高性能嵌入式以太网模块

FMCore　FM020 开发套件

用户手册

RM0001-11

2008-1-16

版本信息

时间	版本	内容	备注
2007-07	RM0001-10	文档建立	
2008－01	RM0001-11	增加实验 5.8 修正图 4.1-1 中一个标注错别字	

目 录

1 模块总体

FM020 高性能核心模块是嵌入式控制系统的核心，其集成的高精度 12 位 AD、10M 以太网接口为数据采集控制系统提供了全新的低成本、高性能解决方案。

本手册描述了 FM020 系列核心模块，系列中不同特性将进行特殊的描述。

FM020 集成了运行在 22.1M 的高性能 C8051F02X 系列单片机，高达 256K＋32K 的 RAM,16M 串行 Flash,实时时钟和集成 10M 以太网接口。2 组 40 引脚的插针将 AD,DA, 数字 IO 口等资源引出。

FM020 采用 5V 供电，引脚兼容 CMOS、TTL 的电平。

提供配套验证板进行模块快速调试、学习。

1.1 模块特点

- 小尺寸： 60mm×42mm×22mm(长×宽×高)
- 处理器： C8051F020
- 主频： 22.1184MHZ
- 程序 FLASH：64K
- 以太网： 集成 10M 以太网接口
- 标准 RAM： 可选 4K，32K
- 扩展 RAM： 可选 128K,256K
- 数据 FLASH：16M 串行 FLASH
- 串行接口： 2 UART/I2C/SPI/ETHERNET
- 实时时钟： PCF8563
- 数字 IO： 16 个双向可按位访问 IO；8 路与模拟共用可选 IO 口；
- 模拟部分：
  - 8 路 12 位 AD
  - 8 路 10 位 AD（可设置为数字 IO 口）
  - 2 路 12 位 DA
  - 2 路比较器
- 外部复位电路
- 内部看门狗

- 内置温度传感器

1.2 模块特色

- 资源丰富的数字模块，混合模块，高可靠 4 层 PCB 设计。
- 集成 10M 以太网接口，提供免费 TCP/IP 协议库，简捷快速的网络协议开发。
- 创新灵活 RAM 扩展模式，兼容使用方便性与容量扩展性。
- 高性价比模块，快速开发系统。

1.3 开发调试工具

模块提供完整的开发工具,包括 SILAB IDE 和 4K 代码限制的 KEIL C51 编译器,JTAG 接口的在线实时调试器。51 兼容的指令和完整的开发调试环境是您可以快速进行基于微处理器的的嵌入式系统开发。

以太网协议生成向导可以生成包括 IP,TCP,UDP,HTTP,FTP 等网络主流协议，是您轻松实现基于以太网的嵌入式系统。

提供配套验证板，进行板上资源和主要功能的演示程序，帮助你快速完成程序开发和验证。

1.4 开发套件内容

- 核心模块
- 配套底板
- 9V 直流电源
- 一条交叉网线，一条直连网线。
- 演示程序
- 相关开发软件
- JTAG 在线调试器（可选）

1.5 核心模块定购信息

产品信号命名：FM020-SAA-TBB-FCC-ET

其中，AA：00－不含 32K RAM 62LV256

01－含 32K RAM 62LV256

BB：00－不含分页大容量 RAM

01－TRAM 为 128K

02－TRAM 为 256K

CC：00－不含数据 FLASH

01－FLASH 容量为 4Mbit

02－FLASH 容量为 8Mbit

04－FLASH 容量为 16Mbit

例：

产品型号 FM020-S01-T02-F03-ET,表明定购的模块含有 32K 的 SRAM 和 256K 的分页 TRAM 以及 8Mbit 的数据 FLASH.

2 使用入门

本章详细描述 FM020 核心模块并说明开发套件的使用方法，逐步讲解核心模块和验证板的使用步骤。如果您单独购买了核心模块，请参考相关内容或后面章节对模块的详细描述。

2.1 连接

要使用 SILAB IDE 调试核心模块需要经过 4 个步骤，分别描述如下：

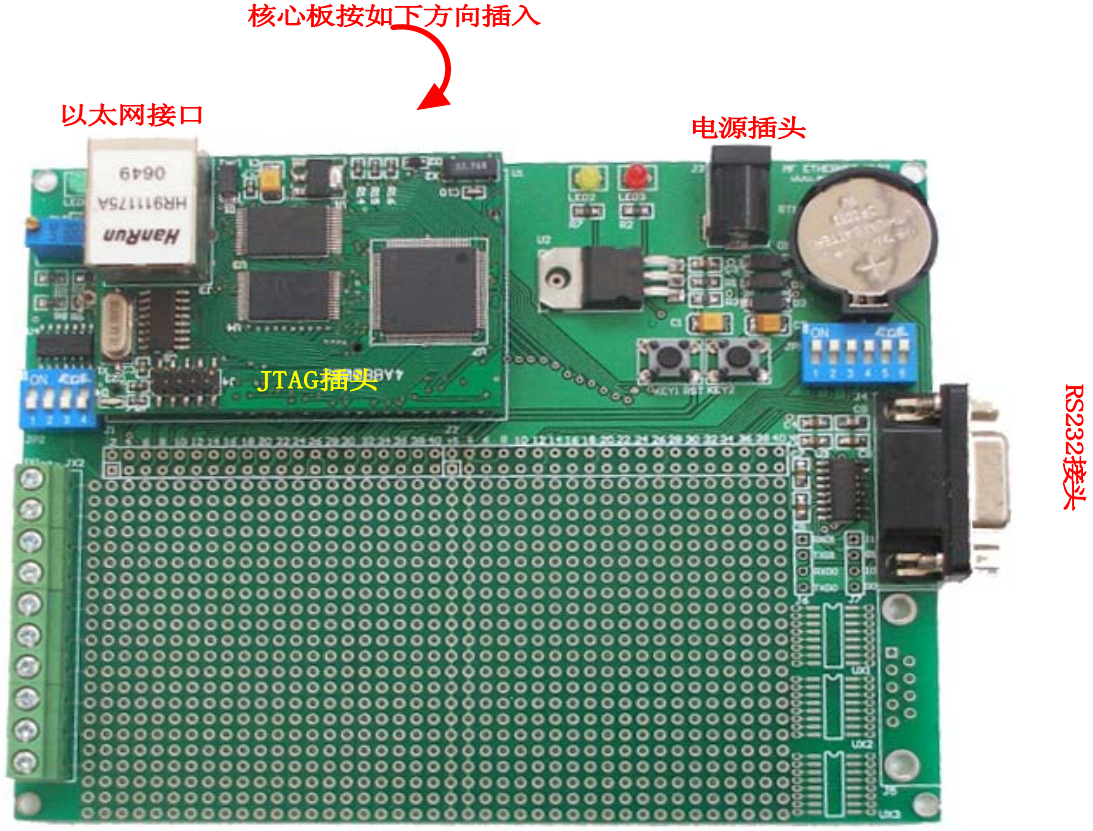
1．将核心模块 FM020 插到验证板上。

2．将调试器的 JTAG 接口插到核心模块的 J4 上。

3．用 9V 直流电源给底板供电（J3）。

4．启动 C8051FXXXX IDE，选择相应的程序，下载并进行调试。

连接后的电路板如下图：



图表 1　主板与核心板连接图

注意：**JTAG 的插入方向要注意，标有 J4 一侧缺一个引脚，该引脚应该与调试器上被堵上的孔相对应。**

2.2 运行演示程序

核心板演示程序可以在光盘的根目录下的/coreModule/App/下找到，其中包括了串口、外部中断、实时时钟、外部 XRAM、串行 FLASH、以太网 HTTP 等程序。演示程序中的工程除 java_keil 程序是 Keil uVision3 工程，其它均在 C8051FXXX IDE 环境下生成，可以使用光盘中的该软件进行编译调试。

2.2.1 安装集成开发环境

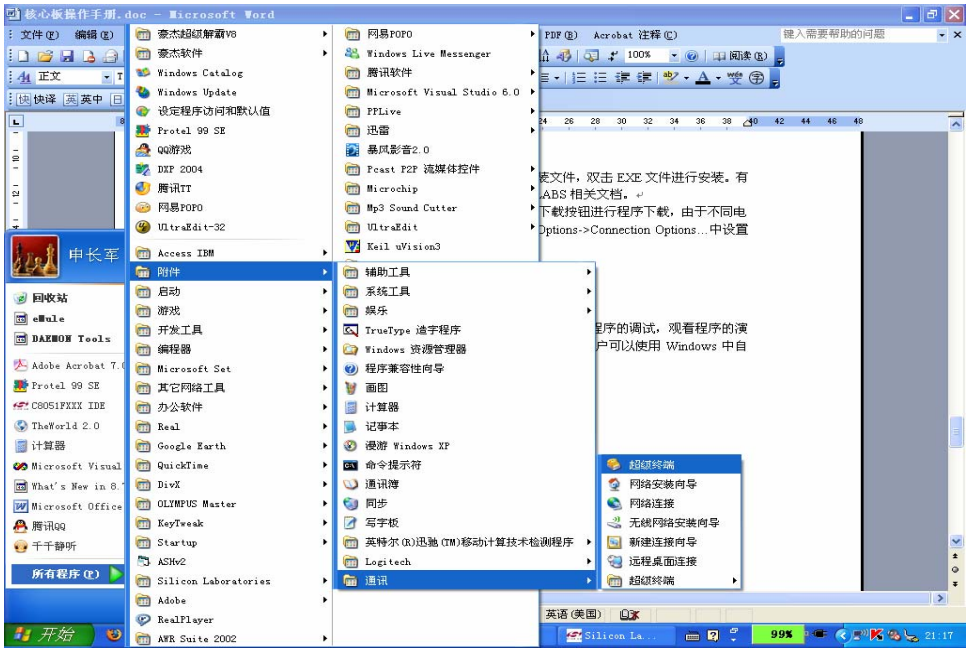
在光盘/software/IDE 下找到 C8051FXXX IDE 的安装文件，双击 EXE 文件进行安装。有关 IDE 环境的安装和调试器驱动的安装过程请参考 SILABS 相关文档。

在 IDE 和调试器安装完成后，当前演示程序，点击下载按钮进行程序下载，由于不同电脑上的调试器安装后的实际串口不同，注意在 IDE 的 Options->Connection Options…中设置调试器使用的串口。

2.2.2 运行演示程序

IDE 和调试器安装完成后，用户即可以打开相应的程序进行程序的调试，观看程序的演示运行效果，其中，演示程序中多数程序使用串口进行通信，用户可以使用 Windows 中自带的超级终端进行串口通信，具体设置过程如下：

