配置请参考‘Socket n 发送写指针寄存器(Sn_TX_WR)’ 以及 Socket n 发送读指针寄存器(Sn_RX_WR)。然而，这 16 位偏移地址会自动转化为指定的

16K B 发送 内存的 物理地 址，如 图 20 所示。 请参考 ‘ 4 .2 章节’ 中，关 于

Sn_TX_WR & Sn_TX_RD 的介绍。

16KB 的接收内存中分配了对应 Socket n 的接收缓存区，用于为来自网络传输的数据做 缓存。Socket n 的接收缓存区。Socket n 接收缓存区的 16 位偏移地址支持 64KB 的寻址范 围（从 0×000 到 0xFFFF），关于他的配置请参考‘Socket n 接受读指针寄存器(Sn_RX_RD)’ 以及 Socket n 接受写指针寄存器(Sn_RX_WR)。然而，这 16 位偏移地址会自动转化为指定的

16K B 接收 内存的 物理地 址，如 图 20 所示。 请参考 ‘ 4 .2 章节’ 中，关 于

Sn_RX_RD & Sn_RX_WR 的介绍。

4 寄存器描述

4.1 通用寄存器

MR (模式寄存器- Mode Register) [R/W] [0x0000] [0x00]2

该寄存器用于 S/W 复位，ping block 模式和 PPPoE 模式。

7	6	5	4	3	2	1	0
RST	Reserved	WOL	PB	PPPoE	Reserved	FARP	Reserved

表格 6 通用寄存器描述

位	符号	说明
7	RST	如果该位(bit)为'1'，内部寄存器将被初始化。它会在复位后自动清零。
6	Reserved	保留位
5	WOL	**网络唤醒 Wake on LAN** 0：关闭网络唤醒 1：开启网络唤醒 如果启用网络唤醒，且正常接收到 UDP 发来的 Magic Packet，将会使中断(INTn)引脚拉低。当使用网络唤醒，无论是任何源端口号都需要对 UDP Socket 端口开启。（具体请参考 Socket n 模式寄存器（Sn_MR）开启 Socket 部分） 注意：W5500 支持的 Magic Packet 通过 UDP 传输，UDP 负载包括 6 字节的同步流（'0xFFFFFFFFFFFF'）和 16 个目标 MAC 地址流。关于密码的设置将被忽略。你可以使用任何 UDP 源端口作为网络唤醒使用。
4	PB	**Ping Block 模式** 0: 关闭 Ping block 1: 启用 Ping block 如果该位(bit)设置为'1'，ping 请求时就没有响应。
3	PPPoE	**PPPoE 模式** 0: 关闭 PPPoE 模式 1: 启用 PPPoE 模式 如果你想使用 ADSL，改为需要设为 '1' 。
2	Reserved	保留位
1	FARP	**强迫 ARP 模式** 0：关闭强迫 ARP 模式 1：启用强迫 ARP 模式 在强迫 ARP 模式下，无论是否发送数据都会强迫发送 ARP 请求。
0	Reserved	保留位

GAR (网关 IP 地址寄存器) [R/W] [0x0001 – 0x0004] [0x00]

该寄存器用来设置默认网关地址。 例） 例如：“192.168.0.1”

0x0001	0x0002	0x0003	0x0004
192 (0xC0)	168 (0xA8)	0 (0x00)	1 (0x01)

SUBR (子网掩码寄存器) [R/W] [0x0005 – 0x0008] [0x00]

该寄存器用来设置子网掩码地址.

例） 例如： “255.255.255.0”

0x0005	0x0006	0x0007	0x0008
255 (0xFF)	255 (0xFF)	255 (0xFF)	0 (0x00)

SHAR (源 MAC 地址寄存器) [R/W] [0x0009 – 0x000E] [0x00]

该寄存器用来设置源 MAC 地址。 例） 例如：“00.08.DC.01.02.03”

0x0009	0x000A	0x000B	0x000C	0x000D	0x000E
0x00	0x08	0xDC	0x01	0x02	0x03

SIPR (源 IP 地址寄存器) [R/W] [0x000F – 0x0012] [0x00]

该寄存器用来设置源 IP 地址。 例）例如：“192.168.0.2”

0x000F	0x0010	0x0011	0x0012
192 (0xC0)	168 (0xA8)	0 (0x00)	2 (0x02)

INTLEVEL (低电平中断定时器寄存器) [R/W] [0x0013 – 0x0014] [0x0000]

该寄存器用于设置中断生效等待的时间(IAWT)。当下一个中断触发，中断引脚将会在 IN TLEVEL 时间后，拉低中断引脚（INTn）。

$$I_{AWT} = (INTLEVEL + 1) \times PLL_{CLK} \times 4 \text{ (when INTLEVEL > 0)}$$

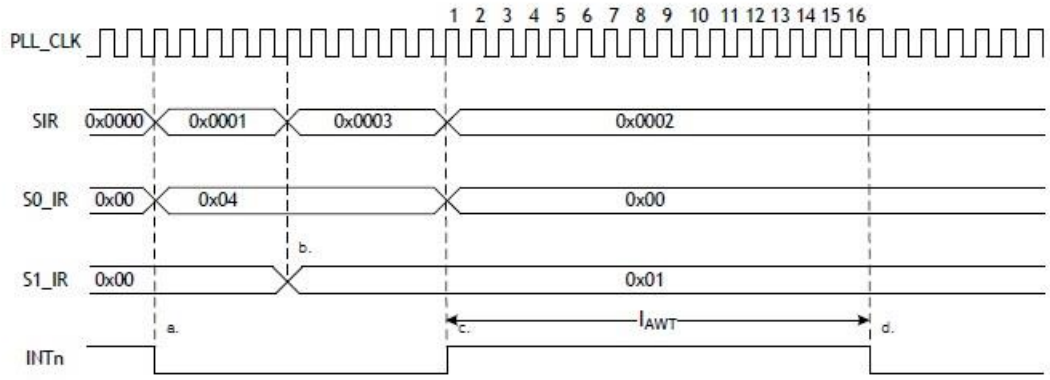


图 21 INTLEVEL 时序

A. 当 Socket 0 的超时中断被触发，在 INTn 引脚被拉低后，S0_IR[3] & SIR[0]设置为

‘1’。

B. 当 Socket 1 的连接中断在前一个中断未处理完成之前被触发，则 INTn 引脚仍然位 低，S1_IR[0] & SIR[1]位设置为‘1’。

C. 如果主机完全是通过清理 S0_IR[3]位来完成之前的中断，则 INTn 引脚拉高，但是

S1_IR[0] & SIR[1]仍然为‘1’。

D. 即 使 S1_IR[0] & SIR[1]位被设置为‘1’，但是在 INTLEVEL 期 间 ，

INTn 不能被拉低。只有过了 INTLEVEL 时间，INTn 才能被拉低。

IR (中断寄存器**) [R/W] [0x0015] [0x00]**

中断寄存器（IR）指明了中断的状态。IR 的每一位都是‘0’，直到被主机写为‘1’.

