

SH3-VER2 开发板用户手册

清华大学电机系

VER 1.00

Dec. 16, 2006

注意:

- 1、手册中的图与实际产品略有不同，请以实际产品为准。
- 2、我们尽力避免手册中的错误，如果出现，敬请谅解。
- 3、如没有作者的授权，请勿将此手册用做商业之目的。

1 使用前必读

1. 系统组件

- SH3-VER2 开发板
- RS232孔到针9芯直通串口线
- AC 220V转DC 5V电源适配器
- 系统初始化FLASH
- 光盘（包含使用说明及示例代码）

2. 连接

- RS232接口一般不支持热插拔，请在断电后插拔串口线。
- 在系统上电前，请确保接线及设置正确，确保电源极性正确。J11极性如图1所示。



Figure 1: J11极性示意

电源端子J10极性在PCB板上有标示，扩展接口接入电源请参见35页表25。三种电源接入方法中只能选择一种给开发板供电。

3. 使用

- 系统运行时，板上的CPU和电源部分会发热，CPU运行在128MHz时，发热尤为明显，请尽量避免在系统运行时接触PCB。
- 请避免在高温、高湿、强电磁干扰、太阳光直射等的恶劣环境中使用该开发板。
- 与扩展板连接时，可以选择从扩展接口引入电源，避免使用两路电源分别给母板和扩展板供电，请不要使用扩展接口中的3.3V电源给其它电路供电。
- 系统配备的监控程序能实现基本的调试功能，当需要跟踪、源码调试等功能时请连接仿真器。

Contents

1 使用前必读-----	2
2 系统软硬件配置-----	5
2.1 硬件配置-----	6
2.2 软件配置-----	6
3 存储器地址空间分配-----	7
4 开发板设置-----	9
4.1 复位开关-----	9
4.2 拨码开关-----	10
4.3 跳线-----	11
4.4 LED-----	11
4.5 测试点-----	11
5 使用监控程序-----	12
5.1 监控程序配置-----	12
5.2 如何与主机连接-----	13
5.3 监控程序命令表-----	14
5.4 监控程序命令使用详解-----	15
5.4.1 ML(Memory Load)下载用户程序-----	15
5.4.2 FL(FLASH Load)烧写主FLASH-----	16
5.4.3 RR(Register Read)读取CPU寄存器-----	17
5.4.4 RC(Register Clear)清除CPU寄存器-----	18
5.4.5 RW(Register Write)写CPU寄存器-----	19
5.4.6 ME(Memory Edit)编辑内存指定单元-----	20
5.4.7 MF(Memory Fill)填充内存块-----	21
5.4.8 MD(Memory Dump)显示内存块的内容-----	22
5.4.9 DA(Disassemble)反汇编-----	23
5.4.10 G(Go)执行程序-----	24
5.4.11 S(Step)单步执行程序-----	25
5.4.12 BS(Breakpoint Set)设置断点-----	26
5.4.13 BC(Breakpoint Clear)清除所有断点-----	27
5.4.14 BR(Breakpoint Read)显示断点-----	28
5.4.15 BD(Breakpoint Delete)删除断点-----	29
5.4.16 BE(Breakpoint Enable)使能断点-----	30

5.4.17 BI(Breakpoint Ignore)无效断点-----	31
5.4.18 H(Help)帮助-----	32
5.5 烧写监控程序-----	33
5.6 上电执行用户程序-----	33
6 扩展接口-----	34
6.1 扩展接口位置-----	34
6.2 扩展接口引脚定义-----	35
A 附录-----	38
A.1 开发板实物照片-----	38
A.2 超级终端设置分步图解-----	39
A.3 图表目录-----	44

2 系统软硬件配置

开发板采用瑞萨公司的SH3系列单片机SH7709S ,性能优秀, 主要应用于各种嵌入式课程教学实验或培训中。开发板引出了CPU所有的信号, 也可以用于开发或小批量的生产该单片机相应产品时的样机制作。表1是系统的软硬件配置。

Table 1: 系统软硬件配置一览

条目		说明
板上MCU		HD6417709SF133B
系统时钟		MCU运行频率128MHz或64MHz(可选), 外设时钟32MHz,外接8MHz的晶振
用户RAM		32Mbyte SDRAM(HY57V281620HCT-H)
ROM	主FLASH ROM	4Mbyte(AM29LV320DB-90EI)
	备用FLASH ROM	1Mbyte(SST39VF040 70-4C-NH)
串口		1路, 最高支持250Kbps的通信速率
H-UDI		1路, 与E10A仿真器连接
RTC		内部时钟, 日历功能, 外接32.768KHz的晶振
监控程序	主机	装有Windows操作系统的PC
	主机接口	9针RS232标准接口, 支持38400bps的波特率
	监控命令	17条命令 RR(Register Read),RC(Register Clear), G(Go)等等
调试指示		8位调试LED指示特定地址空间的低8位数据
CPU状态指示		四位LED分别指示CPU的NORMAL, STANDBY SLEEP, RESET状态
复位开关		RESET和NMI
跳线		跳线选择是否连接仿真器, A/D是否使用外部电源以及备用FLASH与主FLASH地址空间交换
使用环境温度		0-40°C
PCB尺寸		127 x 97 (mm) 4层印刷电路板
电源		DC 5V \pm 5%
消耗电流		400mA

2.1 硬件配置

开发板为127 x 97 mm 的四层印制电路板，图2是开发板的组成示意图，实物可参照38页图27。其中U11和U12为SDRAM, U9为系统主FLASH, U10为系统备用FLASH。J5,J6 和J8分别是为RESET, NMI和GND信号安排的测试点。JP1可以设置主FLASH和备用FLASH地址空间互换，JP3为仿真器选择跳线，JP4是A/D电源选择跳线。