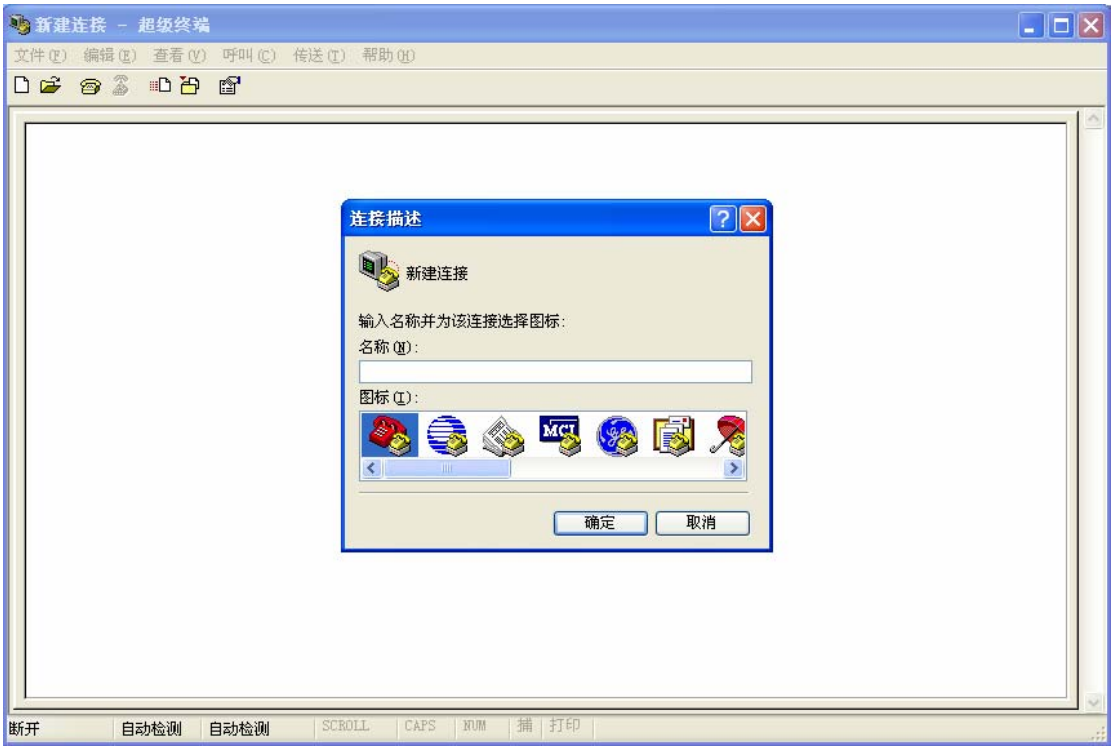
1. 从系统的”开始”处，在如下路径中找到超级终端：



2．启动后，如果出现如下画面：

点击”取消”按钮，在随后出现的提示对话框中选择”是”。

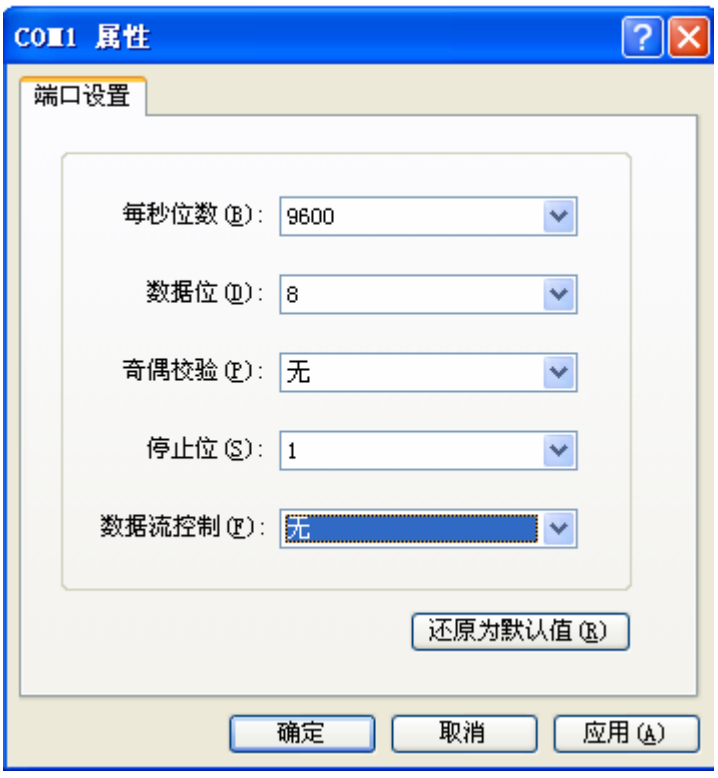
3．在新建连接中输入一个连接名称（任意）:单击确定按钮



4．如果随后再次出现位置信息对话框，依然选择取消并忽略其后的提示。

5．在连接到的对话框中选择与验证板连接的串口，单击确定：

6．在串口属性对话框中，按如下参数设置串口:



7．单击确定后端口打开。

8．为了使用键盘上的退格键（Backspace）,需要单击工具栏上的属性按钮，在出现的对话框上中选择设置页面，然后按如下设置 Backspace 键的发送方式:

单击确定即可。

9．通过调试器下载并运行 Uart0 程序，可以在超级终端的窗口上看到：C8051F02x core Module EC1,v1.0

用户在超级终端上输入任何字符，应该可以看到该字符。如果用户按键盘后没有相应，请检查验证板程序是否在运行、串口设置是否正确等。

演示程序中的绝大多数程序都用到了串口，每个演示程序中都有不同的控制命令，具体请参考软件参考部分。

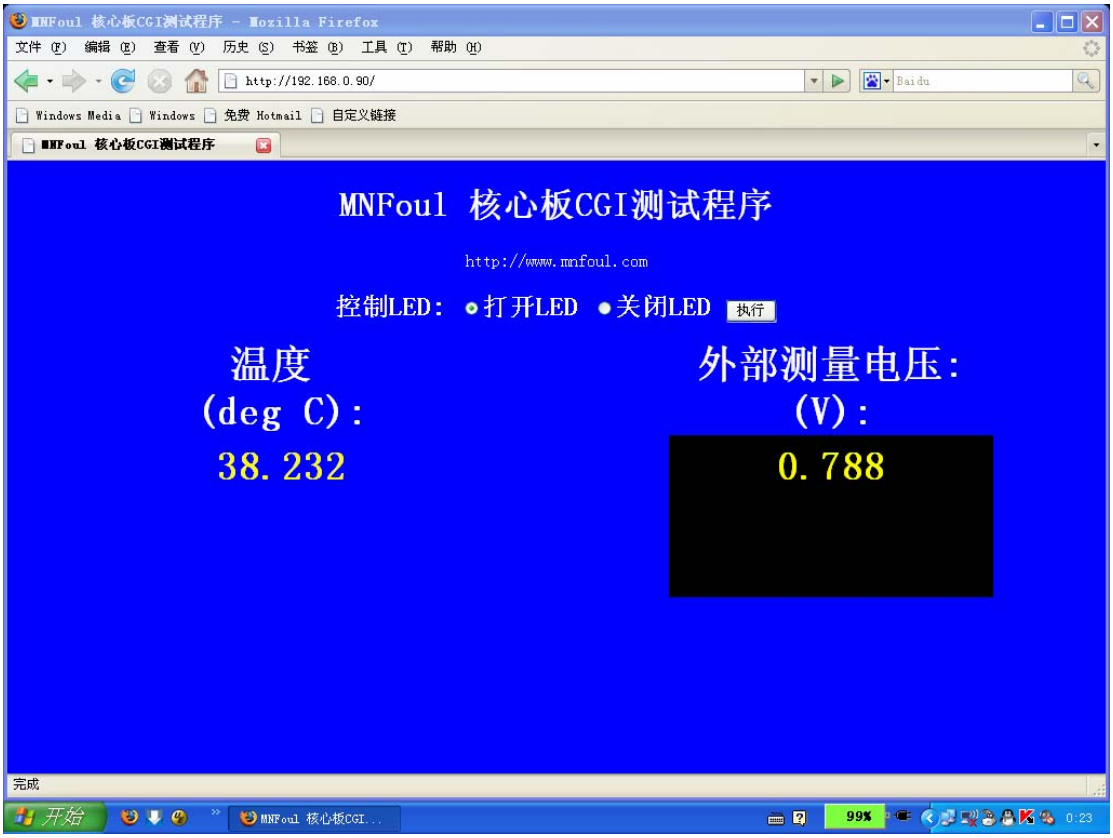
2.2.3 以太网演示程序

也许，现在你已经很想看到我们最强大的 WEB 服务器功能了，所以，我们首先给你一个机会看到简单强大的以太网应用。请您按以下步骤进行安装：

1．将 java_keil 文件夹中的 cgi.hex 通过调试器下载到核心板中。（如果您想重新编译该程序，需要在 keil uVision3 中打开工程，并需要安装 silabs 的 Keil uVison3 驱动以便在 uVision3 中通过调试器在线调试程序，要求 Keil C51 是 8.0 或以上版本）。

2．使用一条交叉网线将计算机与验证板连接。（如果您要通过 HUB 或交换机连接，则应该使用直连网线。）

3. 演示程序中核心模块的 IP 地址为 192.168.0.90,请将您的计算机的 IP 地址设置为 192.168.0.100(最后的一个字节可以为任意，但不能为 90)。子网掩码设置为 255.255.255.0,默认网关为 192.168.0.1。
4. 将验证板上的拨码开关 JP1,JP2 的所有位设置为 ON 状态。
5. 现在请将调试器去掉，并重新给验证板上电。
6. 观察计算机的网络连接指示或网口的指示灯，他们会亮起来。
7. 打开浏览器，输入 http://192.168.0.90,浏览器显示如下画面：



可以通过单击执行按钮控制验证板上的 LED2。

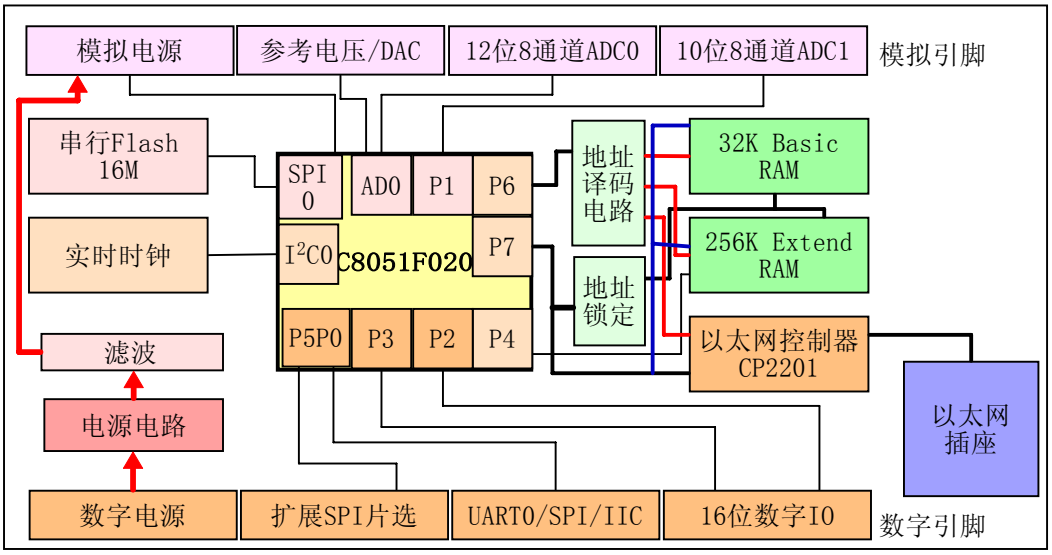
温度和外部测量电压显示电路板上的测量值。

网页每 2 秒自动刷新一次，实时显示测量的温度和电压，可以调节验证板的 R5 来改变测量电压，可以看到网页上的信息实时改变。

3 核心模块参考

本章描述了核心模块的硬件结构，通过本章您可以了解模块的硬件配置和使用方法。

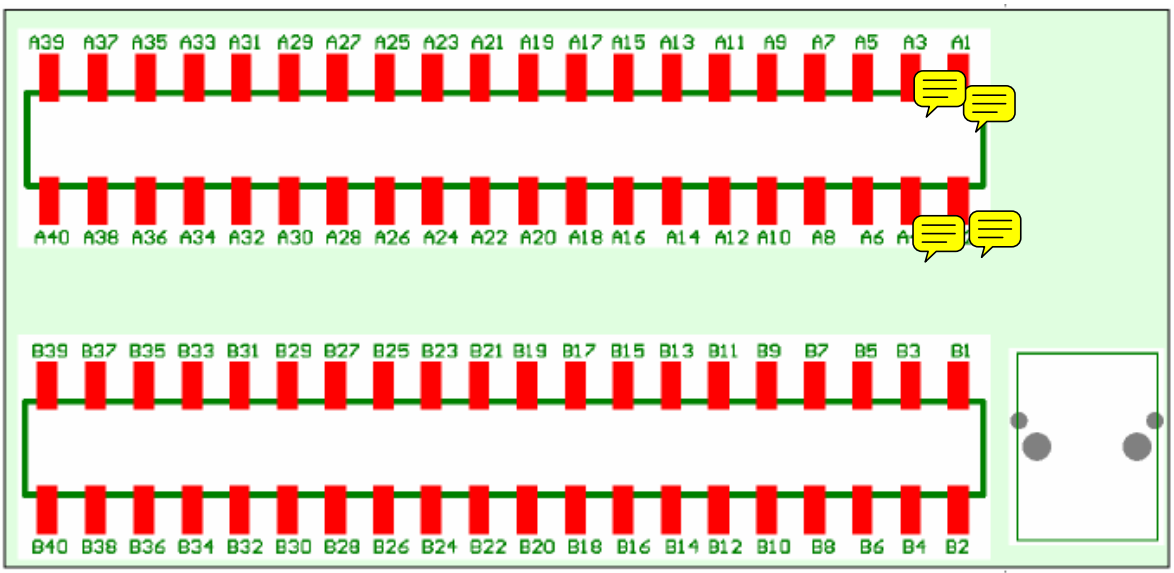
3.1 模块结构



FM020 模块以 C8051F020 为核心，集成了数据 FLASH、实时时钟、扩展 RAM、以太网控制器等外设，构成了一个完整的功能强大的模块。

3.2 模块引脚

模块**正视图**如下，引脚顺序及引脚功能描述如下：



引脚描述：

1．模拟引脚

引脚	名称	方向	功能
A1	AV	P [A]P O/I	模拟电源，当选择片上模拟电源时，该引脚输出片上模拟电源;可通过改变滤波电阻使该引脚作为输入，由外部为模拟部分提供电源。模拟电源与数字电源电压差必须小于 0.5V[1]。 输出：3.3V 输入：3.3V
A2	AGND	[A]P	模拟电源地
A3	VREF	[A]O	模拟参考电压;软件可设置为 1.2V 或 2.4V
A4	AGND	[A]P	模拟地
A5	DAC0	[A]O	12 位 DAC0 电压输出引脚，不可用来驱动负载
A6	DAC1	[A]O	12 位 DAC1 电压输出引脚，不可用来驱动负载
A7	CP1N	[A]I	比较器 1 负输入端
A8	CP1P	[A]I	比较器 1 正输入端
A9	CP0N	[A]I	比较器 0 负输入端
A10	CP0P	[A]I	比较器 0 正输入端
A11	AV	[A]P	模拟电源，与 A1 引脚相通。
A12	VREF0	[A]I	ADC0 参考电压，最大 3.3V
A13	VREFD	[A]I	DAC 参考电压
A14	AGND	[A]P	模拟地
A15	VREF1	[A]I	ADC1 参考电压，最大 3.3V
A16	AGND	[A]P	模拟地
A17	AGND	[A]P	模拟地
A18	AIN0	[A]I	12 位 ADC0 输入通道 0
A19	AIN1	[A]I	12 位 ADC0 输入通道 1
A20	AIN2	[A]I	12 位 ADC0 输入通道 2
A21	AIN3	[A]I	12 位 ADC0 输入通道 3
A22	AIN4	[A]I	12 位 ADC0 输入通道 4
A23	AIN5	[A]I	12 位 ADC0 输入通道 5
A24	AIN6	[A]I	12 位 ADC0 输入通道 6
A25	AIN7	[A]I	12 位 ADC0 输入通道 7
A26	AGND	[A]P	模拟地
A27	P17	[A] [D]IO	8 位 ADC1 输入通道 7 数字 IO 口 P17
A28	P16	[A]I [D]IO	8 位 ADC1 输入通道 6 数字 IO 口 P16
A29	P15	[A]I [D]IO	8 位 ADC1 输入通道 5 数字 IO 口 P15
A30	P14	[A]I [D]IO	8 位 ADC1 输入通道 4 数字 IO 口 P14
A31	P13	[A]I	8 位 ADC1 输入通道 3

