京微齐力
Hercules Microelectronics

HME-M1 系列

数据手册

2018 年 10 月

京微齐力（北京）科技有限公司

一些产品信息来自京微雅格

注意

手册版本号

HME-M1DSC04

联系我们

如果您在使用我们的产品过程中有任何疑问或问题，请与京微齐力（北京）科技有限公司联系，或发送邮件至：
sales@hercules-micro.com

声明

本手册中包含的信息已经仔细检查并认为是完全可靠的。但是，不对手册中可能或潜在的错误负责。京微齐力（北京）科技有限公司保留停止发布或修改手册而不事先通知的权利。为确保获得最新的产品信息，建议用户及时更新手册版本。

本手册介绍的产品并没有被授权用作为生命保障设备或系统中的关键部件。在此使用到的术语有如下定义：1.生命保障设备或系统是满足以下条件的设备或系统，(a)被通过手术植入人体内或（b)用来保障或维持生命，当按照标签上的使用说明正确使用时，有理由认为其工作的中断将会给使用者带来巨大的伤害。2.所谓关键部件是指生命保障设备或系统中满足以下条件的部件，即我们有理由认为该部件中断工作将会导致整个生命保障设备或系统中断工作，或者是影响到后者的安全性和有效性。

环境保护

本产品中包含的某些物质可能会对环境或人体健康有害，为避免将有害物质释放到环境中或危害人体健康，建议采用适当的方法回收本产品，以确保大部分材料可正确地重复使用或回收。有关处理或回收的信息，请与当地权威机构联系。

版本控制

发布日期	版本	修订记录
2011 年 5 月	1.0	首次发布
2012 年 8 月	1.1	■ 表 3-35：将 ckcon 6 修改为 ckcon 2； ckcon 5 修改为 ckcon 1； ckcon 4 修改为 ckcon 0。
2013 年 5 月	CME-M1DSC03	■ 统一命名规则 ■ 手册改版 ■ 修订“**配置模式**”图 ■ 将器件名称由 Astro Ⅱ 更改为 CME-M1 ■ 更新“**封装信息**”等
2014 年 3 月	CME-M1DSC04	■ 新增“**QFN-68 封装引脚列表**”和“**QFN68 封装规格**”； ■ 更新“**HME-M1 选型表**”； ■ 更新“**LQFP144 封装规格**”和“**TQFP100 封装规格**”尺寸； ■ 更新温度范围： 商业级由(TJ = 0℃to 85℃)更改为：(0℃to 85℃)；工业级由（TJ = -40℃to +100℃)更改为：(-40℃to 100℃) ■ 更新“**器件型号实例**”。
2018 年 10 月	HME-M1DSC05	■ 更新“**器件型号实例**”。

目录

注意 **1**

版本控制 **2**

目录 **3**

开始前准备 **6**

关于本手册 6

1 概述 **7**

1.1 HME-M1FPGA 简介 7
1.2 HME-M1 FPGA 主要功能 7
1.3 HME-M1 系统示意图 9
1.4 HME-M1 选型表 9

2 FPGA 特性 **11**

2.1 FPGA 概述 11
2.2 现场可编程逻辑 12
2.3 绕线资源 14
2.4 嵌入式存储器模块 EMB9K 14
2.4.1 emb9k_tdp 15
2.4.2 emb9k_sdp 16
2.4.3 emb9k_sp 18
2.5 I/O 19
2.5.1 I/O 特性 19
2.5.2 I/O 描述 19
2.6 时钟网络 20
2.6.1 时钟网络通路 20
2.6.2 PLL 22
2.6.3 PLL 接口 22
2.6.4 PLL 属性 23

3 MSS 子系统 **26**

3.1 8051 CPU 27
3.1.1 寻址方式 27
3.1.2 指令集 28
3.1.3 寻址方式助记 28
3.1.4 按功能划分的指令集 29
3.1.5 十六进制编码排序指令 32
3.1.6 读-改-写指令 34
3.1.7 扩展指令 34
3.1.8 指令周期计算 35

3.2 存储器 35
3.2.1 内存和 SFR 35
3.2.2 RAM 存储器 36
3.2.3 存储 Banking 37
3.2.4 外部存储器接口（EMIF） 38
3.3 特殊功能寄存器（SFRs） 39
3.3.1 SFR 存储单元 39
3.3.2 SFR 复位值 40
3.3.3 SFR 说明 42
3.4 扩展特殊功能寄存器 63
3.5 乘-除单元 MDU 65
3.5.1 概述 65
3.5.2 MDU 操作说明 66
3.6 中断控制器 68
3.6.1 中断源 68
3.6.2 中断优先级 70
3.6.3 中断源概要 70
3.7 定时器 71
3.7.1 定时器 0 71
3.7.2 定时器 1 73
3.7.3 定时器 2（带有比较/捕获单元） 74
3.8 看门狗 76
3.8.1 概述 76
3.9 I2C 77
3.9.1 I2C 概述 77
3.9.2 说明 77
3.10 SPI 85
3.10.1 概述 85
3.10.2 说明 86
3.11 UART 87
3.11.1 UART0 87
3.11.2 UART1 89
3.12 P 端口 89
3.12.1 概述 89
3.12.2 端口引脚的多路复用 90
3.13 电源管理 91
3.13.1 节电方式 91
3.13.2 退出节电方式 91
3.14 MSS 系统控制 91
3.14.1 MSS 系统时钟电源管理 91
3.14.2 MSS SPI 操作 92
3.14.3 MSS I^2C I/O 设置 92
3.14.4 MSS 在系统重配置 92
3.14.5 MSS 在系统更新 93

3.15 8051 实例化 93
3.15.1 8051 宏模块端口列表 94
4 配置 96
4.1 配置模式 96
4.1.1 AS 模式 96
4.1.2 PS 模式 97
4.1.3 JTAG 模式 98
4.2 非易失性存储器 98
4.3 ISC 99
4.4 安全 99
4.4.1 配置数据加密 99
4.4.2 FLASH 的操作控制 99
4.5 HME-M1 上电复位 99
5 电气特性 101
5.1 LVCMOS33 D.C. 规格 101
5.2 LVTTL33 D.C. 规格 101
6 引脚及封装 102
6.1 引脚定义 102
6.2 引脚列表 103
6.2.1 TQFP-100 封装引脚列表 104
6.2.2 LQFP-144 封装引脚列表 105
6.2.3 QFN-68 封装引脚列表 107
6.3 封装信息 109
6.3.1 LQFP144 封装规格 109
6.3.2 TQFP100 封装规格 110
6.3.3 QFN68 封装规格 111
7 开发工具 112
8 商务指南 113
8.1 HME-M1 系列产品列表 113
8.2 产品命名规则 113
9 缩写 115

开始前准备

关于本手册

本手册只是 **HME-M1 系列 FPGA** 所有手册中的其中一个。旨在帮助用户了解并查知该系列器件的核心功能及参数。

如需了解产品其它信息，请登录 http://www.hercules-micro.com。

1 概述

1.1 HME-M1 FPGA 简介

HME-M1 是一款集成了增强型 8051 处理器硬核和 FPGA 等资源于一体的智能型器件，能够实现完全可定制系统设计和 IP 保护能力，而且易于使用。设计者可以便捷地利用 HME 的 Fuxi 进行 FPGA 设计，支持第三方 EDA 工具 Keil ™ 进行嵌入式设计。HME-M1 的单芯片系统比传统专属功能微控制器具有更大的灵活性、比现有使用软核处理器的 FPGA 具有更低的成本。HME-M1 的 ISP 功能可以实现在系统更新配置镜像，ISC 功能可以实现在系统用其他配置镜像重新配置 HME-M1。HME-M1 系列产品广泛应用于工业、医疗设备、通信系统和消费类电子等多种应用领域。

1.2 HME-M1 FPGA 主要功能

- **基于 SRAM 的 0.13 微米工艺 FPGA Fabric**
  - 4 输入查找表+基于 DFF 的可编程逻辑单元 LC
  - 1024 个可编程逻辑单元
  - 专用的算术进位链
  - 层次化结构布线资源
- **嵌入式 RAM Block 存储器**
  - 2x9Kbit 可配置双端口 DPRAM 存储器 EMB9K
- **时钟网络**
  - 8 个 De-skew 全局时钟
  - 1 个支持频率合成、相移、De-skew 的 PLL（输入时钟频率范围：5～350MHz、输出时钟频率范围：10～350MHz）
  - 4 个外部时钟输入、1 个外部振荡反馈时钟输入
- **I/O 和操作电压**
  - 3.3V LVCMOS/LVTTL
  - 支持 5V 输入
  - 最高可达 150MHz
  - I/O 属性可编程
  - 3.3V I/O 电压、1.2V 内核电压

微控制子系统（MSS）

- **增强型 8051MCU**
  - 精简指令周期，12 倍于标准 8051 的 MIPS，频率最高可达 150MHz
  - 兼容标准 8051 的指令系统
  - 硬件支持乘法、除法指令
  - 支持扩展指令：MOV A,ACC
  - 支持 16 位乘法、32/16 位除法硬件
  - 片上调试系统 OCDS，支持 JTAG 在线调试

  - 8 个外部中断源，总共 13 个中断源
  - 支持高达 8M 数据/代码存储器
- **嵌入式 SRAM 存储器**
  - 32KByte 单端口存储器 SPRAM
  - 4KByte 双端口 DPRAM（Fabric 连 B 端口，MSS 连 A 端口）
  - 数据/代码统一编址、存储大小灵活配置
- **外设**
  - 3 个 16 位定时时器，定时器 2 可用作比较捕获单元
  - 1 个 16 位硬件看门狗
  - 1 个 I^2C 接口
  - 1 个 SPI 接口，可控制 4 个从设备
  - 2 个全双工异步串行接口
- **停止、空闲模式电源管理**
- **芯片系统管理**
  - ISC 控制
  - ISP 控制
  - 系统动态频率切换
  - 系统动态 PLL 控制
  - 系统 FPGA 时钟关断控制

配置

- **配置模式**
  - JTAG 模式
  - AS 模式
  - PS 模式
- **JTAG 接口**
  - JTAG 芯片配置
  - JTAG 8051 调试
  - 芯片配置 8051、调试共用同一 JTAG
- **支持多映像配置文件**
- **可选 4Mbit 内部 SPI-FLASH**
- **ISC**
- **安全机制**
  - 配置文件数据加密
  - 基于密钥的 SPI 操作保护
  - 访问保护机制
  - 配置 memory①保护

① Memory 包括 MSS 8051 的 RAM code

1.3 HME-M1 系统示意图

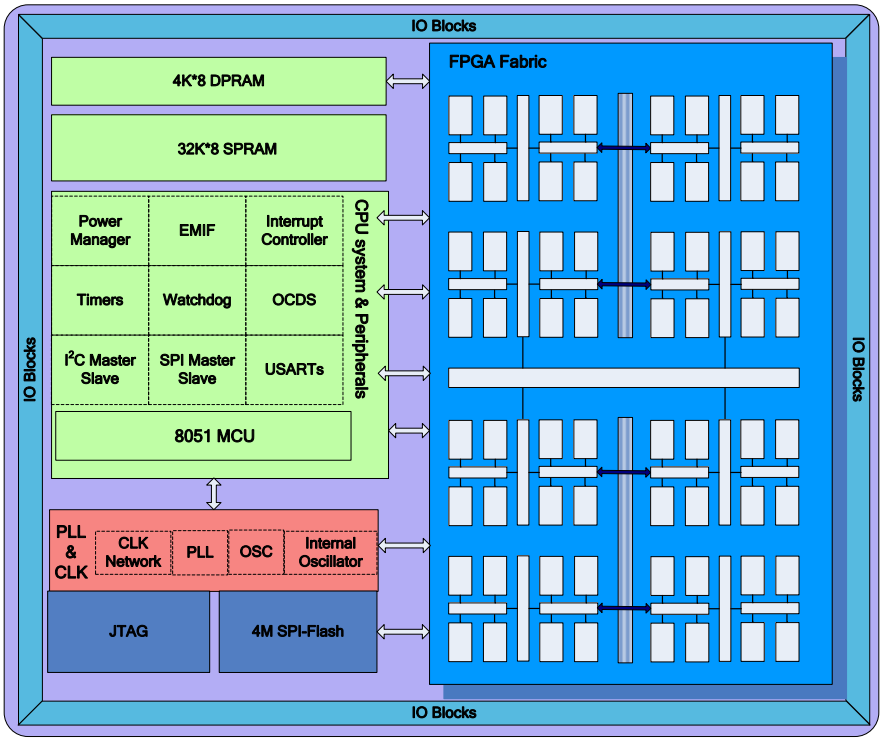


图 1-1 HME-M1 系统示意图

1.4 HME-M1 选型表

表 1-1 HME-M1 系列产品列表

Part Number		M1C01N3	M1C01N0
FPGA Fabric	LC	1024	1024
	RAM Blocks	2	2
MSS	16-bit Timer	3	3
	Memory	32K+4KByte	32K+4KByte
	Watch Dog Timer	1	1
	I2C	1	1
	SPI	1	1
	USART	2	2
SPI Flash		4Mbit	
I/Os		111	111
Speed		7	7
Temp		Commercial (0℃, 70℃) Industry (-40℃, 100℃)	Commercial (0℃, 70℃) Industry (-40℃, 100℃)
Package		Max User IO	
LQFP100		71	71
LQFP144		111	111
QFN68		48	48

2 FPGA 特性

2.1 FPGA 概述

下图为 HME-M1 整个 FPGA 的结构。嵌入式存储器模块 EMB9K 内嵌在 Fabric 中，MSS、GCLK 时钟网络以及 IOB 等通过绕线资源联到 Fabric 上。图中的“连接”表示绕线资源。

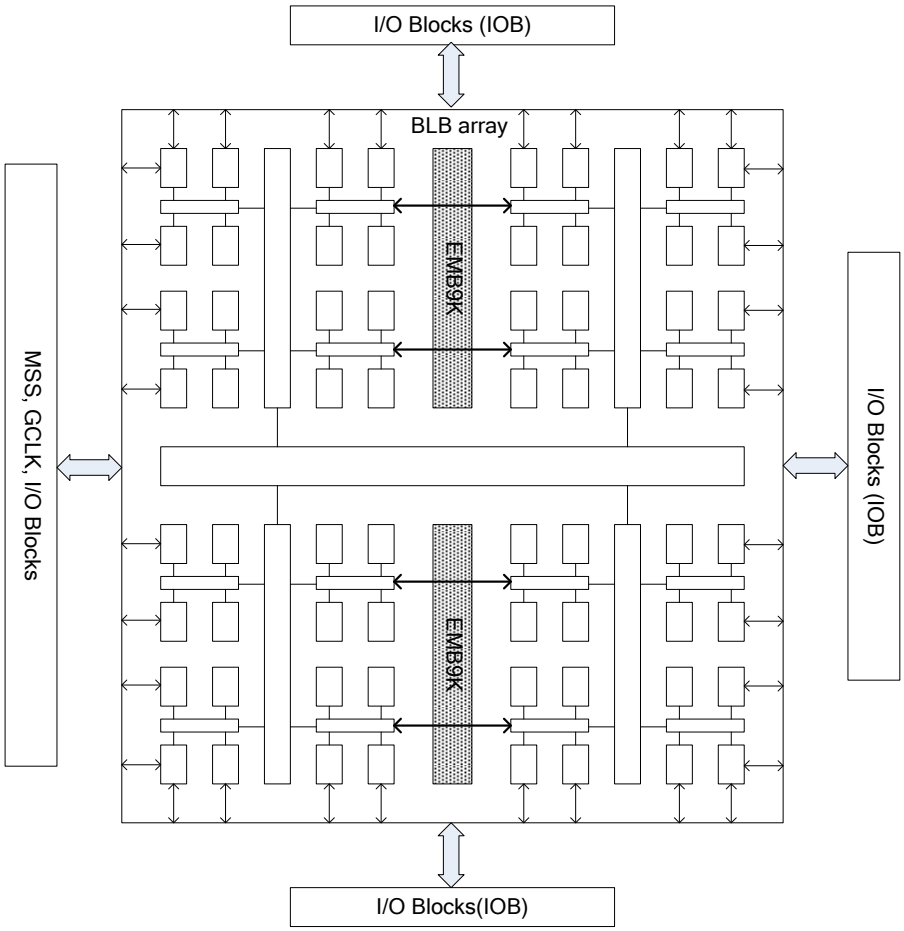


图 2-1 FPGA 概况

FPGA Fabric 的基本模块是 BLB（Basic Logic Block）。64 个 BLB 阵列，分四个层次通过上下绕线资源和和跨层次的进位链资源连接成整个 FPGA Fabric。16 个逻辑单元 LC（Logic Cell）组成一个 BLB，即 HLB1，4 个 HLB1 组合为一个 HLB2。同理，4 个 HLB2 组合成 1 个 HLB3。HME-M1 包含 4 个 HLB3，可以视为一个 HLB4,共计 1024 个逻辑单元（64 个 BLB）。

HME-M1 四周分布 IOB（I/O Block），它们通过交叉互联线与 FPGA 逻辑阵列相连接。芯片的左边有 7 个 IOB，其余三边都有 8 个 IOB。每个 IOB 包含 4 个 IOC（I/O Cell）。

2.2 现场可编程逻辑

现场可编程逻辑单元（LC）作为现场可编程块（图 2-2）中最小的单元，有如下特征：

- 一个 4 输入查找表可以实现四个变量的任何逻辑功能
- 进位链的特点可以用于加法器/减法器
- 或非门链和 WLUT 链可以扩展 LUT 功能
- 一个可编程寄存器
- 支持寄存器反馈回 LUT

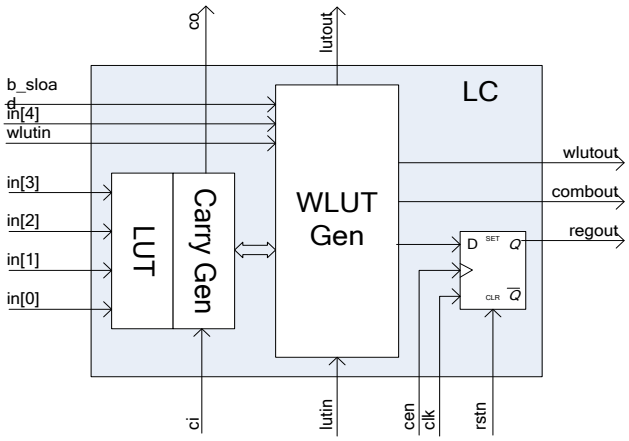


图 2-2 现场可编程逻辑单元

下图为 LC 的串联图。进位链经由进位链输入输出信号串联在一起。与非门链经由 lutin 和 lutout 串联；WLUT 链经由 wlutin 和 wlutout 串联。

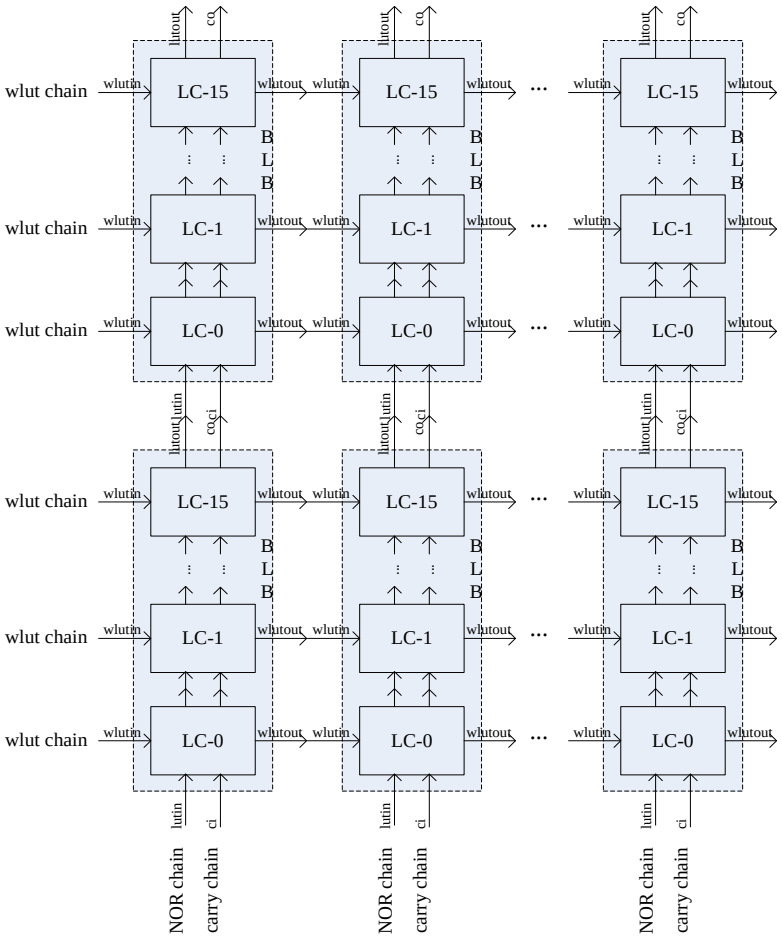


图 2-3 进位链，NOR 链和 WLUT 链

基本逻辑单元块（BLB）是由 16 个 LC 和 BLB_ctrl 块组成的。BLB_ctrl 为 16 个逻辑单元提供：clk,cen,rstn 和 sload，四个公用信号。图 2-4 描述了 BLB 的构造。

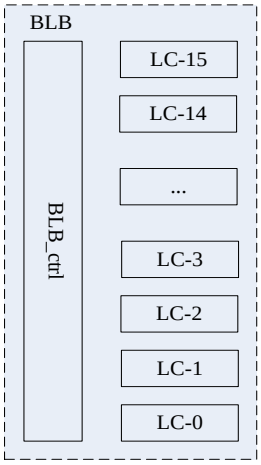


图 2-4 BLB 和 LC

2.3 绕线资源

绕线资源是由一个以四叉树方式绕线多路复用结构和交联绕线结构组成的。分层四叉树绕线资源为所有 BLB 提供充足的绕线路径，其中 BLB 是树的最小节点并且包含 16 个 LC。交联绕线资源在 BLB 之间提供快捷的绕线路径。图 2-5 描述了简化的交联绕线资源和分层绕线资源。