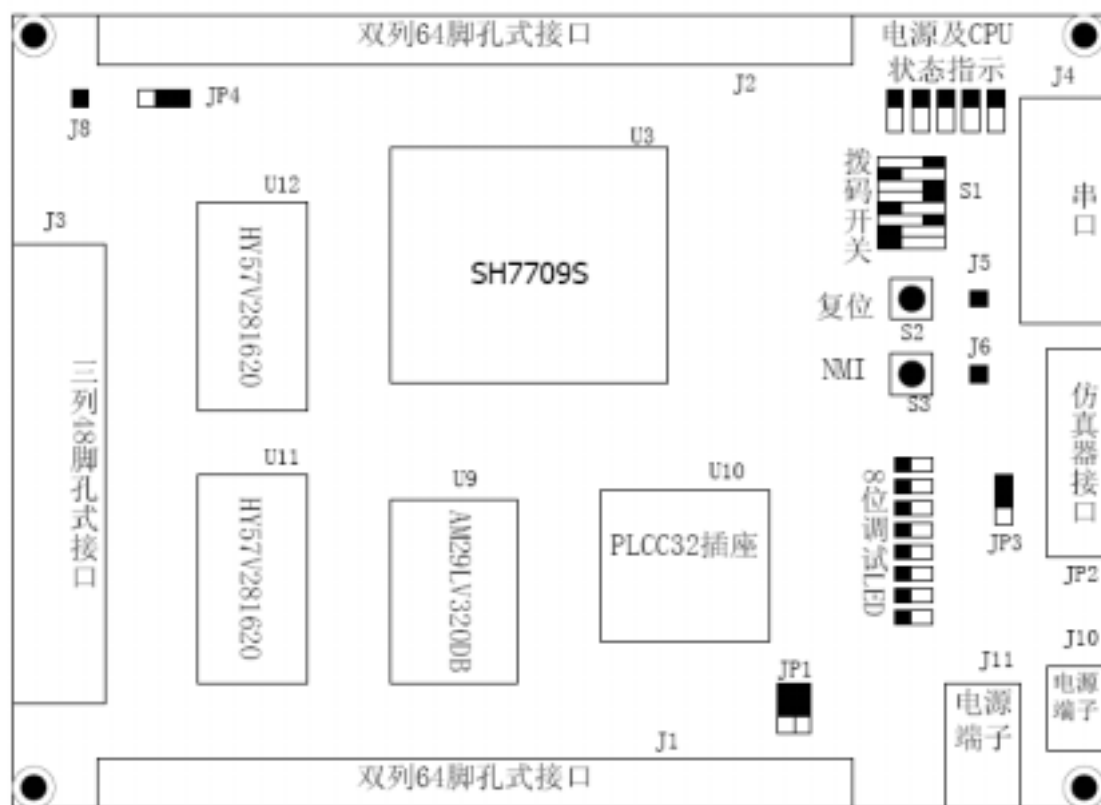


Figure 2: 开发板组成示意图

2.2 软件配置

开发板配备了功能完备的监控程序，临控程序有如下调试功能：

- 显示、修改内存或CPU寄存器。

在执行用户程序前，可以按用户要求修改CPU的通用寄存器；可以按字节、字或双字形式显示或修改内存和寄存器。

- 下载、执行或调试用户程序。

按生成MOT中指定的地址下载程序到内存，可以执行用户、单步执行或设断

点执行用户程序，当下列情况出现时，用户程序停止：

- 遇到用户设置的断点
- 用户按下RESET或NMI复位开关

- 反汇编，把内存的内容显示为SH3的指令。
- 在系统烧写FLASH，固化用户程序。

监控程序可以将用户程序下载到开发板的内存中。用户主机的操作系统需要配置有串口连接软件(比如Windows下的超级终端)以及对应的编译器和链接器。图3显示了开发板与主机连接时的软件配置。

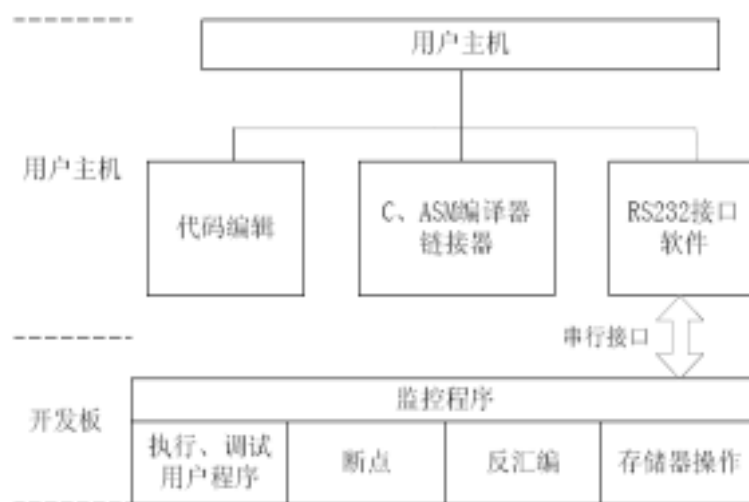


Figure 3: 开发板连接到主机时的软件配置

3 存储器地址空间分配

表3中列出了开发板存储器分配情况。主FLASH和备用FLASH空间可以互换，用于更新监控程序。因主FLASH和备用FLASH位宽不同，更换地址空间的同时要改变0区的位宽。主FLASH工作在0区时，备用FLASH可以做为额外之ROM空间，对它的操作用户需自己编程完成，此时5区的位宽只能为8位，如果取出备用FLASH，则用户可以自行定制5区的使用。

8个调试LED地占用了8MB的地址，向对应地址写入数据时，8个LED显示出低8位数据总线上的数据。在0区不同位宽时对LED的操作不同，0区位宽8位时，写入8位数据即可，当0区位宽为16位时，写入16位数据的低8位即可以显示在LED上，此时，如果是内存是big-endian对齐方式，向该段地址空间内的任意奇地址写入8位数据即可以显示在LED上，如果是little-endian对齐方式，向该段地址空间内的任意偶地址写入8位数据即可以显示在LED上。

Table 3: 开发板存储器分配表

存储区号	主要用途	位宽	地址范围	器件	备注
0区	主FLASH, 调试 LED	16位	分配8MB, H'00000000 - H'007FFFFFFF, 实际 4MB H'00000000 - H'003FFFFFFF	FLASH AM29LV320DB 16位 4MB	1、主 FLASH和 备用 FLASH的地 址空间可以交 换, 因主 FLASH与备用 FLASH 位宽不 同, 交换地址空 间同时要改变0区 的位宽。 2、LED接在数据 线的低8位。
			8位调试 LED 分配 8MB H'00800000 - H'00FFFFFFF		
			用户区 48MB H'01000000 - H'03FFFFFFF		
1区	内部 I/O区	—	—	—	—
2区	用户扩展 区	8, 16, 32位	64MB H'08000000 - H'0BFFFFFFF		参考数据手册
3区	SDRAM区	32位	64MB H'0C000000 - H'0FFFFFFF 实际 32MB H'0C000000 - H'0DFFFFFFF	HY57V281620 HCT-H 2片	外接8M晶振时 CLK = 32MHz, 外接16M晶振时 CLK = 64MHz

4区	用户扩展区	8, 16, 32位	64MB H'10000000 - H'13FFFFFF		参考数据手册
5区	备用FLASH区	8位	分配8MB, H'14000000 - H'147FFFFFF, 实际 1MB H'14000000 - H'140FFFFFF	FLASH SST39VF040 8位 1MB	1、备用FLASH和主FLASH的地址空间可以交换，因备用FLASH与主FLASH位宽不同，交换地址空间同时要改变0区的位宽。 2、如果取出备用FLASH，则5区位宽可由用户任意设定，且用户可以使用5区全部地址空间
			用户区 56MB H'14800000 - H'14FFFFFF		
6区	用户扩展区或PCMLA卡扩展区	8, 16, 32位	64MB H'18000000 - H'1BFFFFFF		参考数据手册

4 开发板设置

设置开发板时，请断开开发板电源。

4.1 复位开关

板上有两个复位开关，S2和S3，S2控制单片机的复位(reset)信号，在用户reset单片机或连接到仿真器时使用，S3控制NMI信号，可以中断由监控程序启动的用户程序。

4.2 拨码开关

板上有一个8位的拨码开关S1，如图4所示，拨码开关的功能在表4中列出。

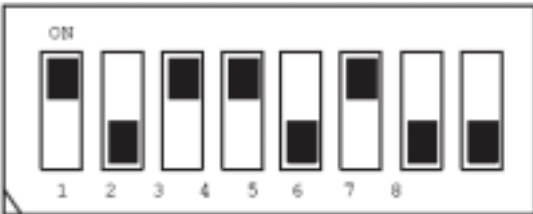


Figure 4: 拨码开关

Table 4: 拨码开关设置

	名称	功能																														
S1-1, S1-2, S1-3	时钟模式 设定	<div>S1-3, S1-2, S1-1与MD[2..0]相连，控制单片机的时钟模式。</div> <table><tr><td>S1-3</td><td>S1-2</td><td>S1-3</td><td>MD[2..0]</td><td>模式</td><td>备注</td></tr><tr><td>ON</td><td>OFF</td><td>ON</td><td>010</td><td>2</td><td>默认设置</td></tr><tr><td>ON</td><td>ON</td><td>ON</td><td>000</td><td>0</td><td>禁止</td></tr><tr><td>ON</td><td>ON</td><td>OFF</td><td>001</td><td>1</td><td>禁止</td></tr><tr><td>OFF</td><td>OFF</td><td>OFF</td><td>111</td><td>7</td><td>禁止</td></tr></table>	S1-3	S1-2	S1-3	MD[2..0]	模式	备注	ON	OFF	ON	010	2	默认设置	ON	ON	ON	000	0	禁止	ON	ON	OFF	001	1	禁止	OFF	OFF	OFF	111	7	禁止
S1-3	S1-2	S1-3	MD[2..0]	模式	备注																											
ON	OFF	ON	010	2	默认设置																											
ON	ON	ON	000	0	禁止																											
ON	ON	OFF	001	1	禁止																											
OFF	OFF	OFF	111	7	禁止																											
S1-5, S1-4	0区总线位 宽设定	<div>S1-5, S1-4对应MD4和MD3，0区的总线宽度可以设为8位或16位。</div> <table><tr><td>S1-5</td><td>S1-4</td><td>位宽</td></tr><tr><td>ON</td><td>ON</td><td>—</td></tr><tr><td>ON</td><td>OFF</td><td>8</td></tr><tr><td>OFF</td><td>ON</td><td>16</td></tr><tr><td>OFF</td><td>OFF</td><td>32</td></tr></table>	S1-5	S1-4	位宽	ON	ON	—	ON	OFF	8	OFF	ON	16	OFF	OFF	32															
S1-5	S1-4	位宽																														
ON	ON	—																														
ON	OFF	8																														
OFF	ON	16																														
OFF	OFF	32																														
S1-6	S1-5对 应MD5, 内存对齐 方式	S1-6 ON:Big-endian, S1-5 OFF: Little-endian，默认为ON																														
S1-7	主 FLASH 写保护	S1-7对应主 FLASH AM29LV320DB 的14脚#WP, S1-7 ON: 主FLASH的地址最低端两个8KB的扇区写保护，S1-6 OFF: 无写保护，默认为OFF。																														