		[D]IO	数字 IO 口 P13
A32	P12	[A]I [D]IO	8 位 ADC1 输入通道 2 数字 IO 口 P12
A33	P11	[A]I [D]IO	8 位 ADC1 输入通道 1 数字 IO 口 P11
A34	P10	[A]I [D]IO	8 位 ADC1 输入通道 0 数字 IO 口 P10
A35	AGND	[A]P	模拟地
A36	**RTCCLK**	[D]O	RTC 可编程时钟脉冲输出
A37	**RTCINT**	[D]O	RTC 中断输出，开漏输出，低电平有效
A38	**P44**	[D]O	输出端口 P44，不能按位操作
A39	**RESET**	[D]IO	单片机复位；开漏引脚
A40	**GND**	[D]P	数字地

2．数字引脚

引脚	名称	方向	功能
B1	VIN	[D]P	模块输入电压，默认情况下用来产生 3.3V 的数字和模拟电压 输入电压范围：4.5~5.5V
B2	VIN	[D]P	模块输入电压，与 B1 相同。
B3	GND	[D]P	数字地
B4	GND	[D]P	数字地
B5	BAT	[D]P	RTC 备用电源，电压范围 2.0～6.0，通常使用纽扣电池
B6	VOUT	[D]P	数字电源输出,3.3V
B7	VOUT	[D]P	数字电源输出,3.3V
B8	GND	[D]P	数字电源地
B9	SPICS0	[D]O	SPI 总线片选信号 0
B10	CP2201Rst	[D]IO	以太网控制器 CP2201 复位引脚；低电平复位；必须使用单片机 IO 引脚控制该引脚
B11	CP2201Int	[D]O	以太网控制器 CP2201 中断输出引脚；必须与单片机外部中断 INT0 连接
B12	SPICS4	[D]O	SPI 总线片选信号 4
B13	SPICS3	[D]O	SPI 总线片选信号 3
B14	SPICS2	[D]O	SPI 总线片选信号 2
B15	SPICS1	[D]O	SPI 总线片选信号 1
B16	TXD0	[D]O	UART0 发送引脚
B17	RXD0	[D]I	UART0 接收引脚
B18	SPICLK	[D]IO	SPI0 总线时钟引脚。 作为主控器件时为输出；作为从器件时为输入
B19	SPIMISO	[D]IO	SPI0 总线数据引脚。 作为主控器件时为输入；作为从器件时为输出
B20	SPIMOSI	[D]IO	SPI0 总线数据引脚。 作为主控器件时为输出；作为从器件时为输入
B21	SPIMSS	[D]I	SPI0 总线从机选择引脚。 当模块作为 SPI 从机时，主控器件使用该引脚选择模块。

			当该引脚为低时模块处于从机模式；当引脚为高时，模块退出从机模式。
B22	I2CSDA	[D]IO	I2C 总线数据总线,开漏引脚。
B23	I2CSCL	[D]I	I2C 总线 SCL
B24	P30	[D]IO	数字 IO，可编程为推挽输出、开漏输出、输入，可编程弱上拉，可按位操作。
B25	P31	[D]IO	数字 IO，可编程为推挽输出、开漏输出、输入，可编程弱上拉，可按位操作。
B26	P32	[D]IO	数字 IO，可编程为推挽输出、开漏输出、输入，可编程弱上拉，可按位操作。
B27	P33	[D]IO	数字 IO，可编程为推挽输出、开漏输出、输入，可编程弱上拉，可按位操作。
B28	P34	[D]IO	数字 IO，可编程为推挽输出、开漏输出、输入，可编程弱上拉，可按位操作。
B29	P35	[D]IO	数字 IO，可编程为推挽输出、开漏输出、输入，可编程弱上拉，可按位操作。
B30	P36	[D]IO	数字 IO，可编程为推挽输出、开漏输出、输入，可编程弱上拉，可按位操作。
B31	P37	[D]IO	数字 IO，可编程为推挽输出、开漏输出、输入，可编程弱上拉，可按位操作。
B32	P20	[D]IO	数字 IO，可编程为推挽输出、开漏输出、输入，可编程弱上拉，可按位操作。
B33	P21	[D]IO	数字 IO，可编程为推挽输出、开漏输出、输入，可编程弱上拉，可按位操作。
B34	P22	[D]IO	数字 IO，可编程为推挽输出、开漏输出、输入，可编程弱上拉，可按位操作。
B35	P23	[D]IO	数字 IO，可编程为推挽输出、开漏输出、输入，可编程弱上拉，可按位操作。
B36	P24	[D]IO	数字 IO，可编程为推挽输出、开漏输出、输入，可编程弱上拉，可按位操作。
B37	P25	[D]IO	数字 IO，可编程为推挽输出、开漏输出、输入，可编程弱上拉，可按位操作。
B38	P26	[D]IO	数字 IO，可编程为推挽输出、开漏输出、输入，可编程弱上拉，可按位操作。
B39	P27	[D]IO	数字 IO，可编程为推挽输出、开漏输出、输入，可编程弱上拉，可按位操作。
B40	GND	[D]IO	数字地

注：[1] 见电源部分详细解释

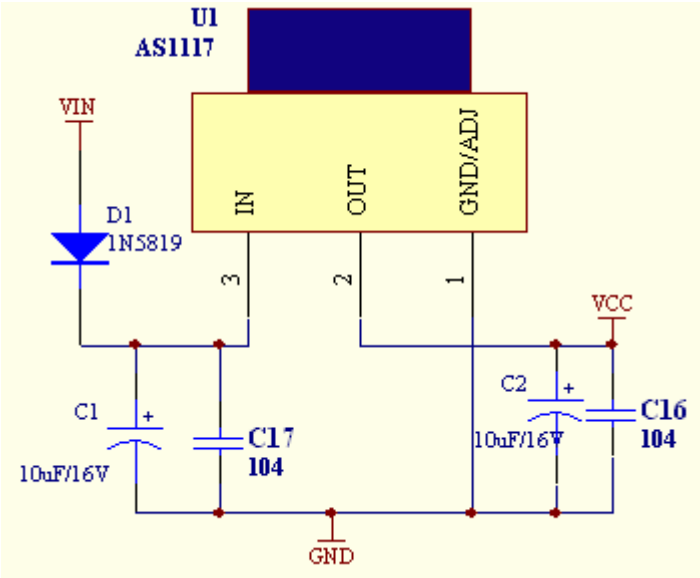
[2] 方向符号标识：[A]：模拟功能；[D]：数字功能；I:输入；O：输出；IO：输入输出

P：电源

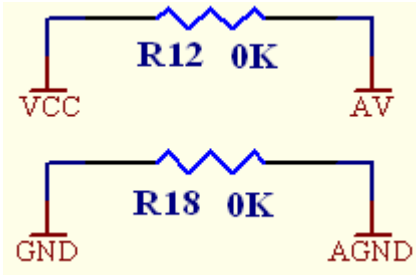
3.3 外设资源

FM020 核心模块是以 C8051F020 为核心的混合信号模块，模块在 C8051F020 的基础上集成了丰富的外设功能。有关 C8051F020 的信息请参考相应的数据手册。本节描述外设的结构及使用方法。

3.3.1 电源



如图所示为系统数字电源产生电路图。模块模拟电源的产生方法如下：



模块出厂时，R12,R18 均焊接，因此，模拟电源引脚 AV,AGND 为输出，可以给模拟外设提供 3.3V 的模拟电压。如果用户需要从外部提供性能更好的模拟电源，需要定购模块时说明或将模块上的 R12,R18 取下。用户外供模拟电源必须为 3.3V,与数字电压不可有超过 0.5V 的偏差。

3.3.2 IO 端口

交叉开关被分配到 P4-P7 高端端口，P0 端口已经进行了如下的外设分配，用户使用时必须在交叉开关使能这些外设：

外设	功能引脚	分配的引脚
UART0	TXD0	P0.0
	RXD0	P0.1
SPI	SPICLK	P0.2
	SPIMISO	P0.3
	SPIMOSI	P0.4
	SPIMSS	P0.5
SMBUS	SDA	P0.6
	SCL	P0.7

用户需要使用的其它外设如：UART1，外部中断 0，外部中断 1 等可以根据程序需要配置的适当的引脚。

模块将 P1,P2,P3 端口完整的引出。

单片机 P5 端口配置为外设的输出，具体分配如下：

单片机引脚	功能
P50	FlashReset, 串行 Flash 的复位引脚
P51	FlashProtect, 串行 Flash 的写保护引脚
P52	FlashCS, 串行 Flash 的片选信号
P53	SPICS0, 引出到模块 B9 引脚
P54	SPICS1, 引出到模块 B15 引脚
P55	SPICS2, 引出到模块 B14 引脚
P56	SPICS3, 引出到模块 B13 引脚
P57	SPICS4, 引出到模块 B12 引脚

3.3.3 RAM

FM020 核心模块数据总线上扩展了 62LV256、62LV2568(或 62LV1024)、CP2201 上个设备。C8051F020 的 EMIF 端口使用高位复用端口模式，外扩设备的地址空间为：

- 62LV256: 0x0000-0x7FFF
- CP2201: 0x8000-0xBFFF
- 62LV1024（62LV2568）: 页内地址为 0xC000-0xFFFF

  62LV1024 总存储空间为 128K，被分为 8 页(62LV2568 为 256K，分为 16 页)，每页空间 16K，页地址使用 P43,P42,P41,P40 四条地址线选择。

  P43 为页面选择最高地址位；P40 为页面选择最低地址位

程序访问该存储空间需要首先使用 P43-P40 选择相应的页面，然后使用 XDATA 设定相应的地址进行访问。

3.3.4 FLASH

模块使用 SPI 总线扩展了一块 16M 的串行 FLASH AT45DB161(或 AT45DB041,AT45DB081，根据模块型号),SPI 引脚需要根据 3.3.2 节描述的方式设置交叉开关。AT45DB161 引脚连接方式如下：

AT45DB 引脚	**C8051F020 引脚**
SI(1)	P0.4(SPIMOSI)
SCK(2)	P0.2(SPICLK)
RESET(3)	P5.0(FlashReset)
CS(4)	P5.2(FlashCS)
WP(5)	P5.1(FlashProtect)
VCC(6)	VCC(数字电源)
GND(7)	GND(数字地)
SO(8)	P0.3(SPIMISO)

当用户外扩 SPI 设备时，应该使用模块的 SPICS0,SPICS1,SPICS2,SPICS3,SPICS4 引脚作为设备的片选信号，并在使能片选信号时必须保证 SPICS0 到 SPICS4 及 FlashCS 信号中只能有一个有效，以防止 SPI 总线冲突。

3.3.5 RTC

模块内含 RTC 实时时钟 PCF8563，使用 I2C 总线连接，PCF8563 引脚连接如下：

PCF8563 引脚	C8051F020 或模块引脚
INT(3)	模块的 A37 引脚（RTCINT）
SDA(5)	P0.6(SDA)
SCL(6)	P0.7(SCL)
CLK	模块的 A36 引脚（RTCCLK）

如果用户需要使用 RTC 的 INT 和 CLK 引脚，应该在外部将相应的引脚与单片机的引脚相连。RTC 的备用电源由模块的 BAT(B5)引脚提供,通过该引脚可以为 RTC 提供备用电源，以实现实时时钟调电后可以继续运行。