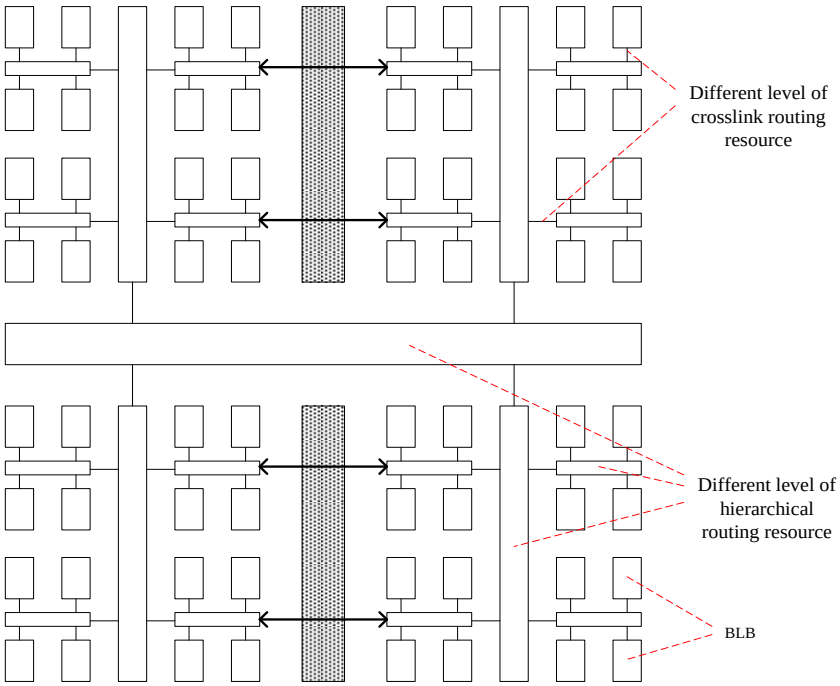


图 2-5 绕线资源

2.4 嵌入式存储器模块 EMB9K

HME-M1 中共 2 个 EMB9K 块。EMB9K 模块是一个同步真双端口存储器，具有如下特征：

- **9,216 bits**
- **混合时钟方式**
- **支持字节使能**

写端口数据宽度为 8、9、16、18、32 或 36 位时，EMB9K 支持字节写操作。字节使能允许屏蔽输入数据，这样 EMB9K 可以写入特定字节。未写入的字节保持之前写入的值。

表 2-1 支持字节的EMB9K 块

we [3..0]	d [15..0]	d [17..0]	d [31..0]	d [35..0]
[0] = 1	[7..0]	[8..0]	[7..0]	[8..0]
[1] = 1	[15..8]	[17..9]	[15..8]	[17..9]
[2] = 1	–	–	[23..16]	[26..18]
[3] = 1	–	–	[31..24]	[35..27]

- **奇偶校验位**
  - 支持奇偶校验码

EMB9K 模块的每个字节都有奇偶校验位，但需要逻辑配合实现。奇偶校验位也可用于存储用户自定义的控制位。

- **A、B 数据宽度可独立配置**
- **支持直通或寄存器读输出**

直通读（同步—1 clock)：在读有效的情况下，读的数据在同一个时钟周期里驱动到读数据总线 q 上。

寄存器读（同步—2 clock)：在读有效的情况下，内部读的数据在下一个时钟周期里放到输出寄存器里并驱动到读数据总线 q 上。

通过设置 output_mode，可以选取直通读模式或寄存器读模式。在寄存器读模式下，可通过 is_clk_qx_inverted 参数来选择时钟上升沿读或时钟下降沿读。

- **支持初始化**

初始化文件格式为.hex 或.dat（每行一个 16 进制数，行数为 EMB9K 的深度）。初始化文件在配置过程中初始化 EMB9K 的存储器。

- **Memory 模式**

依据应用模式可配置成如下几种模式：

- emb9k_tdp
- emb9k_sdp
- emb9k_sp

在 emb9k_tdp、emb9k_sdp 模式下不能同时读写同一个地址。

2.4.1 emb9k_tdp

EMB9K 支持任何组合的双端口操作：不同的端口时钟下的两端口读，两端口写，或一端口读一端口写。图 2-6 显示了真双端口存储配置。

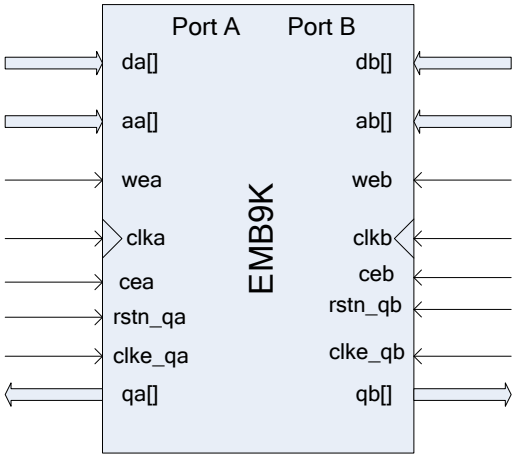


图 2-6 真双端口存储方式

表 2-2 真双端口存储方式端口描述

端口名称	类型	描述
aa (b)	Input	Port A (B) Address.
da (b)	Input	Port A (B) Data Input.
qa (b)	Output	Port A (B) Data Output.
wea (b)	Input	Port A (B) Write Enable. Data is written into the dual-port SRAM upon the rising edge of the clock when both wea (b) and cea (b) are high.
cea (b)	Input	Port A (B) Enable. When cea (b) is high and wea (b) is low, data read from the dual-port SRAM address aa (b). If cea (b) is low, qa (b) retains its value.
clka (b)	Input	Port Clock.
rstn_qa (b)	input	read register reset，low active
clke_qa (b)	input	read clock enable，high active

表 2-3 真双端口存储器方式直通模式下的真值表

输入			输出	
cea(b)	wea (b)	clk	Status	qa (b)
0	X	X	HOLD	Data stored in the memory is retained.
1	0	↑	READ	Data is read from the memory location specified by the address bus.
1	1	↑	WRITE	da (b)

表 2-4 真双端口可能的数据宽度组合

Port A	Port B							
	8K × 1	4K × 2	2K × 4	1K × 8	512 × 16	9 K × 1	1K × 9	512 × 18
8K × 1	√	√	√	√	√			
4K × 2	√	√	√	√	√			
2K × 4	√	√	√	√	√			
1K × 8	√	√	√	√	√			
512 × 16	√	√	√	√	√			
9K × 1						√		
1K × 9							√	√
512 × 18							√	√

2.4.2 emb9k_sdp

EMB9K 存储器也支持简单双端口存储方式：一端口读一端口写。图 2-7 显示了简单双端口存储配置。

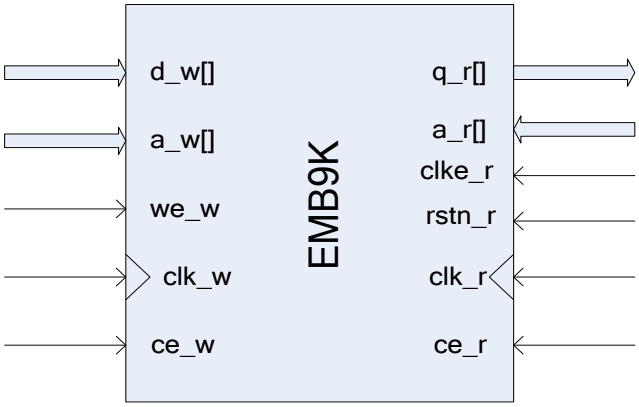


图 2-7 简单双端口存储方式

表 2-5 简单双端口存储方式端口描述

端口名称	类型	描述
d_w	Input	Write Data
a_w	Input	Write Address.
we_w	Input	Write Enable，high active
clk_w	Input	Write Clock.
ce_w	Input	Write Port Enable. high active
q_r	Output	Read Data
a_r	Input	Read Address.
ce_r	Input	Read Enable. high active
clk_r	Input	Read Clock.
rstn_r	Input	read register reset，low active
clke_r	Input	read clock enable，high active

表 2-6 简单双端口数据宽度组合

WritePort	Read Port									
	8K×1	4K×2	2K×4	1K×8	512×16	256×32	9K×1	1K×9	512×18	256×36
8K × 1	√	√	√	√	√	√				
4K × 2	√	√	√	√	√	√				
2K × 4	√	√	√	√	√	√				
1K × 9								√	√	√
1K × 8	√	√	√	√	√	√				
512 × 16	√	√	√	√	√	√				

WritePort	Read Port									
	8K×1	4K×2	2K×4	1K×8	512×16	256×32	9K×1	1K×9	512×18	256×36
512×18								√	√	√
256 × 32	√	√	√	√	√	√				
256×36								√	√	√
9K × 1							√			

2.4.3 emb9k_sp

EMB9K 存储块也支持单端口存储方式，如下图 2-8 所示：

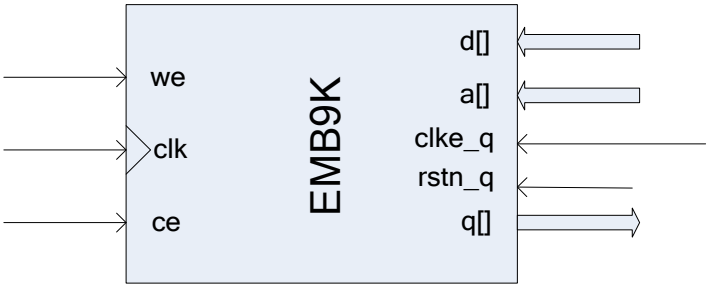


图 2-8 单端口存储方式

表 2-7 单端口存储方式端口描述

端口名称	类型	描述
d	Input	Write Data
a	Input	Write Address.
we	Input	Write Enable，high active
clk	Input	Write Clock.
ce	Input	Port Enable. high active
q	Output	Read Data
rstn_r	Input	read register reset，low active
clke_r	Input	read clock enable，high active

表 2-8 简单双端口可能的数据宽度组合

Port								
8K×1	4K×2	2K×4	1K×8	512×16	256×32	9 K×1	1K×9	512×18	256×36

2.5 I/O

2.5.1 I/O 特性

- 可承受 5V 输入
- 支持 LVCMOS33/LVTTL33 I/O 标准
- 独立三态输出控制
- 内部可编程上拉电阻: 62~112 KΩ
- I/O 驱动电流固定为: 8mA
- 支持异步直通输入输出
- 支持同步输入输出

2.5.2 I/O 描述

IOC是I/O的最小基本单元，一个IOC控制一个I/O。IOC加上绕线资源组成IOB0。下图描述了IOB0的结构。

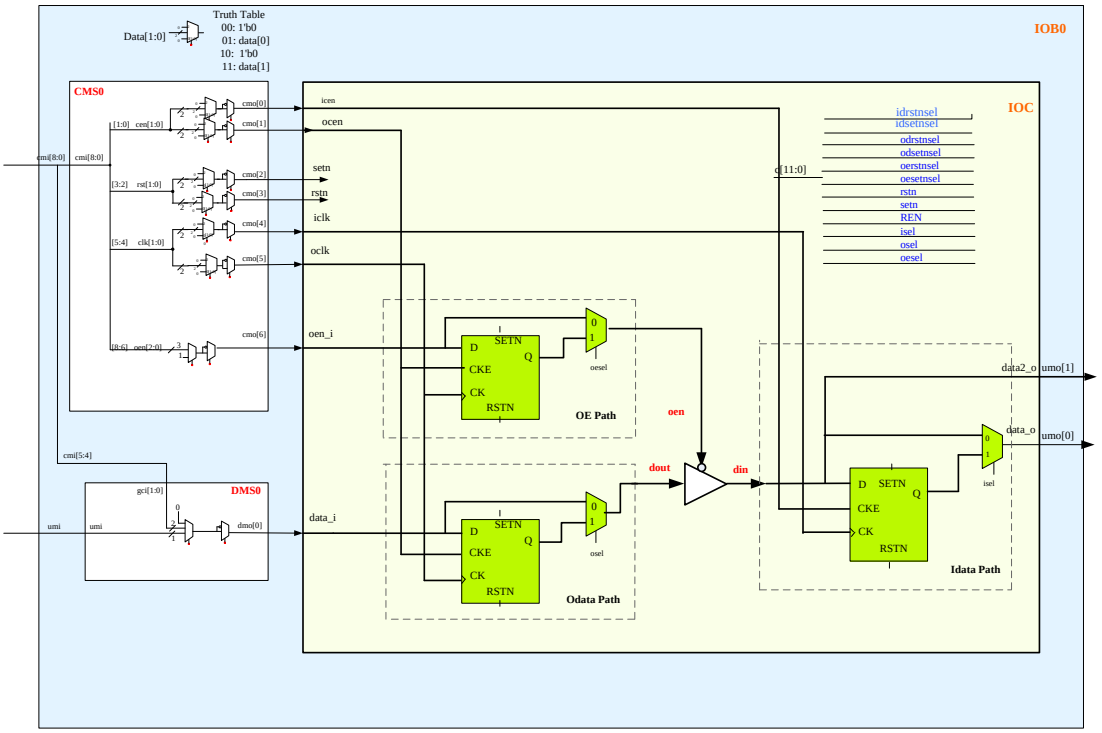


图 2-9 IOB0 结构

4 个 IOB0 和互联绕线资源 HSB1 组成一个 IOB1。下图描述了 IOB1 的结构。7 个 IOB1 控制 HME-M1 的左边 I/O，其余三边各有 8 个 IOB1。

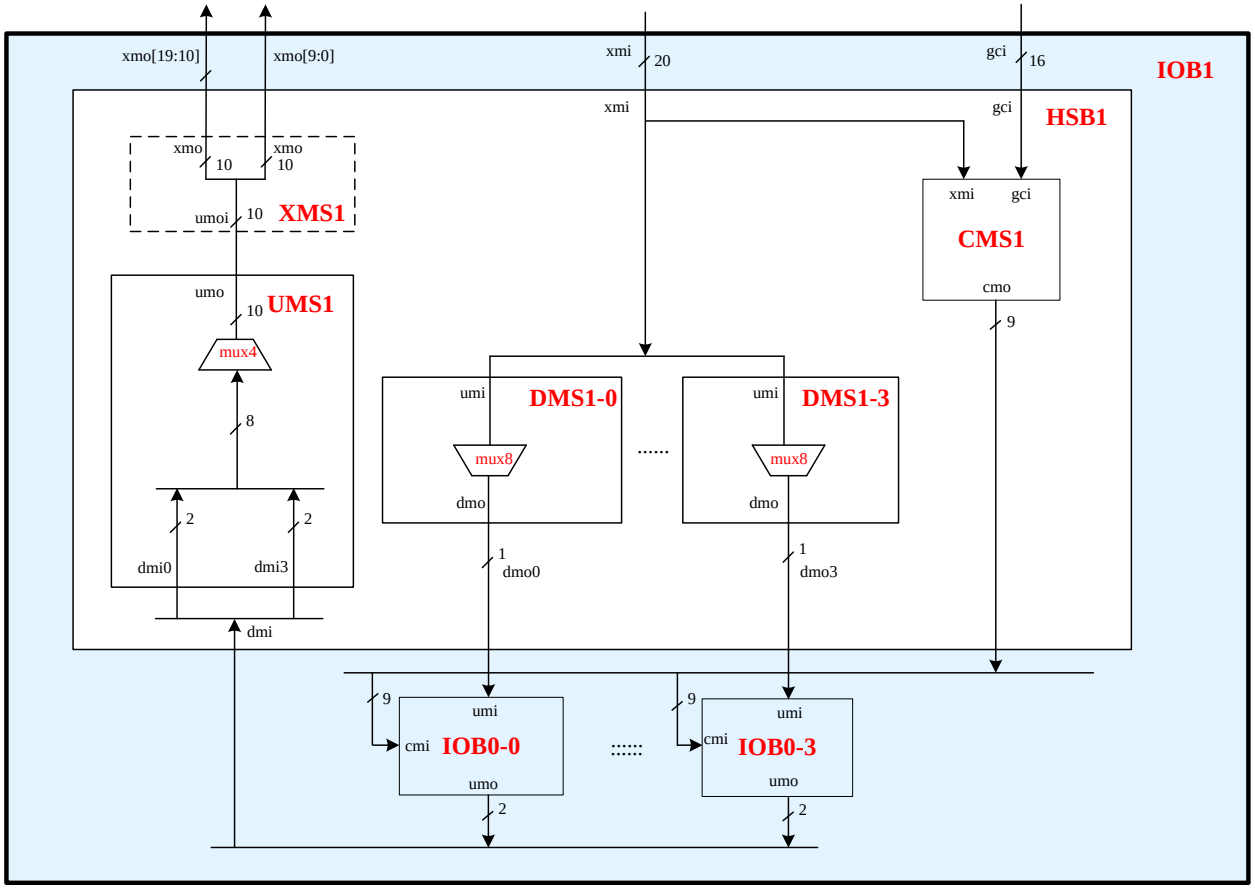


图 2-10 IOB1 结构

2.6 时钟网络

HME-M1 专用的全局时钟网络可以为 MSS 系统和 FPGA 的 fabric 提供灵活的低延迟、低抖动的时钟。

2.6.1 时钟网络通路

下图为 HME-M1 的全局时钟网络通路图。

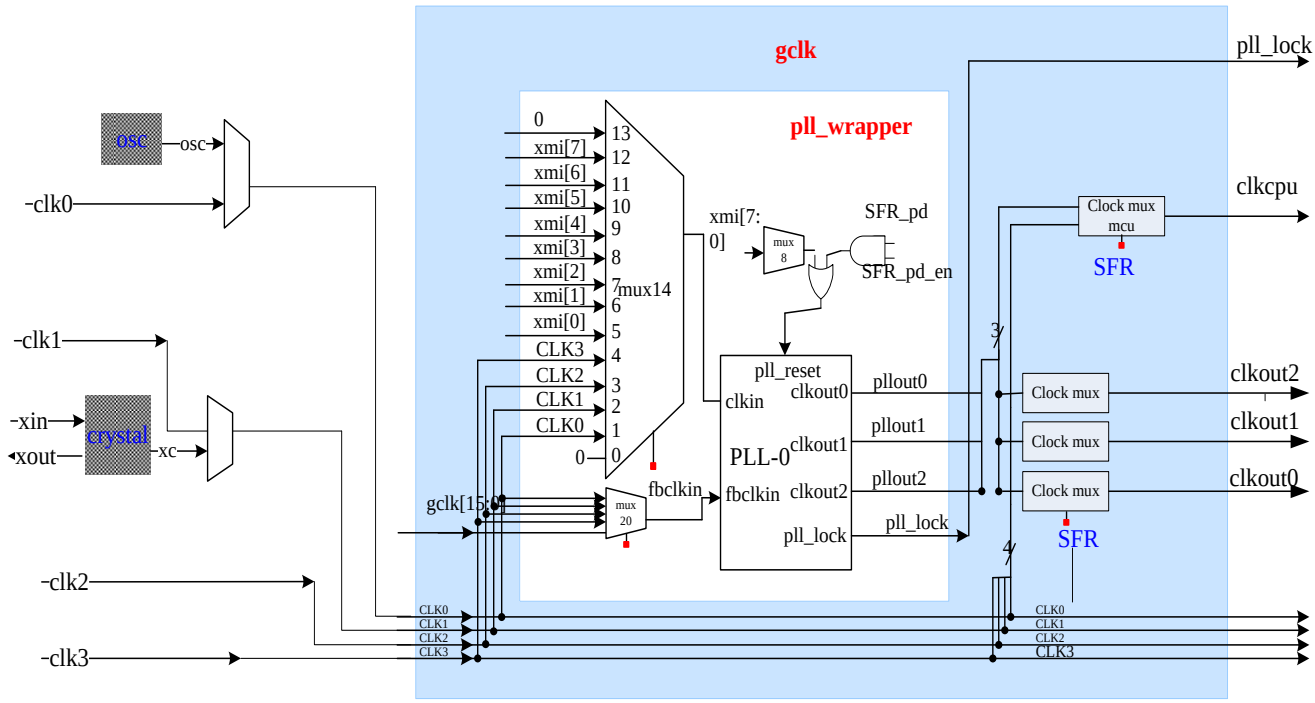


图 2-11 全局时钟网络通路

HME-M1 有五个时钟输入引脚：clk0, clk1, clk2,clk3 和 xin；一个时钟输出引脚：xout；1 个 PLL。

- clk0,clk1,clk2,clk3 这四个引脚既可以作为全局时钟输入引脚，也可以作为用户 I/O。
- xin 和 xout 分别为外部晶体振荡器输入引脚和输出引脚，Xin 只能用作全局时钟输入引脚，输入频率 10～20MHz。当 xin 作为外部时钟输入时，输入时钟通过 xin 接入全局时钟树上，xout 悬空。
- xin 与 clk1 不能同时用作全局时钟输入。
- OSC 为内部振荡器，典型值为 15MHz，随温度、电压变化而变化，变化范围：9~20MHz。
- OSC 为 AS 配置提供时钟，不推荐用户设计里使用 OSC。
- HME-M1 的设计必须在 Fuxi 里用 PLL Wizard 例化 PLL 才能为 MSS 和 Fabric 提供时钟。
- PLL Wizard 中配置对应的 Clock mux 的值，为 clkcpu 和 clkout0、clkout1、clkout2 选择全局时钟源。
- HME-M1Fabric 有 8 个全局时钟。clk0～clk3 和 clkcpu、clkout0、clkout1、clkout2 都可以连到全局时钟总线上。
- PLL 的输入 clkin 可以来自外部 clk0~clk3、外部晶体以及 fabric，PLL 的 de-skew 模式的反馈可以来自 Fabric 的 8 个全局时钟和 clk0～clk3。
- MSS 的 8051 可以通过扩展的 SFR 动态控制 PLL 的电源的关开、clkcpu 和 clkout0、clkout1、clkout2、PLL 输出关开以及分频，详见 MSS 系统。

2.6.2 PLL

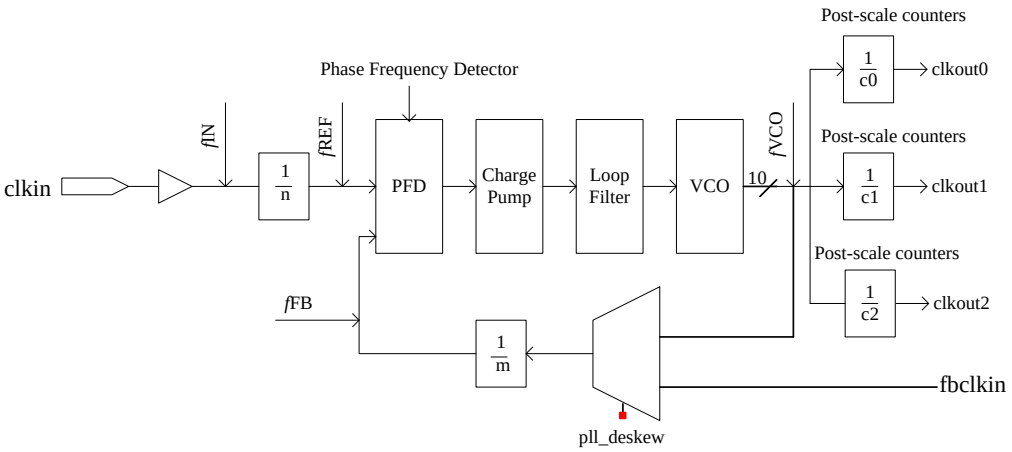


图 2-12 PLL

fREF = fIN / n;
fVCO = fREF * m = fIN * (m / n)
fFB = fVCO / m;
fclkout0 = fVCO / c0 = fIN * (m / (n * c0))
fclkout1 = fVCO / c1 = fIN * (m / (n * c1))
fclkout2 = fVCO / c2 = fIN * (m / (n * c2))