S1-8	上电启动用户程序	S1-8对应单片机的119脚PTE1，监控程序初始化系统后，读取PTE1端口电平，如果为低，则转到地址0x10000处执行，否则继续执行监控程序。
------	----------	---

4.3 跳线

- JP1接1, 2时，如图5主FLASH的地址为H'14000000 - H'143FFFFFF，备用FLASH的地址为H'00000000 - H'000FFFFFF，此时可以通过备用FLASH对主FLASH编程，用来更新监控程序。JP1 接2, 3时，主FLASH地址为H'00000000 - H'003FFFFFF，备用FLASH的地址为H'14000000 - H'140FFFFFF，系统从主FLASH启动。



Figure 5: 跳线JP1设置示意图

- 系统正常运行时JP3接1, 2，连接仿真器时JP3接2, 3。
- A/D使用板上3.3V电源时JP4接2, 3，使用外部电源时JP4接1, 2，外部电源的引入管脚请参照手册34页6节，。

4.4 LED

8位调试LED的操作请参见7页3节。

POWER指示板上3.3V电源，NORMAL, STANDBY, SLEEP, RESET指示CPU的四种工作状态，用户不可操作，四种状态的具体含义请参见单片机的数据手册。

4.5 测试点

板上安排了几个信号的测试点：J5与RESET相连，J6与NMI相连，J8与GND相连，3.3V与1.8V可以通过测U7和U8靠板外边缘的贴片管脚获得，另外支撑板的四个铜柱都是接地的。

5 使用监控程序

5.1 监控程序配置

监控程序运行需要 64KB 的 ROM 和 64KB 的 RAM，监控 ROM 区是 h'00000000-h'0000FFFF，监控RAM区是h'0DFF0000 - h'0DFFFFFF，如图6所示，请不要操作这两部分内存。

Monitor Program Area	0x00000000	
User Flash Area	0x0000FFFF Flash Area	
Debug LED Area	0x003FFFFF	
	0x00800000	
	0x00FFFFFF	Area 0
Vacant Area		
Internal I/O Area	0x04000000	Area 1
Vacant Area	0x08000000	Area 2
User RAM Area	0x0C000000	
	SDRAM Area	
	0x0DFFFFFF	
Monitor Program Area	0x0DFF0000	Aera 3
Vacant Area	0x0E000000	
Vacant Area	0x10000000	Aera 4
Back FLASH Area	0x14000000	
Vacant Area	0x14100000	Aera 5
Vacant Area	0x18000000	Aera 6

Figure 6: 监控程序占用内存示意图

5.2 如何与主机连接

首先检查开发板的硬件设置:S1-3, S1-2, S1-1为ON, OFF, ON; S1-4, S1-3为OFF, ON ; S1-6的为ON; S1-7, S1-8为OFF; JP1接2-3, JP3接1-2。

其次用孔对针直通的串口线连接板上和主机串口，然后设置主机上的串口通信软件，使用的串口通信软件不限，推荐Windows下的超级终端，需对软件做表5中所示设置。

数据长度	8位
奇偶校验	无
停止位	1位
数据流控制	Xon/Xoff
波特率	38400bps

Table 5: 串口通信软件设置参数

```
=====
SH7709S Debugger Ver 1.0B (big endian)
-----

Developed by Yu Xiaojian, Tsinghua Univ., 2005
=====
type H[elp] for help messages...
Ready>
```

Figure 7: 监控程序启动界面

设置与连接完成后接入电源 ,如果在屏幕上显示如图7所示的界面 ,则连接成功!

5.3 监控程序命令表

表6列出了监控程序的所有命令。

Table 6: 监控程序命令表

类别	名称	解释
与主机接口命令	ML(Memory Load)	下载程序
烧写FLASH	FL(Flash Load)	烧写FLASH
寄存器操作	RR(Register Read)	读寄存器
	RC(Register Clear)	清寄存器
	RW(Register Write)	写寄存器
内存操作	ME(Memory Edit)	编辑内存
	MD(Memory Dump)	显示内存块
	MF(Memory Fill)	填充内存
	DA(Disassemble)	反汇编
调试程序	G(Go)	运行
	S(Step)	单步执行
	BS(Breakpoint Set)	设置断点
	BC(Breakpoint Clear)	清除断点
	BR(Breakpoint Read)	显示断点
	BD(Breakpoint Delete)	删除断点
	BE(Breakpoint Enable)	使能断点
	BI(Breakpoint Ignore)	忽略断点
其它	H(Help)	帮助

5.4 监控程序命令使用详解

输入监控的命令时，字母的大小写不限。

5.4.1 ML(Memory Load)下载用户程序

Table 7: ML命令详解

命令		功能	
	ML(Memory Load)		下载用户程序到RAM
选项			
	偏置量(offset)		
格式:			
ML (offset address)			
示例:			
Ready>ML			
Ready>ML 0			
<ul style="list-style-type: none">该命令可以将用户生成的扩展名为“MOT”的程序文件按MOT文件中的地址或加上偏置量后的地址下载到开发板的RAM中。在提示符后输入ML，会显示“Please Send A S-Format File, S-Format即.mot文件，此时利用通信软件的发送功能把.mot文件发送出去，以超级终端为例: 传送, 发送文本文件, 所有文件*.*。生成的代码经常会使用绝对地址，所以一般情况下不要使用offset。			

下面是实际操作的示例。

```
Ready>ml
Please Send A S-format Record

Start Address = 0C000000
End Address   = 0C000038

Transfer complete
```

Figure 8: ML命令示例

5.4.2 FL(FLASH Load)烧写主FLASH

Table 8: FL命令详解

命令		功能	
	FL(Memory Load)		固化用户程序到主 FLASH
选项			
	偏置量(offset)		
格式:			
FL (offset address)			
示例:			
Ready>FL			
Ready>FL 0			
<ul style="list-style-type: none"> 该命令可以将用户生成的.MOT程序文件按MOT文件中的地址或加上偏置量后的地址固化到开发板的主FLASH中。在提示符后输入FL，会显示>Please Send A S-Format File, S-Format即.mot文件，此时利用通信软件的发送功能把.mot文件发送出去，以超级终端为例：传送，发送文本文件，所有文件*.*。 注意： 请不要使用监控程序所占用的FLASH (H'00000000 - H'0000FFFF)。 待烧写的的数据先被存放在RAM中，起始地址 H'0D000000，在烧写FLASH时，尽量避免操作此处的RAM。 			

下面是实际操作的示例。

```
Ready>fl
Please Send A S-format Record

Start Address = 00010000
End Address   = 00010038

Transfer complete
erase flash sectors: complete
program :complete
.....
```

Figure 9: FL命令示例

5.4.3 RR(Register Read)读取CPU寄存器

Table 9: RR命令详解

命令		功能	
RR(Register Read)		读取CPU寄存器	
选项			
格式:			
RR			
示例:			
Ready>RR			
● 显示CPU所有寄存器的内容			