3.3.6 以太网

模块内含 CP2201 作为以太网控制器。CP2201 是并口设备，分配在 0x8000-0xBFFF 地址空间内。CP2201 有两个引脚连接到模块引脚上，用户在使用时需要将引脚与模块的适当的 IO 口连接，这两个引脚如下：

- B10 引脚 CP2201Rst：以太网控制器复位引脚，应该与单片机 IO 口连接
- B11 引脚 CP2201Int：以太网控制器中断输出引脚，该引脚必须与 C8051F020 的 INT0 连接。

3.3.7 模拟部分

FM020 模块内核两个 ADC，2 个 DAC,2 个模拟比较器。其中 ADC0 是 12 位 8 通道转换器，ADC1 是一个可选择使能的 8 位 8 通道转换器。

1．模拟电源：

AV,AGND：模拟电源

用户可以从外部提供性能更好的模拟电压，具体见 3.3.1 部分。

2．ADC0 硬件资源

- 8 输入通道：模块引脚 A18-A25(AIN0-AIN7)
- 参考电压：模块引脚 A12(VREF0)
- 可编程增益：16，8，4，2，1，0.5
- 内置温度传感器

3．ADC1 硬件资源

- 8 输入通道：模块引脚 A27-A34(P17-P10)
- 参考电压：模块引脚 A15(VREF1)
- 可编程增益：4，2，1，0.5

4．可编程的输出参考电压，VREF

用户可从引脚 VREF0,VREF1 提供参考高性能参考电压，也可以使用模块 VREF 引脚输出的 1.2V/2.4V 参考电压。

5．DAC0/DAC1:

FM020 模块内含两个 12 位 DAC。

有关模拟部分的具体使用方法，可以参加 C8051F020 的数据手册。

3.4 使用要点

- 模块主电源供电为：4.2-5.5V，最大电流 300mA。超过供电电压可能引起永久损坏，低于该电压可能引起模块工作不正常。
- 模块输出电源：3.3V，可提供最大电流 100mA,外部设备不能操作该值，否则可能引起永久损坏。
- 如果给模拟部分从单独供电，必须保证模拟电源与数字电源的差不超过 0.5V,数字电源电压为 3.3V
- 模块数字引脚耐压 5V，超过该电压可能引起模块永久损坏

- 模块模拟引脚输入电压必须小于模拟电源电压
- 其它的 C8051F020 手册中列出的可能引起单片机永久损坏的信息。

4 验证板参考

MOFM020 底板为配合 FM020 核心模块而开发的。MOFM020 验证板为核心模块的测试、调试、开发、使用提供环境。MOFM020 提供了以下验证功能：

- IO 口，输入输出
- 串口通信
- 实时时钟
- 数据存储
- ADC
- DAC
- 以太网
- 可扩展的面包板
- 3 个 SO-16 芯片扩展位置
- 10 位输出端子

用户使用该验证板可以快速体验 FM020 核心模块的强大功能，并可以实现用户自定义功能的验证和开发，配套提供的测试程序则为用户的开发提供了快速通道。

4.1 验证板 MOFM020 总体结构

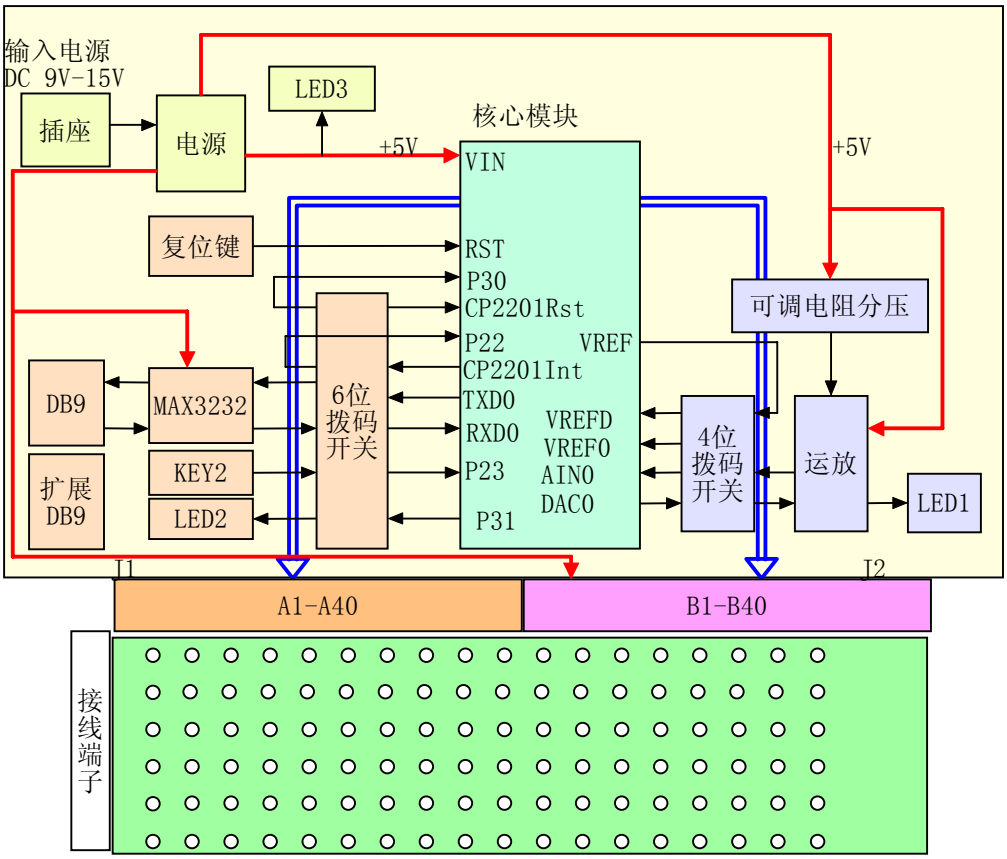


图 4.1-1 验证板结构框图

4.2 验证板功能

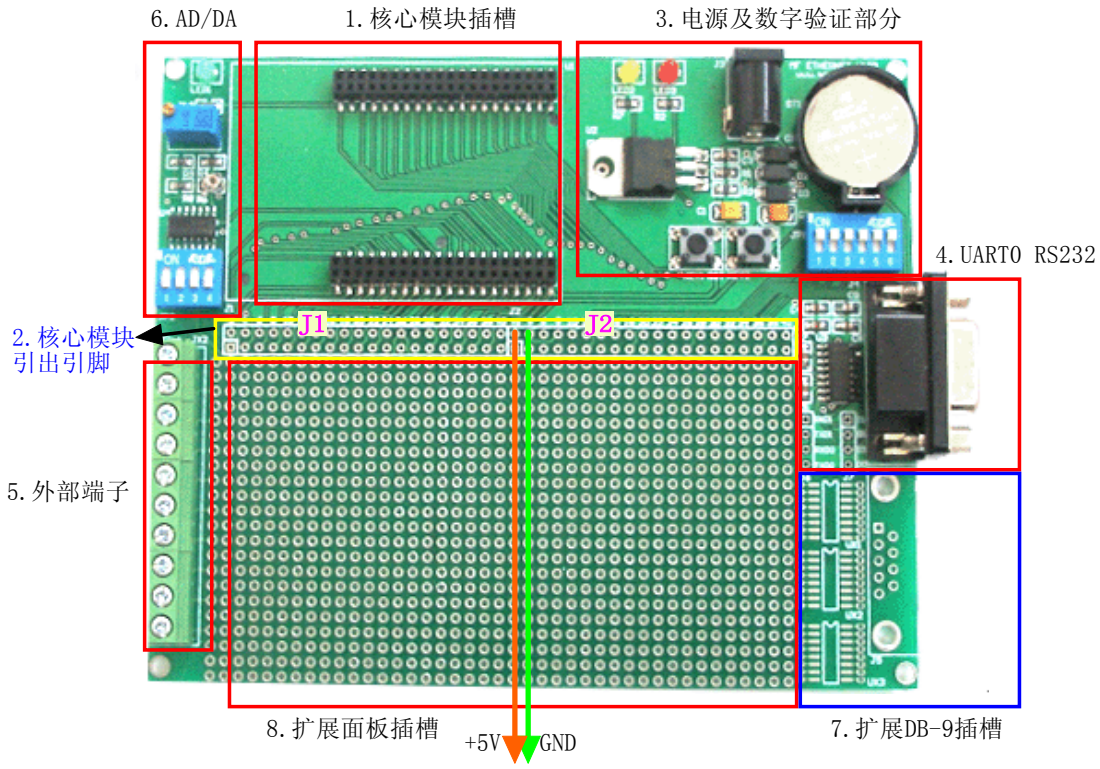


图 4.2-1 验证板布局及外形图

1. 核心模块插槽

两排 40 针的插座提供了 FM020 的插入位置，将核心模块的以太网口向上对齐插座插入。

2. 核心模块引脚引出区

图 4.2-1 中黄色区域内的两排焊盘为核心模块的引脚。左侧 J1 的 40 个双排插孔是模拟部分的 40 个引脚，它与核心模块的 A1-A40 引脚对应。具体如下：

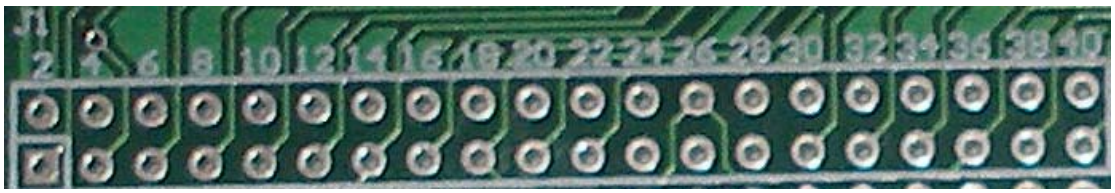


图 4.2-2 模拟引脚引出焊盘

图中方形焊盘为核心模块的 A1 引脚，上方标有 2 的引脚为 A2 引脚，依次类推。这些焊盘与下侧的扩展面板插槽对应，方便用户扩展演示功能。

右侧 J2 的 40 个双拼焊盘对应数字部分的 40 个引脚，它与核心模块的 B1-B40 对应。具体如下：

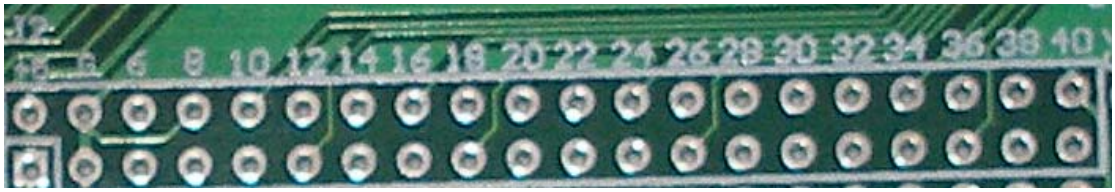


图 4.2-3 数字引脚引出焊盘

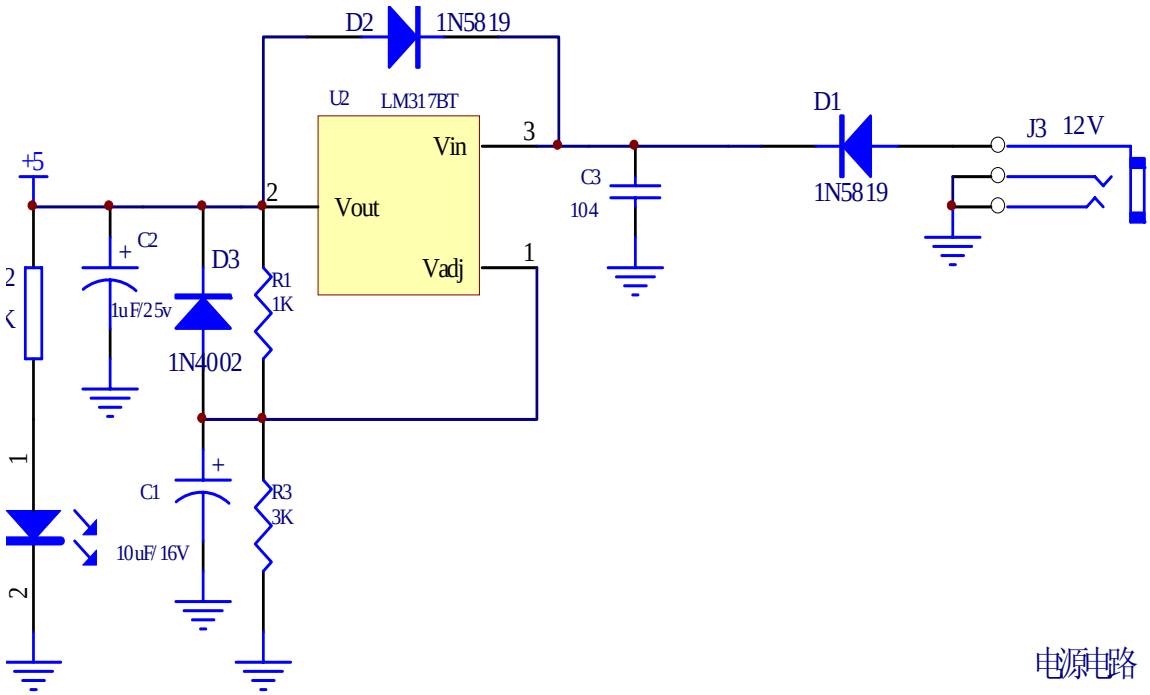
图中方形焊盘为核心模块的 B1 引脚，上方标有 2 的引脚为 B2 引脚，依次类推。左侧的 B1,B2 引脚为核心模块的输入电压＋5V，该列（图中桔红色线的标注）均为 5V，相邻的右侧的 B3,B4 上方标有 G，表明该列（图中绿色线的标注）均为 GND。