如果 IR 不等于‘0×00’，INTn 引脚将会被拉低。直到其变为‘0×00’时，INTn 才会被 拉高。

7	6	5	4	3	2	1	0
CONFLICT	UNREACH	PPPoE	MP	Reserved	Reserved	Reserved	Reserved

表格 7 IR 说明

位	符号	说明
7	CONFLICT	**IP 冲突** 在收到 APR 请求时，发现发送方 IP 于本地 IP 重复，该位将被置‘1’。
6	UNREACH	**目标不可抵达** 当接收到 ICMP（目的端口不可达）包后，该位置‘1’. 当该位为‘1’时，通过相应的 UIPR & UPORTR，可能查询到目标信息。如：IP 地址和端口号。
5	PPPoE	**PPPoE 连接关闭** 当 PPPoE 模式下，PPPoE 连接断开时，该位生效；
4	MP	**Magic Packet** 当网络唤醒模式启用并通过 UDP 接收到 Magic Packet 网络唤醒包时，该位生效；
3~0	Reserved	保留位

IMR (中断屏蔽寄存器**) [R/W][0x0016][0x00]**

中断屏蔽寄存器（IMR）用来屏蔽中断源。每个中断屏蔽位对应中断寄存器（IR）中的 一个位。

如果中断屏蔽位被置“1”时，无论何时 IR 对应的位也置”1”，中断即会产生。换而言之， 当 IMR 中屏蔽位被清“0”。即使对应的 IR 中断位置“1”，也不会产生中断。

7	6	5	4	3	2	1	0
IM_IR7	IM_IR6	IM_IR5	IM_IR4	Reserved	Reserved	Reserved	Reserved

表格 8 IMR 说明

位	符号	说明
7	IM_IR7	**IP 冲突中断屏蔽** 0：关闭 IP 冲突中断 1：启用 IP 冲突中断
6	IM_IR6	**目的地址不能抵达中断屏蔽** 0：关闭目的地址不能抵达 1：开启目的地址不能抵达
5	IM_IR5	**PPPoE 关闭中断屏蔽** 0：关闭 PPPoE 关闭中断 1：开启 PPPoE 关闭中断
4	IM_IR4	**Magic Packet 中断屏蔽** 0：关闭 Magic Packet 中断 1：开启 Magic Packet 中断
3~0	Reserved	保留位

SIR (Socket 中断寄存器) [R/W] [0x0017] [0x00]

SIR 指明了 Socket 的中断状态。该寄存器的每一位直到被主机置‘1’前均为‘0’。

如果 Sn_IR 不等于‘0×00’, 那么意味着 SIR 对应的第 n 位为‘1’。INTn 只有在 SIR 为

‘0×00’时才能被拉低。

7	6	5	4	3	2	1	0
S7_INT	S6_INT	S5_INT	S4_INT	S3_INT	S2_INT	S1_INT	S0_INT

表格 9 SIR 描述

位	符号	说明
7 ~ 0	Sn_INT	当 Socket n 的中断触发，则 SIR 寄存器对应位变为 ‘1’

SIMR (Socket 中断屏蔽寄存器) [R/W] [0x0018] [0x00]

SIMR 寄存器中的每一位都对应 SIR 的相应位。当 SIMR 的一位为‘1’，而 SIR 的对应 位为‘1’时，中断将被触发。换而言之，如果 SIMR 的一位为‘0’，那么即使 SIR 对应 位为’1’,中断将不会被触发。

7	6	5	4	3	2	1	0
S7_IMR	S6_IMR	S5_IMR	S4_IMR	S3_IMR	S2_IMR	S1_IMR	S0_IMR

表格 10 SIMR 描述

位	符号	说明
7 ~ 0	Sn_IMR	**Socket n 中断屏蔽** 0：关闭 Socket n 中断 1：启用 Socket n 中断

RTR (重试时间值寄存器) [R/W] [0x0019 – 0x001A] [0x07D0]

RTR 配置了重传超时的时间值。每一单位数值为 100 微秒。初始化时值设为 2000

（0x07D0），即相当于 200 毫秒（100us X 2000）。

在 RTR 配置 的时 间内 ， W5500 等待 Sn-CR(CONNECT, DISCON, CLOSE, SEND, SEND_MAC, SEND_KEEP command)传输后，来自对方的回应。如果在 RTR 时间段内没 有回应，W5500 进行包重传或触发超时中断。

例） 当超时周期别设置为 400ms 时，RTR = (400ms / 1ms) X 10 = 4000(0x0FA0)

0×0019 0x001A
0x0F 0xA0

RCR (重试计数寄存器**) [R/W] [0x001B] [0x08]**

该寄存器是设置重新传送的次数。当第‘RCR+1’次重传时，超时中断就会置‘1’。

（中断寄存器（Sn_IR）的 ‘中断’位(‘TIMEOUT’ bit)设置为’1′）。

例**)** RCR = 0×0007

0x001B
0×07

W5500 的超时可以用 RTR 和 RCR 来配置。W5500 的超时包括地址解析协议(ARP)和

TCP 重新传 送超时。

在 ARP 的重新传送超时（请参阅 RFC 826 http://www.ietf.org/rfc.html)，W5500 会 自动发 送 ARP 请求去对方(peer)的 IP 地址，从而获取 MAC 地址信息（IP、UDP 或 TCP 用于通信）。至 于等待对方(peer)的 ARP 响应方面，如在 RTR 中设置了重新传送时间 时，对方(peer)的 ARP 没 有响应，超时发生和 ARP 将会请求重新传送。一直重复此步 骤达’RCR+1′次。 如果在 ARP 重复请求重新传送次数达到’RCR+1′次时，仍然没有得到 ARP 响应，就会触发最终超时中断，Sn_IR(TIMEOUT)会变为’1′。

ARP 请求的最终超时值(ARPTO)如下:

$$ARP_{TO} = (RTR \times 0.1ms) \times (RCR + 1)$$

在 配 置 了 RTR 和 RCR 期间，发生 TCP 数据包 重 传 超时，W5500 就会发送 TCP

数据包(SYN、FIN、RST、数据包）和等待确认（ACK）。如果没有对方(peer)的 ACK 响 应，就会触发一个临时超时中断且 TCP 数据包（较早前传送的）会重新传送。直到重新 传送’RCR+1′次。即使 TCP 数据包重新传送’RCR+1′次，如果对方(peer)仍然没有的 ACK 回应，就会触发最终超时中断且同一时间 Sn_IR(TIMEOUT)=’1′。

$$TCP_{TO} = \left(\sum_{N=0}^{M} (RTR \times 2^N) \quad + \quad ((RCR - M) \times RTR_{MAX}) \right) \times 0.1ms$$

N : Retransmission count, $0 \le N \le M$

M : Minimum value when RTR x $2^{(M+1)}$ > 65535 and $0 \le M \le RCR$

RTRMAX : RTR x 2^M

例**)** 当 RTR = 2000(0x07D0), RCR = 8(0×0008),

ARPTO = 2000 X 0.1ms X 9 = 1800ms = 1.8s

TCPTO = (0x07D0+0x0FA0+0x1F40+0x3E80+0x7D00+0xFA00+0xFA00+0xFA00+0xFA00) X 0.1ms

= (2000 + 4000 + 8000 + 16000 + 32000 + ((8 – 4) X 64000)) X 0.1ms

= 318000 X 0.1ms = 31.8s

PTIMER (PPP 连接控制协议请求定时寄存器**) [R/W] [0x001C] [0x0028]**

LCP Echo 请求的期限。PTIMER 的值为 1 时，大约是 25 毫秒(ms) 。 例) 如果 PTIMER 是 200，

例**)** 如果 PTIMER 为 200,

200 * 25(ms) = 5000(ms) = 5 seconds

PMAGIC (PPP 连接控制协议幻数寄存器**) [R/W] [0x001D] [0x00]**

该寄存器配置了用于 LCP 回应请求的 4 字节幻数(Magic number)。

例） PMAGIC = 0×01

0x001D

0×01

LCP Magic number = 0×01010101

PHAR (PPPoE 模式下目标 **MAC** 寄存器**) ß[R/W] [0x001E-0x0023] [0x0000]**

PHAR 需要在 PPPoE 连接过程中写入 PPPoE 服务器所需的 MAC 地址。 例） 例如目标 MAC 地址为 00:08:DC:12:34:56

0x001E	0x001F	0x0020	0x0021	0x0022	0x0023
0x00	0x08	0xDC	0x12	0x34	0x56

PSID (PPPoE 模式下会话 **ID** 寄存器**) [R/W] [0x0024-0x0025] [0x0000]**

PSID 需要填入 PPPoE 连接过程中需要的 PPPoE 服务器会话 ID。 例） 例如会话 ID 为 0×1234

0×0024 0025

18 (0×12) 52(0×34)