下面是实际操作的示例。

```
Ready>rr
---General Registers-----
  R0 =00000000   R1 =00000000   R2 =00000000   R3 =00000000
  R4 =00000000   R5 =00000000   R6 =00000000   R7 =00000000
  R8 =00000000   R9 =00000000  R10=00000000  R11=00000000
  R12=00000000  R13=00000000  R14=00000000  R15=0C100000
ROBANK=00000000 R1BANK=00000000 R2BANK=00000000 R3BANK=00000000
R4BANK=00000000 R5BANK=00000000 R6BANK=00000000 R7BANK=00000000
---Control Registers-----
  SSR=60000000   SPC=00000000   GBR=00000000   YBR=00000000
  SR =60000000           MD RB BL M Q I S T =1 1 0 0 0 0 0 0
---System Registers-----
  MACH=00000000  MACL=00000000   PR =00000000   PC =0C000000
Ready>
```

Figure 10: RR命令示例

5.4.4 RC(Register Clear)清除CPU寄存器

Table 10: RC命令详解

命令		功能	
	RC(Register Clear)		清除CPU寄存器
选项			
格式:	RC		
示例:	<div>Ready>RC</div> <ul style="list-style-type: none">把除R15外的CPU通用寄存器内容清零, R15可由"RW"命令设定。只有当运行用户程序时, 寄存器的设置才真正起作用。		

下面是实际操作的示例。

```
Ready>rc
---General Registers-----
  R0 =00000000    R1 =00000000    R2 =00000000    R3 =00000000
  R4 =00000000    R5 =00000000    R6 =00000000    R7 =00000000
  R8 =00000000    R9 =00000000    R10=00000000   R11=00000000
  R12=00000000   R13=00000000   R14=00000000   R15=0C100000
ROBANK=00000000 R1BANK=00000000 R2BANK=00000000 R3BANK=00000000
R4BANK=00000000 R5BANK=00000000 R6BANK=00000000 R7BANK=00000000
---Control Registers-----
  SSR=60000000    SPC=00000000    GBR=00000000    YBR=00000000
  SR =60000000          MD RB BL M Q I S T =1 1 0 0 0 0 0 0
---System Registers-----
  MACH=00000000   MACL=00000000    PR =00000000    PC =0C000000
Ready>
```

Figure 11: RC命令示例

5.4.5 RW(Register Write)写CPU寄存器

Table 11: RW命令详解

命令		功能	
RW(Register Write)		写CPU寄存器	
选项			
格式:			
RW <寄存器名称> <16进制数>			
示例:			
Ready>RW R15 0D000000			
<ul style="list-style-type: none">• 写入指定CPU寄存器指定的数。• 只有当运行用户程序时，寄存器的设置才真正起作用。			

下面是实际操作的示例。

```
Ready>rw r15 0D000000
---General Registers---
  R0 =00000000   R1 =00000000   R2 =00000000   R3 =00000000
  R4 =00000000   R5 =00000000   R6 =00000000   R7 =00000000
  R8 =00000000   R9 =00000000  R10=00000000  R11=00000000
  R12=00000000  R13=00000000  R14=00000000  R15=0D000000
ROBANK=00000000 R1BANK=00000000 R2BANK=00000000 R3BANK=00000000
R4BANK=00000000 R5BANK=00000000 R6BANK=00000000 R7BANK=00000000
---Control Registers---
  SSR=60000000   SPC=00000000   GBR=00000000   YBR=00000000
  SR =60000000
MD RB BL M Q I S T =1 1 0 0 0 0 0 0
---System Registers---
  MACH=00000000  MACL=00000000   PR =00000000   PC =0C000000
Ready>_
```

Figure 12: RW命令示例

5.4.6 ME(Memory Edit)编辑内存指定单元

Table 12: ME命令详解

命令		功能	
	ME(Memory Edit)		编辑内存指定单元
选项			
	-B(默认) -W -L		
格式:			
ME <内存地址> <选项>			
示例:			
Ready>ME 00800001			
Ready>ME 00800000 -W			
Ready>ME 00800001 -L			
<ul style="list-style-type: none">按字节、字或长字的方式编辑内存内容。输入 '/' 可以返回到前一内容。输入 '.' 结束。			

下面是实际操作的示例。

```
Ready>me 0c001000
0C001000 62 ?55
0C001001 F6 ?/
0C001000 55 ?00
0C001001 F6 ?.
Ready>me 0c001000 -w
0C001000 00F6 ?5555
0C001002 0009 ?.
Ready>me 0c001000 -l
0C001000 55550009 ?5555aaaa
0C001004 0DFF084C ?.
Ready>
```

Figure 13: ME命令示例

5.4.7 MF(Memory Fill)填充内存块

Table 13: MF命令详解

命令		功能	
	MF(Memory Fill)		填充内存块
选项			
格式:			
MF <起始地址> <结束地址> - <16进制数>			
示例:			
Ready>MF 0C000000 -FF			
Ready>MD 0C000000 0C000200 -FF			
<ul style="list-style-type: none">• 如果没有结束地址，默认填充256个字节。• 在16进制数前要加'-'。			

下面是实际操作的示例。

```
Ready>mf 0c000000 -55
Fill with 0x55
Start Address = 0C000000
End Address   = 0C0000FF
Successful!!
Ready>_
```

Figure 14: MF命令示例

5.4.8 MD(Memory Dump)显示内存块的内容

Table 14: MD命令详解

命令		功能	
MD(Memory Dump)		显示内存块的内容	
选项			
格式:			
MD <起始地址> <结束地址>			
示例:			
Ready>MD 0C000000			
Ready>MD 0C000000 0C000200			
<ul style="list-style-type: none">按每行16个字节显示内存内容, 如果没有结束地址, 默认显示256个字节。有对应的ASCII码。输入一次后, 再输入回车即可重复。			

下面是实际操作的示例。

```
Ready>md 0c000000
0C000000 55 55 55 55 55 55 55 55 55 55 55 55 55 55 55 55  UUUUUUUUUUUUUUUUU
0C000010 55 55 55 55 55 55 55 55 55 55 55 55 55 55 55 55  UUUUUUUUUUUUUUUUU
0C000020 55 55 55 55 55 55 55 55 55 55 55 55 55 55 55 55  UUUUUUUUUUUUUUUUU
0C000030 55 55 55 55 55 55 55 55 55 55 55 55 55 55 55 55  UUUUUUUUUUUUUUUUU
0C000040 55 55 55 55 55 55 55 55 55 55 55 55 55 55 55 55  UUUUUUUUUUUUUUUUU
0C000050 55 55 55 55 55 55 55 55 55 55 55 55 55 55 55 55  UUUUUUUUUUUUUUUUU
0C000060 55 55 55 55 55 55 55 55 55 55 55 55 55 55 55 55  UUUUUUUUUUUUUUUUU
0C000070 55 55 55 55 55 55 55 55 55 55 55 55 55 55 55 55  UUUUUUUUUUUUUUUUU
0C000080 55 55 55 55 55 55 55 55 55 55 55 55 55 55 55 55  UUUUUUUUUUUUUUUUU
0C000090 55 55 55 55 55 55 55 55 55 55 55 55 55 55 55 55  UUUUUUUUUUUUUUUUU
0C0000A0 55 55 55 55 55 55 55 55 55 55 55 55 55 55 55 55  UUUUUUUUUUUUUUUUU
0C0000B0 55 55 55 55 55 55 55 55 55 55 55 55 55 55 55 55  UUUUUUUUUUUUUUUUU
0C0000C0 55 55 55 55 55 55 55 55 55 55 55 55 55 55 55 55  UUUUUUUUUUUUUUUUU
0C0000D0 55 55 55 55 55 55 55 55 55 55 55 55 55 55 55 55  UUUUUUUUUUUUUUUUU
0C0000E0 55 55 55 55 55 55 55 55 55 55 55 55 55 55 55 55  UUUUUUUUUUUUUUUUU
0C0000F0 55 55 55 55 55 55 55 55 55 55 55 55 55 55 55 55  UUUUUUUUUUUUUUUUU
Ready>_
```

Figure 15: MD命令示例

5.4.9 DA(Disassemble)反汇编

Table 15: DA命令详解

命令		功能	
	DA(Disassemble)		反汇编
选项			
格式:	DA <起始地址>		
示例:	<div>Ready>DA 0C000000</div> <ul style="list-style-type: none">• 以指令的形式显示内存内容• 输入一次后，再输入回车即可重复。		