3. 电源及数字验证部分：

该部分包括了验证板主电源产生电路及按键、指示灯、备用电池等功能。

1）主板电源

验证板使用输出电压可调的 LM317 稳压芯片作为电源电路的核心，电源电路输出 5V 电压，作为核心板和其它外围芯片的供电。扩展插槽区域中的 5V,GND 列也是由该电源



产生的。

图 4.2-4 电源电路

2）按键电路

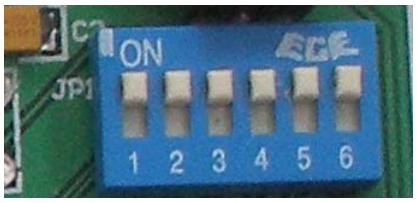
验证板上提供了两个按键，分别是 KEY1 和 KEY2。其中 KEY1 为复位按键，与核心模块的 RST 引脚连接。KEY2 通过拨码开关 JP1 的第 6 位与核心模块的 P23 引脚连接，该引脚可被配置为单片机的外部中断 1。当拨码开关 JP1 的第 6 位处于 ON 状态时，按键 KEY2 与 P23 接通。

3）指示灯

验证板上提供了三个 LED 作为输出指示，其中 LED1 为模拟部分输出电压指示，见下文模拟部分描述。核心模块的 P31 引脚通过拨码开关 JP1 的第 4 位驱动 LED2，可以用来作为数字部分的输出指示。

4）拨码开关

验证板上有 JP1,JP2 两个拨码开关，它们起到跳线的作用。当某一位处于 ON 状态时，核心板上的引脚与验证板的外设电路接通，用户可以方便的进行程序调试和测试。如果用户需要在扩展区域中构建自己的电路并需要使用这些引脚时，就可以把拨码开关的相应位设置位 OFF 状态，并从 J1,J2 处引出这些引脚。



JP1 共有 6 位，如图中拨码开关上标识的 1—6 所示。这些位对应的核心板引脚和验证板外围电路如下：

JP1 位	核心模块引脚	外设电路	说明
1	TXD0(B16)	U3-J4 组成的 RS232 接口	DB9 端子 J4 为 RS232 接口，该部分电路形成异步串行电路。串口调试部分必须设置。
2	RXD0(B17)		
3	P30（B24）	CP2201 的复位引脚	该位 ON 时可以通过 P30 控制 CP2201 的复位引脚，以太网功能必须设置为 ON。
4	P31(B25)	LED2	驱动外部 LED，指示部分必须使用。
5	P22(B34)	CP2201 中断输出引脚	P22 配置为单片机的 INT0，接收外部中断信号。以太网功能必须。
6	P23(B35)	按键 KEY2 输入	P23 配置为单片机的 INT1，按键输入信号。

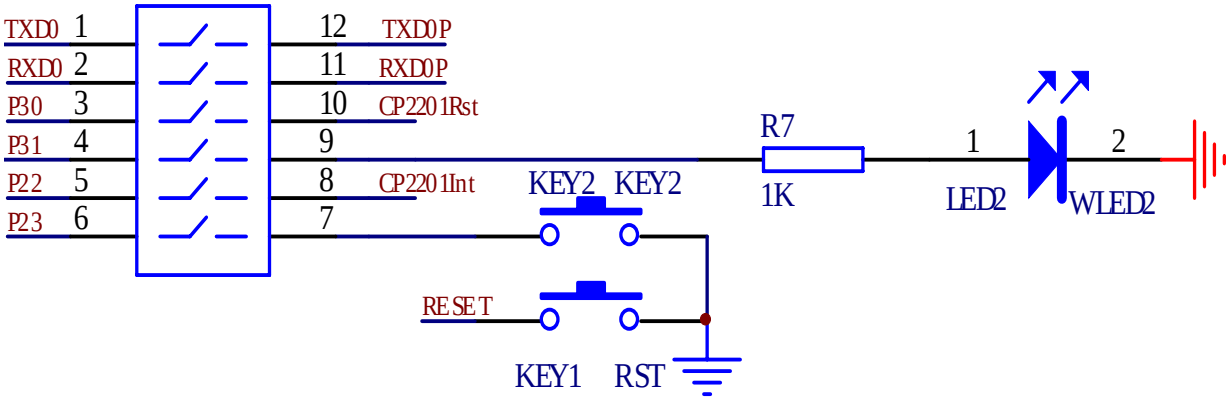


图 4.2-5 拨码开关及外围电路

5）后备电池

后备电池 BT1 使用 CR2032 电池为核心模块的 RTC 供电，保证验证板调电后核心模块的 RTC 依然正常运行，电池的正极与核心模块的 VBAT 连接。

4. 串行接口区域：

1）与核心模块 UART0 连接的 RS232：DB-9 母头的 RS232 接口 J4，可以直接与计算机等设备连接。使用 J4 时需要将 JP1 的 1,2 位设置为 ON 以便与核心模块的 TXD0,RXD0 连接。

2）预留的 DB9 端子 J5：验证板上有一个没有焊接器件的 DB9 接头 J5，该接头作为扩展使用。由于核心模块上没有定义 UART1，所以用户可以根据需要定义 UART1，并把相应的发送和接收引脚与 J6 的 1，2 连接，J6 的 1，2 已经与芯片 U6(MAX3232)的第二组转换通道连接，这样，在 J7 的 1，2 处便可以得到相应的 RS232 信号，用户可以将 J7 的 1，2 与 J5 的相应引脚连接，这样用户就可以得到第二个串行通信接口。当然，用户也可以不使用 U6 的转换功能，自己在扩展的 SO-16 芯片焊盘上(UX1,UX2,UX3)焊接 MAX485 等 RS485 转换芯片，以实现一个 485 通道。

下面举例说明预留的 DB9 通道的连线方法：假设用户将 P1 口作为了模拟接口，将 UART1 的 TXD1,RXD1 引脚分别分配到了 P20,P21 引脚，那么验证板上扩展的母头 DB9 连线方法如下：(图中有绿点的地方表示线的起始位置)。

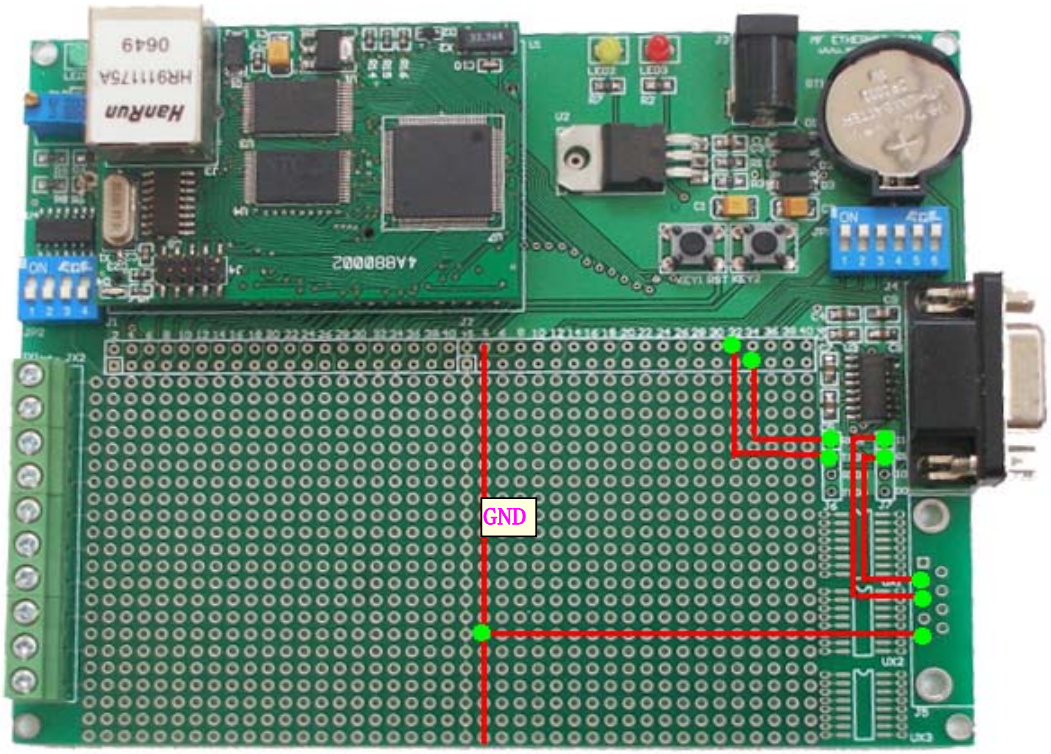
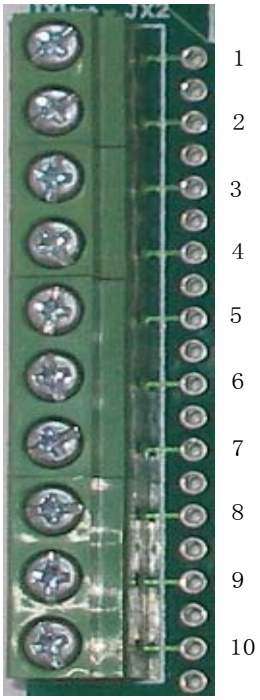


图 4.2-6 扩展 DB9 使用方法



5. 外部端子：

区域 5 中的 10 个接线端子可以作为对外部设备控制和采集的接口。接线端子已经与其后侧的焊盘连接，如下图所示：

外侧的 10 个端子与标有 1，2，…10 的 10 个焊盘相连接，可以直接使用这 10 个焊盘与其后侧的扩展区域连接。

6. AD/DA 验证区

AD/DA 验证区包括了一个用于切换线路连接的 4 位拨码开关、输入电压调节的可调电阻、电压跟随的运放 LM324 及用于输出电压指示的 LED。

模拟验证区使用 ADC0 的输入通道 1 采集经可调电阻分压的电压并通过 DAC0 输出采集的电压，经运放放大后以 2 倍的 DAC 输出电压驱动 LED。模拟部分使用核心模块输出的 VREF 作为参考电压。

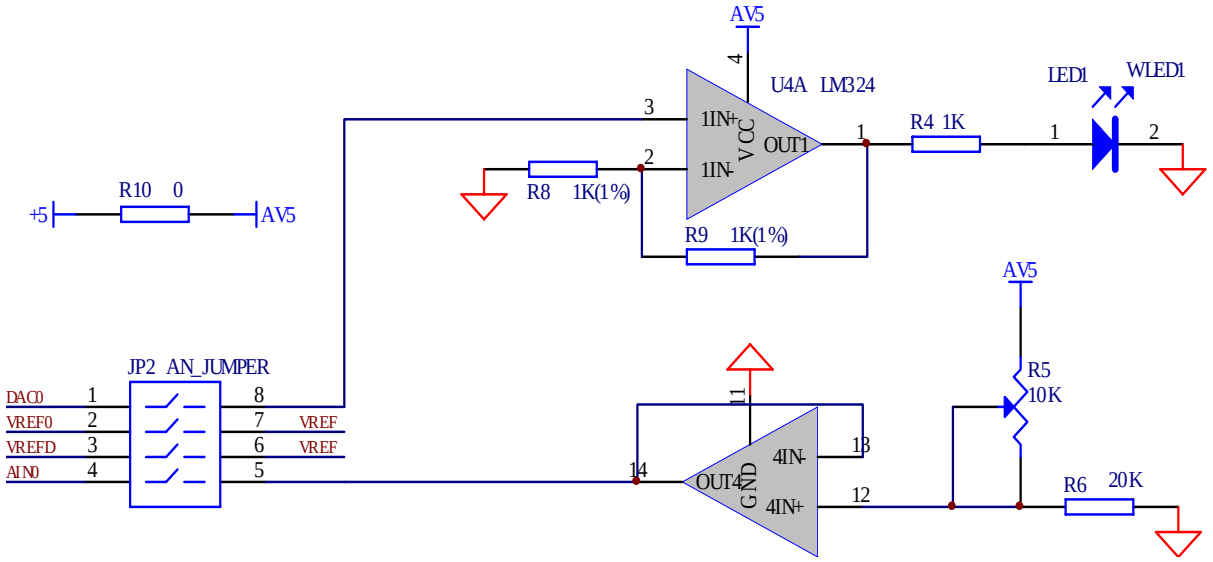


图 4.2-7 模拟区电路

拨码开关 JP2 用于连接核心模块的模拟引脚与验证板外设电路，各位对应关系如下：

JP2 位	核心模块引脚	外设电路	说明
1	DAC0	运放输出驱动电路	ON 状态 DAC0 驱动外围
2	VREF0	VREF,核心模块输出的参考电压	ON 状态后，ADC0 将使用 VREF 作为参考电压
3	VREFD	VREF,核心模块输出的参考电压	ON 状态后，DAC0 将使用 VREF 作为参考电压
4	AIN0	经分压后的电压信号	将可调电阻分压后的电压输入该引脚进行转换