VCO 输出为 300MHz～800MHz。输出占空比为：50%±5%。

PLL 使用 m/（n×c）换算频率，为 PLL 输出端口提供时钟频率合成。输入时钟信号先被前置分频器（pre-divider）的系数 n 整除，然后再与反馈环路分频器（feedback-loop divider）的系数 m 相乘。控制回路将 VCO 的输出信号频率与 fIN×(m/n)进行匹配。输出级 C0，C1，C2 将高频 VCO 分频后输出。

前置分频器（pre-divider）系数 n 的范围是 1 到 32。反馈环路分频器（feedback-loop divider）的系数 m 的范围是 1 到 512。输出级分频器（post-scale divider）的 c0/c1/c2 的范围是 1 到 32。

2.6.3 PLL 接口

表 2-9 PLL 接口

信号	I/O	描述
clkin	I	Reference clock 5-350MHz
fbclkin	I	Feedback clock: used in de-skew mode
pll_reset	I	PLL reset: when reset = 1 , PLL reset
clkout0	O	Main output clock 10~350Mhz support dynamic clock switch
clkout1	O	First shifted phase out clock 10~350Mhz

		Can run at different freq in synthesize mode support dynamic clock switch
clkout2	O	Second shifted phase out clock 10~350Mhz Can run at different freq in synthesize mode support dynamic clock switch
pll_lock	O	PLL Lock status output. When pll_lock=1, PLL is in lock status
clk_cpu	O	Mcu clock out 10~350Mhz, support dynamic clock switch

2.6.4 PLL 属性

表 2-10 PLL 属性

类型	描述	典型值
pll_pwrmode	pll_pwrmode =always_on, always power on(default) pll_pwrmode =comb , MCU and FP control mode enable pll_pwrmode =FP , FP control mode enable pll_pwrmode =MCU , MCU control mode enable pll_pwrmode = always_off, always power off	stream
pll_deskew	pll_deskew=1,deskew mode enable pll_deskew=0, frequency synthesize,use VCO as feedback	0,1
pll_bypass	pll_bypass=1,pass input reference clock to clkout0/1/2	0,1
pll_o0en	pll_o0en=1, enable individual clock output clkout0	0,1
pll_o1en	pll_o1en=1, enable individual clock output clkout1	0,1
pll_o2en	pll_o2en=1, enable individual clock output clkout2	0,1
pll_ph1en	pll_ph1en=1,phase shift mode enable for clock output clkout1	0,1
pll_ph2en	pll_ph2en=1,phase shift mode enable for clock output clkout2	0,1
pll_refdiv	input divider	1:32
pll_fbdiv	feedback divider	1:512
pll_odiv0	output divider for clock output clkout0	1:32
pll_odiv1	output divider for clock output clkout1	1:32
pll_odiv2	output divider for clock output clkout2	1:32
pll_phsel1	phase shift control for clock output clkout1	Accoreding to the value of pll_odiv1
pll_phsel2	phase shift control for clock output clkout2	Accoreding to the value of pll_odiv2
pll_fsl	pll_fsl=1,link the phase shift between clkout0 and clkout1/ clkout2	0,1
clkcpu_mux	Clkcpu_mux = in0, divider disable and clk source is clk_in0 (default) Clkcpu_mux = in1, divider disable and clk source is	stream

	clk_in1 Clkcpu_mux = dyn, divider enable and clk source is dynamic Clkcpu_mux = dis, clk source is '0'	
clkcpu_in0	clkcpu_in0 = po0, select pllout0 (default) clkcpu_in0 = po1, select pllout1 clkcpu_in0 = po2, select pllout2	stream
clkcpu_in1	clkcpu_in0 = ck0, select clk0 (default) clkcpu_in0 = ck1, select clk1 clkcpu_in0 = ck2 select clk2 clkcpu_in0 = ck3 , select clk3	stream
clkout0_mux	clkout0_mux = in0, divider disable and clk source is clk_in0 (default) clkout0_mux = in1, divider disable and clk source is clk_in1 clkout0_mux = dyn, divider enable and clk source is dynamic clkout0_mux = dis, clk source is '0'	stream
clkout0_in0	clkout0_in0 = po0, select pllout0 (default) clkout0_in0 = po1, select pllout1 clkout0_in0 = po2, select pllout2	stream
clkout0_in1	clkout0_in0 = ck0, select clk0 (default) clkout0_in0 = ck1, select clk1 clkout0_in0 = ck2, select clk2 clkout0_in0 = ck3 , select clk3	stream
clkout1_mux	clkout1_mux = in0, divider disable and clk source is clk_in0 (default) clkout1_mux = in1, divider disable and clk source is clk_in1 clkout1_mux = dyn, divider enable and clk source is dynamic clkout1_mux = dis, clk source is '0'	stream
clkout1_in0	clkout1_in0 = po1, select pllout1 (default) clkout1_in0 = po2, select pllout2 clkout1_in0 = po0, select pllout0	stream
clkout1_in1	clkout1_in0 = ck0, select clk0 (default) clkout1_in0 = ck1, select clk1 clkout1_in0 = ck2, select clk2 clkout1_in0 = ck3 , select clk3	stream

clkout2_mux	clkout2_mux = in0, divider disable and clk source is clk_in0(default) clkout2_mux = in1, divider disable and clk source is clk_in1 clkout2_mux = dyn, divider enable and clk source is dynamic clkout2_mux = dis, clk source is '0'	stream
clkout2_in0	clkout2_in0 = po2, select pllout2 (default) clkout2_in0 = po0, select pllout0 clkout2_in0 = po1, select pllout1	stream
clkout2_in1	clkout2_in0 = ck0, select clk0 (default) clkout2_in0 = ck1, select clk1 clkout2_in0 = ck2, select clk2 clkout2_in0 = ck3 , select clk3	stream
ck0_src	ck0_src=PAD, select clk0 PAD as input(default) ck0_src=OSC, select internal osc as input	stream
ck1_src	ck1_src=PAD, select clk1 PAD as input ck1_src=XC, select crstral pin as input	stream

3 MSS 子系统

MSS 子系统由 150MHz 增强型 8051 处理器、集成的外设和集成的 SRAM 组成。具有如下特征：

- **增强型 8051MCU**
  - 精简指令周期，12 倍于标准 8051 的 MIPS，频率最高可达 150MHz
  - 兼容标准 8051 的指令系统
  - 片上调试系统（OCDS），支持 JTAG 在线调试
  - 支持高达 8M 数据/代码存储器
- **嵌入式 SRAM 存储器**
  - 32KByte 单端口存储器 SPRAM
  - 4KByte 双端口 DPRAM（Fabric 连 B 端口，8051 连 A 端口）
  - 数据/代码统一编址、存储大小灵活配置
- **外设**
  - 一个算术协处理器 MDU
  - 3 个 16 位定时器，定时器 2 可用作比较捕获单元
  - 1 个 16 位硬件看门狗
  - 1 个 I^2C 接口
  - 1 个 SPI 接口
  - 2 个全双工异步串行接口
- **停止、空闲模式电源管理**
- **芯片系统管理**
  - ISC 控制
  - IAP 控制
  - 在系统动态频率切换
  - 在系统动态 PLL 控制
  - 在系统 FPGA 时钟关断控制

MSS 子系统的 EMIF、P 端口、SPI 以及 DPRAM4K 的 B 端口连到了 FPGA 的 Fabric 上，I^2C 和扩展的 SFR 连到了 HME-M1 的硬件模块上。MSS 通过这些通路可以智能的控制 HME-M1 的运行。图 3-1 描述的是 MSS 系统的功能和连接关系。

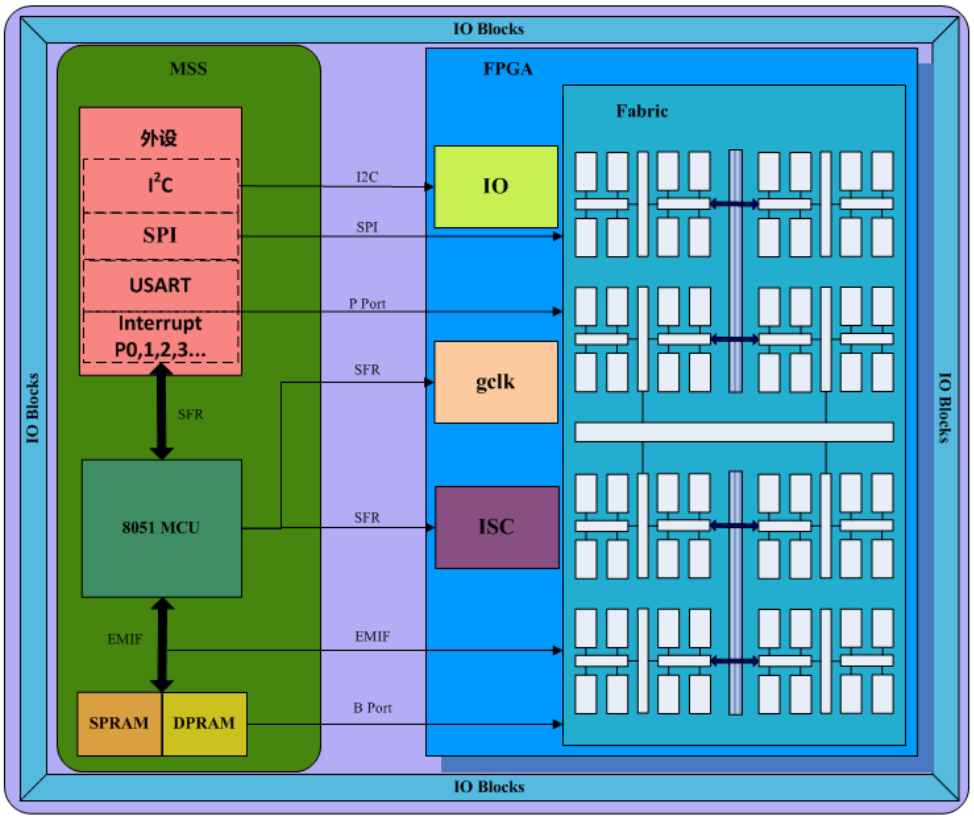


图 3-1 MSS 功能连接图

3.1 8051 CPU

HME-M1 使用单周期增强型 8051 CPU 作为中央处理器，指令集与标准 ASM51 完全兼容。

8051CPU 主要特性：

- 精简指令周期，12 倍于标准 8051 的 MIPS，频率最高可达 150MHz
- 兼容标准 8051 的指令系统
- 硬件支持乘法、除法指令
- 支持扩展指令：MOV A,ACC
- 片上调试系统（OCDS），JTAG 在线调试
- 支持 16 位乘法、32/16 位除法硬件
- 8 个外部中断源，共计 13 个中断源
- 77 个 SFR
- 支持高达 8MB 程序/数据存储器

3.1.1 寻址方式

8051 支持的寻址方式如下表 3-1 所示。

表 3-1 寻址方式

Addressing mode	Description
Direct Addressing	The operand is specified by a direct 8-bit address field. Only the internal RAM and the SFRs can be accessed using this mode.
Indirect Addressing	The instruction specifies the register which contains the address of the operand. The registers R0 or R1 are used to specify the 8-bit address, while the Data Pointer (DPTR) register is used to specify the 16-bit address.
Register Addressing	Certain instructions access one of the registers (R0-R7) in the specified register bank. These instructions are more efficient because there is no need for an address field.
Register Specific Instructions	Some instructions are specific to certain registers. For example, some instructions always act on the accumulator. In this case, there is no need to specify the operand.
Immediate Constants	Some instructions carry the value of the constants directly instead of an address.
Indexed Addressing	This type of addressing can be used only for a read of the program memory. This mode uses the Data Pointer as the base and the accumulator value as an offset to read a program memory.
Bit Addressing	In this mode, the operand is one of 256 bits.

3.1.2 指令集

8051 指令与工业标准 8051 指令完全兼容。下面几个表为指令集，包括寻址方式助记，按功能划分的指令集和十六进制编码排序指令，以及指令耗时计算。

3.1.3 寻址方式助记

下面 2 个表（表 3-32 和表 3-3）描述了应用在指令集表中的助记。

表 3-2 数据寻址方式助记

Symbol	Description
Rn	Working register R0-R7
direct	One of 128 internal RAM locations or any Special Function Register
@Ri	Indirect internal or external RAM location addressed by register R0 or R1
#data	8-bit constant included in instruction (immediate operand)
#data 16	16-bit constant included as bytes 2 and 3 of instruction (immediate operand)
bit	One of 128 software flags located in internal RAM, or any flag of bit- addressable Special Function Registers, including I/O pins and status word
A	Accumulator

表 3-3 程序寻址方式助记

Symbol	Description
addr16	Destination address for LCALL or LJMP, can be anywhere within the 64-Kbyte page of program memory address space
addr11	Destination address for ACALL or AJMP, within the same 2-Kbyte page of program memory as the first byte of the following instruction
Rel	SJMP and all conditional jumps include an 8-bit offset byte. Its range
	is +127/-128 bytes relative to the first byte of the following instruction

3.1.4 按功能划分的指令集

下表按照指令的功能进行分类，描述了指令编码的十六进制格式，每条指令占用的字节以及执行每条指令所需的机器周期。需注意，显示的周期数以无程序存储等待状态为前提。

表 3-4 按功能划分的指令

Mnemonic	Description	Code	Bytes	Cycles
Arithmetic operations				
ADD A,Rn	Add register to accumulator	0x28-0x2F	1	2
ADD A,direct	Add directly addressed data to accumulator	0x25	2	3
ADD A,@Ri	Add indirectly addressed data to accumulator	0x26-0x27	1	4
ADD A,#data	Add immediate data to accumulator	0x24	2	2
ADDC A,Rn	Add register to accumulator with carry	0x38-0x3F	1	2
ADDC A,direct	Add directly addressed data to accumulator with carry	0x35	2	3
ADDC A,@Ri	Add indirectly addressed data to accumulator with carry	0x36-0x37	1	4
ADDC A,#data	Add immediate data to accumulator with carry	0x34	2	2
SUBB A,Rn	Subtract register from accumulator with borrow	0x98-0x9F	1	2
SUBB A,direct	Subtract directly addressed data from accumulator with borrow	0x95	2	2
SUBB A,@Ri	Subtract indirectly addressed data from accumulator with borrow	0x96-0x97	1	4
SUBB A,#data	Subtract immediate data from accumulator with borrow	0x94	2	2
INC A	Increment accumulator	0x04	1	1
INC Rn	Increment register	0x08-0x0F	1	3
INC direct	Increment directly addressed location	0x05	2	4
INC @Ri	Increment indirectly addressed location	0x06-0x07	1	5
INC DPTR	Increment data pointer	0xA3	1	1
DEC A	Decrement accumulator	0x14	1	1
DEC Rn	Decrement register	0x18-0x1F	1	3

Mnemonic	Description	Code	Bytes	Cycles
DEC direct	Decrement directly addressed location	0x15	2	4
DEC @Ri	Decrement indirectly addressed location	0x16-0x17	1	5
MUL AB	Multiply A and B	0xA4	1	4
DIV	Divide A by B	0x84	1	4
DA A	Decimally adjust accumulator	0xD4	1	1
Logic operations				
ANL A,Rn	AND register to accumulator	0x58-0x5F	1	2
ANL A,direct	AND directly addressed data to accumulator	0x55	2	3
ANL A,@Ri	AND indirectly addressed data to accumulator	0x56-0x57	1	4
ANL A,#data	AND immediate data to accumulator	0x54	2	2
ANL direct,A	AND accumulator to directly addressed location	0x52	2	4
ANL direct,#data	AND immediate data to directly addressed location	0x53	3	4
ORL A,Rn	OR register to accumulator	0x48-0x4F	1	2
ORL A,direct	OR directly addressed data to accumulator	0x45	2	3
ORL A,@Ri	OR indirectly addressed data to accumulator	0x46-0x47	1	4
ORL A,#data	OR immediate data to accumulator	0x44	2	2
ORL direct,A	OR accumulator to directly addressed location	0x42	2	4
ORL direct,#data	OR immediate data to directly addressed location	0x43	3	4
XRL A,Rn	Exclusive OR register to accumulator	0x68-0x6F	1	2
XRL A,direct	Exclusive OR directly addressed data to accumulator	0x65	2	3
XRL A,@Ri	Exclusive OR indirectly addressed data to accumulator	0x66-0x67	1	4
XRL A,#data	Exclusive OR immediate data to accumulator	0x64	2	2
XRL direct,A	Exclusive OR accumulator to directly addressed location	0x62	2	4
XRL direct,#data	Exclusive OR immediate data to directly addressed location	0x63	3	4
CLR A	Clear accumulator	0xE4	1	1
CPL A	Complement accumulator	0xF4	1	1
RL A	Rotate accumulator left	0x23	1	1
RLC A	Rotate accumulator left through carry	0x33	1	1
RR A	Rotate accumulator right	0x03	1	1
RRC A	Rotate accumulator right through carry	0x13	1	1
SWAP A	Swap nibbles within the accumulator	0xC4	1	1
Data transfer operations				
MOV A,Rn	Move register to accumulator	0xE8-0xEF	1	1
MOV A,direct	Move directly addressed data to accumulator	0xE5	2	3
MOV A,@Ri	Move indirectly addressed data to accumulator	0xE6-0xE7	1	4
MOV A,#data	Move immediate data to accumulator	0x74	2	2

Mnemonic	Description	Code	Bytes	Cycles
MOV Rn,A	Move accumulator to register	0xF8-0xFF	1	1
MOV Rn,direct	Move directly addressed data to register	0xA8-0xAF	2	4
MOV Rn,#data	Move immediate data to register	0x78-0x7F	2	2
MOV direct,A	Move accumulator to direct	0xF5	2	2
MOV direct,Rn	Move register to direct	0x88-0x8F	2	3
MOV direct1,direct2	Move directly addressed data to directly addressed location	0x85	3	4
MOV direct,@Ri	Move indirectly addressed data to directly addressed location	0x86-0x87	2	5
MOV direct,#data	Move immediate data to directly addressed location	0x75	3	3
MOV @Ri,A	Move accumulator to indirectly addressed location	0xF6-0xF7	1	3
MOV @Ri,direct	Move directly addressed data to indirectly addressed location	0xA6-0xA7	2	4
MOV @Ri,#data	Move immediate data to in directly addressed location	0x76-0x77	2	3
MOV DPTR,#data16	Load data pointer with a 16-bit immediate	0x90	3	3
MOVC A,@A+DPTR	Load accumulator with a code byte relative to DPTR	0x93	1	4
MOVC A,@A+PC	Load accumulator with a code byte relative to PC	0x83	1	4
MOVX A,@Ri	Move external RAM (8-bit addr.) to accumulator②	0xE2-0xE3	1	5~12
MOVX A,@DPTR	Move external RAM (16-bit addr.) to accumulator②	0xE0	1	4~11
MOVX @Ri,A	Move accumulator to external RAM (8-bit addr.)②	0xF2-0xF3	1	6~13
MOVX @DPTR,A	Move accumulator to external RAM (16-bit addr.)②	0xF0	1	5~12
PUSH direct	Push directly addressed data onto stack	0xC0	2	4
POP direct	Pop directly addressed location from stack	0xD0	2	3
XCH A,Rn	Exchange register with accumulator	0xC8-0xCF	1	2
XCH A,direct	Exchange directly addressed location with accumulator	0xC5	2	3
XCH A,@Ri	Exchange indirect RAM with accumulator	0xC6-0xC7	1	4
XCHD A,@Ri	Exchange low-order nibbles of indirect and accumulator	0xD6-0xD7	1	5
Program branches				
ACALL addr11	Absolute subroutine call	xxx10001b	2	4
LCALL addr16	Long subroutine call	0x12	3	4
RET	Return from subroutine	0x22	1	5

② The MOVX instructions perform one of two actions depending on the state of 'pmw' bit (pcon.4). For more information refer to the Program Memory Write mode.