下面是实际操作的示例。

```
Ready>da 10000
00010000 0009      NOP
00010002 0009      NOP
00010004 0009      NOP
00010006 0009      NOP
00010008 D106      MOY.L @(6,PC),R1
0001000A 9209      MOY.W @(9,PC),R2
0001000C 2121      MOY.W R2,@R1
0001000E B00B      BSR B
00010010 0009      NOP
00010012 D104      MOY.L @(4,PC),R1
00010014 9205      MOY.W @(5,PC),R2
00010016 2121      MOY.W R2,@R1
00010018 B006      BSR 6
0001001A 0009      NOP
0001001C AFF4      BRA FF4
0001001E 0009      NOP
Ready>
```

Figure 16: DA命令示例

5.4.10 G(Go)执行程序

Table 16: G命令详解

命令		功能	
	G(Go)		执行用户程序
选项			
格式:	G <起始地址>		
示例:	<div>Ready>G 0C000000</div> <ul style="list-style-type: none">• 开始招待用户程序。• 执行用户程序时，可以按NMI复位键中断程序执行。		

下面是实际操作的示例。

Ready>g 10000

Figure 17: G命令示例

5.4.11 S(Step)单步执行程序

Table 17: S命令详解

命令		功能	
	S(Step)		单步执行程序
选项			
格式:	S <起始地址>		
示例:	Ready>S 0C000000		
	<ul style="list-style-type: none">• 单步执行程序。• 输入一次后，再输入回车即可重复。		

下面是实际操作的示例。

```
Ready>s 10000
00010000 0009      NOP
Ready>
00010002 0009      NOP
Ready>_
```

Figure 18: S命令示例

5.4.12 BS(Breakpoint Set)设置断点

Table 18: BS命令详解

命令		功能	
	BS(Breakpoint Set)		设置断点
选项			
格式:	BS <断点地址>		
示例:	Ready>BS 0C000100		
	● 在指定的地址设置断点		

下面是实际操作的示例。

```
Ready>bs 100A
=====
< Enabled Break Point >
=====
NO.    Address
[01] : 0000100A
=====
Ready>_
```

Figure 19: BS命令示例

5.4.13 BC(Breakpoint Clear)清除所有断点

Table 19: BC命令详解

命令		功能	
	BS(Breakpoint Clear)		清除所有断点
选项			
格式:	BC		
示例:	Ready>BC <ul style="list-style-type: none">清除所有断点		

下面是实际操作的示例。

```
Ready>bc
=====
< Enabled Break Point >
=====
  NO.   Address
=====
Ready>
```

Figure 20: BC命令示例

5.4.14 BR(Breakpoint Read)显示断点

Table 20: BR命令详解

命令		功能	
	BR(Breakpoint Read)		显示断点
选项			
格式:	BR		
示例:	<p>Ready>BR</p> <ul style="list-style-type: none"> • 显示所有断点。 • 断点按地址的升序排序，序号用于BD命令中。 		

下面是实际操作的示例。

```
Ready>br
=====
< Enabled Break Point >
=====
  NO.   Address
[01] : 00001006
[02] : 0000100A
=====
Ready>_
```

Figure 21: BR命令示例

5.4.15 BD(Breakpoint Delete)删除断点

Table 21: BD命令详解

命令		功能	
BD(Breakpoint Delete)		删除断点	
选项			
格式:			
BD <断点对应序号>			
示例:			
Ready>BD 2			
<ul style="list-style-type: none">● 删除第2个断点。● 断点的序号由BR命令中得出。			

下面是实际操作的示例。

```
Ready>bd 2
=====
< Enabled Break Point >
=====
  NO.   Address
[01] : 00001006
=====
Ready>
```

Figure 22: BD命令示例

5.4.16 BE(Breakpoint Enable)使能断点

Table 22: BE命令详解

命令		功能	
	BE(Breakpoint Enable)		使能断点
选项			
格式:	BE		
示例:	Ready>BE		
	● 使能所有已设断点。		

下面是实际操作的示例。

```
Ready>be
=====
< Enabled Break Point >
=====
  NO.   Address
[01] : 00001006
=====
```

Figure 23: BE命令示例

5.4.17 BI(Breakpoint Ignore)无效断点

Table 23: BI命令详解

命令		功能	
	BI(Breakpoint Ignore)		无效断点
选项			
格式:	BI		
示例:	Ready>BI		
	● 使所有已设断点无效。		

下面是实际操作的示例。

```
Ready>bi
=====
< Ignored Break Point >
=====
  NO.   Address
[01] : 00001006
=====
Ready>_
```

Figure 24: BI命令示例

5.4.18 H(Help)帮助

Table 24: H命令详解

命令		功能	
	H(Help)		帮助
选项			
格式:	H <序号>		
示例:	<div>Ready>H</div> <div>Ready>H 2</div> <div>Ready>H M</div> <ul style="list-style-type: none">• 显示所有命令或命令格式。		

下面是实际操作的示例图。

```
Ready>h
-----
Debugger Help : address or data must be specified in hexadecimal format
(need not H')
-----
[1] General          --- H
[2] Register         --- RC   RR   RW
[3] Break Point      --- BS   BR   BD   BC   BI   BE
[4] Memory           --- ML   ME   MD   MF   FL
[5] Disassemble      --- DA
[6] Start User Program --- G   S
.....H[elp] number(or class), for more information.
-----
Ready>h 4
-----
Debugger Help : [4] Memory
-----
Memory Load      : M[em]L[oad]
Memory Edit      : M[em]E[dit] startAddr [size(-W,-L)]
Memory Dump      : M[em]D[ump] [startAddr] [endAddr]
Memory Fill      : M[em]F[ill] [startAddr] [endAddr] [-Data]
Flash Load       : F[lash]L[oad] [offsetAddr]
-----
Ready>_
-----
```

Figure 25: H命令示例

5.5 烧写监控程序

当主FLASH中的监控程序损坏时，可由下述方法更新：

1. JP1置1-2. S1-1, S1-2, S1-3为ON, OFF, ON. S1-4, S1-5, S1-6, S1-7, S1-8分别为OFF, ON, ON, OFF, OFF, JP3置1-2。
2. 把附带的备用FLASH放入PLCC32插座中。
3. 连接主机和开发板的串口，主机的串口软件设置同启动监控程序一样，设置完成后，接入电源。
4. 如果设置正确，在通信软件上会显示出提示符，此时输入FL。
5. 程序先会把主FLASH中的所有数据在RAM中做一备份，并提示："Flash ROM data copy to RAM".
6. 备份结束后，提示："Please Send A S-format File". 此时用通信软件的发送功能发送光盘中附带的监控程序.MOT文件(B8M102.mot)。
7. 系统完成对FLASH的擦除和编程工作后会重新出现提示符，此时编程结束。
8. 关闭电源，设置JP1为2-3, S1-4, S1-5为ON, OFF。
9. 重新接入电源，如果出现监控程序界面，即完成了对主FLASH中临控程序的更新。

注意：更新完毕后，除监控程序外，FLASH中其它的内容不会被改变。

用这种方法也可以把用户程序放入由监控程序占用的64KB 主FLASH空间。

5.6 上电执行用户程序

设置S1-8为ON，系统上电后，监控程序完成对系统的基本初始化，就转到地址H'00010000开始执行，把用户程序的入口地址放入该地址，则系统上电后就可以执行用户程序，此时监控程序完成了如下的初始化工作。

1. SDRAM控制器，用户可以直接使用SDRAM。
 2. 系统外设的时钟为32MHz。
- 其它的初始化工作，需由用户程序完成。

6 扩展接口

板上有三个扩展接口，引出了CPU的所有信号。

6.1 扩展接口位置

图中三个扩展连接器相邻管脚的水平或垂直距离都为2.54mm/100mil，由34页图26可以得出扩展接口的相对位置，制作扩展板时三个扩展接口的位置须仔细按照图中所示的尺寸来设定。