7. 扩展面板插槽

验证板上的扩展面板焊盘为进行自定义的扩展和开发提供了方便的途径。

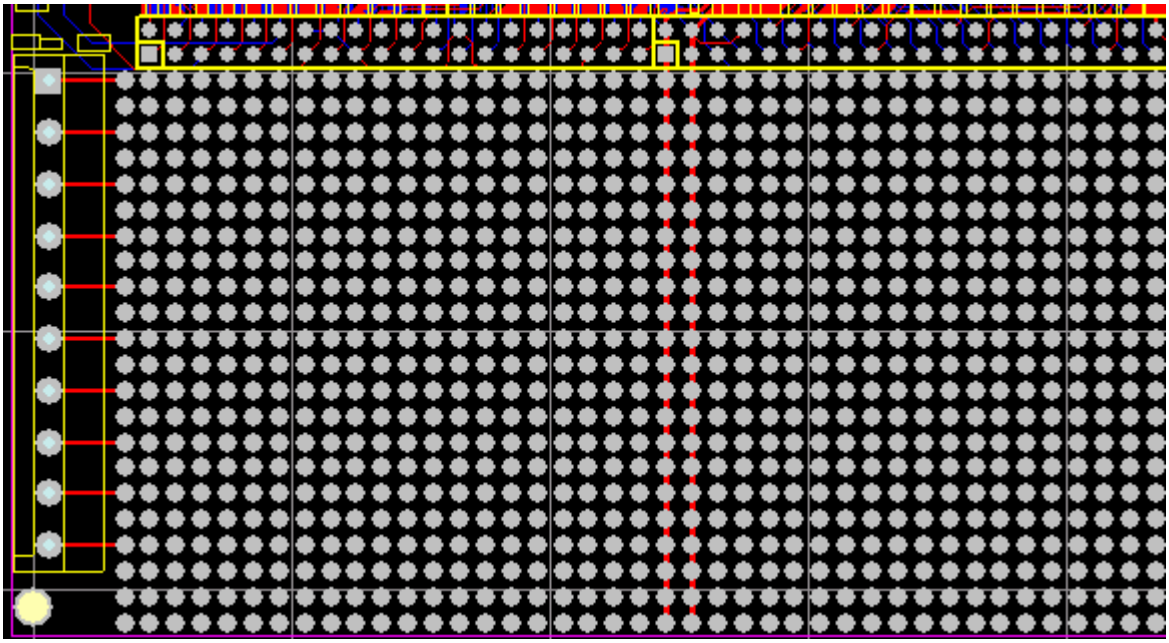


图 4.2-8 扩展区域

扩展区域共 40×22 个焊盘，焊盘间距为标准的 100mil，可以用来焊接标注的 DIP 芯片，右侧的扩展的 3 个 SO-16 焊盘可以用来焊接贴片芯片。扩展区域需要注意几点：

1） 与 J2 的第一、二列对应的两列焊盘分别时＋5V 和 GND 信号。即图中两条红色的线穿过的焊盘。

2） 最左侧的一列焊盘已经与接线端子连接，由于接线端子的焊盘间距为 200mil，所以连接的焊盘为每隔一个使用一个，如图中左侧的水平红线连接的部分。

3） 图中上部的黄色框中的区域分别为核心模块的 A,B 引脚，所以在使用时要小心引脚顺序，不要将高压加到核心模块引脚上，以免造成模块损坏。

5 软件参考

开发套件中包含了 8 个测试程序，分别涵盖了外部中断、串口通信、RAM 使用、实时时钟、SPI 总线（串行 FLASH）、AD/DA、以太网等。

5.1 外部中断

1. 程序文件夹：\prog\exint
2. 源文件：
   - exint.c：主文件
   - F020.h：头文件
3. 编译环境：C8051FXXX IDE，Keil C51
4. 功能：通过交叉开关，将 INT0,INT1 外边配置到 P22,P23 引脚上，从而在外边中断 1 的驱动下，实现底板上发光二极管 LED2 随按键 KEY2 的按下而亮/灭
5. 实验条件：1.底板上拨码开关 JP1 的 4,6 位设置到 ON 状态
6. 注意事项：底板上 INT0 是位 CP2201 的 INT 设计的，所以本实验中 INT0 没有使用，中断必须被关闭

5.2 串口通信

1. 程序文件夹：\prog\uart0
2. 源文件：
   - Uart0.c：主文件
   - F020.h：单片机寄存器头文件
3. 编译环境：C8051FXXX IDE,Keil C51
4. 功能：通过交叉开关，将 UART0 配置到 P00,P01 引脚上,实现.通过计算机向模块发送命令控制底板上的 LED 灯
5. 实验条件：底板上 JP1 的 1，2，4，6 设置为 ON 状态
6. 注意事项：使用超级终端进行串口调试，注意设置串口参数为 9600,8,N,1；.串口通信协议是基于字符的，支持 BACKSPACE 键，但是不支持其它功能键和方向键。
7. 通信命令：（所有命令以回车结束）
   1）setled=2,on：点亮 LED2
   2）setled=2,off：关闭 LED2

5.3 RAM

1. 程序源文件夹：\prog\xram
2. 源文件：
   - Xram.c
   - F020.h

3. 编译环境：C8051FXXX IDE,Keil C51
4. 功能：
   - 通过计算机向模块发送命令控制底板上的 LED 灯。
   - 向指定的 RAM 发送读写命令。
   - 发送命令进行外部 RAM 测试。
5. 实验条件：底板上 JP1 的 1，2，4,6 设置为 ON
6. 注意事项：

   1）底板上外边扩展的 XRAM 地址空间分布：
   - SRAM：U3(62LV256): 0-0x7FFF，32K
   - TRAM：U4(62LV2568/62LV1024), 通过分页方式访问：

     页内地址：0xC000-0xFFFF

     页地址： HRAM3:HRAM0
   - 以太网控制器：CCP2201: 0x8000-0xBFFF

   2）根据使用的核心模块的型号不同，检查 U4 实际焊接的芯片，需要更改程序中 TRAM_TYPE 的宏定义：

   //#define TRAM_TYPE T62LV2568

   #define TRAM_TYPE T62LV1024
7. 通信命令：

   1）命令说明：

   字符通信协议，所有的命令以回车符结束，命令字符不区分大小写

   键盘控制字符支持 Backspace，其它控制键不支持

   2）应答：命令操作成功，最后都将返回\r\nOK\r\n

   命令错误，返回\r\nError\r\n

   3）具体命令：
   - 控制 LED

     命令格式：setled=灯序号,开关状态 \r

     参数： 灯序号： 2,LED2

     开关状态：ON,打开灯；OFF,关闭灯

     应答： \r\nOK\r\n
   - 测试 SRAM,U2

     命令格式：testsram\r

     参数： 无

     应答： 发送测试的结果；如果测试中发现错误，可能出现了芯片焊接问题或损坏问题
   - 测试 TRAM,U4

     命令格式：testtram\r

     参数： 无

     应答： 发送每个页面的测试结果；测试过程中会修改 RAM 中保存的内容
   - 向指定页面写入字符串

     命令格式：writetram=页面\r

     参数： 页面，0-15

     应答： \r\nOK\r\n;\r\nError\r\n

     说明： 写入的字符串均在页面的 0 偏移处开始写入，即 0xC000 地址空间；用户发送的字符串长度不要超过接收缓冲区的长度
   - 读取指定页面从 0 偏移处开始的指定长度数据

     命令格式：readtram=页面，长度\r

参数： 页面序号：欲写入的页面,0-15；
长度：要读出的字节个数
应答： 16 进制的方式显示 264 个字节的内容；同时每个 16 进制后的括号内显示该字符；字节间用空格隔开
\r\nOK\r\n

5.4 实时时钟

1. 程序文件夹：\prog\rtc
2. 源文件：
   - rtc_main.c：主文件
   - rtc.c：SMBUS 操作函数实现文件
   - rtc.h：SMBUS 操作头文件
   - datatype.h：数据类型定义文件
   - F020.h：单片机寄存器头文件
3. 编译环境：C8051FXXX IDE,Keil C51
4. 功能：通过交叉开关，将 SMBUS 配置到 P05,P06 引脚上,实现通过 IIC 总线访问 PCF8563，通过计算机向模块发送命令获取和设置 PCF8563 的实时时钟。
5. 实验条件：底板上 JP1 的 1，2，4，6 设置为 ON 状态。
6. 注意事项：无。
7. 通信命令：（所有命令以回车结束）
   - 控制 LED

   命令格式：setled=灯序号,开关状态 \r
   参数： 灯序号： 2
   开关状态：ON,打开灯；OFF,关闭灯
   应答： \r\nOK\r\n
   - 读取当前时间

   命令格式：gettime
   参数： 无
   应答： HH:MM:SS YYYY/MM/DD\r\n \r\nOK\r\n
   - 设置当前时间

   命令格式：settime=YYYYMMDDHHMMSS
   参数： 年月日时分秒，具体格式如 20070505110203
   应答： HH:MM:SS YYYY/MM/DD\r\n \r\nOK\r\n

5.5 数据 FLASH

1. 程序文件夹：\prog\at45db
2. 源文件：
   - At45main.c：主文件
   - At45.c： AT45DB 操作
   - At45.h： AT45DB 操作头文件
   - datatype.h： 数据类型定义文件
   - F020.h： 单片机寄存器头文件

3. 编译环境：C8051FXXX IDE,Keil C51
4. 功能：通过交叉开关，将 SPI 配置到 P02,P03,P04 引脚上,实现通过 SPI 总线访问 AT45DB041B/081B/161D，通过计算机向模块发送命令对 FLASH 进行操作。
5. 实验条件：底板上 JP1 的 1，2，4，6 设置为 ON 状态。
6. 注意事项：根据核心模块的型号不同，U6 可能焊接不同的芯片，本测试程序启动时会自动检测 U6 的类型。
7. 通信命令：（所有命令以回车结束）

1）.控制 LED

命令格式：setled=灯序号,开关状态 \r

参数：　灯序号:　2

开关状态：ON,打开灯；OFF,关闭灯

应答：　\r\nOK\r\n

2）.擦除指定页面

命令格式：erasepage=页面序号\r

参数：　页面序号，要擦除的页面的序号，从 0 到器件的页面最大编号，由使用的器件决定

应答：　\r\nOK\r\n

3）.擦除指定的块

命令格式：eraseblock=块序号\r

参数：　块序号，要擦除的块的序号，每个块包含 8 个 PAGE

应答：　\r\nOK\r\n;\r\nError\r\n

4）.向指定的页面写入字符串,该页面原有内容均被删除掉

命令格式：writepage=页面序号，要写入的字符串

参数：　页面序号：欲写入的页面；

要写入的字符串：命令总长度不能超过 RECVBUF_LEN;字符长度不超过页面大小

应答：　\r\nOK\r\n

5）.读指定页面

命令格式：readpage=页面序号

参数：　页面序号：要读取的页面的序号。

应答：　16 进制的方式显示 264 个字节的内容；同时每个 16 字节后的括号内显示该字符；字节间用空格隔开

5.6 AD/DA

1　程序文件夹：\prog\AD_DA

2　源文件：

- ADDA.c：主文件
- F020.h：头文件

3　编译环境：C8051FXXX IDE，Keil C51

4　功能：测试电路板 AD,DA 功能,实现：

- 使用 ADC0 测量 AIN0 引脚电压，每秒通过串口发送测量的电压和温度值。
- 将 AD 的输入电压在 DA 上输出，经 2 倍放大后驱动 LED1

5　实验条件：

- 底板上 JP1 的 1，2，4，6 设置为 ON 状态
- 底板上 JP2 的 1，2，3，4 设置为 ON 状态

6 注意事项：

- 常量AD_VREF是单片机输出的参考电压,不同单片机会有变化,具体见常量AD_VREF定义处,其值应该与 J1 的 2,3 间电压保持一致
- AD 测量的电压可以通过用万用表测量 J1 的 2,18 间的电压验证

5.7 以太网

1 程序文件夹：\prog\cp2201\java_keil

2 源文件：

- Main.c：主文件
- 其它文件由 TCP/IP 配置向导产生，需要注意的文件有：
  - mn_userconst.h: 程序中的常量定义，主要要注意最后的的 BASE_ADDRESS 和 EMIF_TIMING。前者必须为 0x80,后者必须满足核心板上外部 RAM 和 CP2201 的操作时序要求，不能太快。
  - \VFILE_DIR:该目录中的文件为 HTTP 网页文件
    - Index.c：HTML 网页编译后生成的数组文件
    - Index.h：头文件
    - mn_defs.h/mn_stackconst.h：和主目录下的这两个文件相同。
  - Mn_stack_bank_039.lib/Mn_stack_bank_0393.lib: TCP/IP 配置向导生成的 TCP/IP 程序库。