Mnemonic	Description	Code	Bytes	Cycles
RETI	Return from interrupt	0x32	1	5
AJMP addr11	Absolute jump	xxx00001b	2	3
LJMP addr16	Long jump	0x02	3	4
SJMP rel	Short jump (relative address)	0x80	2	3
JMP @A+DPTR	Jump indirect relative to the DPTR	0x73	1	3
JZ rel	Jump if accumulator is zero	0x60	2	3
JNZ rel	Jump if accumulator is not zero	0x70	2	3
JC rel	Jump if carry flag is set	0x40	2	3
JNC	Jump if carry flag is not set	0x50	2	3
JB bit,rel	Jump if directly addressed bit is set	0x20	3	5
JNB bit,rel	Jump if directly addressed bit is not set	0x30	3	5
JBC bit,rel	Jump if directly addressed bit is set and clear bit	0x10	3	5
CJNE A,direct,rel	Compare directly addressed data to accumulator and jump if not equal	0xB5	3	5
CJNE A,#data,rel	Compare immediate data to accumulator and jump if not equal	0xB4	3	4
CJNE Rn,#data,rel	Compare immediate data to register and jump if not equal	0xB8-0xBF	3	4
CJNE @Ri,#data,rel	Compare immed. to ind. and jump if not equal	B6-B7	3	6
DJNZ Rn,rel	Decrement register and jump if not zero	D8-DF	2	4
DJNZ direct,rel	Decrement directly addressed location and jump if not zero	D5	3	5
NOP	No operation	0	1	1
Boolean manipulation				
CLR C	Clear carry flag	0xC3	1	1
CLR bit	Clear directly addressed bit	0xC2	2	4
SETB C	Set carry flag	0xD3	1	1
SETB bit	Set directly addressed bit	0xD2	2	4
CPL C	Complement carry flag	0xB3	1	1
CPL bit	Complement directly addressed bit	0xB2	2	4
ANL C,bit	AND directly addressed bit to carry flag	0x82	2	3
ANL C,/bit	AND complement of directly addressed bit to carry	0xB0	2	3
ORL C,bit	OR directly addressed bit to carry flag	0x72	2	3
ORL C,/bit	OR complement of directly addressed bit to carry	0xA0	2	3
MOV C,bit	Move directly addressed bit to carry flag	0xA2	2	3
MOV bit,C	Move carry flag to directly addressed bit	0x92	2	4

3.1.5 十六进制编码排序指令

下表显示的按十六进制编码排序的指令集。

表 3-5 十六进制指令

Op-code	Mnemonic
00 H	NOP
01 H	AJMP addr11
02 H	LJMP addr16
03 H	RR A
04 H	INC A
05 H	INC direct
06 H	INC @R0
07 H	INC @R1
08 H	INC R0
09 H	INC R1
0A H	INC R2
0B H	INC R3
0C H	INC R4
0D H	INC R5
0E H	INC R6
0F H	INC R7
10 H	JBC bit,rel
11 H	ACALL addr11
12 H	LCALL addr16
13 H	RRC A
14 H	DEC A
15 H	DEC direct
16 H	DEC @R0
17 H	DEC @R1
18 H	DEC R0
19 H	DEC R1
1A H	DEC R2
1B H	DEC R3
1C H	DEC R4
1D H	DEC R5
1E H	DEC R6
1F H	DEC R7
20 H	JB bit.rel
21 H	AJMP addr11
22 H	RET
23 H	RL A
24 H	ADD A,#data
25 H	ADD A,direct
26 H	ADD A,@R0
27 H	ADD A,@R1
28 H	ADD A,R0
29 H	ADD A,R1
2A H	ADD A,R2
2B H	ADD A,R3
2C H	ADD A,R4
2D H	ADD A,R5
2E H	ADD A,R6
2F H	ADD A,R7
30 H	JNB bit,rel
31 H	ACALL addr11
32 H	RETI
33 H	RLC A
34 H	ADDC A,#data
35 H	ADDC A,direct
36 H	ADDC A,@R0
37 H	ADDC A,@R1
38 H	ADDC A,R0
39 H	ADDC A,R1
3A H	ADDC A,R2
3B H	ADDC A,R3
3C H	ADDC A,R4
3D H	ADDC A,R5
3E H	ADDC A,R6
3F H	ADDC A,R7
40 H	JC rel
41 H	AJMP addr11
42 H	ORL direct,A
43 H	ORL direct,#data
44 H	ORL A,#data
45 H	ORL A,direct
46 H	ORL A,@R0
47 H	ORL A,@R1
48 H	ORL A,R0
49 H	ORL A,R1
4A H	ORL A,R2
4B H	ORL A,R3
4C H	ORL A,R4
4D H	ORL A,R5
4E H	ORL A,R6
4F H	ORL A,R7
50 H	JNC rel
51 H	ACALL addr11
52 H	ANL direct,A
53 H	ANL direct,#data
54 H	ANL A,#data
55 H	ANL A,direct
56 H	ANL A,@R0
57 H	ANL A,@R1
58 H	ANL A,R0
59 H	ANL A,R1
5A H	ANL A,R2
5B H	ANL A,R3
5C H	ANL A,R4
5D H	ANL A,R5
5E H	ANL A,R6
5F H	ANL A,R7
60 H	JZ rel
61 H	AJMP addr11
62 H	XRL direct,A
63 H	XRL direct,#data
64 H	XRL A,#data
65 H	XRL A,direct
66 H	XRL A,@R0
67 H	XRL A,@R1
68 H	XRL A,R0
69 H	XRL A,R1
6A H	XRL A,R2
6B H	XRL A,R3
6C H	XRL A,R4
6D H	XRL A,R5
6E H	XRL A,R6
6F H	XRL A,R7
70 H	JNZ rel
71 H	ACALL addr11
72 H	ORL C,bit
73 H	JMP @A+DPTR
74 H	MOV A,#data

Op-code	Mnemonic
75 H	MOV direct,#data
76 H	MOV @R0,#data
77 H	MOV @R1,#data
78 H	MOV R0.#data
79 H	MOV R1.#data

Op-code	Mnemonic
7A H	MOV R2.#data
7B H	MOV R3.#data
7C H	MOV R4.#data
7D H	MOV R5.#data
7E H	MOV R6.#data

Op-code	Mnemonic
7F H	MOV R7.#data

3.1.6 读-改-写指令

执行“从 SFR 或内部 RAM 读取字节，修改并且重新写回”的指令，叫做“读-改-写指令”。当目标是 I/O 端口（P0-P3）或端口中的一位时，这些指令读取的是输出锁存器而不是引脚。

表 3-6 读-改-写指令

Mnemonic	Description	Code	Bytes	Cycles
ANL direct,A	AND accumulator to direct	0x52	2	3
ANL direct,#data	AND immediate data to direct	0x53	3	4
ORL direct,A	OR accumulator to direct	0x42	2	3
ORL direct,#data	OR immediate data to direct	0x43	3	4
XRL direct,A	Exclusive OR accumulator to direct	0x62	2	3
XRL direct,#data	Exclusive OR immediate data to direct	0x63	3	4
JBC bit, rel	Jump if bit is set and clear bit	0x10	3	4
CPL bit	Complement bit	0xB2	2	3
INC direct	Increment direct	0x05	2	3
INC @Ri	Increment indirect	0x06-0x07	1	3
DEC direct	Decrement direct	0x15	2	3
DEC @Ri	Decrement indirect	0x16-0x17	1	3
DJNZ direct,rel	Decrement and jump if not zero	0xD5	3	4
MOV bit,C	Move carry flag to direct bit	0x92	2	3
CLR bit	Clear bit	0xC2	2	3
SETB bit	Set bit	0xD2	2	3

3.1.7 扩展指令

HME-M1 中的 8051 MCU 兼容英特尔 MCS51 指令集，并扩展了如下指令：

MOV A, ACC

这条指令不会破坏 ACC 寄存器中的数据。

3.1.8 指令周期计算

为精确计算程序执行时间，每个指令的执行周期都需要计算在内。每个指令周期都可使用下面的公式计算：

```
if (BYTES > 1 or CYCLES = 1) then
DURATION = CYCLES + (BYTES+R)*P + X*D
else
DURATION = CYCLES + (2+R)*P + X*D
```

其中：

- BYTE 是指令的字节数（见表 3-4 和表 3-6）
- CYCLE 是无等候状态时的周期数（见表 3-4 和表 3-6）
- R=1 针对 MOVC 指令，否则 R=0
- X=1 针对 MOVX 指令，否则 X=0
- P=程序存储器等待状态数（='ckcon[6:4]'）
- D=数据存储器等待状态数（='ckcon[2:0]'）

在程序存储器写模式下（PMW），MOVX 的公式如下：

```
DURATION = CYCLES + (2+X)*P
```

3.2 存储器

8051 内核采用哈佛结构，程序和数据的地址、数据总线是共享的但有独立的程序和数据控制信号。8051 的存储组织与标准 8051 相同。分三种存储区域：程序空间、外部数据空间和内部数据空间。HME-M1 的 MSS 集成了 256 字节的内部数据空间。

利用页寄存器可以扩展程序和数据空间到 8MB。HME-M1 把 8051 的程序与数据控制信号合并将哈佛体系结构连接为冯诺依曼体系结构，也就是进行统一编址，共享相同的存储空间。需要用户在编程时，区分程序和数据空间，以免重叠。

3.2.1 内存和 SFR

图 3-2 描述的是内存为 256 字节的存储映射。存储空间同时提供 128 字节的特殊功能寄存器。对低于 80H 的地址空间无论进行直接寻址还是间接寻址，访问的都是数据存储器的低 128 字节。对高于 7Fh 的地址空间进行间接寻址访问的是内部数据存储器的高 128 字节，直接寻址访问的是 SFR。低 128 字节包括工作寄存器（00h…1Fh）以及位寻址区（20h…2Fh）。最低 32 字节构成 4 个 bank，每个 bank 都包括八个寄存器（R0-R7）。程序存储器状态字（PSW）的 2 个位选取被使用的 bank。紧接着 16 个字节是位寻址区，可通过 00h-7Fh 地址进行位寻址（参见表 3-10 可位寻址空间）。

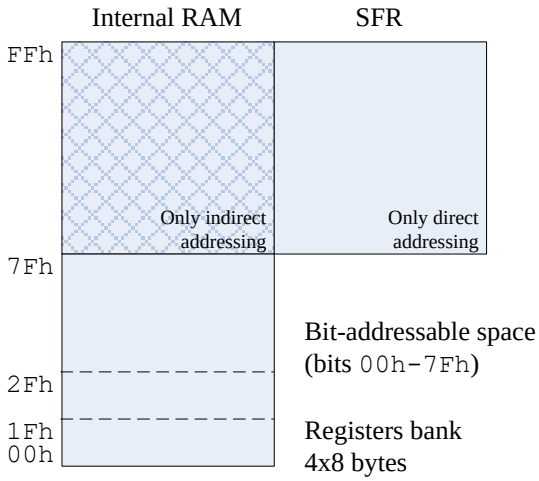


图 3-2 内存映像

3.2.2 RAM 存储器

HME-M1 内部集成了一块 32KB 的 SPRAM 和一块 4KB 的 DPRAM。32KB 的 SPRAM 仅为 MSS 系统可用。8051 操作 SPRAM 和 DPRAM 最高频率可达 150MHz，8051 与 RAM 之间采用同步工作方式。

在 Fuxi 中例化 system wizard 时设置 8051 firmware 的输出.hex 为 MSS 系统内嵌 RAM 的初始化。内部集成的 Memory 用作程序区时，MSS 的程序作为 HME-M1 配置的一部分在配置过程中装载到 MSS 的程序 Memory 里。

4KB 的双端口 RAM 中的 A 端口连线到 MSS 的 Memory 总线，B 端口连到 FPGA 的 Fabric，MSS 和 FPGA 共享 4KB 空间，同时 MSS 和 FPGA 可以独立操作 DPRAM。图 3-3 描述了 HME-M1 中 MSS 的存储映像。

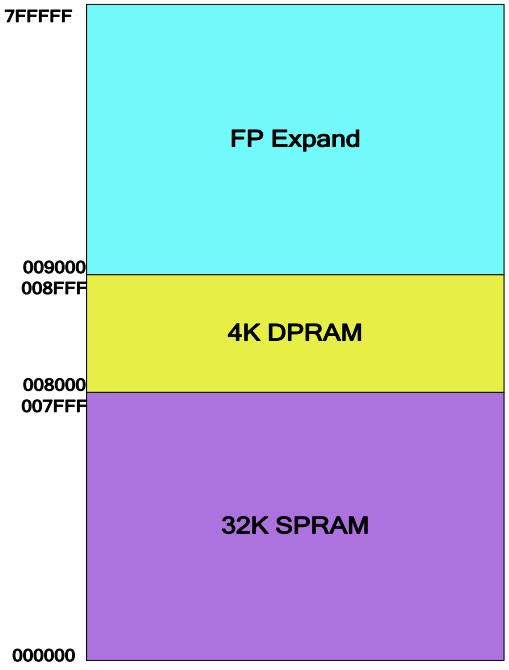


图 3-3 存储地址映像

注意：用户编程必须确保程序和数据空间不重叠，以免程序运行时，对数据的操作改写了程序代码。

4KB 的 DPRAM 存储块 B 端口存储方式，如下图 2-8 所示：

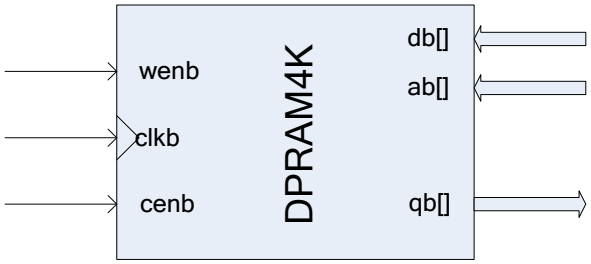


图 3-4 单端口存储方式

表 3-7 4KB 的DPRAM 存储方式端口描述

Port name	Type	Description
db	Input	Write Data，8 bit width
ab	Input	Write Address，12 bit width
wenb	Input	Write Enable，low active
clkb	Input	Write Clock.
cenb	Input	Port Enable. low active
qb	Output	Read Data，8 bit width

3.2.3 存储 Banking

8051 不做 Banking 时，存储空间可达 64KB（0000h～FFFFh）。当用做 Banking 时，存储映像如下所示。具体设置,请参见 9) 和 10) 。

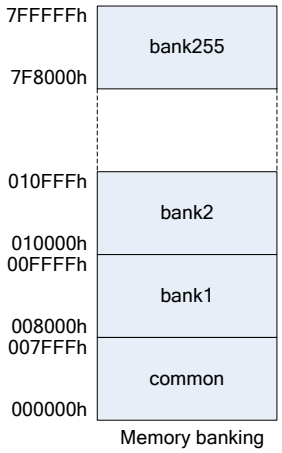


图 3-5 程序/外部数据存储段映射

3.2.4 外部存储器接口（EMIF）

外部存储器接口用于扩展 MSS 系统的存储空间，地址映像位于 9000～7FFFFF，需用 Fabric 去实现 EMIF 接口。MSS 的 8051 时钟端口通过硬连线连接到全局时钟树上，需要在 PLL wizard 中选择一个，不需要 Primace 软件绕线。EMIF 的 clkemif 时钟也来自全局时钟，需要 Fuxi 软件绕线实现时钟的连接。8051 时钟与 EMIF 的 clkemif 时钟可以同频率也可不同频率，两者是异步关系。EMIF 接口实现了 Fabric EMIF 操作与 8051 EMIF 操作的双向同步。8051 对 Fabric 扩展的存储空间每次读写需要大约 4 个 clkemif + 3 个 8051 时钟周期。

表 3-8 EMIF 端口描述

Port name	Type	Width	Description
clkemif	Input	1	Fabric EMIF clock， posedge active
memaddr	Output	23	EMIF Address，MSS to Fabric
memdatai	Input	8	Read Data，Fabric to MSS
memdatao	Output	8	Write data，MSS to Fabric
memrd	Output	1	read Enable. high active
memwr	Output	1	Write Enable，high active
memack	Input	1	Fabric to MSS operation acknowledge

EMIF 包括 23 位宽的地址总线“memaddr”,8 位的输入数据总线 “memdatai”,8 位输出数据总线“memdatao”，控制信号 “memrd”, “memwr”, “memack”以及时钟信号“clkemif”。

控制信号 “memrd”,“memwr”,在 clkemif 的上升沿输出到 Fabric。读取时，在 Fabric 的数据有效的情况下，下一个时钟的上升沿 Fabric 输出一个时钟的“memack”有效到 MSS。写入时，Fabric 接收到数据后，输出一个时钟的“memack”有效到 MSS。

EMIF 的读写时序时序图，如下所示。

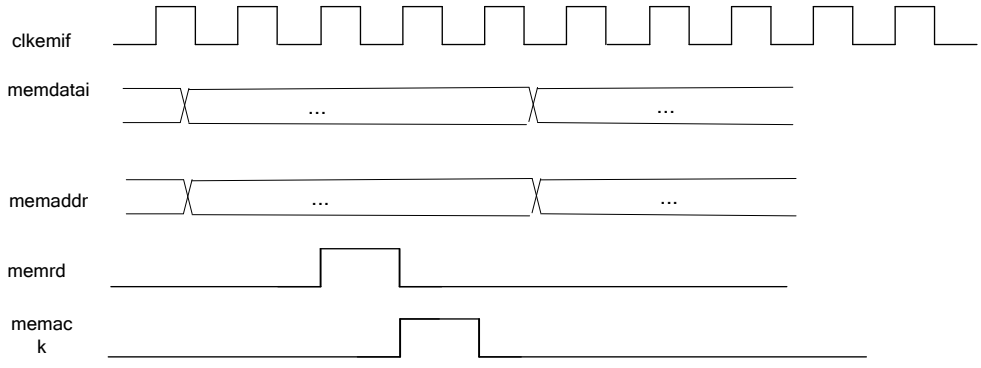


图 3-6 EMIF 读时序

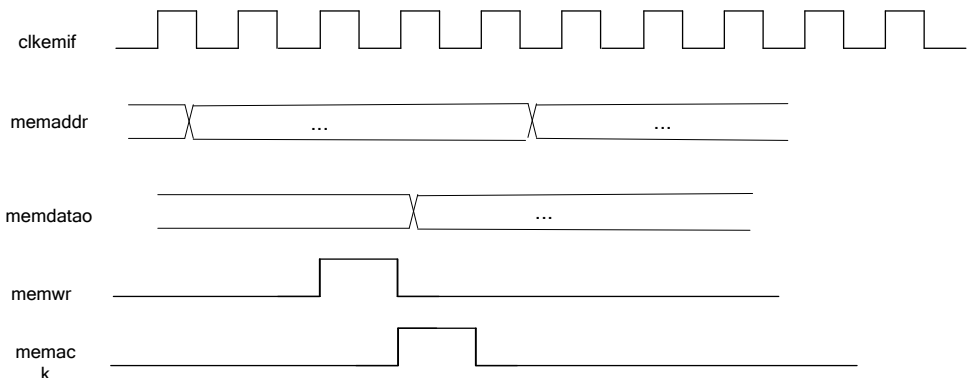


图 3-7 EMIF 写时序

3.3 特殊功能寄存器（SFRs）

SFRs 在内存空间中，地址范围从 7Fh 到 FFh。

3.3.1 SFR 存储单元

如表 3-9 所示的 SFR 表，部分地址已被占用，其余部分为保留的特殊功能寄存器。

表 3-9 SFR 存储单元

Hex/ Bin	X000	X001	X010	X011	X100	X101	X110	X111	Bin/ Hex
F8	misccon	clkcpucon	clko0con	clko1con	clko2con				FF
F0	b							srst	F7
E8		md0	md1	md2	md3	md4	md5	arcon	EF
E0	acc	spsta	spcon	spdat	spssn				E7
D8	adcon		i2cdat	i2cadr	i2ccon	i2csta			DF
D0	psw								D7
C8	t2con		crcl	crch	tl2	th2			CF
C0	ircon	ccen	ccl1	cch1	ccl2	cch2	ccl3	cch3	C7
B8	ien1	ip1	s0relh	s1relh					BF
B0	p3								B7
A8	ien0	ip0	s0rell	i2cspisel	iscaddr0	iscaddr1	iscaddr2	iscaddr3	AF
A0	p2								A7
98	s0con	s0buf	ien2	s1con	s1buf	s1rell			9F
90	p1		dps		pagesel	d_pagesel			97
88	tcon	tmod	tl0	tl1	th0	th1	ckcon		8F
80	p0	sp	dpl	dph	dpl1	dph1	wdtrel	pcon	87

在 SFR 地址空间中有 16 个 SFR 是既可以字节寻址也可以是位寻址的。位寻址的 SFR 是地址以 000 ´ b（80 ´ h , 88 ´ h , 90 ´ h … F8 ´ h）结尾的。16 个寄存器（128 位）和 128 位的内部数据存储器（存储单元为 20 ´ h … 2F ´ h）一起构成位寻址区（请参见表 3-10）。

表 3-10 可位寻址空间

Hex/ Bin	X000	X001	X010	X011	X100	X101	X110	X111	Bin/ Hex
SFR									
F8	misccon.0	misccon.1	misccon.2	misccon.3	misccon.4	misccon.5	misccon.6	misccon.7	FF
F0	b.0	b.1	b.2	b.3	b.4	b.5	b.6	b.7	F7
E8									EF
E0	acc.0	acc.1	acc.2	acc.3	acc.4	acc.5	acc.6	acc.7	E7
D8	adcon.0	adcon.1	adcon.2	adcon.3	adcon.4	adcon.5	adcon.6	adcon.7	DF
D0	psw.0	psw.1	psw.2	psw.3	psw.4	psw.5	psw.6	psw.7	D7
C8	t2con.0	t2con.1	t2con.2	t2con.3	t2con.4	t2con.5	t2con.6	t2con.7	CF
C0	ircon.0	ircon.1	ircon.2	ircon.3	ircon.4	ircon.5	ircon.6	ircon.7	C7
B8	ip.0 /ien1.0	ip.1 /ien1.1	ip.2 /ien1.2	ip.3 /ien1.3	ip.4 /ien1.4	ip.5 /ien1.5	ip.6 /ien1.6	ip.7 /ien1.7	BF
B0	p3.0	p3.1	p3.2	p3.3	p3.4	p3.5	p3.6	p3.7	B7
A8	ien0.0	ien0.1	ien0.2	ien0.3	ien0.4	ien0.5	ien0.6	ien0.7	AF
A0	p2.0	p2.1	p2.2	p2.3	p2.4	p2.5	p2.6	p2.7	A7
98	s0con.0	s0con.1	s0con.2	s0con.3	s0con.4	s0con.5	s0con.6	s0con.7	9F
90	p1.0	p1.1	p1.2	p1.3	p1.4	p1.5	p1.6	p1.7	97
88	tcon.0	tcon.1	tcon.2	tcon.3	tcon.4	tcon.5	tcon.6	tcon.7	8F
80	p0.0	p0.1	p0.2	p0.3	p0.4	p0.5	p0.6	p0.7	87
Internal RAM									
78	2Fh.0	2Fh.1	2Fh.2	2Fh.3	2Fh.4	2Fh.5	2Fh.6	2Fh.7	7F
70	2Eh								77
68	2Dh								6F
60	2Ch								67
58	2Bh								5F
50	2Ah								57
48	29h								4F
40	28h								47
38	27h								3F
30	26h								37
28	25h								2F
20	24h								27
18	23h								1F
10	22h								17
08	21h.0	21h.1	21h.2	21h.3	21h.4	21h.5	21h.6	21h.7	0F
00	20h.0	20h.1	20h.2	20h.3	20h.4	20h.5	20h.6	20h.7	07

3.3.2 SFR 复位值

下表为复位值和简单说明。

表 3-11 SFR 表的复位值

SFR	Addr	Rese	Description
p0	80h	FFh	Port 0
sp	81h	07h	Stack Pointer
dpl	82h	00h	Data Pointer Low
dph	83h	00h	Data Pointer High
wdtrel	86h	00h	Watchdog Timer Reload
pcon	87h	00h	Power Control
tcon	88h	00h	Timer/Counter Control
tmod	89h	00h	Timer Mode Register
tl0	8Ah	00h	Timer 0, low byte
tl1	8Bh	00h	Timer 1, low byte
th0	8Ch	00h	Timer 0, high byte
th1	8Dh	00h	Timer 1, high byte
ckcon	8Eh	71h	Clock Control Register
p1	90h	FFh	Port 1
dps	92h	00h	Data Pointer Select
pagesel	94h	01h	Program Memory Page
d_pagesel	95h	01h	External Data Memory
s0con	98h	00h	Serial Port 0, Control
s0buf	99h	00h	Serial Port 0, Data Buffer
ien2	9Ah	00h	Interrupt Enable Register
s1con	9Bh	00h	Serial Port 1, Control
s1buf	9Ch	00h	Serial Port 1, Data Buffer
s1rell	9Dh	00h	Serial Port 1, Reload
p2	A0h	FFh	Port 2
ien0	A8h	00h	Interrupt Enable Register
ip0	A9h	00h	Interrupt Priority Register
s0rell	AAh	D9h	Serial Port 0, Reload
p3	B0h	FFh	Port 3
ip/ien1	B8h	00h	Interrupt Priority Register
ip1	B9h	00h	Interrupt Priority Register
s0relh	BAh	03h	Serial Port 0, Reload
s1relh	BBh	03h	Serial Port 1, Reload
ircon	C0h	00h	Interrupt Request Control
ccen	C1h	00h	Compare/Capture Enable