PMRU (PPPoE 模式下最大接收单元**) [R/W] [0x0026-0x0027] [0xFFFF]**

PMRU 规定了 PPPoE 模式下的最大接收单元。 例） 例如 PPPoE 模式下最大接收单元为 0×1234

0×0026 0027

18 (0×12) 52 (0×34)

UIPR (无法抵达 **IP** 地址寄存器**) [R] [0x0028-0x002B] [0x00000000]**

UPORTR (无法抵达端口寄存器**) [R] [0x002C-0x002D] [0x0000]**

当 W5500 发送数据给一个未开启或不可抵达的端口号时，接收到一个 ICMP 包（目的 地址无法抵达），IR 变为‘1’，UIPR & UPORTR 会分别记录下目的 IP 地址和端口号。

例） 例如“192.168.0.11”

0x0028	0x0029	0x002A	0x002B
192 (0xC0)	168 (0xA8)	0 (0x00)	11 (0x0E)

例） 例如“0x1234”

0x002C	002D
18 (0x12)	52(0x34)

PHYCFGR (W5500 PHY 配置寄存器) [R/W] [0x002E] [0b10111XXX]

PHYCFGR 配置 PHY 的工作模式及 PHY 重启。此外，PHYCFGR 指明了 PHY 的状态， 比如：全/办双工，速度，链接状态。

表格 11 PHYCFGR 描述

<table>
<tr><th>位</th><th>符号</th><th>说明</th></tr>
<tr><td>7</td><td>RST</td><td>重启
Reset [R/W]
当该位为‘0’时，内部 PHY 重启。
在 PHY 重启后，该位须设置为‘1’。</td></tr>
<tr><td>6</td><td>OPMD</td><td>配置 PHY 工作模式
1：通过 PHYCFGR 的 OPMDC[2:0]位配置；
0：通过硬件引脚配置(PMODE[2:0])
该位通过 OPMDC[2:0]位或者 PMODE[2:0]引脚配置 PHY 的工作模式。当 W5500 通过 POR 或 RSTn 引脚重启，则 PHY 的工作模式默认为通过 PMODE[2:0]引脚配置。
在 POR 或 RSTn 重启后，用户仍然可以重新设置使用 OPMDC[2:0]位来配置 PHY 工作模式。
如果用户想重新设置通过 OPMDC[2:0]位来配置 PHY 工作模式。就需要用户在 PMODE[2:0]引脚配置模式下将该位设置为‘1’之后，并将 RST 位置‘0‘重启 PHY。</td></tr>
<tr><td>5~3</td><td>OPMDC</td><td>工作模式配置位 [R/W]
这些位选定 PHY 的工作模式如下表所示：
<table>
<tr><th>5</th><th>4</th><th>3</th><th>说明</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>10BT 半双工，关闭自动协商</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>10BT 全双工，关闭自动协商</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>100BT 半双工，关闭自动协商</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>100BT 全双工，关闭自动协商</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>100BT 半双工，启用自动协商</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>未启用</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>掉电模式</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>全功能，启用自动协商</td></tr>
</table></td></tr>
<tr><td>2</td><td>DPX</td><td>双工工作状态【只读】
1：全双工
0：半双工</td></tr>
<tr><td>1</td><td>SPD</td><td>速度状态【只读】
1: 100Mpbs based
0: 10Mpbs based</td></tr>
<tr><td>0</td><td>LNK</td><td>连接状态 [只读]
1: 已连接
0: 连接断开</td></tr>
</table>

VERSIONR (W5500 芯片版本寄存器) [R] [0x0039] [0x04]

W5500 的版本寄存器默认为 0×04.

4.2 Socket 端口寄存器

S*n*3_MR (Socket n 模式寄存器) [R/W] [0x0000] [0x00]

该寄存器用于配置所有 SOCKET 的选项或协议类型

7	6	5	4	3	2	1	0
MULTI/ MFEN	BCASTB	ND / MC /MMB	UCASTB MIP6B	P3	P2	P1	P0

表格 12 Sn_MR 描述

位	符号	说明
7	MULTI/ MFEN	**UDP 多播模式** **0：关闭多播** **1：开启多播** 该位只有当 UDP 模式(P[3:0] = '0010')，才能生效。 使用多播模式，需要 Sn_DIPR & Sn_DPORT 在 Socket n 通过 Sn_CR 的打开配置命令打开之前，分开配置组播 IP 地址及端口号。 **在 MACRAW 模式下开启 MAC 地址过滤** 0：关闭 MAC 地址过滤 1：启用 MAC 地址过滤 该位只有在 MACRAW(P[3:0] = '0100')模式期间，才能生效。 如果设置为 '1',W5500 只接受广播包以及发送给他本身的包。当该位设置为 '0'，W5500 可以接收到来自网络的所有包。如果用户希望实现混合 TCP、IP 协议栈，建议设置该位为 '1'，以降低主机处理所有接收包的负载。
6	BCASTB	**MACRAW 和 UDP 模式下的网络阻塞** 0：关闭广播阻塞 1：开启广播阻塞 该位在 UDP 模式(P[3:0] = '0010')可以屏蔽接收广播包。 此外，该位在 MACRAW 模式下(P[3:0] = '0100')同样生效。
5	ND/MC/ MMB	**使用无延时 ACK** **Use No Delayed ACK** 0：关闭无延时 ACK 选项 1：开启无延时 ACK 选项

<table>
<tr><td></td><td></td><td>该位只有在 TCP 模式下(P[3:0] = ‘0001’)才能生效。
当该位设置为 ‘1’ 时，W5500 会在从对端接收到数据包后没有任何延时尽快地回复 ACK 包。当该位为 ‘0’ ，W5500 发送 ACK 包需要 RTR 设定的超时时间做延时。
多播
0：使用 IGMP 版本 2
1：使用 IGMP 版本 1
该位只在 UDP 模式(P[3:0] = ‘0010’)，且 MULTI= ‘1’ 时生效。
他配置了 IGMP 消息的版本（加入/离开/报告）。

MACRAW 模式下的多播阻塞
0：关闭多播阻塞
1：开启多播阻塞
该位只有在 MACRAW(P[3:0] = ‘0100’)模式下才能生效。他可以屏蔽多播 MAC 地址的包传输。</td></tr>
<tr><td>4</td><td>UCASTB
MIP6B</td><td>UDP 模式下的单波阻塞
0： 关闭单波阻塞
1： 开启单波阻塞
在 UDP 模式(P[3:0] = ‘0010’)和 MULTI= ‘1’ 的情况下，该位屏蔽接收单波数据包；

MACRAW 模式下，IPv6 包阻塞
0：关闭 IPv6 包阻塞
1：开启 IPv6 包阻塞
该位只有在 MACRAW 模式下才能生效(P[3:0] = ‘0100’)。他将屏蔽接收 IPv6 包。</td></tr>
<tr><td>3</td><td>P3</td><td rowspan="4">协议 Protocol
该位配置 Socket n 使用的协议模式
<table><tr><th>P3</th><th>P2</th><th>P1</th><th>P0</th><th>含义</th></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>Closed</td></tr><tr><td>0</td><td>0</td><td>0</td><td>1</td><td>TCP</td></tr><tr><td>0</td><td>0</td><td>1</td><td>0</td><td>UDP</td></tr><tr><td>0</td><td>1</td><td>0</td><td>0</td><td>MACRAW</td></tr></table>* MACRAW 只有在 Socket 0 下才可用</td></tr>
<tr><td>2</td><td>P2</td></tr>
<tr><td>1</td><td>P1</td></tr>
<tr><td>0</td><td>P0</td></tr>
</table>

3*n* 是 *Socket* 编号（*0*，*1*，*2*，*3*，*4*，*5*，*6*，*7*）.*n* 设置了 SNUM[2:0]控制位集

n is Socket number (0, 1, 2, 3, 4, 5, 6, 7). *n* is set ‘SNUM[2:0]’ in Control Bits sets.