3 编译环境：Keil uVision3，Keil C51 V8.08

4 功能：从计算机浏览器中访问 http://192.168.0.90,显示数据采集和控制页面

- .可以通过页面控制 LED2
- 页面每两秒自动刷新一次温度和电压数据
- 调解电位器可以观察到网页上的数据变化

5 实验条件：

- 底板上 JP1 和 JP2 的所有位都要设置为 ON 状态

6 注意事项：

- 计算机与核心模块通过交叉网线连接(或两者都使用直联网线连接到 HUB 或路由器上)
- 计算机的 IP 地址必须为 192.168.0.XXX,其中 XXX 可以为 2-254 中的数据,但不能为 90(因为核心模块为 90)
- 常量AD_VREF是单片机输出的参考电压,不同单片机会有变化,具体见常量AD_VREF定义处,其值应该与 J1 的 2,3 间电压保持一致，AD 测量的电压可以通过用万用表测量 J1 的 2,18 间的电压验证
- Main.c 中定义的函数 mn_init_m()是替换向导中生成的 mn_init()函数的，由于目前 TCP/IP 向导生成的代码中有关 C8051F020 的 mn_init()函数存在问题导致程序下载到芯片中后无法上电执行。用户在自己使用 TCP/IP 向导生成新的工程后也要将 main()函数中调用 mn_init()的地方改为 mn_init_m()函数。
- 编译器注意事项：
  - 使用 KEIL C8.0 以上版本 C51 编译器
  - .使用 LARGE RAM 模式
  - 连接选项不使用覆盖分析,即增加连接选项 NOOL

5.8 串口转 TCP

1 程序文件夹：\prog\cp2201\tcp_uart

2 源文件：

- Main.c：主文件
- 其它文件由 TCP/IP 配置向导产生，需要注意的文件有：
  - mn_userconst.h: 程序中的常量定义，主要要注意最后的的 BASE_ADDRESS 和 EMIF_TIMING。前者必须为 0x80,后者必须满足核心板上外部 RAM 和 CP2201 的操作时序要求，不能太快。
  - Mn_stack_bank_039.lib/Mn_stack_bank_0393.lib: TCP/IP 配置向导生成的 TCP/IP 程序库。

3 编译环境：Keil uVision3，Keil C51 V8.08

4 功能：模块作为 TCP 客户端自动与 192.168.0.2 的 2000 端口进行 TCP 连接，并将从串口接收到的数据发送到 TCP 服务器上；同时将 TCP 服务器发送过来的数据发送到串口上。因此，该试验实现了串口与以太网的数据透明传输。

- 如果 TCP 服务器没有启动，则模块会每间隔 2 秒自动进行一次连接；
- 实现条件下每次连续发送的数据长度不能超过 100。（常量定义的原因，可以增加很大的）
- 两次发送的数据间隔小于 1s 时有可能丢失数据。（实验程序为简化而没有处理这些问题，但他们是可以解决的）

5 实验条件：

- 底板上 JP1 和 JP2 的所有位都要设置为 ON 状态
- 计算机使用串口调试和 TCP 调试的软件。

6 注意事项：

- 计算机与核心模块通过交叉网线连接(或两者都使用直联网线连接到 HUB 或路由器上)
- 计算机的 IP 地址必须为 192.168.0.2,TCP 调试软件应该监听 2000 端口。
- 模块程序还可以实现作为 TCP 服务器、UDP 传输等串口与以太网的转换功能。
- Main.c 中定义的函数 mn_init_m()是替换向导中生成的 mn_init()函数的，由于目前 TCP/IP 向导生成的代码中有关 C8051F020 的 mn_init()函数存在问题导致程序下载到芯片中后无法上电执行。用户在自己使用 TCP/IP 向导生成新的工程后也要将 main()函数中调用 mn_init()的地方改为 mn_init_m()函数。
- 编译器注意事项：
  - 使用 KEIL C8.0 以上版本 C51 编译器
  - .使用 LARGE RAM 模式

连接选项不使用覆盖分析,即增加连接选项 NOOL

6 以太网开发参考

FM020 的具有强大的以太网功能，利用免费的 TCP/IP 库，基于开发向导，用户可以实现：

- 基于 TCP 的用户自定义通信
- 基于 TCP 的 MODBUS 协议
- 基于 UDP 的数据传输
- 嵌入式 HTTP 服务器
- 嵌入式 SMTP 协议，实现收发 e-mail
- 嵌入式 FTP 客户和服务器
- 嵌入式 telnet 服务器

FM020 为用户开发提供了丰富的文档和软件。有关文档如下：

- 有关嵌入式以太网开发的基础知识，请参考文档：\手册\参考文档\AN237.PDF,其中对使用 silab TCP/IP 库进行编程开发进行了详细的描述。
- 有关 TCP/IP 库的使用方法，请参考文档：\手册\参考文档\AN292.pdf,其中详细描述了 TCP/IP 库的使用方法。

相关的软件有：

- TCP/IP 生成向导：软件位于：\software\ethernet\TCPIPConfigWizInstaller V3.21,使用 TCP/IP 向导可以生成以太网程序框架，具体使用方法在 AN292 的第 6 部分有讲述。

6.1 FM020 以太网开发流程

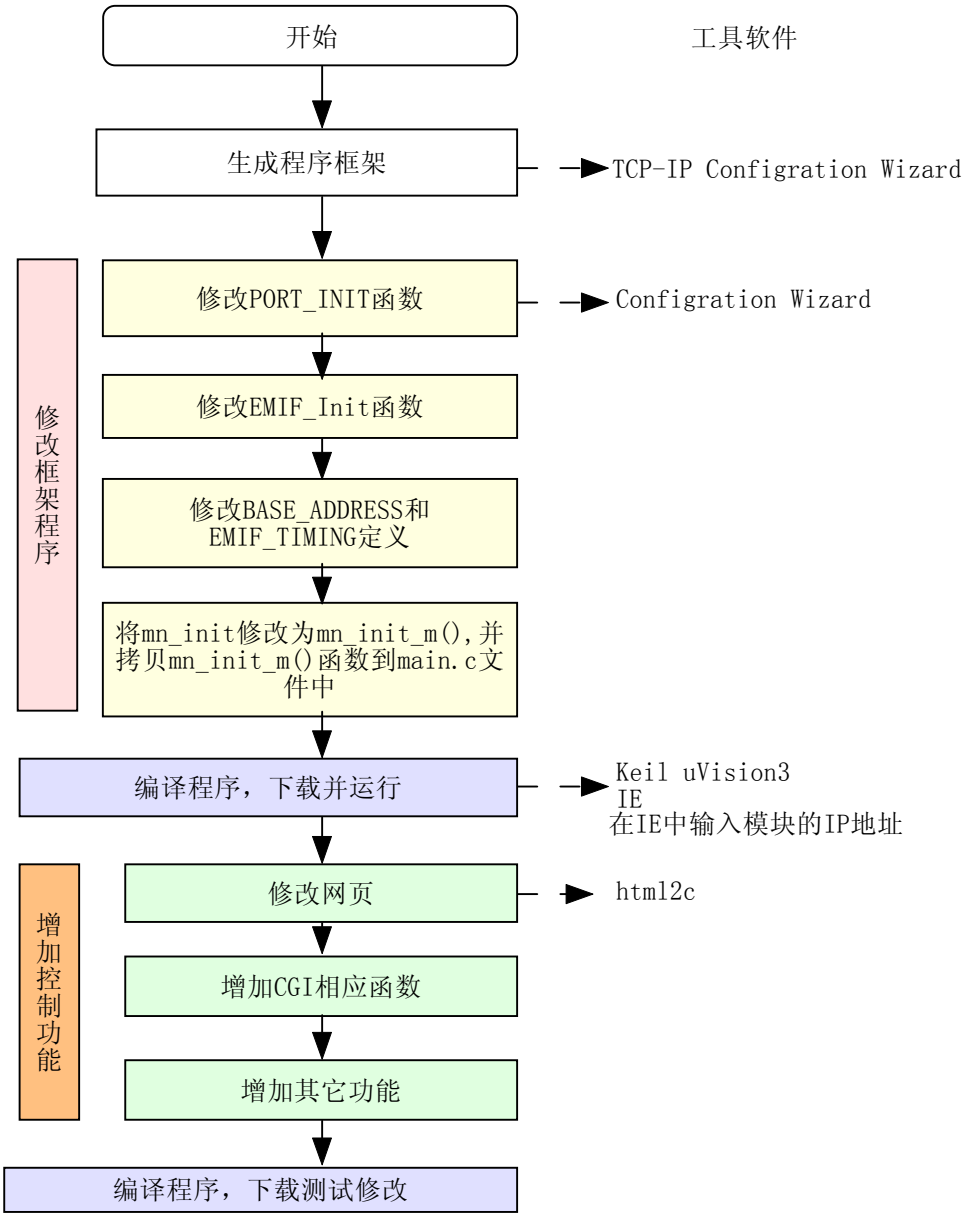


图 6.1-1 FM020 以太网开发流程

6.1.1 生成程序框架

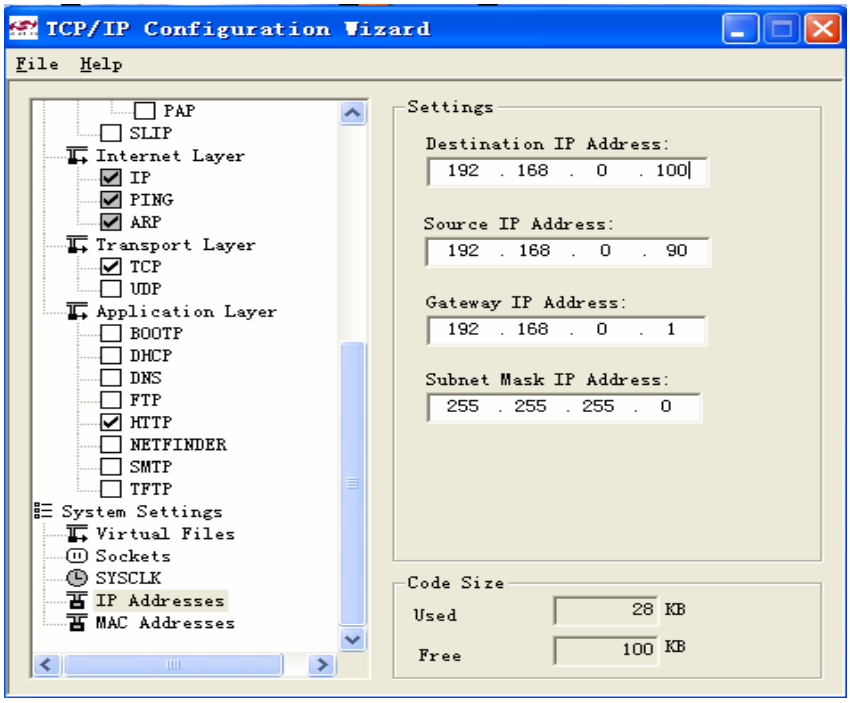
使用 silab 的 TCP-IP Configraion wizard 生成网络程序的框架。有关该向导的具体使用方法可以参考文献 1。下面以生成一个 HTTP 服务器为例说明：

1. Communication Adapter 选择 CP220x,Device Selection 选择 C8051F02x, Link/Physical Layer 选择 Ethernet。
2. Application Layer 选择 HTTP 协议（此时 Transport Layer 的 TCP 会自动被选择，一

定要保证 TCP 协议处于选中状态）。

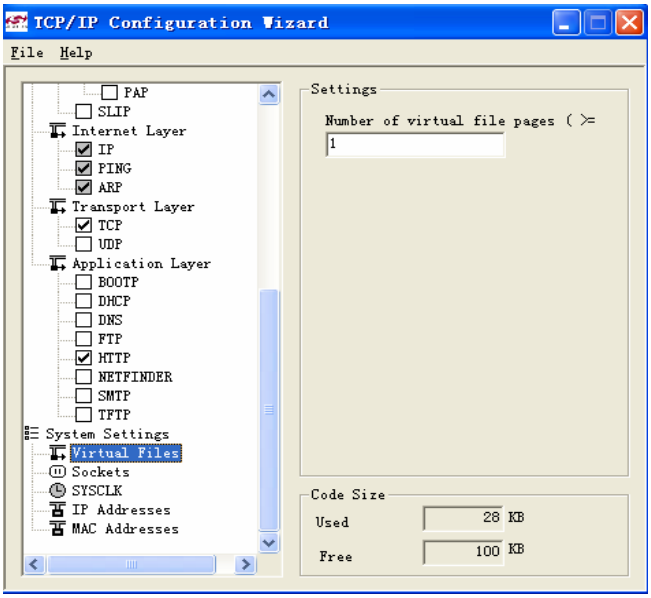
3. SYSCLK 中的系统时钟应该为 22.1184MHZ

4. IP Address 如下图设置：



其中，Source IP Adress 为模块的 IP 地址，Gateway IP Address 为网关 IP，必须为 192.168.x.1,在使用 HTTP 协议时，Destination IP Adress 无效。此处设置的参数也可以在生成的工程中的 mn_userconst.h 中修改。