SFR	Addr	Rese	Description
ccl1	C2h	00h	Compare/Capture
cch1	C3h	00h	Compare/Capture
ccl2	C4h	00h	Compare/Capture
cch2	C5h	00h	Compare/Capture
ccl3	C6h	00h	Compare/Capture
cch3	C7h	00h	Compare/Capture
t2con	C8h	00h	Timer 2 Control Register
crcl	CAh	00h	Compare/Reload/Capture
crch	CBh	00h	Compare/Reload/Capture
tl2	CCh	00h	Timer 2, low byte
th2	CDh	00h	Timer 2, high byte
psw	D0h	00h	Program Status Word
adcon	D8h	00h	Serial Port 0 Baud Rate Select register (only)
i2cdat	DAh	00h	I2C Data Register
i2cadr	DBh	00h	I^2C Address Register
i2ccon	DCh	00h	I^2C Control Register
i2csta	DDh	F8h	I^2C Status Register
acc	E0h	00h	Accumulator
spsta	E1h	00h	Serial Peripheral Status
spcon	E2h	14h	Serial Peripheral Control
spdat	E3h	00h	Serial Peripheral Data
spssn	E4h	FFh	Serial Peripheral Slave
md0	E9h	00h	Multiplication/Division
md1	EAh	00h	Multiplication/Division
md2	EBh	00h	Multiplication/Division
md3	ECh	00h	Multiplication/Division
md4	EDh	00h	Multiplication/Division
md5	EEh	00h	Multiplication/Division
arcon	EFh	00h	Arithmetic Control
b	F0h	00h	B Register
srst	F7h	00h	Software Reset Register

3.3.3 SFR 说明

本节简略描述了所有 SFR。

1) 累加器-ACC

在大多数 8051 指令中使用累加器 A，保存操作数并存储运算结果。在使用累加器的指令中，其助记符是 A，而不是 ACC。

2) B 寄存器-B

B 寄存器在执行乘除法指令时使用。也可以用作临时寄存器用来存储临时数据。

3) 程序状态寄存器-PSW

PSW 寄存器反映 CPU 当前执行状态的状态位。需注意，检验位 P 只能根据 ACC 寄存器状态由硬件修改其值。

表 3-12 PSW 寄存器

Bit	Symbol	Description	Type
psw.7	cy	Carry flag	R/W
		Carry bit in arithmetic operations and accumulator for Boolean operations.	
psw.6	ac	Auxiliary Carry flag	R/W
		Set if there is a carry-out from 3rd bit of Accumulator in BCD Operations	
psw.5	f0	General purpose Flag 0	R/W
		General purpose flag available for user	
psw.4	rs1	Register bank select control bit 1, used to select working register bank	R/W
psw.3	rs0	Register bank select control bit 0, used to select working register bank	R/W
psw.2	ov	Overflow flag	R/W
		Set in case of overflow in Accumulator during arithmetic operations	
psw.1	f1	General purpose Flag 1	R/W
		General purpose flag available for user.	
psw.0	p	Parity flag	R
		Reflects the number of ‘1’s in the Accumulator.	
		P = ‘1’ if Accumulator contains an odd number of ‘1’s	
		P = ‘0’ if Accumulator contains an even number of ‘1’s	

rs1 和 rs0 位的状态按如下方式选择工作寄存器组：

表 3-13 寄存器组存储单元

rs1	rs0	Selected Register Bank	Location
0	0	Bank 0	(00H – 07H)
0	1	Bank 1	(08H – 0FH)
1	0	Bank 2	(10H – 17H)
1	1	Bank 3	(18H – 1FH)

4) 堆栈指针-SP

这个寄存器指向内部数据存储空间的堆栈顶部。用于执行中断服务子程序或子程序调用前保存程序的返回地址。堆栈指针在执行 PUSH 或 CALL 命令之前会递增，执行 POP 或 RET(I)指令之后会递减（通常指向堆栈顶部）。

5) 数据指针-DPH,DPL

MCU 中实现了 2 个数据指针寄存器。DPTR 可以通过 DPL 和 DPH 进行访问。当前使用的数据指针寄存器由 DPS 寄存器选取。

这两个寄存器用于间接寻址时存储 16bit 长的地址，并适用于以下指令：

- MOVX
- MOVC
- JMP（computed branch）

它们可以作为 1 个 16 位的寄存器或者 2 个 8 位的寄存器进行操作。DPH 存储当前间接地址高字节，DPL 存储当前间接地址的低字节。

通常用于访问外部程序空间或数据空间，例如：

- MOVC A, @A+DPTR（程序空间）
- MOVX A, @DPTR（数据空间）

6) 数据指针选择寄存器-DPS

8051 包含 2 个数据指针寄存器。每个寄存器都可作为 16 位的地址源用于间接寻址。DPS 寄存器用于选择当前数据指针寄存器。

- HME-M1 中，DPS 寄存器位于 SFR 存储空间，宽为 1 位。HME-M1 中，有 2 个 DPTR。

表 3-14 DPS 寄存器

Bit	Symbol	Description	Type
dps.7	-	not used, read as 0	R
dps.6	-		
dps.5	-		
dps.4	-		
dps.3	-		R/W

Bit	Symbol	Description	Type
dps.2	-		R/W
dps.1	-		R/W
dps.0	dpsel0	Data Pointer Register select. The content of this bit specifies which DPTR from the two is used as current active DPTR.	R/W

7) 数据指针 1—DPH1,DPL1

DPTR1 寄存器可以通过 DPL1 和 DPH1 进行访问。不论实际 DPS 寄存器选中当前哪个 DPTR，DPL1 和 DPH1 始终指向 DPTR1。当 DPS 寄存器的 LSB 被设置为 1 时，所有 DPTR 相关指令都使用这个 16 位的寄存器，否则 DPTR 来自 DPH 和 DPL。

- DPTR1 特殊功能寄存器位于 SFR 空间，共有 2 个数据指针。

8) 端口—P0,P1,P2,P3

执行写操作后，这些寄存器中的内容可以从芯片的相应引脚处检测到（端口 P0，P1，P2，P3）。向任一的口线写‘1’可使得相应引脚保持高电平，写‘0’会使得相应的引脚保持低电平。

执行读操作时，寄存器端口 P0,P1,P2,P3 的状态反应了 8051 引脚的相应值。

应注意到某些涉及到端口 N 的读-改-写指令实际上是读锁存器（如，INC P0; ANL P2, A），而其他指令则是读引脚（如, MOV A,P1）。

- P0,P1,P2,P3 寄存器位于 SFR 存储器空间。

9) 程序存储页面寄存器—PAGESEL

此寄存器为程序存储器在 bank 实现时提供额外地址。需注意，为了在复位后不再向“pagesel”寄存器写值仍能够生成正确的地址（逻辑地址 8000h 等于对应的物理地址），PAGESEL 的复位值为 1。不能将 0 值赋给 pagesel 寄存器，否则将导致 bank 区域（逻辑地址在 8000h-FFFFh）与物理地址（0000h-7FFFh）内容重叠。

10) 数据存储页面寄存器—D_PAGESEL

此寄存器为数据存储器的组合提供附加地址。需注意，为了在复位后不再向“d_pagesel”寄存器写值仍能够生成正确的地址（逻辑地址 8000h 等于对应的物理地址），PAGESEL 的复位值为 1。不能将 0 值赋给 pagesel 寄存器，否则将导致 bank 区域（逻辑地址在 8000h-FFFFh）与物理地址（0000h-7FFFh）内容重叠。

11) 定时/计数器控制寄存器—TCON

TCON 寄存器反映了 8051 定时器 0 和定时器 1 当前状态，并且控制这些模块的工作。

表 3-15 TCON 寄存器

Bit	Symbol	Description	Type
tcon.7	tf1	Timer 1 overflow flag	R/W
		Bit set by hardware when Timer1 overflows. This flag can be cleared by software and is automatically cleared when interrupt is processed.	
tcon.6	tr1	Timer1 Run control	R/W
		If cleared, Timer 1 stops.	
tcon.5	tf0	Timer 0 overflow flag	R/W
		Bit set by hardware when Timer 0 overflows. This flag can be cleared by software and is automatically cleared when interrupt is processed.	
tcon.4	tr0	Timer 0 Run control	R/W
		If cleared, Timer 0 stops.	
tcon.3	ie1	External interrupt 1 flag	R/W
		Set by hardware, when external interrupt int1 (edge/level, depending on settings) is observed. Cleared by hardware when interrupt is processed.	
tcon.2	it1	External interrupt 1 type control	R/W
		If set, external interrupt 1 is activated at falling edge on input pin. If cleared, external interrupt 1 is activated at low level on input pin.	
tcon.1	ie0	External interrupt 0 flag	R/W
		Set by hardware, when external interrupt int0 (edge/level, depending on settings) is observed. Cleared by hardware when interrupt is processed.	
tcon.0	it0	External interrupt 0 type control	R/W
		If set, external interrupt 0 is activated at falling edge on input pin. If cleared, external interrupt 0 is activated at low level on input pin.	

tf0，tf1(定时器 0 和定时器 1 溢出中断请求标志)，ie0 和 ie1(外部 0 和 1 中断请求标志)，在进入对应的中断服务程序时会被硬件自动清零。

12) 定时器模式寄存器—TMOD

TMOD 寄存器用于配置 8051 的定时器 0 和定时器 1。

表 3-16 TMOD 寄存器

Bit	Symbol	Description	Type
tmod.7	gate	Timer 1 gate control	R/W
		If set, enables external gate control (pin “int(1)”) for Counter 1. When “int(1)” is high, and “tr1” bit is set (Table 18), the Counter 1 is incremented every falling edge on “t1” input pin	
tmod.6	c/t	Timer 1 counter/Timer select	R/W
		Selects Timer or Counter operation. When set to 1, a Counter operation is performed, when cleared to 0, the Timer/Counter 1 will function as a	

Bit	Symbol	Description	Type
		Timer.	
tmod.5	m1	Timer 1 mode	R/W
tmod.4	m0	Selects mode for Timer/Counter 1, as shown in table below.	
tmod.3	gate	Timer 0 gate control	R/W
		If set, enables external gate control (pin “int(0)”) for Counter 0. When “int(0)” is high, and “tr0” bit is set (Table 18), the Counter 0 is incremented every falling edge on “t0” input pin	
tmod.2	c/t	Timer 0 counter/Timer select	R/W
		Selects Timer or Counter operation. When set to 1, a Counter operation is performed, when cleared to 0, the Timer/Counter 0 will function as a Timer.	
tmod.1	m1	Timer 0 mode	R/W
tmod.0	m0	Selects mode for Timer/Counter 0, as shown in table below.	

表 3-17 定时/计数方式

m0	m1	Mode	Function
0	0	Mode 0	13-bit Counter/Timer, with 5 lower bits in tl0 (tl1) register and 8 bits in th0 (th1) register (for Timer 0 or Timer 1, respectively). Note, that unlike in 80C51, the 3 high-order bits of tl0 (tl1) are zeroed whenever Mode 0 is enabled.
0	1	Mode 1	16-bit Counter/Timer.
1	0	Mode 2	8 -bit auto-reload Counter/Timer. The reload value is kept in th0 (th1), while tl0 (tl1) is incremented every machine cycle. When tl0 (tl1) overflows, a value from th0 (th1) is copied to tl0 (tl1).
1	1	Mode 3	For Timer1: Timer1 is stopped.
			For Timer0: Timer 0 acts as two independent 8 bit Timers / Counters – tl0, th0.
			- tl0 uses the Timer0 control bits and sets tf0 flag on overflow
			- th0 operates as Timer. It is enabled by tr1 bit and sets tf1 flag on overflow.

13) 定时器 0—TH0,TL0

寄存器反映了定时器 0 的状态。TH0 存储高字节，TL0 存储低字节。定时器 0 可配置为定时器或计数器。

14) 定时器 1—TH1,TL1

寄存器反映了定时器 1 的状态。TH1 存储高字节，TL1 存储低字节。定时器 0 可配置为定时器或计数器。

15) 定时器 2 控制寄存器—T2CON

T2CON 寄存器反映 8051 中定时器 2 的当前状态，并用于控制定时器 2 工作。

表 3-18 T2CON 寄存器

Bit	Symbol	Description	Type
t2con.7	t2ps	Prescaler select	R/W
		t2ps = 0 – Timer 2 is clocked with 1/12 of the oscillator frequency. t2ps = 1 – Timer 2 is clocked with 1/24 of the oscillator frequency.	
t2con.6	i3fr	Active edge selection for external interrupt “int3”, (used also as compare and capture signal)	R/W
		0 - falling edge	
		1 - rising edge	
t2con.5	i2fr	Active edge selection for external interrupt “int2”	R/W
		0 - falling edge	
		1 - rising edge	
t2con.4	t2r1	Timer 2 reload mode selection:	R/W
		0X – reload disabled 10 – Mode 0 11 – Mode 1	
t2con.3	t2r0		
t2con.2	t2cm	Timer 2 compare mode selection	R/W
		0 – Mode 0	
		1 – Mode 1	
t2con.1	t2i1	Timer 2 input selection: (t2i1, t2i0)	R/W
		00 Timer 2 stopped	
		01 input frequency f/12 or f/24	
		10 Timer 2 is incremented by falling edge detection at pin “t2”	
t2con.0	t2i0	11 input frequency f/12 or f/24 gated by external pin “t2”	

16) 定时器 2—TH2,TL2

寄存器反映了定时器 2 的状态。TH2 存储高字节，TL2 存储低字节。定时器 2 可以被配置工作在比较/捕获/重载模式。

17) 比较/捕获功能配置寄存器—CCEN

CCEN 寄存器为与定时器 2 相关的比较/捕获单元的配置寄存器（详细描述参见 3.7.3）。

表 3-19 CCEN 寄存器

Bit	Symbol	Description			Type
ccen.7	cocah3	compare/capture mode for CC3 register			R/W
ccen.6	cocal3	**cocah3**	**cocal3**	**Description**	R/W
		0	0	compare/capture disabled	
		0	1	capture on rising edge at pin cc0	
		1	0	compare enabled	

Bit	Symbol	Description			Type
		1	1	capture on write operation into register cc3	
ccen.5	cocah2	compare/capture mode for CC2 register			R/W
ccen.4	cocal2	**cocah2**	**cocal2**	**Description**	R/W
		0	0	compare/capture disabled	
		0	1	capture on rising edge at pin cc1	
		1	0	compare enabled	
		1	1	capture on write operation into register ccl2	
ccen.3	cocah1	compare/capture mode for CC1 register			R/W
ccen.2	cocal1	**cocah1**	**cocal1**	**Description**	R/W
		0	0	compare/capture disabled	
		0	1	capture on rising edge at pin cc2	
		1	0	compare enabled	
		1	1	capture on write operation into register cc1	
ccen.1	cocah0	compare/capture mode for CRC register			R/W
ccen.0	cocal0	**cocah0**	**cocal0**	**Description**	R/W
		0	0	compare/capture disabled	
		0	1	capture on falling/rising edge at pin cc3	
		1	0	compare enabled	
		1	1	capture on write operation into register crcl	

18) 比较/捕获寄存器—CC1,CC2,CC3

比较/捕获寄存器（CC1,CC2,CC3）为 16 位的寄存器，用于定时器 2 相关的比较/捕获单元的操作（详细描述参考 3.7.3）。CCn 寄存器中，CCHn 存储高字节，CCLn 存储低字节。

- CCL1,CCH1,CCL2,CCH2,CCL3,CCH3 寄存器位于 SFR 存储器空间。

19) 比较/重装入/捕获寄存器—CRCH,CRCL

比较/重装入/捕获寄存器 CRC 为 16 位宽的寄存器，用于与定时器 2 相关的对比/捕获单元的操作。（详细描述参见 3.7.3）。CRCH 存储高字节，CRCL 存储低字节。

20) 串口 0 控制寄存器—S0CON

S0CON 寄存器控制串口 0 的功能。

表 3-20 S0CON 寄存器

Bit	Symbol	Description	Type
s0con.7	sm0	Serial Port 0 mode select	R/W
s0con.6	sm1	(see 表 ***3-21 串口 0 工作方式与波特率***)	
s0con.5	sm20	Multiprocessor communication enable (see 2) “UART0 多处理器通信”).	R/W
s0con.4	ren0	Serial reception enable	R/W

		If set HIGH serial reception at Serial Port 0 is enabled. Otherwise serial reception at Serial Port 0 is disabled.	
s0con.3	tb80	Transmitter bit 8	R/W
		This bit is used while transmitting data through Serial Port 0 in Modes 2 and 3. The state of this bit corresponds with the state of the 9th transmitted bit (e.g. parity check or multiprocessor communication). It is controlled by software.	
s0con.2	rb80	Received bit 8	R/W
		This bit is used while receiving data through Serial Port 0 in Modes 2 and 3. It reflects the state of the 9th received bit.	
		In Mode 1, if multiprocessor communication is enabled (sm20 = 0), this bit is the stop bit that was received (参见 2) “UART0 多处理器通信”).	
		In Mode 0 this bit is not used.	
s0con.1	ti0	Transmit interrupt flag	R/W
		It indicates completion of a serial transmission at Serial Port 0.	
		It is set by hardware at the end of bit 8 in mode 0 or at the beginning of a stop bit in other modes. It must be cleared by software.	
s0con.0	ri0	Receive interrupt flag	R/W
		It is set by hardware after completion of a serial reception at Serial Port 0.	
		It is set by hardware at the end of bit 8 in mode 0 or in the middle of a stop bit in other modes.	
		It must be cleared by software.	

表 3-21 串口 0 工作方式与波特率

sm0	sm1	Mode	Description	Baud Rate	
0	0	Mode 0	shift register	Fclk/12	
0	1	Mode 1	8-bit UART	Variable (details below the table)	
1	0	Mode 2	9-bit UART	Depends on smod (pcon.7) value	
				smod	**Baud Rate**
				0	Fclk/64
				1	Fclk/32
1	1	Mode 3	9-bit UART	Variable (details below the table)	

串口 0 在方式 1 或 3 下工作的波特率为：

bd(adcon.7=0):

$$\text{波特率}=\frac{2^{SMOD}*Fclk}{32}*(\textit{定时器 1 overflow rate})$$

bd(adcon.7=1):

$$波特率 = \frac{2^{SMOD} * Fclk}{64 * \left(2^{10} - s0rel\right)}$$

图 3-8 UART0 波特率的计算

- smod(pcon.7)　　串口 0 波特率选择标
- s0rel　　S0REL 寄存器的值（s0relh,s0rell）(见 P50)
- bd(adcon.7)　　“adcon”寄存器最高有效位

21) 串口 0 数据缓冲寄存器—S0BUF

将数据写入该寄存器时，在串口输出缓冲设置数据，并通过串口 0 进行发送。从 S0BUF 读取数据时，从串口接收缓冲读取数据。

22) 串口 0 重装入寄存器—S0RELH,S0RELL

串口 0 重装入寄存器用于生成串口 0 波特率。（参见图 3-8 UART0 波特率的计算）。共占用了 10 位，其中低 8 位来自 S0RELL，高 2 位来自 S0RELH (s0relh.1, s0relh.0)。

23) 串口 1 控制寄存器—S1CON

S1CON 寄存器控制串口 1 的功能。

表 3-22 S1CON 寄存器

Bit	Symbol	Description	Type
s1con.7	sm	Serial Port 1 mode select	R/W
		sm = 0: Mode A selected for Serial Port 1 - 9-bit UART sm = 1: Mode B selected for Serial Port 1 - 8-bit UART	
s1con.6	-	not used, read as 0	R
s1con.5	sm21	Multiprocessor communication enable (see2) UART1 多处理器通信).	R/W
s1con.4	ren1	Serial reception enable	R/W
		If set HIGH serial reception at Serial Port 1 is enabled. Otherwise serial reception at Serial Port 1 is disabled.	
s1con.3	tb81	Transmitter bit 8	R/W
		This bit is used while transmitting data through Serial Port 1 in Mode A. The state of this bit corresponds with the state of the 9th transmitted bit (e.g. parity check or multiprocessor communication). It is controlled by software.	
s1con.2	rb81	Received bit 8	R/W
		This bit is used while receiving data through Serial Port 1 in Mode A. It reflects the state of the 9th received bit.	
		In Mode B, if multiprocessor communication is enabled (sm21 = 0), this bit is the stop bit that was received (see 2) “UART0 多处理器通信”).	
s1con.1	ti1	Transmit interrupt flag	R/W
		It indicates completion of a serial transmission at Serial Port 1.	

Bit	Symbol	Description	Type
		It is set by hardware at the beginning of a stop bit in mode A or B. It must be cleared by software.	
s1con.0	ri1	Receive interrupt flag	R/W
		It is set by hardware after completion of a serial reception at Serial Port 1.	
		It is set by hardware in the middle of a stop bit in mode A or B. It must be cleared by software.	

串口 1 波特率：

$$\text{波特率} = \frac{\text{Fclk}}{32 * \left(2^{10} - s1rel\right)}$$

图 3-9 UART1 波特率计算

- s1rel　　S1REL 寄存器的值（“s1relh”，“s1rell”）(参见 P51)

24) 串口 1 数据缓冲—S1BUF

将数据写入该寄存器，在串口输出缓冲设置数据，并经过串口 1 发送。读取 S1BUF 时，从串口接收缓冲接收数据。

25) 串口 1 重装入寄存器—S1RELH,S1RELL

串口重装入寄存器用于产生串口 1 波特率。（参见图 3-9 UART1 波特率计算）。共占用了 10 位，其中低 8 位来自 S1RELL，高 2 位来自 S1RELH(s1relh.1 ,s1relh.0)。

26) 看门狗重装入寄存器—WDTREL

WDTREL 寄存器存储看门狗定时器高 7 位的重载值（见 3.8 看门狗），同时配置看门狗定时器的预分频。

表 3-23 WDTREL 寄存器

Bit	Symbol	Description	Type
wdtrel.7	-	Prescaler select	R/W
		When set, the watchdog is clocked through an additional divide- by-16 prescaler.	
wdtrel.6 wdtrel.5 wdtrel.4 wdtrel.3 wdtrel.2 wdtrel.1 wdtrel.0	-	Watchdog reload value	R/W
		Reload value for the highest 7 bits of the watchdog Timer. This value is loaded to the Watchdog Timer when a refresh is triggered by a consecutive setting of bits wdt (ien0.6) and swdt (ien1.6). For details see Watchdog Timer description (3.8 看门狗).	