S*n*_CR (Socket n 配置寄存器) [R/W] [0x0001] [0x00]

该寄存器用于设置 Socket n 的配置命令如 OPEN、CLOSE、CONNECT、LISTEN、END 和 RECEIVE。经 W5500 识别这一命令后，Sn_CR 寄存器会自动清零为 0×00。尽管 Sn_CR 被清零为 0×00，但命令仍在处理中。为了验证该命令是否完成，请检查 Sn_IR 或 Sn_SR 寄存器。

表格 13 S*n*_CR 描述

<table>
<tr><th>值</th><th>符号</th><th>说明</th></tr>
<tr><td>0x01</td><td>OPEN</td><td>按照 Sn_MR(P3:P0)的协议选择来初始化和打开(open) Socket n-th。下表显示了 Sn_SR 和 Sn_MR 的对应值。
<table>
<tr><th>Sn_MR (P[3:0])</th><th>Sn_SR</th></tr>
<tr><td>Sn_MR_CLOSE ('0000')</td><td>-</td></tr>
<tr><td>Sn_MR_TCP ('0001')</td><td>SOCK_INIT (0x13)</td></tr>
<tr><td>Sn_MR_UDP ('0010')</td><td>SOCK_UDP (0x22)</td></tr>
<tr><td>S0_MR_MACRAW ('0100')</td><td>SOCK_MACRAW (0x02)</td></tr>
</table></td></tr>
<tr><td>0x02</td><td>LISTEN</td><td>该位只在 TCP 模式(Sn_MR(P3:P0) = Sn_MR_TCP)下生效。在这种模式下，Socket n 被配置为一个 TCP 服务器,它是等待“TCP 客户端”的连接请求（SYN 数据包）。该 Sn_SR 寄存器由 SOCK_INIT 改变为 SOCK_LISTEN。
当一个 TCP 客户端的连接请求成功后，该 Sn_SR 寄存器由 SOCK_LISTEN 改变为 SOCK_ESTABLIESHE，与此同时 Sn_IR(0)会变为'1'。另一方面，当连接失败时，Sn_IR(3)被设置为'1'，Sn_SR 改变为 SOCK_CLOSED。</td></tr>
<tr><td>0x04</td><td>CONNECT</td><td>此模式只适用于 TCP 模式且运行 Socket n 作为 TCP 客户端。通过与存储在目的地址寄存器（Sn_DIPR）和端口号寄存器（Sn_DPORT）中的 IP 地址和端口号进行连接，一个连接请求被发送到 TCP 服务器。当一个客户端的连接请求成功后，Sn_SR 寄存器改为 SOCK_ESTABLIESHED，Sn_IR(0)会变为'1'。
以下三种情况意味着连接请求失败：
1. ARP_{TO} 发生超时(Sn_IR(s)='1')。因为目的地的 MAC 地址不能通过 ARP 过程中获取。
2. 当没有收到 SYN/ACK 数据包，而引起 TCPTO(Sn_IR(3))被设置为'1'时。
3. 当 RST 数据包而不是 SYN/ACK 数据包被接收时。
以上三种情况下，Sn_SR 会改为 SOCK_CLOSED。</td></tr>
<tr><td>0x08</td><td>DISCON</td><td>只在 TCP 模式下有效：
不论“TCP 服务器”或“TCP 客户端”，都使用 DISCON 断开。
主动关闭：它传输断开请求（FIN 数据包）到所连接的对方(peer) 。</td></tr>
</table>

		被动关闭：当从对方(peer)收到 FIN 数据包时，回复一个 FIN 数据包到对方(peer) 。 当成功的执行断开连接操作（FIN/ACK 数据包被接收）时，Sn_SR 改为 SOCK_CLOSED。 当断开请求没有收到 ACK 时，TCP_{TO} 超时中断就会发生(Sn_IR(3)='1')，Sn_SR 改为 SOCK_CLOSED。 比照>如果用 CLOSE 命令来代替 disconnect 命令的话，意味着直接将 Sn_SR 变为 SOCK_CLOSED，将不再执行 FIN/ACK 这种断开机制，从而没有断开连接过程。如果当通讯期间从一个对方(peer)接收到一个 RST 数据包，Sn_SR 将无条件地更改为 SOCK_CLOSED。
0x10	CLOSE	关闭 Socket n. Sn_SR 改为 SOCK_CLOSED。
0x20	SEND	发送(SEND)Socket n 发送(TX)内存中的所有缓冲数据。欲想了解更多详情，请参阅 Socket n 发送(TX)自由尺寸寄存器(Sn_TX_FSR)，Socket n 发送(TX)写指针寄存器(Sn_TX_WR)和 Socket n 发送(TX)读指针寄存器 (Sn_TX_RD)。
0x21	SEND_MAC	只在 UDP 模式下有效： 基本操作是与发送(SEND)相同的。一般来说，发送(SEND)数据通常需要先通过自动 ARP（地址解析协议）请求获得目的地 MAC 地址才能进行传输。而 SEND_MAC 却不需要使用自动 ARP 请求。此时，目的 MAC 地址使用的是主机 Sn_DHAR 的设置，而不是通过 ARP 过程获取的。
0x22	SEND_KEEP	只在 TCP 模式下有效： 通过发送 1 字节在线心跳包来检查连接状态。 如果对方不能在超时计数期内反馈在线心跳包，这个连接将会被关闭并触发超时中断。
0x40	RECV	通过使用接收读指针寄存器(Sn_RX_RD)来判定 Socket n 接收缓存是否完成接收处理。 详情参考，Socket n 接收读大小寄存器(Sn_RX_RSR)，Socket n 接收写指针寄存器(Sn_RX_WR)以及 Socket n 接收读指针寄存器(Sn_RX_RD)。

S*n*_IR (Socket n 中断寄存器**) [R] [0x0002] [0x00]**

Sn_IR 寄 存器用于提供给 Socket n 中断类型信息， 如建立(Establishment)、 终止

(Termination)、接收数据(Receiving data)和超时(Timeout)。当触发一个中断即 Sn_IMR

的对应位是'1'的时候，Sn_IR 的对应位也将会变成'1'。 如果想把 Sn_IR 位清零的话，主机应该将该位置'1'。

7	6	5	4	3	2	1	0
Reserved	Reserved	Reserved	SEND_OK	TIMEOUT	RECV	DISCON	CON

表格 14 Sn_IR 描述

位	符号	说明
7~5	Reserved	保留位
4	SEND_OK	**Sn_IR(SENDOK)中断** 当 SEND 命令完成时，此位生效。
3	TIMEOUT	**Sn_IR(TIMEOUT) 中断** 当 ARP_{TO} 或者 TCP_{TO} 超时被触发时，此位生效。
2	RECV	**Sn_IR(RECV) 中断** 无论何时，只要接收到了对方数据，此位生效。
1	DISCON	**Sn_IR(DISCON) 中断** 当接收到对方的 FIN or FIN/ACK 包时，此位生效。
0	CON	**Sn_IR(CON) 中断** 当成功与对方建立连接，且 Sn_SR 变为 SOCK_ESTABLISHED 状态时，此位生效。