5. Virtual Files 中设置的为程序中虚拟文件的个数，在 HTTP 服务器中就是使用的网页的个数，可以根据需要设置，增加该量将会增加 RAM 的使用量。



6. 选择菜单 File->Generate Project…生成工程即可。

6.1.2修改框架程序

向导生成的为 C8051FXXX IDE 工程文件，若用户要在该程序中编译程序，则要保证设置的编译器为 Keil C51 V8.0 及其以上版本，且必须为正式版本。如果用户要将工程添加到 Keil uVision3 中，则需要将生成的所有文件（包括两个.lib 文件）都加入到工程中。

1. 修改 PORT_INIT

函数 PORT_INIT 定义了单片机的引脚分配和数据总线的位置，用户必须根据核心模块与底板间的连接方式来修改该函数。可以使用 Configration Wizard 软件来生成该函数。配置引脚时需要注意一下几点：

- 数据总线在高端口，且使用复用方式
- 使用验证板时，INT0 必须被分配到 P22
- 使用验证板时，P30 与 CP2201 的 RST 连接
- P4-P7 端口要设置为输出状态

考虑到以上几点，交叉开关的分配必须使能 UART0,SPI0,SMBUS0,UART1,并且定义 P1 口为模拟端口,用户可以参考示例程序中的函数代码。

2. 修改 EMIF_Init()函数

EMIF_Init 函数中需要修改 EMI0CF = 0x2B,即设置外部数据端口的位置。

3. 修改常量

修改 CP2201 的地址空间位置和数据总线的接口时序。这两个常量在 mn_userconst.h，常量要按如下方式定义：

```
#define    BASE_ADDRESS      0x80
#define EMIF_TIMING              0xFF
```

此处定义的 EMIF_TIMING 为最慢的速度，可以调整该值，但必须满足 SRAM,TRAM,和 CP2201 三个设备的接口时序。

4. 修改 mn_init（）函数调用

由于目前 TCP/IP 向导生成的库函数中 mn_init 存在有关 C8051F020 的问题，导致程序必须在使用调试器复位后才能运行，因此，开发套件中提供了一个 mn_init_m()函数来替换该函数。用户在 main 函数中原来调用 mn_init 的地方修改为调用 mn_init_m 函数，并将开发套件示例程序中的 mn_init_m()函数复制到 main.c 文件中。注意，复制函数时应该连同函数前面的头文件包含、变量声明及函数声明一起复制，该函数应该在 main 函数前面。

```
//-----------------------------------------------------------------------------------------------------------
// mn_init_m 修正函数使用的变量声明,复制函数时应包括这些内容
//-----------------------------------------------------------------------------------------------------------
#include <string.h>
extern byte code NUM_SOCKETS;
extern byte code DEVICE_ID;
extern byte code NUM_VF_PAGES;
extern byte code NUM_POST_FUNCS;
int    mn_ip_init();
void mn_timer_init();
void mn_http_init();
void mn_arp_init();
```

6.1.3增加控制功能

1. 修改网页

使用向导生成的基本工程中包含了一个静态的网页，用户可以修改该网页以增加采集、控制功能。

在工程目录下的 VFILE_DIR 目录中的 index.html 文件即为核心模块显示的网页，用户可以使用各种网页编辑工具编辑该网页，然后使用该目录下的 html2c.exe 程序将网页转换为 index.c 和 index.h 文件。

- 与我们熟悉的服务器相同，嵌入式 HTTP 服务器的默认网页的名称也必须是 index.html，这是用户访问该 IP 地址的默认页面。
- 如果用户要增加多个网页，则需要首先编辑这些网页，使这样网页可以在计算机上的浏览器中正常访问，然后使用 HTML2C.exe 转换所有的这些网页。最后，在 main 函数中的如下部分添加这些网页到虚拟文件中：

```
// Connect to the network
establish_network_connection();

// Add web page to virtual file system.
// The main page MUST be called index.htm or index.html.
mn_vf_set_entry((byte *)"index.html", INDEX_SIZE, index_html, VF_PTYPE_FLASH);
//  添加第二个网页
mn_vf_set_entry((byte *)”voltage.html”,VOLTAGE_SIZE,voltage_html,VF_PTYPE_FLASH);
```

注意，此时必须修改 mn_userconst.h 中的 num_vf_pages 常量的定义，该值应该大于等于网页的个数。

- 应该注意检查生成的程序的大小，过多的网页会占用大量的程序空间。

2. 增加数据传输和控制功能

建立一个 HTTP 服务器的目的一定是要实现参数设置、数据采集、控制的，所以在实际使用的过程中一定不会简单的使用静态网页的。在 HTTP 服务器中，实现浏览器与服务器数据交换的接口主要是 CGI（Common GateWay Interface）。如果要实现浏览器端数据的自动更新，则用户需要在网页代码中添加 java script 代码来控制浏览器自动刷新数据。有关 HTML 和 java script 的相关知识，用户可以参考其它相关资料，以下以示例程序中的采集和控制功能来解释实现方法。有关这些操作的具体实现方法请参考文档[1]。

我们要实现通过网页来控制验证板上 LED 的状态，并要求浏览器能够每 2 秒自动更新一次测量的电压和温度数据。

1）网页编辑

要实现浏览器定时刷新数据的功能，我们在网页中增加了如下的 java script 函数：

```
<script type="text/javascript">
var delay=2000
var sel = 1;
function loadframes()
{
    t = 0;

    if(sel){
        document.getElementById("temperature").src="get_data?type=temp&format=html&";
        sel = 0;
    }else{
        document.getElementById("voltage").src="get_data?type=voltage&format=html&";
        sel = 1;
    }
    setTimeout("loadframes()", delay);
}

</script>
```

上文中，蓝色部分控制着函数执行的周期和动作。setTimeout 控制函数 loadframes 每 2 秒执行一次。在 loadframes 函数中，轮流执行红色部分的代码。其中定义了两个 ID 分别为 temperature 和 voltage 的子窗体，他们的执行的操作分别是其后的 src=””中的代码。

下面的代码中，分别定义了两个 IFRAME 来显示 temperature 和 voltage：

```
<table width="100%"      cellspacing=0 border=0    cellpadding=0    bgcolor="blue">

 <tr>
<!-------------------------------------------------------------------------------------------------------->
   <td>
     <span  style="font-family:  sans-serif;  font-size:  28pt;  font-weight:  bold;  vertical-align: middle;">
     <center>  温度  <br> (deg C):<br>
     <IFRAME SRC="about:blank" id="temperature" FRAMEBORDER="0"></IFRAME>
     </center>
     </span>
   </td>
<!-------------------------------------------------------------------------------------------------------->
   <td>
     <span  style="font-family:  sans-serif;  font-size:  28pt;  font-weight:  bold;  vertical-align: middle;">
     <center>
     外部测量电压:<br>(V):<br>
     <IFRAME SRC="about:blank" id="voltage" FRAMEBORDER="0"></IFRAME>
     </center>
     </span>
   </td>
<!-------------------------------------------------------------------------------------------------------->
 </tr>
 </table>
```

可以看到，两条蓝色线中间的部分分别是一个 IFRAME,他们的 ID 即为上文中 document.getElementById 的参数，这使得这两个子窗体显示的内容每 2 秒轮流被刷新显示。他们发送的 CGI 参数为 get_data,读取温度时参数为 type=temp,而读取电压时为

type=voltage。所以我们在核心模块的程序中增加 CGI 函数 get_data 后，处理这两种参数就可以了。

有关控制 LED 的部分 HTML 代码为：

```
<form action="set_led" target="_blank">
    <span style="font-family: sans-serif; font-size: 18pt; font-weight: bold; vertical-align: middle;">
        <center>
        控制 LED:
        <input type="radio" name="led" value="0" checked >打开 LED
        <input type="radio" name="led" value="1">关闭 LED
        <input type="submit" value="执行">
        </center>
    </span>
</form>
```

这个 form 中的按钮被单击时将提交 set_led 的命令，根据单选按钮的状态带 led=0 或 led=1 的操作。

2）增加 CGI 函数

在网页中添加这些内容后，我们需要在 HTTP 服务器中添加 CGI 函数来相应客户端的请求。

- 首先，在 main 函数中在添加 mn_vf_set_entry 后添加如下代码：

```
mn_pf_set_entry(
  (byte*)"get_data",              // Script Name (ASCII)
  get_data                        //  函数指针，get_data 函数处理客户读取温度和电压的请求。
);
mn_pf_set_entry(
  (byte*)"set_led",                 // Script Name (ASCII)
  set_led                          //  函数指针，set_led 函数处理客户设置 led 的请求。
);
```

注意，应该在 mn_userconst.h 中修改 num_post_funcs 的值，该值应大于等于使用 mn_pf_set_entry 添加的项目。

- 其次，添加 get_data 和 set_led 函数

```
void get_data (PSOCKET_INFO socket_ptr);
```

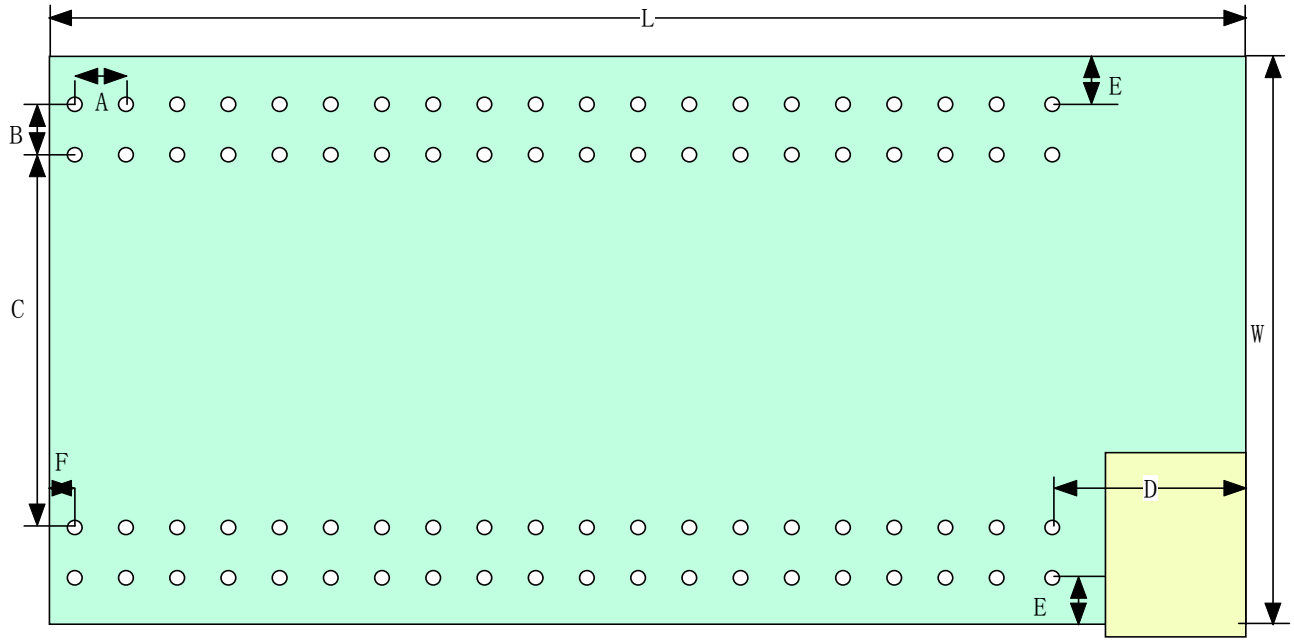
void set_led(PSOCKET_INFO socket_ptr);

有关这两个函数中的具体代码，可以见示例程序，函数中用到的 TCP/IP 库中的函数也可以在参考文档[1],[2]中找到。

7 参考文档

[1] SILABS, AN292,EMBEDDED ETHERNET SYSTEM DESIGN GUIDE, REV.0.2 6/06
[2] SILABS,AN237,TCP/IP LIBRARY PROGRAMMER’S GUIDE,REV.0.6 6/06
[3] SILABS,CP2200/1 SINGLE-CHIP ETHERNET CONTROLLER,REV.0.2 1/06

附录 A 核心模块机械尺寸图



尺寸表格：

参数	尺寸(mm)	尺寸(mil)
A	2.00	78.74
B	5.30	208.66
C	26.96	1061.34
D	20.00	788.97
E	2.18	85.67
F	1.88	73.97
L	59.92	2359.00
W	49.91	1650.00

附录 B 验证版电路图