27) 中断允许寄存器 0——IEN0

- 实现了外部中断使能标志位。
- 实现了看门狗刷新标志位。
- 实现了定时器 0 中断使能标志位。
- 实现了定时器 1 中断使能标志位。
- 实现了定时器 2 中断使能标志位。
- 实现了串口 0 中断使能标志位。

表 3-24 IEN0 寄存器

Bit	Symbol	Description	Type
ien0.7	eal	Interrupts enable	R/W
		When set to 0 – all interrupts are disabled Otherwise enabling each interrupt is done by setting the corresponding interrupt enable bit	
ien0.6	wdt	Watchdog Timer refresh flag	R/W
		Set to initiate a refresh of the watchdog Timer. Must be set directly before swdt (ien1.6) is set to prevent an unintentional refresh of the watchdog Timer. The wdt bit is cleared by hardware after the next instruction executed after the one that had set this bit, so that watchdog refresh can be done only with direct sequence of setting wdt and swdt.	
ien0.5	et2	Timer 2 interrupt enable	R/W
		When et2=0 Timer 2 interrupt is disabled. When et2=1 and eal=1 Timer 2 interrupt is enabled.	
ien0.4	es0	Serial Port 0 interrupt enable	R/W
		When es0=0 Serial Port 0 interrupt is disabled. When es0=1 and eal=1 Serial Port 0 interrupt is enabled.	
ien0.3	et1	Timer 1 overflow interrupt enable	R/W
		When et1=0 Timer 0 overflow interrupt is disabled. When et1=1 and eal=1 Timer 1 overflow interrupt is enabled.	
ien0.2	ex1	External interrupt 1 enable	R/W
		When ex1=0 external interrupt 1 is disabled. When ex1=1 and eal=1 external interrupt 1 is enabled.	
ien0.1	et0	Timer 0 overflow interrupt enable	R/W
		When et0=0 Timer 0 overflow interrupt is disabled. When et0=1 and eal=1 Timer 0 overflow interrupt is enabled.	
ien0.0	ex0	External interrupt 0 enable	R/W
		When ex0=0 external interrupt 0 is disabled. When ex0=1 and eal=1 external interrupt 0 is enabled.	

28) 中断允许寄存器 1—IEN1

- 实现了外部中断使能标志位。

- 实现了定时器 2 中断使能标志位。
- 实现了看门狗定时器启动/刷新标志位。

表 3-25 IEN1 寄存器

Bit	Symbol	Description	Type
ien1.7	exen2	Timer 2 external reload interrupt enable When exen2=0, Timer 2 external reload interrupt 2 is disabled. When exen2=1 and eal=1, Timer 2 external reload interrupt 2 is enabled.	R/W
ien1.6	swdt	Watchdog Timer start/refresh flag. Set to activate/refresh the watchdog Timer. When set directly after setting wdt (ien0.6), a watchdog Timer refresh is performed. This bit is immediately cleared by hardware.	R/W
ien1.5	ex6	External interrupt 6 enable When ex6=0 external interrupt 6 is disabled. When ex6=1 and eal=1 external interrupt 6 is enabled.	R/W
ien1.4	ex5	External interrupt 5 enable When ex5=0 external interrupt 5 is disabled. When ex5=1 and eal=1 external interrupt 5 is enabled.	R/W
ien1.3	ex4	External interrupt 4 enable When ex4=0 external interrupt 4 is disabled. When ex4=1 and eal=1 external interrupt 4 is enabled.	R/W
ien1.2	ex3	External interrupt 3 enable When ex3=0 external interrupt 3 is disabled. When ex3=1 and eal=1 external interrupt 3 is enabled.	R/W
ien1.1	ex2	External interrupt 2 enable When ex2=0 external interrupt 2 is disabled. When ex2=1 and eal=1 external interrupt 2 is enabled.	R/W
ien1.0	ex7	External interrupt 7 enable When ex7=0 external interrupt 7 is disabled. When ex7=1 and eal=1 external interrupt 7 is enabled.	R/W

29) 中断允许寄存器 2—IEN2

- 实现了串口 1 中断使能标志位。

表 3-26 IEN2 寄存器

Bit	Symbol	Description	Type
ien2.7	-	not used, read as 0	R
ien2.6	-		
ien2.5	ex12	External interrupt 12 enable Not used in the HME-M1 device, read as 0	R

Bit	Symbol	Description	Type
ien2.4	ex11	External interrupt 11 enable	R
		Not used in the HME-M1 device	
ien2.3	ex10	External interrupt 10 enable	R
		Not used in the HME-M1 device, read as 0	
ien2.2	ex9	External interrupt 9 enable	R
		Not used in the HME-M1 device, read as 0	
ien2.1	ex8	External interrupt 8 enable	R
		Not used in the HME-M1 device, read as 0	
ien2.0	es1	Serial Port 1 interrupt enable	R/W
		When es1=0 Serial Port 1 interrupt is disabled. When es1=1 and eal=1 Serial Port 1 interrupt is enabled.	

30) 中断优先级寄存器——IP0,IP1

13 个中断源可分为 6 个优先组。每个组都有 4 个优先级可供选择。通过设定 IP0 和 IP1 寄存器内相应的值，可以实现选定的优先级。中断优先级寄存器中的值按照下表对每个中断源进行了优先级的定义。

- 实现了看门狗定时器状态标志位。

表 3-27 IP0 寄存器

Bit	Symbol	Description	Type
ip0.7	-	not used, read as 0	R
ip0.6	wdts	Watchdog Timer status flag	R/W
		Set by hardware when the watchdog Timer reset occurs.	
ip0.5	-	Interrupt priority	R/W
ip0.4	-	Each bit together with corresponding bit from IP1 register specifies the priority level of the respective interrupt priority group.	
ip0.3	-		
ip0.2	-		
ip0.1	-		
ip0.0	-		

表 3-28 IP1 寄存器

Bit	Symbol	Description	Type
ip1.7	-	not used, read as 0	R
ip1.6	-		
ip1.5	-	Interrupt priority	R/W
ip1.4	-	Each bit together with corresponding bit from IP0 register specifies the priority level of the respective interrupt priority group.	
ip1.3	-		
ip1.2	-		

Bit	Symbol	Description	Type
ip1.1	-		
ip1.0	-		

表 3-29 优先组

Group	Corresponding interrupt bits	Highest priority → lowest priority (in the same interrupt group)			
0	ip1.0, ip0.0	External interrupt 0	Serial channel 1 interrupt	Not used	External interrupt 7
1	ip1.1, ip0.1	Timer 0 Interrupt	Not used	Not used	External interrupt 2
2	ip1.2, ip0.2	External interrupt 1	Not used	Not used	External interrupt 3
3	ip1.3, ip0.3	Timer 1 Interrupt	Not used	Not used	External interrupt 4
4	ip1.4, ip0.4	Serial channel 0 interrupt	Not used	Not used	External interrupt 5
5	ip1.5, ip0.5	Timer 2 Interrupt	Not used	Not used	External interrupt 6

表 3-30 优先级

ip1.x	ip0.x	
0	0	Level 0 (lowest)
0	1	Level 1
1	0	Level 2
1	1	Level 3 (highest)

X 是优先组的组号。

31) 中断请求控制寄存器——IRCON

- 实现了外部中断允许标志位。
- 实现了定时器 2 溢出/重装入标志位。

表 3-31 IRCON 寄存器

Bit	Symbol	Description	Type
ircon.7	exf2	Timer 2 external reload flag	R/W
ircon.6	tf2	Timer 2 overflow flag	R/W
ircon.5	iex6	External interrupt 6 edge flag	R/W
ircon.4	iex5	External interrupt 5 edge flag	R/W
ircon.3	iex4	External interrupt 4 edge flag	R/W

Bit	Symbol	Description	Type
ircon.2	iex3	External interrupt 3 edge flag	R/W
ircon.1	iex2	External interrupt 2 edge flag	R/W
ircon.0	iex7	External interrupt 7 edge flag	R/W

32) 电源控制寄存器—PCON

- “pmw”用于实现程序存储器写模式。

表 3-32 PCON 寄存器

Bit	Symbol	Description	Type
pcon.7	smod	Serial Port 0 baud rate select (see 表 *3-21*) (baud rate doubler)	R/W
pcon.6	wdt_tm	Watchdog Timer Test Mode flag	R/W
		When set to 1, the fclk/12 divider at the input of the Watchdog Timer is skipped	
pcon.5	isr_tm	Interrupt Service Routine Test Mode flag	R/W
		When set to 1, the interrupt vectors assigned to Timer 0 & 1, Serial Port 0 & 1, SPI and I^2C interfaces can be triggered only with the use of external inputs of the core	
pcon.4	pmw	Program memory write mode	R/W
		Setting this bit enables the program memory write mode	
pcon.3	p2sel	High-order address byte configuration bit	R/W
		Chooses the higher byte of address (“memaddr[15:8]”) during MOVX @Ri operations;	
		when 0, the “memaddr[15:8]” = “p2reg” when 1, the “memaddr[15:8]” = ADDR_HIGH_RI	
		The “p2reg” is the contents of Port2 output register. The ADDR_HIGH_RI = 8’h0 is the parameter defined before synthesis.	
pcon.2	gf0	General Purpose Flag	R/W
pcon.1	stop	Stop mode control	R/W
		Setting this bit activates the Stop Mode. This bit is always read as 0	
pcon.0	idle	Idle mode control	R/W
		Setting this bit activates the Idle Mode. This bit is always read as 0	

33) 时钟控制寄存器—CKCON

寄存器的值定义了在读取/写入外部数据、程序存储器时内部生成的等待状态的个数。

表 3-33 CKCON 寄存器

Bit	Symbol	Description	Type
ckcon.7	-	not used, read as 0	R
ckcon.6	-	Program memory wait state control	R/W
ckcon.5	-		
ckcon.4	-		
ckcon.3	-	not used, read as 0	R
ckcon.2	-	External data memory stretch cycle control	R/W
ckcon.1	-		
ckcon.0	-		

表 3-34 代码存储器等待状态周期的宽度

CKCON Register			Wait Value	Read Signals Width		Write Signals Width	
ckcon.6	ckcon.5	ckcon.4		memaddr	mempsrd	memaddr	mempsrd
0	0	0	0	1	1	2	1
0	0	1	1	2	2	3	1
0	1	0	2	3	3	4	2
0	1	1	3	4	4	5	3
1	1	0	4	5	5	6	4
1	0	1	5	6	6	7	5
1	1	0	6	7	7	8	6
1	1	1	7	8	8	9	7

表 3-35 外部数据存储器等待状态周期的宽度

CKCON Register			Strech Value	Read Signals Width		Write Signals Width	
ckcon.2	ckcon.1	ckcon.0		memaddr	memrd	memaddr	memwr
0	0	0	0	1	1	2	1
0	0	1	1	2	2	3	1
0	1	0	2	3	3	4	2
0	1	1	3	4	4	5	3
1	0	0	4	5	5	6	4
1	0	1	5	6	6	7	5
1	1	0	6	7	7	8	6
1	1	1	7	8	8	9	7

34) 串口 0 波特率选择寄存器—ADCON

寄存器的最高有效位用于生成串口 0 的波特率（参见表 3-21）。

表 3-36 ADCON 寄存器

Bit	Symbol	Description	Type
adcon.7	bd	Serial Port 0 baud rate select (in modes 1 and 3)	R/W
		When 1, additional internal baud rate generator is used, otherwise Timer 1 overflow is used	
adcon.6	-	not used, read as 0	R
adcon.5	-		
adcon.4	-		
adcon.3	-		
adcon.2	-		
adcon.1	-		
adcon.0	-		

35) 乘法/除法寄存器—MD0,MD1,MD2,MD3,MD4,MD5

MD0—MD5 是用于 MDU 操作的寄存器。关于 MDU 功能的详细描述，请参见 3.5 乘-除单元。

36) 算术控制寄存器—ARCON

ARCON 寄存器用于控制 MDU 操作，并报告 MDU 当前状态。

表 3-37 ARCON 寄存器

Bit	Symbol	Description	Type
arcon.7	mdef	MDU Error flag MDEF	R
		Indicates an improperly performed operation (when one of the arithmetic operations has been restarted or interrupted by a new operation).	
arcon.6	mdov	MDU Overflow flag MDOV	R
		Overflow occurrence in the MDU operation.	
arcon.5	slr	Shift direction	R/W
		slr = 0 - shift left operation. slr = 1 - shift right operation.	
arcon.4	sc.4	Shift counter	R/W
arcon.3	sc.3	When set to all ‘0’s, normalize operation is selected. After normalization, the “sc.0” … “sc.4” contain the number of normalizing shifts performed.	
arcon.2	sc.2		
arcon.1	sc.1		
arcon.0	sc.0	When at least one of these bit is set high shift operation is selected. The number of shifts performed is determined by the number written to “sc.4”.., “sc.0”, where “sc.4” is the MSB.	

37) I²C 数据寄存器—I2CDAT

I2CDAT 寄存器包含一个从 I²C 总线发送或接收的字节。当这一 SFR 未处于移位状态时，CPU 可以通过直接寻址的方式读写该 8 位 SFR。由于 I2CDAT 寄存器没有影子寄存器或双重缓冲，用户只能在 I²C 中断

服务程序中读取 I2CDAT。有关 I²C 端口功能的详尽描述，请见 3.9.1 节 I2C。

38) I²C 地址寄存器—I2CADR

I2CADR 寄存器存储 8051 I2C 作为从设备时的地址。此地址用于识别 I²C 总线上是否有外部设备试图寻址该 8051 从设备。I²C 接口功能描述请参见 3.9.1 节 I2C。

表 3-38 I2CADR 寄存器

Bit	Symbol	Description	Type
i2cadr.7	adr	Own I²C slave address (7 bit)	R/W
i2cadr.6			
i2cadr.5			
i2cadr.4			
i2cadr.3			
i2cadr.2			
i2cadr.1			
i2cadr.0	gc	General Call Address Acknowledge	R/W
		If this bit is set, the general call address is recognized; otherwise it is ignored.	

39) I²C 控制寄存器 — I2CCON

I2CCON 寄存器控制 I²C 接口的操作。CPU 可以通过直接寻址的方式读写该 8 位 SFR。其中，寄存器中 2 位受 I²C 硬件影响：当串行中断请求发生时，硬件置 1“si”，I²C 总线规定的 STOP 状态在出现在总线时硬件清 0“sto”位。I²C 接口功能的详细描述请参见 3.9.1 节“ I2C”。关于时钟率设置，请参见 3) 。

表 3-39 I2CCON 寄存器

Bit	Symbol	Description	Type
i2ccon.7	cr2	Clock rate bit 2	R/W
i2ccon.6	ens1	I²C enable bit	R/W
		When ens1='0' the "sdao" and "sclo" outputs are set to 1, that drives the output pads of the chip in high impedance, and "sdai" and "scli" input signals are ignored.	
		When ens1='1' I²C component is enabled.	
i2ccon.5	sta	START Flag	R/W
		When sta='1', the I²C component checks the I²C bus status and if the bus is free a START condition is generated.	
i2ccon.4	sto	STOP Flag	R/W
		When sto='1' and I²C interface is in master mode, a STOP condition is transmitted to the I²C bus.	
i2ccon.3	si	Serial Interrupt Flag	R/W
		The "si "is set by hardware when one of 25 out of 26 possible I²C states is entered (2.4.46). The only state that does not set the "si" is state	

Bit	Symbol	Description	Type
		F8h, which indicates that no relevant state information is available. The “si” flag must be cleared by software. In order to clear the “si” bit, ‘0’ must be written to this bit. Writing a ‘1’ to si bit does not change value of the “si”.	
i2ccon.2	aa	Assert Acknowledge Flag	R/W
		When aa=’1’, an “acknowledge” will be returned when: - the “own slave address” has been received - the general call address has been received while gc bit in i2cadr register was set - a data byte has been received while I²C was in master receiver mode - a data byte has been received while I²C was in slave receiver mode	
		When aa=’0’, an “not acknowledge” will be returned when: - a data byte has been received while I²C was in master receiver mode - a data byte has been received whileI²C was in slave receiver mode	
i2ccon.1	cr1	Clock rate bit 1	R/W
i2ccon.0	cr0	Clock rate bit 0	R/W

40) I2C 状态寄存器 — I2CSTA

I2CSTA 寄存器的值反映了 I²C 接口电路当前状态。有关 I²C 接口功能详细描述，请参考 3.9.1 节 I2C。

表 3-40 I2CSTA 寄存器

Bit	Symbol	Description	Type
i2csta.7 i2csta.6 i2csta.5 i2csta.4 i2csta.3	-	I²CStatus Code	R
i2csta.2 i2csta.1 i2csta.0	-	Not implemented, read as 0	R

41) SPI 串口外设状态寄存器 — SPSTA

SPSTA 寄存器包含多个标志位，有数据传送完成标志，写冲突状态标志，“ssn”(从选择)管脚（模式错误异常）逻辑电平不一致。

表 3-41 SPSTA 寄存器

Bit	Symbol	Description	Type
spsta.7	spif	Serial Peripheral Data Transfer Flag	R
		Set by hardware upon data transfer completion.	

Bit	Symbol	Description	Type
		Cleared by hardware when data transfer is in progress. Can be also cleared by reading the “spsta” register with the “spif” bit set, and then reading the “spdat” register.	
spsta.6	wcol	Write Collision Flag	R
		Set by hardware upon write collision to “spdat”.	
		Cleared by hardware upon data transfer completion when no collision has occurred. Can be also cleared by an access to “spsta” register and an access to “spdat” register.	
spsta.5	sserr	Synchronous Serial Slave Error Flag	R
		Set by hardware when “ssn” input is de-asserted before the end of receive sequence.	
		Cleared by disabling the SPI module (clearing “spen” bit in “spcon” register).	
spsta.4	modf	Mode Fault Flag	R
		Set by hardware when the “ssn” pin level is in conflict with actual mode of the SPI controller (configured as master while externally selected as slave).	
		Cleared by hardware when the “ssn” pin is at appropriate level. Can be also cleared by software by reading the “spsta” register with “modf” bit set.	
spsta.3 spsta.2 spsta.1 spsta.0	-	not used, read as 0	R

42) SPI 串行外设控制寄存器 — SPCON

串行外设控制寄存器用于配置 SPI 模块。SPCON 选择主时钟频率，将模块配置为主设备或从设备，选择串行时钟极性和相位，使能”ssn”输入，使能/禁止整个 SPI 模块的工作。

表 3-42 SPCON 寄存器

Bit	Symbol	Description	Type
spcon.7	spr2	Serial Peripheral Rate 2	R/W
		Together with “spr1” and “spr0” defines the clock rate in master mode.	
spcon.6	spen	Serial Peripheral Enable	R/W
		When cleared disables the SPI interface. When set enables the SPI interface.	
spcon.5	ssdis	SS Disable	R/W
		When cleared enables the “ssn” input in both Master and Slave modes. When set disables the “ssn” input in both Master and Slave modes.	
		In Slave mode, this bit has no effect if “cpha”=0. When “ssdis” is set, no	

<table>
<tr><th>Bit</th><th>Symbol</th><th>Description</th><th>Type</th></tr>
<tr><td></td><td></td><td>“modf” interrupt request will be generated.</td><td></td></tr>
<tr><td>spcon.4</td><td>mstr</td><td>Serial Peripheral Master</td><td>R/W</td></tr>
<tr><td></td><td></td><td>When cleared configures the SPI as a Slave. When set configures the SPI as a Master.</td><td></td></tr>
<tr><td>spcon.3</td><td>cpol</td><td>Clock Polarity</td><td>R/W</td></tr>
<tr><td></td><td></td><td>When cleared, the “sck” is set to 0 in idle state. When set, the “sck” is set to 1 in idle state.</td><td></td></tr>
<tr><td>spcon.2</td><td>cpha</td><td>Clock Phase</td><td>R/W</td></tr>
<tr><td></td><td></td><td>When cleared, data is sampled when the “scki”/”scko” leaves the idle state (see “cpol”). When set, data is sampled when the “scki”/”scko” returns to idle state (see “cpol”).</td><td></td></tr>
<tr><td>spcon.1</td><td>spr1</td><td>Serial Peripheral Rate</td><td>R/W</td></tr>
<tr><td>spcon.0</td><td>spr0</td><td>Together with “spr2” specify the serial clock rate in Master mode.</td><td>R/W</td></tr>
</table>

spr2	spr1	spr0	Serial Peripheral Rate
0	0	0	Fclk / 2
0	0	1	Fclk / 4
0	1	0	Fclk / 8
0	1	1	Fclk / 16
1	0	0	Fclk / 32
1	0	1	Fclk / 64
1	1	0	Fclk / 128
1	1	1	the master clock is not generated (when “cpol” = ‘1’ on the “scko” output is high level, otherwise is low level)

43) SPI 串行外设数据寄存器 — SPDAT

SPDAT 是 “接收数据”寄存器的读/写缓冲。当向 SPDAT 写数据时，数据被直接放置到移位寄存器（没有发送缓冲器）。读取 SPDAT 时，返回位于接收缓存中的值，而不是移位寄存器中的值。

44) SPI 串行外设从选择寄存器 — SPSSN

对 SPSSN 寄存器低 4 位的读/写用于控制 CPU 输出信号“spssn [3:0]”。对“spssn [3:0]”写入数据可以立即出现在 SPSSN[3:0]输出信号上。输出每一位都可以选择一个外部独立的 SPI 从设备。

45) 软件复位寄存器 — SRST

软件复位可通过 SRST SFR 寄存器实现。SRST 寄存器中的内容如下。

表 3-43 SRST 寄存器

Bit	Symbol	Description	Type
srst.7	-	not used, read as 0	R
srst.6	-		
srst.5	-		
srst.4	-		
srst.3	-		
srst.2	-		
srst.1	-		
srst.0	srstreq	Software reset request.	R/W
		Writing ‘0’ value to this bit will have no effect. Single writing ‘1’ value to this bit will have no effect. Double writing ‘1’ value (in two consecutive instructions) will generate an internal software reset.	
		Reading this bit will inform about the reset source:	
		if ‘0’ – source of last reset sequence was not a software reset (hardware, watchdog or debugger reset); if ‘1’ – source of last reset sequence was a software reset (caused by double writing ‘1’ value to the “srstreq” bit).	