S*n*_SR (Socket n 状态寄存器**) [R] [0x0003] [0x00]**

Sn_SR 指示了 Socket n 的状态，并根据 Sn_CR 或者一些 TCP 模式下的特殊控制包，

如 SYN，FIN 包而改变。

表格 15 Sn_SR 状态描述

值	符号	说明
0x00	SOCK_CLOSED	该位指示了 Socket n 处于关闭状态，资源被释放。 当 DICON，CLOSE 命令生效或当触发超时中断时，W5500 对应的 Socket n 会无视之前的状态，变为 SOCK_CLOSED。
0x13	SOCK_INIT	该位指示了 Socket n 端口打开并处于 TCP 工作模式。 当 Sn_MR (P[3:0]) = '0001'且 OPEN 命令生效时，Sn_SR 变为 SOCK_INIT。之后，用户才可以使用 LISTEN 或 CONNECT 命令。
0x14	SOCK_LISTEN	此位指示着 Socket n 工作在 TCP 服务器模式下，且等待对方（TCP 客户端）的连接请求（SYN Packet）。 当连接请求被成功接收以后，Socket_SR 会变为 SOCK_ESTALBLISHED 状态。 否则将会在触发 TCP_{TO} 超时中断之后，变为 SOCK_CLOSED 状态。
0x17	SOCK_ESTABLISHED	指示了 Socket n 的连接状态。 SOCK_LISTEN 状态下，当 TCP 服务器处理 TCP 客户端的 SYN 请求包或当 CONNECT 命令配置成功时，变为 SOCK_ESTABLISHED。 在此状态下，可以使用 SEND 或者 RECV 命令进行数据包传输。
0x1C	SOCK_CLOSE_WAIT	指示了 Socket n 接收到了来自连接对方发来的断开连接请求（FIN packet）。这是一个半关闭状态，可以进行数据传输。若要全部关闭，需要使用 DISCON 命令。而如果是要关闭 Socket，需要使用 CLOSE 命令。
0x22	SOCK_UDP	指示了 Socket n 处于 UDP 模式下(Sn_MR(P[3:0]) = '0010')。 当 Sn_MR(P[3:0]) = '0010'且 OPEN 命令生效时，Sn_SR 改变为 SOCK_UDP。 不同于 TCP 模式，在这个模式下，数据包可以在无连接过程的情况下传输。
0x02	SOCK_MACRAW	指示了 Socket 0 工作在 MACRAW 模式下(S0_MR(P[3:0]) = '0100')。MACRAW 模式仅在 Socket 0 下生效。 当 S0_MR(P[3:0] = '0100'且 OPEN 命令生效时，Sn_SR 改变为 SOCK_MACRAW。 如 UDP 模式一样，Socket 0 工作在 MACRAW 模式下时，也能在无连接过程的情况下，实现 MAC 数据包（以太网帧）传输。

下表显示了 Socket n 状态改变时的临时状态。

表格 16 Sn_SR 临时状态描述

值	符号	说明
0x15	SOCK_SYNSENT	指示了 Socket n 已经发送连接请求（SYN Packet）到对方。 他显示了发送 CONNECT 命令后，Sn_SR 从 SOCK_INIT 到 SOCK_ESTABLISHED 的临时状态。 如果此时，收到了来自对方的接受连接请求（SYN/ACK packet）则，变为 SOCK_ESTABLISHED。 否则，在 TCP_{TO} 超时(Sn_IR[TIMEOUT] = '1')中断之后，转变为 SOCK_CLOSED。
0x16	SOCK_SYNRECV	指示 Socket n 成功的从对方收到了连接请求包(SYN packet)。 如果 Socket n 成功的给对方发送了连接应答(SYN/ACK packet)，将转变为 SOCK_ESTABLISHED 状态。 否则，在触发超时中断(Sn_IR[TIMEOUT] = '1')后，变为 SOCK_CLOSED。
0x18	SOCK_FIN_WAIT	这些状况表示 SOCKET n 正在关闭。 这显示的是断开连接（主动关闭或被动关闭）的过程。 当断线程序成功完成或 TCPTO(Sn_IR(超时)='1')发生时，它便会更改为 SOCK_CLOSED。
0x1A	SOCK_CLOSING	
0X1B	SOCK_TIME_WAIT	
0X1D	SOCK_LAST_ACK	指示了 Socket n 在被动关闭状态下，正在等待对断开连接请求(FIN packet)做出回应(FIN/ACK packet)。 当 Socket n 成功接收到了断开连接请求的回应或出发超时中断，则变为 SOCK_CLOSED 状态。

S*n*_PORT (Socket n 源端口寄存器) [R/W] [0x0004-0x0005] [0x0000]

该寄存器配置了 Socket n 的源端口号。当 Socket n 工作在 TCP 或 UDP 模式下，该寄

存器生效。注意：必须在 OPEN 命令生效前，完成对该寄存器的设置。 例) 如 SOCKET 0 的端口=5000(0×1388)，配置应如下，

0x0004	0x0005
0x13	0x88

S*n*_DHAR (Socket n 目的 MAC 地址寄存器) [R/W] [0x0006-0x000B] [0xFFFFFFFFFFFF]

Sn_DHAR 寄存器指示的为：UDP 模式下，使用 Send_MAC 配置命令，配置 Socket n

的目标主机 MAC 地址；或者 CONNECT/SEND 配置命令，ARP 过程获取到的 MAC 地址。

例) 如 Socket 0 的目标 MAC 地址 = 08.DC.00.01.02.10，配置应如下，

0x0006	0x0007	0x0008	0x0009	0x000A	0x000B
0x08	0xDC	0x00	0x01	0x02	0x0A

S*n*_DIPR (Socket 目标 IP 地址寄存器) [R/W] [0x000C-0x000F] [0x00000000]

Sn_DIPR 配置或指示的为 Socket n 的目标主机 IP 地址，在 TCP/UDP 模式下生效。

在 TCP 客户端模式下，在 CONNECT 配置命令前，该寄存器设置了 TCP 服务器的 IP

地址。

在 TCP 服务器模式下，他显示了在成功建立连接后，TCP 客户端的 IP 地址；

在 UDP 模式下，他配置了对方主机的 IP 地址以供 SEND 或 SEND_MAC 配置命令后接 收 UDP 包。

例） 如 Socket 0 的目标 IP 地址= 192.168.0.11， 配置应如下，

0x000C	0x000D	0x000E	0x000F
192 (0xC0)	168 (0xA8)	0 (0x00)	11 (0x0B)

S*n*_DPORT (Socket n 目标端口寄存器) [R/W] [0x0010-0x0011] [0x00]

Sn_DPORT 配置或指示了 Socket n 的目标主机端口号，在 TCP/UDP 模式下生效。

在 TCP 客户端模式下，在 CONNET 配置命令前，该寄存器配置了 TCP Server 监听的 端口号。

在 TCP 服务器模式下，他显示了在成功建立连接后，TCP 客户端的端口号；

在 UDP 模式下，他配置了对方主机的端口号以供 SEND 或 SEND_MAC 配置命令后接收

UDP 包。

例) 如 Socket 0 的目标端口号 = 5000(0×1388) ，配置应如下，

0x0010	0x0011
0x13	0x88

S*n*_MSSR (Socket n-th 最大分段寄存器) [R/W] [0x0012-0x0013] [0x0000]

该寄存器配置或显示了 Socket n 的最大传输单元 MTU(Maximum Transfer Unit)。

在 TCP/UDP 模式下，默认该寄存器设定的最大传输单元生效。

然而，在 PPPoE 模式下(MR[PPPoE] = ‘1’)，该寄存器将取决于 PPPoE 的最大传输单元。

表格 17 S*n*_MSSR 描述

Mode	Normal (MR(PPPoE)=‘0’)		PPPoE (MR(PPPoE)=‘1’)	
	Default MTU	Range	Default MTU	Range
TCP	1460	1 ~ 1460	1452	1 ~ 1452
UDP	1472	1 ~ 1472	1464	1 ~ 1464
MACRAW	1514			

当 Socket n 处于 MACRAW 模式时，由于 MTU 不在内部处理，默认的 MTU 将会生效，

因此，当传输的数据比默认的 MTU 大时，主机需要手动的将数据划分成默认 MTU 大小 单元进行传输。

当 Socket n 处于 TCP/UDP 模式，而传输的数据比 MTU 大时，数据将会被自动的划分 成默认 MTU 单元大小传输。

在 UDP 模式下，由于不像 TCP 模式那样涉及到一些连接过程，所以使用了 MTU 配置。 当不同大小的 MTU 数据传输给对方是时，可能会收到 ICMP 包（MTU 分片）。这样的话 IR(FMTU)置‘1’，对方的信息如 MTU 大小以及 IP 地址将分别由 FMTUR 和 UIPR 指定。 如果 IR[MTU] = ‘1’，用户不能发送数据到对方。如果要重新恢复与对方的通讯，可以按 照以下操作：

1. 通过 CLOSED 配置命令关闭 Socket。

2. 设置 Sn_MSS 指定 FMTUR 中的 MTU。

3. 通过 OPEN 配置命令打开 Socket n。

4. 重新与对方通信

例) 如 Socket 0 的 MSS = 1460(0x05B4) ， 配置应如下，

0x0012	0x0013
0x05	0xB4

S*n*_TOS (Socket IP 服务类型寄存器) [R/W] [0x0015] [0x00]

该寄存器设置在 IP 层里 IP header 的 TOS(Type of Service – 服务类型) 字段。它应在执 行 OPEN 命令之前设置。请参考 http://www.iana.org/assignments/ip-parameters.