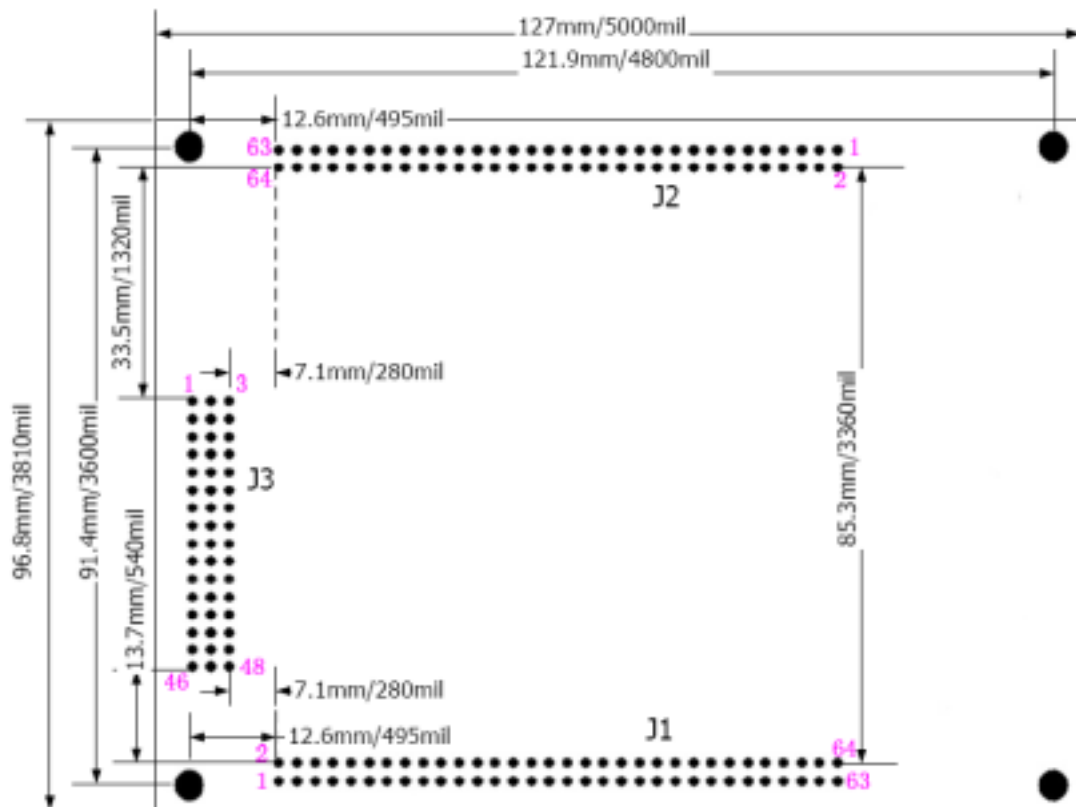


Figure 26: 扩展接口的相对位置尺寸图

6.2 扩展接口引脚定义

下面分别列出了每个扩展接口的管脚与CPU管脚的对应关系。

Table 25: J1管脚定义

序号	CPU管脚	信号名称	序号	CPU管脚	信号名称
1		VCC ¹	2	60	A5
3	61	A6	4	62	A7
5	63	A8	6	64	A9
7	65	A10	8	66	A11
9	67	A12	10	68	A13
11		GND	12	70	A14
13	72	A15	14	73	A16
15	74	A17	16	75	A18
17	76	A19	18	77	A20
19	78	A21	20	80	A22
21	82	A23	22	84	A24
23	86	A25	24		GND
25	87	BS	26	88	RD#
27	89	WE0#	28	90	WE1#
29	91	WE2#	30	92	WE3#
31	93	WR#	32	94	PTE7
33	96	CS0#	34	98	CS2#
35	99	CS3#	36	100	CS4#
37	101	CS5#	38	102	CS6#
39		GND	40	103	PTE4
41	104	PTE5	42	105	CKE
43	106	RAS3L#	44	107	PTJ1
45	108	CASL#	46	110	CASU#
47	112	PTJ4	48	113	PTJ5
49	114	DACK0	50	115	DACK1
51	116	PTE6	52	117	PTE3
53	118	RAS3U#	54	119	PTE1
55	120	TDO	56	121	BACK#
57		GND	58	122	BREQ#
59	123	WAIT#	60	124	RESETM#
61	125	ADTRG#	62	126	PTG7
63		GND	64		EXPOWER ²

¹VCC=3.3V

²可由此处引入5V电源

Table 26: J2管脚定义

序号	CPU管脚	信号名称	序号	CPU管脚	信号名称
1	127	PTG6	2	128	PTG5
3	129	PTG4	4	130	PTG3
5	131	PTG2	6	133	PTG1
7	135	PTG0	8	136	TRST#
9	137	TMS	10	138	TDI
11	139	TCK	12		GND
13	140	PINT11	14	141	PINT10
15	142	PINT9	16	143	PINT8
17	151	AUDCK	18		VCC ³
19	157	STATUS0	20	158	STATUS1
21	159	TCLK	22	160	IQOUT#
23	162	CKIO	24	164	TXD0
25	165	SCK0	26	166	TXD1
27	167	SCK1	28		GND
29	168	TXD2	30	169	SCK2
31	170	RTS2#	32	171	RXD0
33		GND	34	172	RXD1
35	174	RXD2	36	176	CTS2#
37	177	PTC7	38	178	PTC6
39	179	PTC5	40	180	PTC4
41	182	WAKEUP#	42	184	RESETOUT#
43	185	PTC3	44	186	PTC2
45	187	PTC1	46	188	PTC0
47		GND	48	189	DRAK0
49	190	DRAK1	50	191	DREQ0#
51	192	DREQ1#	52	193	RESETP#
53	194	CA	54	198	ACSSIN ⁴
55		GND	56	199	AN0
57	200	AN1	58	201	AN2
59	202	AN3	60	203	AN4
61	204	AN5	62	206	AN6
63	207	AN7	64	205 ⁵	AVCCIN

³VCC=3.3V⁴该信号通过一串接的磁珠与系统的GND相连⁵跳线JP4选择是否使用板上的VCC或者AVCCIN

Table 27: J3管脚定义

序号	CPU 管脚	信号	序号	CPU 管脚	信号	序号	CPU 管脚	信号
1	7	NMI	2	8	IRQ0	3	9	IRQ1
4	10	IRQ2	5	11	IRQ3	6	12	IRQ4
7		VCC ⁶	8	13	D31	9	14	D30
10	15	D29	11	16	D28	12	17	D27
13	18	D26	14	20	D25	15		GND
16	22	D24	17	23	D23	18	24	D22
19	25	D21	20	26	D20	21	28	D19
22	30	D18	23	31	D17	24		GND
25	32	D16	26	34	D15	27	36	D14
28	37	D13	29	38	D12	30	39	D11
31		GND	32	40	D10	33	41	D9
34	42	D8	35	43	D7	36	44	D6
37	46	D5	38	48	D4	39	49	D3
40	50	D2	41	51	D1	42	52	D0
43		GND	44	53	A0	45	54	A1
46	55	A2	47	56	A3	48	58	A4