3.4 扩展特殊功能寄存器

HME-M1 扩展了一些专用 SFR。下表对扩展的 SFR 进行了详细描述。

表 3-44 外部特殊功能寄存器映射

register	location	attribute	Reset value	description
I2CSPISEL	ABh	R/W	8’H00	Bit0: SPI interface source. must be 1 Bit1：I²C IO select, contro the device IO attribute 0: the IO is used as general user IO 1: the IO is used as fixed I²C IO Bit[7:2] Reserved
ISCADDR0	Ach	R/W	8’H00	Reconfiguration start address [7:0]
ISCADDR1	ADh	R/W	8’H00	Reconfiguration start address [15:8]
ISCADDR2	AEh	R/W	8’H00	Reconfiguration start address [23:16]
ISCADDR3	AFh	R/W	8’H00	Reconfiguration start address [31:24]
MISCCON	F8h	R/W	8’H00	Bit0 ISCEN: reconfiguration enable, be used to trigger the reconfiguration sequence 0: disable 1:enable BIT1 DPRAMWP: DPRAM write protect from

				FP 0:disable write 1:enable write Bit2 PLLLOCK: read only pll_lock, the status of pll 0: not lock 1: lock Bit3 PLLPWD: pll_pwrdn, pll power down 0: open the pll power 1: close the pll power Bit4 CLKCPUTRIG: Triggers CLKCPUON and make it active when changes from 0 to 1. Bit5 CLKO0TRIG: Triggers CLKO0CON and make it active when changes from 0 to 1. Bit6 CLKO1TRIG: Triggers CLKO1CON and make it active when changes from 0 to 1. Bit7 CLKO2TRIG: Triggers CLKO2CON and make it active when changes from 0 to1.
CLKCPUCON	F9h	R/W	8'H00	Clk configuration for clkcpu or clkper Bit[2:0] the ratio of divider 000: bypass 001: divided by 2 010: divided by 4 011: divided by 8 100: divided by 16 101: divided by 32 110: divided by 64 111: divided by 128 Bit3:switch_gclk 0: from CLKN(clock pin or OSC) 1: from PLL Bit[7:4] Reserved
CLKO0CON	FAh	R/W	8'H00	Clk configuration for clkout0 Bit[2:0] the ratio of divider 000: bypass 001: divided by 2 010: divided by 4 011: divided by 8 100: divided by 16 101: divided by 32 110: divided by 64 111: divided by 128 Bit3:switch_gclk

				0: from CLKN(clock pin or OSC) 1: from PLL Bit[7:4] Reserved
CLKO1CON	FBh	R/W	8'H00	Clk configuration for clkout1 Bit[2:0] the ratio of divider 000: bypass 001: divided by 2 010: divided by 4 011: divided by 8 100: divided by 16 101: divided by 32 110: divided by 64 111: divided by 128 Bit3:switch_gclk 0: from CLKN(clock pin or OSC) 1: from PLL Bit[7:4] Reserved
CLKO2CON	FCh	R/W	8'H00	Clk configuration for clkout2 Bit[2:0] the ratio of divider 000: bypass 001: divided by 2 010: divided by 4 011: divided by 8 100: divided by 16 101: divided by 32 110: divided by 64 111: divided by 128 Bit3:switch_gclk 0: from CLKN(clock pin or OSC) 1: from PLL Bit[7:4] Reserved

注意：

请谨慎使用 PLL 电源开关功能。关掉 PLL 电源时，MISCCON [7:4]必须有一位同时置位。如果重新打开 PLL 电源时，MISCCON [7:3]必须同时清零。

3.5 乘-除单元 MDU

3.5.1 概述

MDU——乘法-除法器，是片上算术协处理器，允许 8051 可以执行扩展的算术操作。乘法-除法器提供 32 位的除法和 16 位的乘法，以及移位和规格化操作。这些操作都是无符号的整数操作。

MDU 由 7 个寄存器控制，这 7 个寄存器存储映像为特殊功能寄存器（SFR）。算术单元允许与 CPU 之间独立的并行操作。

操作数和操作结果保存在“md0”…“md5”寄存器中（参见 35) 第 58 页）。这一模块由”arcon”寄存器（36) P58）控制。MDU 的计算都会覆盖它的操作数。

3.5.2 MDU 操作说明

MDU 操作包含三个阶段：

1) 装载 mdx 寄存器

MDC 执行的运算类型由“mdx”寄存器写入顺序决定。

表 3-45 MDU 寄存器写入顺序

Operation	32 bit/16 bit	16 bit / 16 bit	16 bit x 16 bit	shifting normalizing
first write	md0 D’endL	md0 D’endL	md0 M’andL	md0 LSB
	md1 D’end	md1 D’endH	md4 M’erL	md1
	md2 D’end		md1 M’andH	md2
	md3 D’endH			md3 MSB
	md4 D’orL	md4 D’orL		
last write	md5 D’orH	md5 D’orH	md5 M’erH	arcon

写入“md0”可启动后续操作。后面的写操作必须按照“表 3-45 MDU 寄存器写入顺序”中的执行，用于决定特定的 MDU 操作。最后一次的写操作启动所选的操作。

SFR 控制器检测上述某些操作并将控制传输到算术操作 FSM。写入操作访问“md0”最后到达“md5”，其中一个写访问发生在“md2”或“md3”时，会选中“32 位/16 位的除法”。其他情况下，在最终写入“md5”之前，写操作发生在“md1”或”md4”，则会选择一个“16 位/16 位的除法或乘法”。写入“md4”选择“16 位/16 位除法”，写入“md1”选择“乘法”。在复位之后， MDU 将设置“乘法”作为默认操作。任何写入“arcon”的访问都会强制移位或规格化。

2) 执行计算

在执行操作的过程中，MDU 并行于 CPU 独立工作。

表 3-46 MDU 操作执行次数

Operation	Number of clock cycles	
Division 32bit/16bit	17 clock cycles	
Division 16bit/16bit	9 clock cycles	
Multiplication	11 clock cycles	
Shift	min 3 clock cycles (sc = 01h)	max 18 clock cycles (sc = 1Fh)
Normalize	min 4 clock cycles (sc <- 01h)	max 19 clock cycles (sc <- 1Fh)

3) 从 mdx 寄存器中读取结果

表 3-47 MDU 寄存器读取顺序

Operation	32 bit/16 bit	16 bit / 16 bit	16 bit x 16 bit	Shifting normalizing
first read	md0 QuoL	md0 QuoL	md0 PrL	md0 LSB
	md1 Quo	md1 QuoH	md1 Pr	md1
	md2 Quo		md2 Pr	md2
	md3 QuoH			
	md4 RemL	md4 RemL		
last read	md5 RemH	md5 RemH	md3 PrH	md3 MSB

“mdx”寄存器读取的顺序并不重要，但是最后的读取操作（从“md5” -除法和“md3”-乘法，移位或规格化）决定了整个计算过程的结束（阶段三的最终点）。

4) 规格化

左移操作移除存储在”md0”…“md3”寄存器中 32 位无符号整数变量所有前导的 0。当“md3”寄存器中的最高有效位是一个‘1’时，整个操作完成。规格化之后，“arcon.4”（MSB）…“arcon0”（LSB）包含已完成的左操作次数。

5) 移位

在移位操作中，存储在“md0”…“md3”寄存器（后者包含 MSB）中 32 位的整数变量被左移或右移特定位数。“slr”位（“arcon.5”）定义移位方向，“arcon.4”…“arcon0”位指定移位个数（一定为非 0）。在移位操作中，右移时 0 从最左边进入“md3”，左移时 0 从最右边进入“md0”。

6) “mdef”标志（arcon,7-表 3-37 ARCON 寄存器）

“mdef”错误标志表明运行了不正确的操作（当某个算术操作重启或被新操作打断）。第一个写操作到“md0”时错误判断机制自动使能，在最后从阶段三 “md3”（乘法或移位/规格）或 “md5”（除法）读取指令时错误判断机制禁用。

错误标志设置条件：

- MDU 操作的阶段二(重启或运算中断)，写入“mdx”寄存器（“md0”…“md5”和“arcon”中的任一个）
- 错误标志机制使能，MDU 操作的第二阶段，读取“mdx”寄存器。这种情况下会设置错误标志但是计算不会中断。

只有读取“arcon”寄存器之后，错误标志才会被复位。错误标志位是只读的。

7) “mdov”标志（arcon.6-表 3-37 ARCON 寄存器）

“mdov”溢出标志在如下情况之一发生时设置：

- 除 0
- 乘法结果大于 0000 FFFFH
- “md3”的 MSB 设置后（“md3.7”=‘1’）,开始规格化 MDU 中不符合上述情况的任一操作都会清零溢出标志。需注意，溢出标志只受硬件控制，软件不可写。

3.6 中断控制器

8051 有 13 个中断源：

- 8 个外部中断（有些与比较/捕获单元中断相结合）
- 3 个定时器中断（Timer 0，1 和 2）
- 2 个串口中断（serial 0，1）

13 个中断源分为 6 个优先组。每个优先组有 4 个优先级，可以任选其一。每个中断源都有独立标志，向量和使能位。这些中断可以全局使能或禁用。

3.6.1 中断源

1) 外部中断 0 和 1

外部中断 0 和 1 可编程为低电平触发或下降沿事件触发，这取决于 IT0 和 IT1 位。寄存器 TCON 中的 IE0 和 IE1 位是检测产生中断的标志。

在边沿触发方式中，INTx 输入在每个机器周期中被采样。如果在一个周期中采样为高，而下一个周期中为低，则检测到了下降沿事件，需设置 IEx 标志中断请求。同理，上升沿事件也是这样检测出来的。标志位请求中断。由于外部中断是在每个机器周期被采样，因此需要至少一整个机器周期的高采样结果或低采样结果。IEx 标志在中断服务程序被调用时自动清零。

若选择了低电平触发方式，则请求源会保持管脚低电平直到中断响应。IEx 标志不会由于进入中断服务程序而被硬件清零。如果继续保持中断低电平甚至在服务程序结束后还是低电平，则处理器会从相同的中断源发送另一个中断请求，在这种情况下 ISR 中必须能够强制外部设备释放中断请求信号。

中断可以通过在软件上设置中断请求响应标志位来触发。

2) 外部中断 2

外部中断 2 可以编程为上升沿事件触发或下降沿事件触发，这取决于 T2CON 寄存器中的 I2FR 位。T2CON 寄存器中的 I2FR 位是检测产生中断的标志。

调用请求服务程序时，标志位被自动清零。

3) 外部中断 3

外部中断 3 可以编程为上升沿事件触发或下降沿事件触发，这取决于 T2CON 寄存器中的 I3FR 位。IRCON 寄存器中的 I3FR 位是检测产生中断的标志。

此外，标志位可以由比较/捕获单元（0）设置（以及中断调用）。比较/捕获方式设置为比较方式时（CCEN 寄存器中标志 cocal0=0,cocah0=1）且定时器 2 中的值与比较寄存器 CRC 中的值相等时，比较/捕获单元设置标志位。外部中断配置（上升沿或下降沿）在这里也可以应用。当产生中断时，I3FR 定义边沿，这意味着当定时器 2 中的值与 CRC 寄存器的值不相等时也可能生成中断。

只有在 cocal0=0 且 cocah0=1 时，标志位由比较/捕获单元设置。在其他情况下，只有当外接管脚检测到相应的边沿时才能设置标志位。不能同时使用外部中断和 CCU 中的中断。

调用服务程序时，标志位被自动清零。

4) 外部中断 4

外部中断 4 仅仅由上升沿事件触发产生，寄存器 IRCON 中的 IEX4 位是检测中断生成的标志。

此外，标志位可以由比较/捕获单元（0）设置（以及中断调用）。当比较/捕获方式设置为比较方式（CCEN 寄存器中标志 cocal0=0,cocah0=1）且定时器 2 中的值与比较寄存器 CC1 中的值相等时，比较/捕获单元设置标志位。

调用中断服务程序时，标志位被自动清零。

5) 外部中断 5

外部中断 5 仅仅由上升沿事件触发产生，寄存器 IRCON 中的 IEX5 位是检测中断生成的标志。

此外，标志位可以由比较/捕获单元（0）设置（以及中断调用）。当比较/捕获方式设置为比较方式（CCEN 寄存器中标志 cocal0=0,cocah0=1）且定时器 2 中的值与比较寄存器 CC2 中的值相等时，比较/捕获单元设置标志位。

调用中断服务程序时，标志位被自动清零。

6) 外部中断 6

外部中断 6 仅仅由上升沿事件触发产生，寄存器 IRCON 中的 IEX6 位是检测中断生成的标志。

此外，标志位可以由比较/捕获单元（0）设置（以及中断调用）。当比较/捕获方式设置为比较方式（CCEN 寄存器中标志 cocal0=0,cocah0=1）且定时器 2 中的值与比较寄存器 CC3 中的值相等时，比较/捕获单元设置标志位。

7) 外部中断 7

外部中断 7 仅仅由上升沿事件触发产生，寄存器 IRCON 中的 IEX7 位是检测中断生成的标志。

调用中断服务程序时，标志位被自动清零。

8) 定时器中断

定时器 0 和 1 由 TF0 和 TF1 标志生成。这些标志由定时器 0 和 1 中的溢出 int 设置。当定时器中断响应时，TF0 和 TF1 被硬件自动清零。定时器 2 中断是由 TF 2 中断与 EXF 2 中断逻辑或产生的。这些标志由定时器 2 操作中的溢出或捕获/重载事件设置。当定时器 2 中断执行时，硬件不会清零标志。软件需要查询 TF2 和 EXF2 判断中断的来源并清零恰当的标志。

9) 串口中断

串口 0 在接收或发送时能够产生中断。接收或发送完成后，硬件自动把 S0CON SFR 中的 RI0 或 TI0 位置为 1。这些位不会被硬件自动清零，需要用户使用软件清零。

串口 1 可以在接收或发送时产生中断。串口 1 有两种中断源，分别被 S1CON SFR 中的 RI1 和 TI1 位获得。这些位不会被硬件自动清零，需要用户使用软件清零。

10) 使能/禁用位

设置或清空在 IE0, IE1, IE2 SFR 中相应的位，可以独立的使能或禁用每个中断源。IE0 有一个全局使能/禁用位 EAL，清零 EAL 位会即刻禁用所有中断。

3.6.2 中断优先级

8051 的中断源可分为 6 个中断组。每组都有 4 个优先级可供选择。给 IP0 和 IP1 寄存器设置恰当值可以设定 6 组中断源的优先级。详情请参见“表 3-29 优先组”。

中断组内部的中断优先结构由硬件确定。第一列中断源有最高优先级（限于本组内），第二列中断源有中等优先级，第三列中断源的优先级最低。组内中断优先级不可更改。

中断组之间也有中断优先结构。group0 有最高优先级，group5 的优先级最低。中断组之间的优先权可以通过改变优先级（优先级可从 0 到 3 设置）来实现。详情参见表 3-30 优先级。

调用 1 个以上的中断时，所有优先类型都要考虑在内。最重要的是优先级由 IP0 和 IP1 寄存器设置的，其次是组之间的自然优先级，最后是组内部的优先权。

赋予中断最高优先权（以第一顺序实现），需要以下步骤：

- 从各组之中选择拥有最高优先权的几个组
- 最高优先权组中选取一个具有最高自然优先级的组
- 再从最高优先权组中选择内部最高优先权的中断

重要的是当前运行的中断服务子程序只能被更高优先级的中断打断。相同优先级或低优先级不能打断正在运行的中断服务子程序。因此服务程序中最多可以同时有 4 个中断。

注意：当中断优先级为 X 的中断提出请求时，即使在被调用的中断服务子程序中改变了该中断源的中断优先级，该中断服务也仅仅能被优先级高于 X 的中断请求打断。

3.6.3 中断源概要

表 3-48 描述了中断源、标志位、向量地址、使能位、中断组、优先权位和组优先级。

表 3-48 中断源概要

Source	Flag	Vector address	Enable bit	Interrupt Group	Interrupt Priority	Priority in group	Flag cleared by
External Interrupt 0	IE0	0003H	EX0	0	IP1.0, IP0.0	Highest	Hardware, software
Timer0 overflow	TF0	000BH	ET0	1	IP1.1, IP0.1	Highest	Hardware, software
External Interrupt 1	IE1	0013H	EX1	2	IP1.2, IP0.2	Highest	Hardware, software
Timer1 overflow	TF1	001BH	ET1	3	IP1.3, IP0.3	Highest	Hardware, software

Source	Flag	Vector address	Enable bit	Interrupt Group	Interrupt Priority	Priority in group	Flag cleared by
Serial Port 0 interrupt	RI0 + TI0	0023H	ES0	4	IP1.4, IP0.4	Highest	Software
Timer2 interrupt	TF2	002BH	ET2	5	IP1.5, IP0.5	Highest	Software
External Interrupt 7	IEX7	0043H	EX7	0	IP1.0, IP0.0	Lowest	Hardware, software
External Interrupt 2	IEX2	004BH	EX2	1	IP1.1, IP0.1	Lowest	Hardware, software
External Interrupt 3	IEX3	0053H	EX3	2	IP1.2, IP0.2	Lowest	Hardware, software
External Interrupt 4	IEX4	005BH	EX4	3	IP1.3, IP0.3	Lowest	Hardware, software
External Interrupt 5	IEX5	0063H	EX5	4	IP1.4, IP0.4	Lowest	Hardware, software
External Interrupt 6	IEX6	006BH	EX6	5	IP1.5, IP0.5	Lowest	Hardware, software
Serial Port 1 interrupt	RI1 + TI1	0083H	ES1	0	IP1.0, IP0.0	Middle	Software

3.7 定时器

HME-M1 中有三个定时器（定时器 0, 定时器 1 和 定时器 2）和一个看门狗（WDT）。

3.7.1 定时器 0

1) 概述

定时器 0 是 16 位的寄存器，可以配置为计数器或定时器工作模式。同样可以当做 SFR 访问： “th0”和“tl0”(P46 13) 定时器 0—TH0, TL0)。

2) 功能方式

定时器 0 支持多方式。

- **方式 0 和方式 1**

方式 0 中，定时器 0 配置成 13 位的寄存器（“tl0”=5 位，“th0”=8 位）。“tl0”中较高的 3 位固定可以忽略。
方式 1 中，定时器 0 配置成 16 位的寄存器

- **方式 2**

此方式下定时器 0 配置为带有常数自动重装入的 8 位寄存器。

- **方式 3**

方式 3 中，定时器 0 配置为 1 个 8 位的定时/计数器和一个 8 位的定时器。当定时器 0 在方式 3 下工作时，定时器 1 仍可被串口作为波特率发生器继续在其他方式下使用，或者其它不需要调用从定时器 1 中断的应用。

1) 说明

定时器方式下，定时器 0 每 12 个时钟周期中加 1。

在计数器方式下，定时器 0 相应的输入引脚"t0"处检测到下降沿时，定时器 0 递增。由于需要 2 个时钟周期来识别一个 1-to-0 事件，最大输入计数率是振荡器频率的 1/2。占空比没有限制，但为保证正确的识别 0、1 状态，输入 0 或 1 电平应至少在一个时钟周期内保持稳定。

定时器 0 有 4 种工作方式可供选择。2 个特殊功能寄存器："tmod"(参见 P45 12) 定时器模式寄存器)和"tcon"(0)都可用于选择恰当的方式。

- **方式 0 下的定时/计数器 0**

调用此方式需要设置"tmod"寄存器的标志"tmod[1:0]"="00" (参见 P45 12) 定时器模式寄存器)。在此方式中，计数率来自定时器模式中的"clk"输入或计数器模式的"t0"输入。"tmod[2]"置 0 时选定时器工作模式，否则就选计数器工作模式。

定时/计数器可分为一个低字节一个高字节，共计 2 个 8 位的寄存器。低字节在此工作方式下又被分成 2 个部分——低 5 位和高 3 位，低字节的低 5 位与高字节组成为一个 13 位的定时\计数器，且每 12 个时钟周期或当外部信号"t0"检测到下降沿时加 1。

当定时/计数器 0 溢出时，通过输出引脚"tf0"置位标志"tcon[5]"并发出中断请求。当中断响应信号（"int0ack"）到达时，此位被硬件清零。

定时/计数器可以由软件和硬件控制。运行定时器 0 则必须置位"tcon[4]"标志。使能门控标志"tmod[3]"时，可以通过输入信号"int0"停止计数。

- **方式 1 下的定时/计数器 0**

调用此方式需要设置"tmod"寄存器的标志"tmod[1:0]"="01" (参见 P45 12) 定时器模式寄存器)。此方式与方式 0 唯一的不同在于，低字节不会被分为 5 位和 3 位 2 个部分，而是整个低字节作为计数器。定时/计数器 0 在方式 1 下是一个 16 位的计数器。

- **方式 2 下的定时/计数器 0**

调用此方式需要设置"tmod"寄存器的标志"tmod[1:0]"="10" (参见 P45 12) 定时器模式寄存器)。此工作方式中，计数率来自定时器模式中的"clk" 输入或计数器模式的"t0"输入。"tmod[2]"位置 0 选取定时器工作模式，直 1 则选取计数器工作模式。

低字节（"tl0"）每 12 个时钟周期或当外部信号"t0"的值从 1 变为 0 时，加 1。

定时/计数器工作在 8 位常数自动重装入方式下。定时/计数器的低字节溢出时，通过输出引脚"tf0"置位标志"tcon[5]"并发出中断请求。当中断响应确认信号（"int0ack"）到达时，此位被硬件清零。此外，当溢出发生时，低字节（"tl0"）会从高字节（"th0"）重新载入新值。

定时/计数器由软件和硬件控制。运行定时器 0 则必须置位"tcon[4]"标志，使能门控标志"tmod[3]"时，可以通过输入信号 "int0"停止计数。

- **方式 3 下的定时/计数器 0**

调用此方式需要设置“tmod”寄存器的标志“tmod[1:0]”=“00” (参见 P45 12)　定时器模式寄存器)。此工作方式中，计数率来自定时器模式中的“clk”输入或计数器模式的“t0”的输入。通过清零“tmod[2]”标志选择定时器选项，否则选取计数器选项。

低字节（“tl0”）在每 12 个时钟周期或外部信号“t0”的值从 1 变为 0 时加 1。高字节（“tl0”）每 12 个时钟周期加 1。当定时/计数器的低字节溢出时，通过输出引脚“tf0”置位标志“tcon[5]”并发出中断请求。当定时/计数器的高字节溢出时，通过输出引脚“tf1”置位标志“tcon[7]”并发出中断请求。当中断响应确认信号（“int0ack”，“int1ack”）分别到达时，硬件清零相应的位。