S*n*_TTL (Socket IP 生存时间寄存器**) [R/W] [0x0016] [0x80]**

该寄存器设置在 IP 层里 IP 头的 TTL(Time-To-Live – 生存时间) 字段。它应在执行

OPEN 命令之前设置。请参考 Uhttp://www.iana.org/assignments/ip-parametersU . **S*n*_RXBUF_SIZE (Socket n** 接收缓存大小寄存器 **- Socket n RX Buffer Size Register) [R/W] [0x001E] [0x02]**

Sn_RXBUF_SIZE 配置了 Socket n 的接收缓存大小。Socket n 接收缓存区大小可以配置 为 1，2，4，8 和 16Kbytes。如果配置为其他大小，则 W5500 不能正常的从对方主机接 收数据。

即使 Socket n 的接收缓存大小初始 默认为 2Kbytes 。 用户仍 然可 以 使用 Sn_RXBUF_SIZE 重新定义。但是所有 Socket 接收缓存（Sn_RXBUF_SIZE）的总大小不能 超过 16Kbytes。否则，将会使得接收异常。

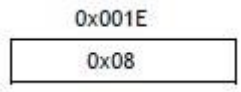
当所有的 Sn_RXBUF_SIZE 配置完成后， 就会按照 Socket 0 到 7 的顺序依次将

16Kbytes 的接收内存分配给各个 Socket 作为接收缓存使用。

不论 Socket n 的接收缓存配置的大小如何，都可以被 16 位的偏移地址寻址找到。（寻 址范围：0×0000 到 0xFFFF）

Value (dec)	0	1	2	4	8	16
Buffer size	0KB	1KB	2KB	4KB	8KB	16KB

例**)** Socket 0 RX Buffer Size = 8KB



S*n*_TXBUF_SIZE (Socket n 发送缓存大小寄存器**) [R/W] [0x001F] [0x02]**

Sn_TXBUF_SIZE 配置了 Socket n 的发送缓存大小。Socket n 发送缓存区大小可 以配置为 1，2，4，8 和 16Kbytes。如果配置为其他大小，则 W5500 不能正常给对 方主机发送数据。

即使 Socket n 的发送缓存大小初始默认为 2Kbytes 。用户仍然可以使用 Sn_RXBUF_SIZE 重新定义。但是所有 Socket 发送缓存的总大小不能超过 16Kbytes。 否则，将会使得发送异常。

当所有的 Sn_TXBUF_SIZE 配置完成后，就会按照 Socket 0 到 7 的顺序依次将

16Kbytes 的发送内存分配给各个 Socket 作为发送缓存使用。

不论 Socket n 的接发送存配置的大小如何，都可以被 16 位的偏移地址寻址找到。

（寻址范围：0×0000 到 0xFFFF）

Value (dec)	0	1	2	4	8	16
Buffer size	0KB	1KB	2KB	4KB	8KB	16KB

例） Socket 0 TX Buffer Size = 4KB

0x001F
0x04

S*n*_TX_FSR (Socket n 空闲发送缓存寄存器) [R] [0x0020-0x0021] [0x0800]

Sn_TX_FSR 显示了 Socket n 发送缓存的空闲空间大小。该寄存器初始化配置为 Sn_TXBUF_SIZE 大小。当传输数据比 Sn_TX_FSR 大时，将不能保存到 Socket n 的发送缓存中。因为多出来的数据会覆盖之前未传输完成的数据。因此在向 Socket n 发送缓保存数据之前，需要先检查一下数据大小是否等于或小于其剩余空间，然后再保存数据到发送缓存并通过 SEND/SEND_MAC 配置命令发送。如果数据比检查到的剩余空间大，需要将数据划分成小于或等于剩余空间的大小之后，再保存数据到 Socket n 发送缓存。

如果 Sn_MR(P[3:0])不是 TCP 模式(‘0001’)，W5500 将计算发送写指针(Sn_TX_WR)和

Socket n 发送读指针之间的空间，并自动将数据划分成相应大小。

如果 Sn_MR(P[3:0])是 TCP 模式(‘0001’)，W5500 将计算发送写指针(Sn_TX_WR)与内部 ACK 指针(指示已经从连接对方接收数据的节点位置)之间的空间。

例) 如 2048(0×0800) 在 S0_TX_FSR 时，

例） In case of 2048(0×0800) in S0_TX_FSR,

0x0020	0x0021
0x08	0x00

Sn_TX_RD (Socket n 发送读指针寄存器) [R] [0x0022-0x0023] [0x0000]

Sn_TX_RD 寄存器可以通过 OPEN 配置命令进行初始化。然而，如果 Sn_MR(P[3:0])是

TCP 模式(‘0001’)，该寄存器将会在 TCP 连接期间，重新进行初始化。 该寄存器初始化之后，会根据 SEND 配置命令自增。SEND 配置命令传输的是 Socket n

发送缓存中，当前 Sn_TX_RD 到 Sn_TX_WR 之间保存的数据。在传输完保存的数据之后，

SEND 配置命令会使得 Sn_TX_RD 等于 Sn_TX_WR。当 Sn_TX_RD 增加的值超出最大值

0xFFFF（大于 0×10000 并产生进位），Sn_TX_RD 会忽略进位，仅使用低 16 位的值。

S*n*_TX_WR (Socket n 发送写指针寄存器) [R/W] [0x0024-0x0025] [0x0000]

Sn_TX_WD 寄存器可以通过 OPEN 配置命令进行初始化。然而，如果 Sn_MR(P[3:0])是

TCP 模式(‘0001’)，该寄存器将会在 TCP 连接期间，重新进行初始化。 该寄存器需要读取或更新如下。

1. 读取发送缓存中将要保存传输数据的首地址。

2. 从 Socket n 的发送缓存对应的首地址开始，保存需要传输的数据；

3. 在保存完传输数据之后，将 Sn_TX_WR 的值增加到传输数据大小。如果增加后， 超过最大值 0xFFFF（比 0×10000 大且产生进位），那么将自动忽略进位，并自动 更新为低 16 位的值。

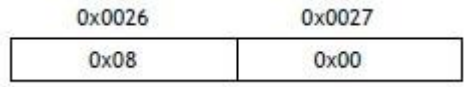
4. 通过使用 SEND 命令发送保存在 Socket n 发送缓存中的数据。

Sn_RX_RSR (Socket n 空闲接收缓存寄存器) [R] [0x0026-0x0027] [0x0000]

Sn_RX_RSR 显示了 Socket n 接收缓存中已接收和保存的数据大小。

Sn_RX_RSR 不会超过 n_RXBUF_SIZE 大小，且计算的为 Socket n 接收写指针(Sn_RX_WR)

和 Socket n 接收读指针之间的空间大小。 例) 如 **2048(0x0800)** 在 **S0_RX_RSR** 时，



S*n*_RX_RD (Socket n 接收读指针寄存器) [R/W] [0x0028-0x0029] [0x0000]

Sn_RX_RD 寄存器可以通过 OPEN 配置命令进行初始化。请确保该寄存器按照以下步 骤读取并更新：

1. 读取保存在接收缓存中数据的首地址；

2. 从保存在 Socket n 接收缓存中数据的首地址开始读取数据；

3. 在读取完毕接收数据，将 Sn_RX_RD 的值更新为所读数据大小。如果增加后的值 超过最大值 0xFFFF，即超过 0×10000 并产生进位，将会忽略进位，只取低 16 位 值。

4. 在接收到 RECV 命令后，将更新后的 Sn_RX_RD 值告知 W5500.