⁶VCC=3.3V

A 附录

A.1 开发板实物照片

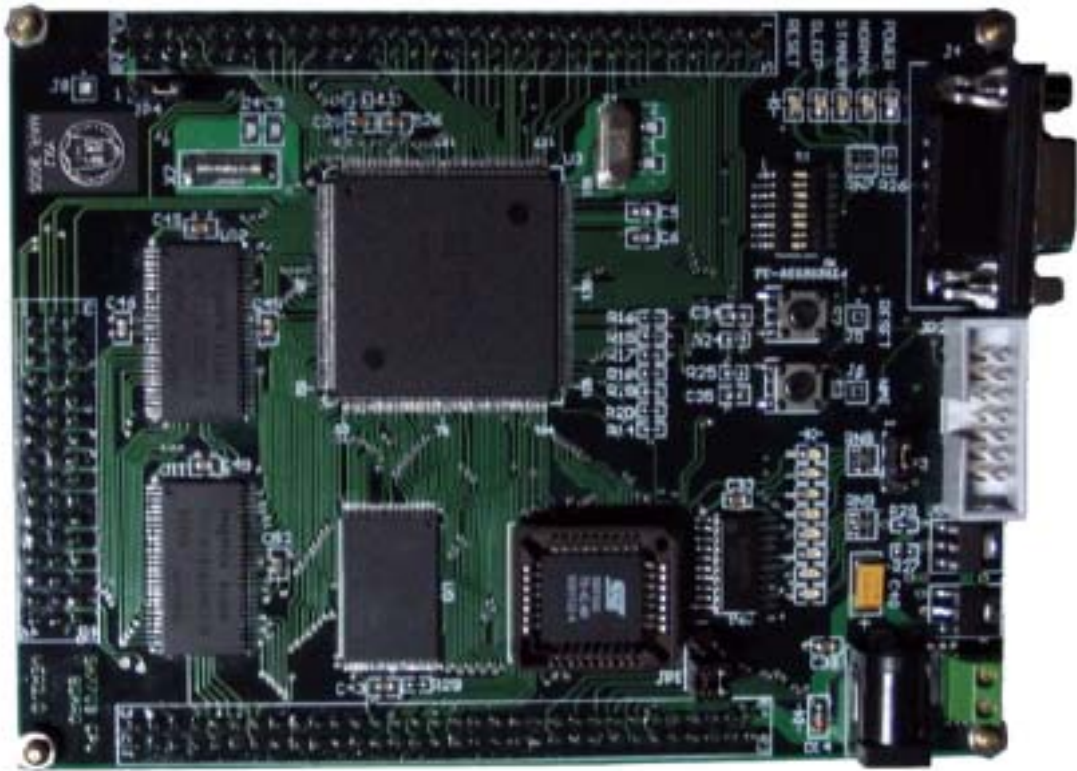


Figure 27: 开发板实物照片

A.2 超级终端设置分步图解

示例中主机所使用的操作系统为Windows XP。

1. 启动超级终端：程序->附件->通讯->超级终端。
2. 新建连接。

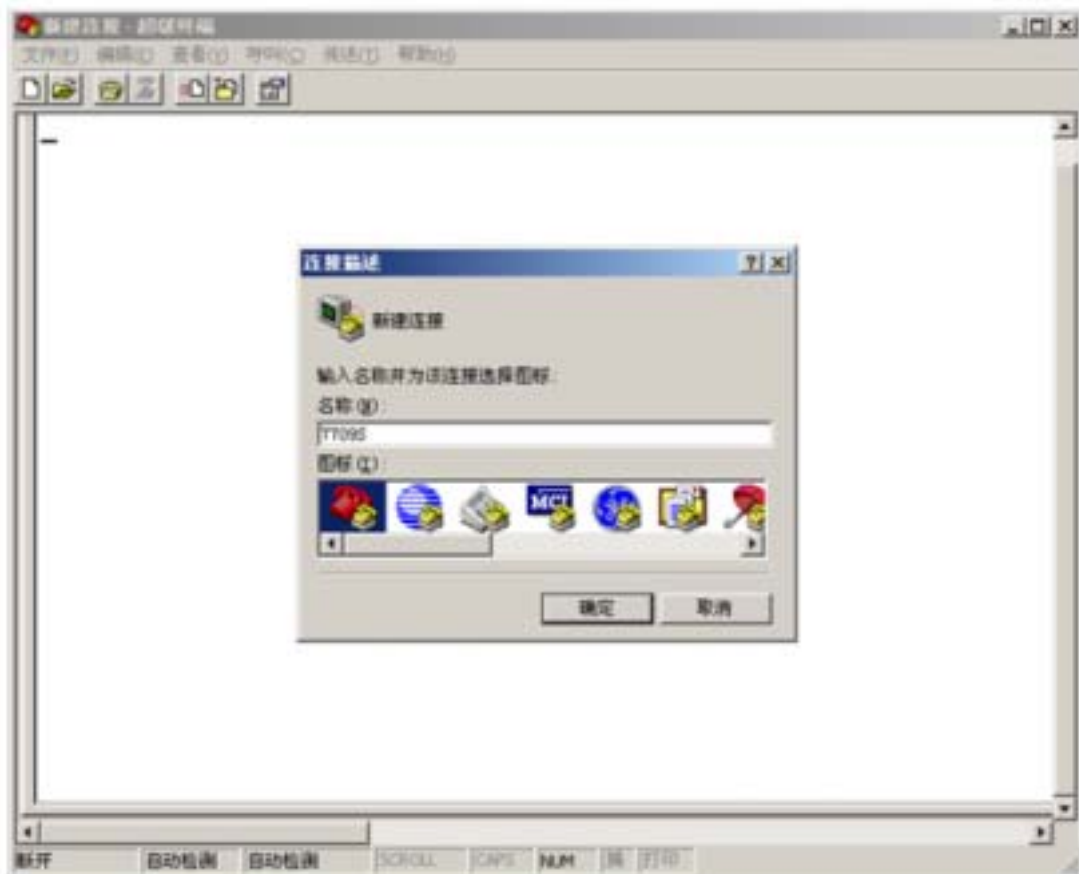


Figure 28: 超级终端设置分步图解2

3. 选择连接到开发板的串口。



Figure 29: 超级终端设置分步图解3

4. 设置主机串口参数。

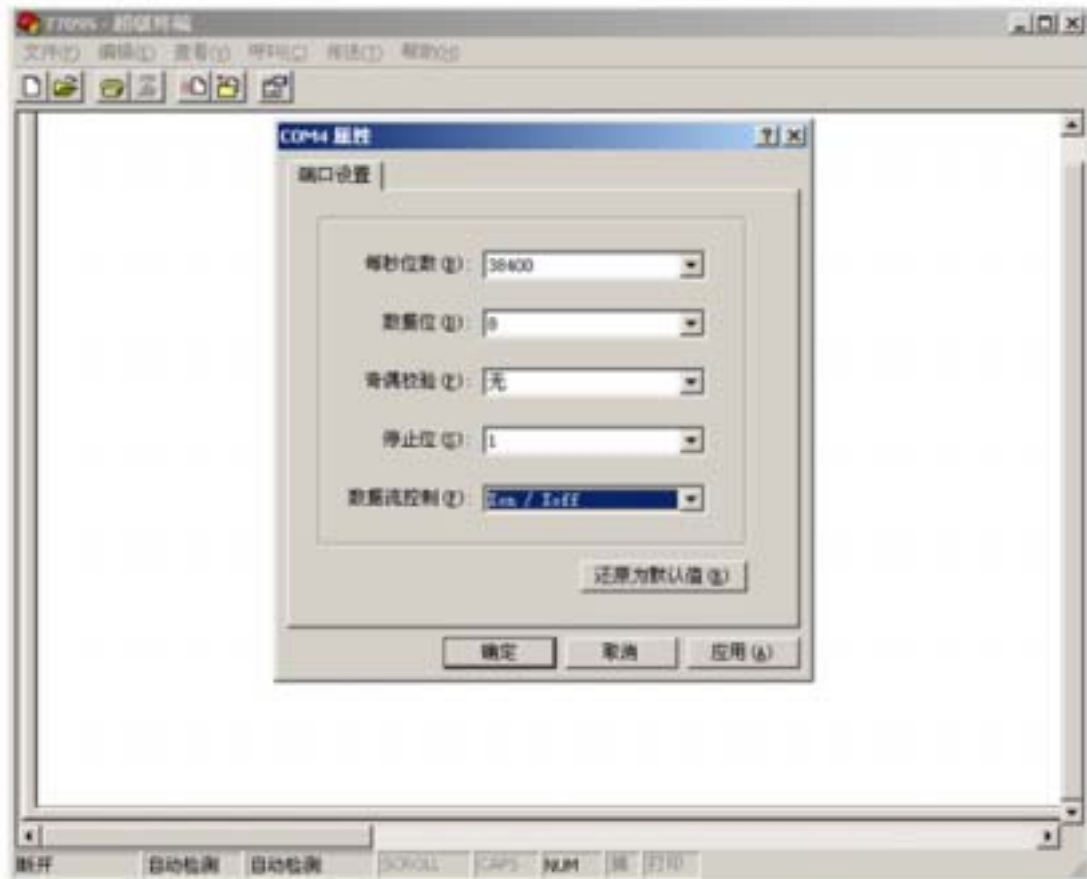


Figure 30: 超级终端设置分步图解4

5. 给开发板上电，即可看到监控程序界面。

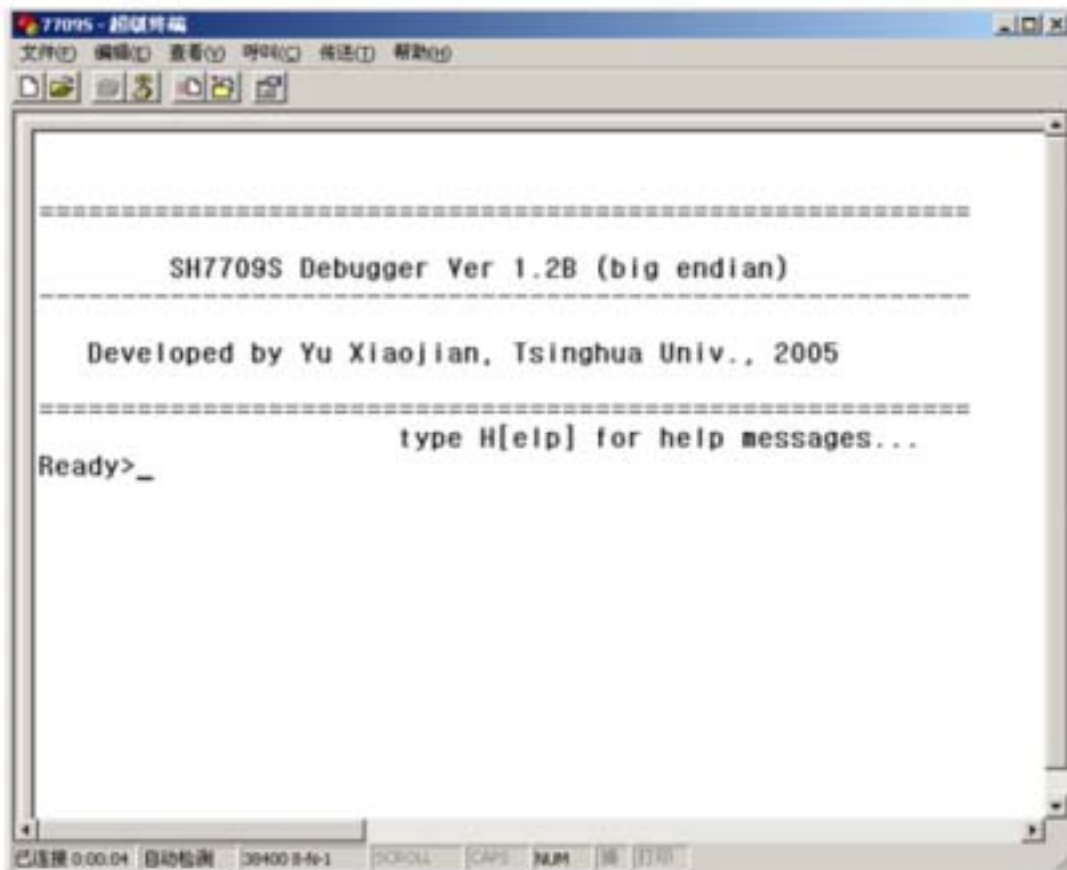


Figure 31: 超级终端设置分步图解5

6. 传送程序文件。

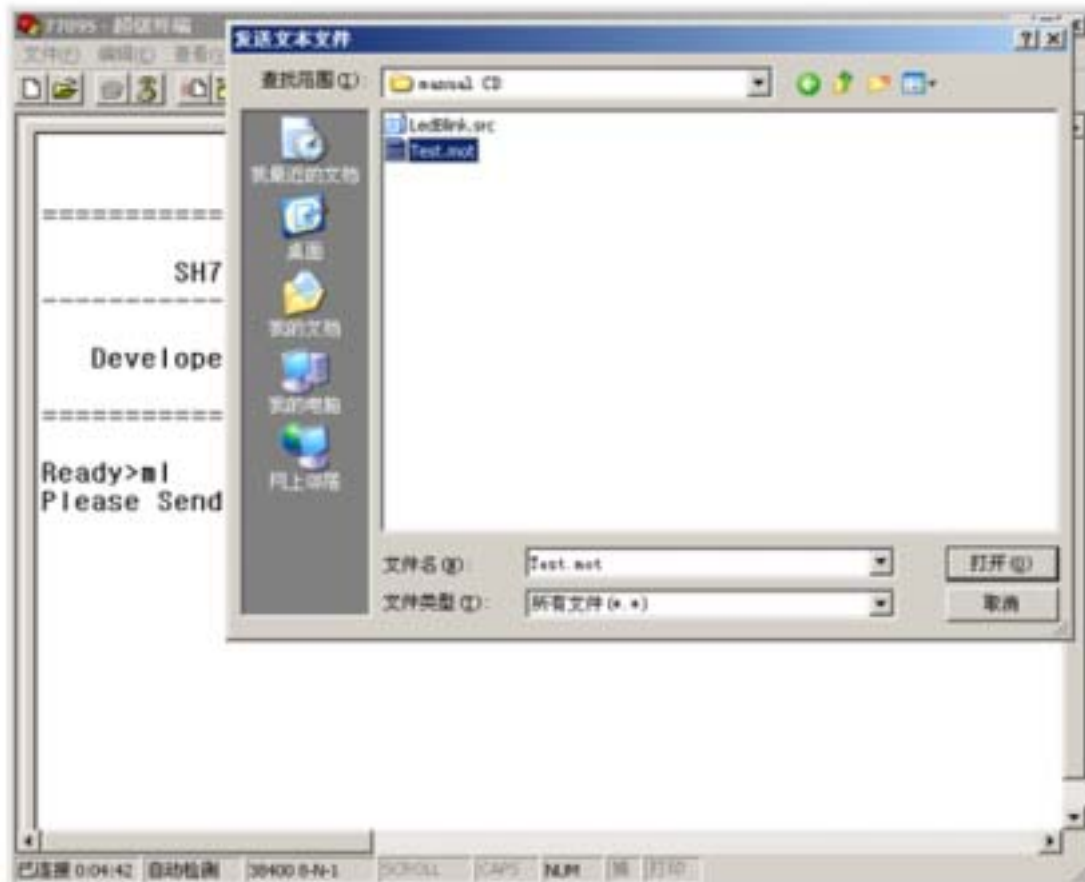


Figure 32: 超级终端设置分步图解6

A.3 图表目录

List of Figures

1 J11极性示意	2
2 开发板组成示意图	6
3 开发板连接到主机时的软件配置	7
4 拨码开关	10
5 跳线JPI设置示意图	11
6 监控程序占用内存示意图	12
7 监控程序启动界面	13
8 ML命令示例	15
9 FL命令示例	16