例) 如 **2048(0x0800)** 在 **S0_RX_RD** 时，

0x0028	0x0029
0x08	0x00

Sn_RX_WR (Socket n 接收写指针寄存器) [R] [0x002A-0x002B] [0x0000]

Sn_RX_WR 寄存器可以通过 OPEN 配置命令进行初始化。并且随着数据接收自动增加。 如果 Sn_RX_WR 的值增长到超过最大值 0xFFFF（即超过 0×10000 并产生进位）,那么

将自动忽略进位，并自动更新为低 16 位的值。 例) 如 2048(0×0800) 在 S0_RX_WR 时，

0x002A	0x002B
0x08	0x00

Sn_IMR (Socket n 中断屏蔽寄存器) [R/W] [0x002C] [0xFF]

Sn_IMR 负责屏蔽 Socket n 的中断。每一位都对应了 Sn_IR 寄存器的相应位。Socket n 的中断触发并且 Sn_IMR 的对应位为‘1’时，Sn_IR 的对应位变为‘1’。如果 Sn_IMR 和 Sn_IR 的对应位均为‘1’且 IR 寄存器的相应为‘1’，INTn 引脚便会拉低并使主机 产生中断。

7	6	5	4	3	2	1	0
Reserved	Reserved	Reserved	SEND_OK	TIMEOUT	RECV	DISCON	CON

表格 18 Sn_IMR 描述

位	符号	说明
7~5	Reserved	保留位
4	SENDOK	Sn_IR(SENDOK) 中断屏蔽
3	TIMEOUT	Sn_IR(TIMEOUT) 中断屏蔽
2	RECV	Sn_IR(RECV) 中断屏蔽
1	DISCON	Sn_IR(DISCON) 中断屏蔽
0	CON	Sn_IR(CON) 中断屏蔽

Sn_FRAG (Socket n 分段寄存器) [R/W] [0x002D-0x002E] [0x4000]

它设置了 IP 层中 IP 报头的分段字段。 例) Sn_FRAG0 = 0×4000 (不要分段)

0x002D	0x002E
0x00	0x00

Sn_KPALVTR (Socket 在线时间寄存器) [R/W] [0x002F] [0x00]

Sn_KPALVTR 配置了 SOCKET n 的‘KEEP ALIVE(KA)’在线验证心跳包传输时间。他只在

TCP 模式下生效，在其他模式下将会被忽略。单位时间为 5 秒。

KA 包会在 Sn_SR 变为 SOCK_ESTABLISHED 且与对方至少进行过一次收或发的通讯后 进行传输。如果‘Sn_KPALVTR > 0’，W5500 在一定时间周期会自动传输 KA 包以检查 TCP 的连接状态（自动在线验证）。如果‘Sn_KPALVTR = 0’，将不会启动自动在线验 证，主机可以通过 SEND_KEEP 配置命令发送 KA 包（手动在线验证）。在‘Sn_KPALVTR

> 0’时，将会无视手动在线验证。

例）Sn_KPALVTR = 10（会每 50 秒自动发送一次在线验证包）

0x002F
0x0A

5 电气规范

5.1 绝对最大额定值

表格 19 绝对最大额定值

符号	参数	阈值	单位
V_{DD}	直流供电电压	-0.5 to 4.6	V
V_{IN}	直流输入电压	-0.5 to 6	V
V_{OUT}	直流输出电压	-0.5 to 4.6	V
I_{IN}	直流输入电流	±5	mA
T_{OP}	工作温度	-40 to +85	°C
T_{STG}	存储温度	-65 to +150	°C

*备注：设备加载超过‘绝对最大额定值’时，可能造成永久性损坏。

5.2 绝对最大额定值 (电气灵敏度)

静电释放

表格 20 ESD

符号	参数	测试环境	等级	最大值（1）	单位
VESD(HBM)	Electrostatic discharge voltage (human body model)	TA = +25 °C conforming to MIL-STD 883F Method 3015.7	2	2000	V
VESD(MM)	Electrostatic discharge voltage (man machine model)	TA = +25 °C conforming to JEDEC EIA/JESD22 A115-A	B	200	V
VESD(CDM)	Electrostatic discharge voltage (charge device model)	TA = +25 °C conforming to JEDEC JESD22 C101-C	III	500	V

静态锁定

表格 21 静态锁定

符号	参数	测试环境	等级	最大值（1）	单位
LU	Static latch-up class	TA = +25 °C conforming to JESD78A	I	≥ ±200	mA

5.3 直流特性

表格 22 直流特性

符号	参数	测试环境	最小	标准	最大	单位
V_{DD}	供应电压	Apply VDD, AVDD	2.97	3.3	3.63	V
V_{IH}	高电平输入电压		2.0		5.5	V
V_{IL}	低电平输入电压		- 0.3		0.8	V
V_T	阈值电压	All inputs except XI	1.30	1.41	1.53	V
V_{T+}	施密特触发 低电平到高电平阈值电压	All inputs except XI	1.53	1.64	1.73	V
V_{T-}	施密特触发 高电平到低电平阈值电压	All inputs except XI	0.95	1.02	1.09	V
T_J	节点温度		0	25	125	°C
I_L	输入漏电流				±1	μA
R_{PU}	上拉电阻	SCSn, RSTn, PMODE[2:0]	62	77	112	Kohm
R_{PD}	下拉电阻	RSVD(Pin 23, Pin 38 ~ Pin 42)	48	85	174	Kohm
V_{OL}	低电平输出电压	IOL = 8mA, All outputs except XO			0.4	V
V_{OH}	高电平输出电压	IOH = 8mA, All outputs except XO	2.4			V
I_{OL}	低电平输出电流	VOL = 0.4V, All outputs except XO	8.6	13.9	18.9	mA
I_{OH}	高电平输出电流	VOH = 2.4V, All outputs except XO	12.5	26.9	47.1	mA
I_{DD1}	电源电流 (正常模式)	VDD=3.3V, AVDD=3.3V, Ta = 25°C		132		mA
I_{DD2}	电源电流 (掉电模式)	PHY Power Down mode, VDD=3.3V, AVDD=3.3V, Ta = 25°C		13		mA

5.4 功耗

(测试环境: VDD=3.3V, AVDD=3.3V, Ta = 25°C)

表格 23 功耗

状态	最小	标准	最大	单位
100M Link	-	128	-	mA
10M Link	-	75	-	mA
Un-Link (Auto-negotiation mode)	-	65	-	mA
100M Transmitting	-	132	-	mA
10M Transmitting	-	79	-	mA
Power Down mode	-	13	-	mA

5.5 交流特性

5.5.1 复位时钟

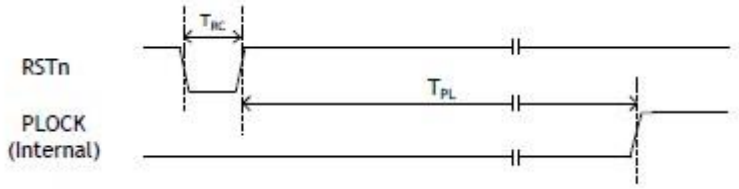


图 22 复位时钟

表格 24 复位时钟

符号	描述	最小	最大
T_{RC}	复位时钟周期	500 us	-
T_{PL}	RSTn to internal PLOCK (PLL 锁相环)	-	1 ms

5.5.2 唤醒时间

整流器的唤醒时间为：**10us**

5.5.3 晶体特性

表格 25 晶体特性

参数	阈值
频率	25 MHz
频率误差 (at 25℃)	±30 ppm
并联电容	7pF Max
驱动功率	59.12uW/MHz
负载电容	18pF
老化速度 (at 25℃)	±3ppm / year Max

5.5.4 SPI 时钟

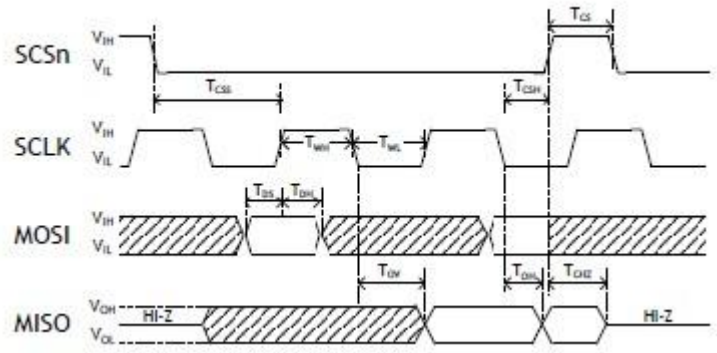


图 23 SPI 时钟

表格 26 SPI 时钟

符号	说明	最小	最大	单位
F_{SCK}	SCK Clock Frequency		80/33.3[4]	MHz
T_{WH}	SCK High Time	6		ns
T_{WL}	SCK Low Time	6		ns
T_{CS}	SCSn High Time	30		ns
T_{CSS}	SCSn Setup Time	5	-	ns

4 理论值速度

即使理论设定值速率为 80MHz，但是高速信号在受电磁串扰和长信号线的影响下可能会失真。目前实测具

有稳定波形的速率，至少能够保证为 33.3MHz.

详情参考 SPI 应用笔记，里面展示了 WIZnet 的测试环境及结果。

T_{CSH}	SCSn Hold Time	5		ns
T_{DS}	Data In Setup Time	3		ns
T_{DH}	Data In Hold Time	3		ns
T_{OV}	Output Valid Time		5	ns
T_{OH}	Output Hold Time	0		ns
T_{CHZ}	SCSn High to Output Hi-Z		2.1[5]	ns

5.5.5 变压器特性

表格 27 变压器特性

参数	发射端	接收端
线匝比	1:1	1:1
电感系数	350 uH	350 uH

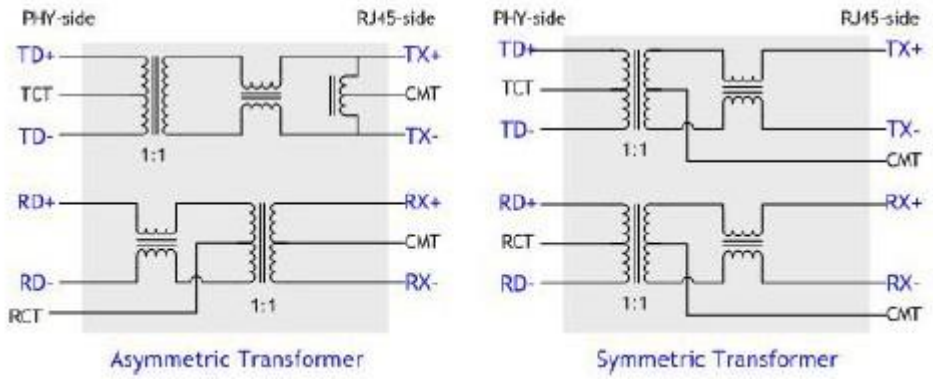


图 24 变压器类型

5.5.6 极性变换 MDIX

W5500 不支持自动极性变换功能。 因此，用户需要使用直连线与交换机或路由器连接，使用交叉线与终端设备（如服务器，工作站 或其他 W5500）相连。 然而，用户可以使用任何一种类型的网线与具有自动极性变换的其他设备相连。对应接口会自动 纠正不正确的布线。

6 封装描述

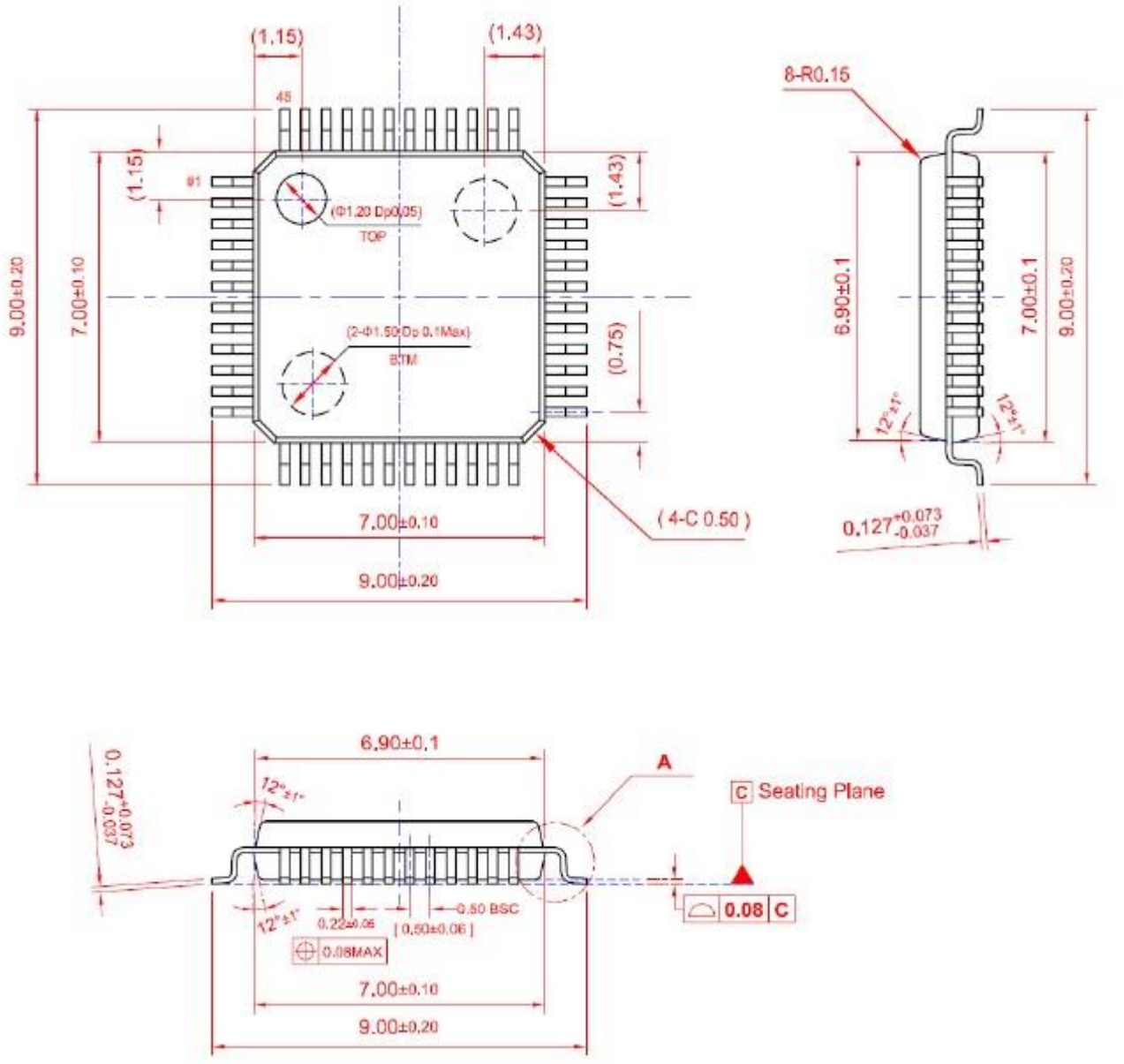


图 25 封装描述

注意：1. 以上尺寸不包括模具突出部分；

2. ( )代表仅供参考；

3. [ ]基于组装质量

4. 单位：mm