10 RR命令示例	17
11 RC命令示例	18
12 RW命令示例	19
13 ME命令示例	20
14 MF命令示例	21
15 MD命令示例	22
16 DA命令示例	23
17 G命令示例	24
18 S命令示例	25
19 BS命令示例	26
20 BC命令示例	27
21 BR命令示例	28
22 BD命令示例	29
23 BE命令示例	30
24 BI命令示例	31
25 H命令示例	32
26 扩展接口的相对位置尺寸图	34
27 开发板实物照片	38
28 超级终端设置分步图解2	39
29 超级终端设置分步图解3	40
30 超级终端设置分步图解4	41
31 超级终端设置分步图解5	42
32 超级终端设置分步图解6	43

List of Tables

1 系统软硬件配置一览	5
3 开发板存储器分配表	8
4 拨码开关设置	10
5 串口通信软件设置参数	13
6 监控程序命令表	14
7 ML命令详解	15
8 FL命令详解	16
9 RR命令详解	17
10 RC命令详解	18
11 RW命令详解	19
12 ME命令详解	20
13 MF命令详解	21
14 MD命令详解	22
15 DA命令详解	23
16 G命令详解	24
17 S命令详解	25
18 BS命令详解	26
19 BC命令详解	27
20 BR命令详解	28
21 BD命令详解	29
22 BE命令详解	30
23 BI命令详解	31
24 H命令详解	32
25 J1管脚定义	35
26 J2管脚定义	36
27 J3管脚定义	37