这种工作方式下，定时/计数器 0 的低字节由“tcon[4]”位和“int0”输入控制。将“tcon[4]”位置 1 可使能定时器工作模式。当强制“int0”为 0 并开启门控位 tmd[3]时，“int0”停止计数。

高字节只由“tcon[6]”位控制，设置后使能计数。

3.7.2 定时器 1

1)　概述

定时器 1 是 16 位寄存器，可以配置为计数器或定时器工作模式。可以作为 SFR 访问：“tl1”，“th1”(P46 14))。

2)　功能方式

- **方式 0 和方式 1**

方式 0 时，定时器 1 配置为 13 位的寄存器（“tl1”=5 位，“th1”=8 位）。“tl1”中较高的 3 个位固定不变可忽略。方式 1 时，定时器 1 被配置为 16 位的计数器。

- **方式 2**

在此方式中，定时器 1 被配置为带有常数自动重装入的 8 位计数器。

- **方式 3**

在方式 3 中，定时器 1 不启用。

3)　说明

在定时器模式下，定时器 1 每 12 个时钟周期加 1。

在计数器模式下，当定时器 1 的相应输入引脚“t1”处检测到下降沿时定时器 1 加 1。由于需要 2 个时钟周期来识别一个 1 到 0 的事件，最大输入计数率为振荡器频率的 1/2。尽管对占空比没有限制，但为确保正确识别 0、1 状态，输入 0 或 1 电平应至少在一个时钟周期内保持稳定。

定时器 1 有四种工作方式可供选择。2 个特殊功能寄存器：　“tmod”(P45)和“tcon”(0)，用于选择适当的工作方式。

- **方式 0 下的定时/计数器 1**

调用此方式需设置“tmod”寄存器的“tm0d[5:4]”=“00”（P45）。在这种方式下，计数率源于定时器模式的　“clk”

输入或计数器模式的“t1”输入。“tmod[6]”位置 0 选定时器模式，置 1 则选计数器模式。

定时/计数器 1 分成一个低字节一个高字节，共计 2 个 8 位的寄存器。低字节又被分成两个部分——低 5 位和高 3 位，低字节的低五位和高字节组成一个 13 位的计数器，且每 12 个时钟周期或外部信号“t1”从 1 变成 0 时加 1。当定时/计数器 1 溢出时，通过输出引脚“tf1”置位标志“tcon[7]”并发出中断请求。当中断响应确认信号（“int1ack”）到达时，硬件清零此位。

定时/计数器 1 可以通过软件或硬件进行控制。运行定时器 1 则必须置位“tcon[6]”标志。使能门控标志“tmod[7]”时，可以通过输入信号“int1”停止计数。

- **方式 1 下的定时/计数器 1**

调用此方式需设置“tmod”寄存器“tm0d[5:4]”=“01”（P45）。此方式仅有一处与方式 0 不同，即低字节不被分成 5 位和 3 位，而是整个低字节作为一个计数器。定时/计数器 1 在方式 1 下是一个 16 位的计数器。

- **方式 2 下的定时/计数器 1**

调用此方式需要置位“tmod”寄存器 “tmod[5:4]”=“10” (P45)。在此方式中，计数率源于定时器模式中的输入“clk”或计数器选模式的输入“t1”。“tmod[6]”置位 0 选择定时器工作模式，置 1 则选计数器工作模式。

定时/计数器工作在 8 位常数自动重装入方式下。12 个时钟周期或外部信号“t0”的值从 1 变为 0 时，低字节（“tl0”）加 1。当定时/计数器的低字节溢出时，通过输出引脚“tf1”置位标志“tcon[7]”并发出中断请求。当中断响应确认信号（“int1ack”）到达时，硬件清零该位。此外，当溢出发生时，低字节（“tl0”）会从高字节（“th0”）重新载入数值。

定时/计数器可以通过软件和硬件控制。运行定时器 1 必须置位 “tcon[6]”标志。使能门控标志“tmod[7]”时，可以通过输入信号“int1”停止计数。

- **方式 3 下的定时/计数器 1**

调用此方式需要设置“tmod”寄存器“tmod[5:4]”=“11” (P45)。在此方式中，定时/计数器 1 禁用（方式 3 中只能运行定时/计数器 0）。

3.7.3 定时器 2（带有比较/捕获单元）

1) 概述

定时器 2 可以配置为计数器或定时工作模式，或者配置为定时器 2 的子模块比较/捕获单元。

2) 定时器 2 说明

定时器 2 可以作为定时器、事件计数器、门控定时器进行操作。

- **定时器方式**

调用这种工作方式需设置寄存器“t2con”标志，“t2i0=1”和“t2i1=0”。（见 P46）。在此方式中，计数率源于定时器选项中的输入“clk”。根据预分频系数，定时器 2 每 12 个时钟周期或 24 个时钟周期加 1。寄存器“t2con”的“t2ps”选择预分频系数（P46）。当“t2ps”=0 时，定时器每 12 个时钟周期自动加 1，否则每 24 个时钟周期加 1。

- **事件计数器方式**

调用此方式需置位寄存器“t2con”标志“t2i0=1”和“t2i1=1”。（见 P46）。此方式下，当外部信号“t2”从 1 变成 0 时，定时器 2 加 1。在每个时钟的上升沿，输入“t2”都进行采样。检测到切换的下一个时钟周期是定时器 2 加 1。最大计数率是时钟频率的 1/2。

- **门控定时器方式**

调用此方式需置位寄存器“t2con”标志“t2i0=1”和“t2i1=1”。（见 P46）。定时器 2 每 12 或 24 个时钟周期加 1（由“t2ps”标志决定），但除此以外还由外部信号“t2”门控。当“t2”=0 时定时器 2 停止工作。“t2”输入被采样寄存后停止定时器 2 自增。

- **定时器 2 自动重装入**

寄存器“crc”中的 16 位常数重装入可以在以下 2 种方式下运行：

- 重装入方式 0：重装入启动信号是由定时器 2 溢出产生的（自动重装入）；
- 重装入方式 1：重装入启动信号是由相应的输入引脚“t2ex”负跳变产生的。

3) 比较功能说明

比较/捕获单元包括四个寄存器“cc1”，“cc2”，“cc3”和“cc4”(P48)和“crc”(P48)。每个寄存器都可以配置在比较器方式下工作。寄存器内存储的值会与定时器 2 中的值相比较。比较器输出驱动“p1”端口（“p1.0”…“p1.3”）的低 4 位:

- “p1.0”是与寄存器“crc”关联的比较输出结果（“ccubus.0”）；
- “p1.1”是与寄存器“cc1”关联的比较输出结果（“ccubus.1”）；
- “p1.2”是与寄存器“cc2”关联的比较输出结果（“ccubus.2”）；
- “p1.3”是与寄存器“cc3”关联的比较输出结果（“ccubus.3”）。

由寄存器“t2con”中的“t2cm”位选择两种比较方式。

- **比较方式 0**

调用此方式需设定寄存器“t2con”标志“t2cm”=0（见 P46）。当定时器 2 中的值等于比较寄存器中的值时，比较器输出由低变高。当定时器 2 溢出时，输出又变低。

此方式下对端口 1（“p1”）的写操作无效，因为内部总线的输入线路和写入寄存器线路不连接。

- **比较方式 1**

调用这种方式，需置 1 寄存器“t2con”的标志“t2cm”（P46）。比较方式 1 中，输出信号的切换由软件决定。一个定时器 2 溢出不会引起输出改变。此方式下，两种信号输出切换都可以被控制。先将值写入“影子寄存器”中，compare 信号有效后，这个值会被传输到输出寄存器中。

- **捕获功能说明**

4 个 16 位的 CCU 寄存器，都可配置在捕获方式下运行。此方式下，定时/计数器实际值会根据外部事件（0 方式）或软件写操作（1 方式）保存在 CCU 寄存器中。

- **捕获方式 0**

方式 0 下，捕获定时器 2 内容的条件：

- 输入“cc1”中检测到上升沿（“cc1”寄存器在捕获方式下运行）

- 输入“cc2”中检测到上升沿（“cc2”寄存器在捕获方式下运行）
- 输入“cc3”中检测到上升沿（“cc3”寄存器在捕获方式下运行）
- 输入“cc0”中检测到上升沿或下降沿，取决于“i3fr”位（“crc”寄存器在捕获方式下运行）

定时器 2 中的内容会被适当的捕获寄存器锁存。此方式下不会产生中断请求。

- **捕获方式 1**

方式 1 中，定时器 2 捕获操作是由任意写入特定捕获寄存器低位字节的操作引起的。此功能与写入捕获寄存器的值无关。定时器 2 中的内容会被恰当的捕获寄存器锁存。此方式不产生中断请求。

3.8 看门狗

3.8.1 概述

看门狗是 15 位的计数器，每 24 或 384 个时钟周期加 1。用于提供系统监管，可防止软、硬件崩溃。如果软件不能在 786336 或 12581376 个时钟周期（使用 12MHz 时钟 65ms 或 1s）内刷新看门狗，会产生内部复位。

1) 说明

看门狗包含一个 15 位的计数器（不能作为 SFR 访问），常数自动重装入寄存器“wdtrel”(P51),前置 2 分频 16 分频和控制逻辑。

看门狗的计数率取决于“wdtrel”寄存器的 MSB。当“wdtrel.7”=1 时，看门狗每 12*32=384 个时钟周期加 1，则整个周期长达 12*32*256*128=12582912 个时钟周期。

当“wdtrel.7”=0 时，看门狗每 12*2=24 个时钟周期加 1，整个周期长达 12*2*256*128=786432 个时钟周期。

当“wdt_tm”测试方式输入置为 1 时，看门狗的计数率为 clkper 时钟频（所有分频器— 1/12，1/8，1/2，1/16 等等都被删除），可缩短看门狗溢出所需要的时间（用于进行验证）。

- **启动步骤**

启动看门狗有两种方式。一种方式是硬件自动启动，在外部复位信号有效时检测输入信号“swd”的电平。“swd”在复位信号变低之前保持高电平，此时看门狗从默认设置下自动运行（所有寄存器都置为 0）。

另一个方法是用软件启动看门狗。置 1“ien1”寄存器中的“swdt”标志来启动看门狗（P52）。仅置位“swdt”标志来启动看门狗，不能自动对重装入看门狗定时器重装入。

一旦启动，只能在硬件使用“reset”引脚复位时禁用“swd”输入引脚，才能停止看门狗。

当看门狗计数器进入 7FFCh 状态，“wdts”输出信号有效，产生内部复位。“ip0”寄存器（P54）的标志“wdts”因看门狗复位而复位，因外部硬件“reset”信号而清零。“wdts”信号不能使运行中的看门狗复位。当“wdts”信号从 7FFFh 溢出到 0000h，“wdts”的输出无效，而“ip0”寄存器的“wdts”标志置 1 从而允许软件查询复位是由外部输入引起或看门狗超时引起。

“ip0”寄存器的标志“wdts”也可由软件修改。

- **刷新看门狗**

看门狗必需定时刷新以防复位请求信号（“wdts”）有效。编程人员需要使用两条俩徐的指令来实现对看门狗的刷新。第一条指令置位“ien0”寄存器中的“wdt”（参照表 3-24 IEN0 寄存器），第二条指令置位““ien1”寄存器中的“swdt”（参见表 3-25 IEN1 寄存器）。 “wdt”和“swdt”之间的置 1 操作最大允许延时为 1 个指令周期（即这 2 个标志的置位指令不允许被其它指令隔开）。置位后,“wdt”标志自动清零。这样无论“swdt”是否置位，都可以防止看门狗自动重装入。看门狗定时器的 7 个高位自动重装入“wdtrel”（参见 P51）的值，“wdtrel”中的值越大，看门狗刷新所需的周期越短。

3.9 I2C

3.9.1 I2C 概述

HME-M1 集成了一个符合 Philips I²C 总线特性的 I²C 总线外设，支持 I²C 总线的所有传输模式。I²C 使用 2 条总线在设备之间传输信息，“scl”为串行时钟线，“sda”为串行数据线。

HME-M1 的 I²C2 线连到 HME-M1 固定的 I/O 上，需要在总线外部用电阻上拉。

3.9.2 说明

与总线连接的每个设备都有唯一的地址。I²C 是真正的多主控总线，能够实现包括冲突、检测、仲裁等功能以防止两个或多个控制同时发出数据传输而造成数据失效。

1) 操作模式

I²C 执行 8 比特的双向数据传输，可以运行下面四种模式：

- **主传输模式**

  “scl”输出连续时钟时，通过“sda”输出连续数据

- **主接收模式**

  “scl”输出连续时钟，通过“sda”接收连续数据

- **从接收模式**

  通过“sda”和“scl”接受连续数据和连续时钟

- **从传输模式**

  连续数据通过“sda”传输，通过“scl”输入连续时钟

2) 仲裁

在主模式中，仲裁逻辑检验“sda”上的每个传输高状态（‘1’），若总线上的其他装置重新写了高状态并将“sda”线拉回低状态（‘0’），仲裁将会丢失，I²C 会立即从主传输模式变为从接收模式。

3) 串行时钟生成器

当 I²C 处在主模式时，可编程的时钟脉冲生成器产生“scl”时钟脉冲。当 I²C 在从属模式下时，时钟生成器被抑制。时钟生成器的功能由“i2ccon”寄存器的“cr0”，“cr1”，“cr2”等比特控制（参见 P59 39) I2C 控制寄存器 — I2CCON）。下表显示了“clko”在主模式下可能的频率。表中的“bclk”输入与 Timer1 输出溢出相关联。这表明 I²C 的波特率可由 Timer1 控制。

表 3-49 I²C 时钟频率比特设置

cr2	cr1	cr0	Bit frequency				Clk divided by
			6 MHz	12 MHz	16 MHz	24 MHz	
0	0	0	23	47	63	92	256
0	0	1	27	54	71	108	224
0	1	0	31	63	83	124	192
0	1	1	37	75	100	148	160
1	0	0	6.25	12.5	17	25	960
1	0	1	50	100	133	200	120
1	1	0	100	200	266	400	60
1	1	1	“bclk”输入 8 分频				

1) 地址比较器

接收到的 7 位大小的从地址会与 I²C 自己的从地址相比较。自身的从地址可以通过“i2cadr”寄存器（参见 P59 38) ）设置。同样，接收到的第一个字节也会和广播地址（00H）进行比较，这取决于“i2cadr”寄存器的“gc”位（参见 P59 38) ）。如果相等，则设置“i2ccon”寄存器的“si”位（参见 P59 39) ）并请求中断。

2) 特殊功能寄存器

微处理器通过四个特殊功能寄存器连接到 I²C 上：“i2ccon”（控制寄存器，第 59 页），“i2csta”（状态寄存器，第 60 页），“i2cdat”（数据寄存器，第 58 页），“i2cadr”（特有从地址寄存器，第 59 页）。

“i2ccon”寄存器包含全局 I2C 启动比特“ens1”，时钟频设置比特(“cr0”,“cr1”,“cr2”,参见表 3-39 I2CCON 寄存器)。“i2ccon”寄存器提供启动将开始停止到 I²C 总线的标志（“sta”和“sto”比特），以及传输确认标志“aa”。

当检测到主控制 FSM 有变化时，“i2ccon”提供由硬件设置的中断请求“si”标志。更多 FSM 细节，请参见“i2csta”寄存器描述。

“i2csta”寄存器反映了 I2C 控制器的主要状态。寄存器的低三位一直是 0。有 26 个可能状态编码，分别在表 3-50…表 3-54 有详述，其中的 25 个有 I2C FSM 状态中任一状态进入时，设置“si” 标志(参见 P59 39))，产生中断请求。唯一不生成中断的状态是 F8h 状态。必须由软件向“si”位中写入‘0’才能清除标志。写入‘1’不会改变“si”中的值。

“i2cdat”寄存器存放 I2C 总线传输的一字节或 I2C 总线接收到的一字节。“i2cdat”寄存器不能两次缓冲，所以仅当 I2C 中断时 MCU 才能正确读取接收到的数据。

“i2cadr”寄存器包含 I2C 组件的特有从地址，以及允许确认广播地址的“gc”标志。

共用相同向量的 I2C 中断作为外部中断 7。“si”标志在进入中断控制器（ISR）之前，与外部中断 7 边沿标志做或运算。为检测真实中断源，“si”标志须被中断服务程序检测。由于外部中断 7 标志在引导服务子程序后会被自动清除，则只有“si”标志带有真实中断源的信息。

下表所示，“SLA”指从地址，“R”指 R/W 位=1 与从地址一起传输，“W”指 R/W 位=0 与从地址一起传输。

表 3-50 主传输器模式下的 I2C 状态

Status code	Status of the I2C	Application software response					Next action taken by the I2C hardware
		to/from I2CDAT	to I2CCON				
			sta	sto	si	aa	
08H	START condition has been transmitted	Load SLA+W	X	0	0	X	SLA+W will be transmitted ACK will be received
10H	Repeated START condition has been transmitted	Load SLA+W or	X	0	0	X	As above
		Load SLA+R	X	0	0	X	SLA+R will be transmitted I^2C will be switched to “master receiver” mode
18H	SLA+W has been transmitted; ACK has been received	Load data byte	0	0	0	X	Data byte will be transmitted; ACK will be received
		or no action	1	0	0	X	Repeated START will be transmitted;
		or no action	0	1	0	X	STOP condition will be transmitted; the “sto” flag will be reset
		or no action	1	1	0	X	STOP condition followed by a START condition will be transmitted; the “sto” flag will be reset
20H	SLA+W has been transmitted; “not ACK” has been received	Load data byte	0	0	0	X	Data byte will be transmitted; ACK will be received
		or no action	1	0	0	X	Repeated START will be transmitted;
		or no action	0	1	0	X	STOP condition will be transmitted; the “sto” flag will be reset
		or no action	1	1	0	X	STOP condition followed by a START condition will be transmitted; the “sto” flag will be reset
28H	Data byte in i2cdat has been transmitted; ACK has been received	Load data byte	0	0	0	X	Data byte will be transmitted; ACK bit will be received
		or no action	1	0	0	X	Repeated START will be transmitted

Status code	Status of the I2C	Application software response					Next action taken by the I2C hardware
		to/from I2CDAT	to I2CCON				
			sta	sto	si	aa	
		or no action	0	1	0	X	STOP condition will be transmitted; the “sto” flag will be reset
		or no action	1	1	0	X	STOP condition followed by a START condition will be transmitted; sto flag will be reset
30H	Data byte in i2cdat has been transmitted	Data byte	0	0	0	X	Data byte will be transmitted; ACK will be received
		or no action	1	0	0	X	Repeated START will be transmitted;
		or no action	0	1	0	X	STOP condition will be transmitted; sto flag will be reset
		or no action	1	1	0	X	STOP condition followed by a START condition will be transmitted; sto flag will be reset
38H	Arbitration lost in SLA+R/W or data bytes	No action	0	0	0	X	I^2C bus will be released; the “not addressed slave” state will be entered
		or no action	1	0	0	X	A START condition will be transmitted when the bus becomes free

表 3-51 主接收器模式下的 I2C 状态

Status code	Status of the I2C	Application software response					Next action taken by the I2C hardware
		to/from	to i2CCON				
		I2CDAT	sta	sto	si	aa	
08H	START condition has been transmitted	Load SLA+R	X	0	0	X	SLA+R will be transmitted; ACK will be received
10H	Repeated START condition has been transmitted	Load SLA+R or	X	0	0	X	As above
		Load SLA+W	X	0	0	X	SLA+W will be transmitted; I^2C will be switched to “master transmitter” mode
38H	Arbitration lost in “not ACK” bit	No action or	0	0	0	X	I^2C bus will be released; I^2C will enter a “slave” mode
		no action	1	0	0	X	A start condition will be transmitted when the bus

Status code	Status of the I2C	Application software response					Next action taken by the I2C hardware
		to/from	to i2CCON				
		I2CDAT	sta	sto	si	aa	
							becomes free
40H	SLA+R has been transmitted; ACK has been received	No action or	0	0	0	0	Data byte will be received;
		no action	0	0	0	1	not ACK will be returned
							Data byte will be received; ACK will be returned
48H	SLA+R has been transmitted; ACK has been received	No action	1	0	0	X	Repeated START condition will be transmitted
		no action	0	1	0	X	STOP condition will be transmitted; the “sto” flag will be reset
		no action	1	1	0	X	STOP condition followed by START condition will be transmitted; the “sto” flag will be reset
50H	Data byte has been received; ACK has been returned	Read data byte or	0	0	0	0	Data byte will be received; “not ACK” will be returned
		read data byte	0	0	0	1	Data byte will be received; ACK will be returned
58H	Data byte has been received; ACK has been returned	Read data byte or	1	0	0	X	Repeated START condition will be transmitted
		read data byte or	0	1	0	X	STOP condition will be transmitted; the “sto” flag will be reset
		read data byte	1	1	0	X	STOP condition followed by START condition will be transmitted; the “sto” flag will be reset

表 3-52 从属接收器模式下 I2C 状态

Status code	Status of the I2C	Application software response					Next action taken by the I2C hardware
		to/from	to i2CCON				
		I2CDAT	sta	sto	si	aa	
60H	Own SLA+W has been received; ACK has been returned	No action or	X	0	0	0	Data byte will be received and “not ACK” will be returned
		no action	X	0	0	1	Data byte will be received and ACK will be returned
68H	Arbitration lost in	No	X	0	0	0	Data byte will be received and “not

Status code	Status of the I2C	Application software response					Next action taken by the I2C hardware
		to/from	to i2CCON				
		I2CDAT	sta	sto	si	aa	
	SLA+R/W as master; own SLA+W has been received, ACK returned	action or					ACK" will be returned
		no action	X	0	0	1	Data byte will be received and ACK will be returned
70H	General call address (00H) has been received; ACK has been returned	No action or	X	0	0	0	Data byte will be received and "not ACK" will be returned
		no action	X	0	0	1	Data byte will be received and ACK will be returned
78H	Arbitration lost in SLA+R/W as	No action or	X	0	0	0	Data byte will be received and "not ACK" will be returned
	master; general call address has been received, ACK returned	no action	X	0	0	1	Data byte will be received and ACK will be returned
80H	Previously addressed with own SLV address; DATA has been received; ACK returned	Read data byte or	X	0	0	0	Data byte will be received and "not ACK" will be returned
		read data byte	X	0	0	1	Data byte will be received and ACK will be returned
88H	Previously addressed with own SLA; DATA byte has been received; "not ACK" returned	Read data byte or	0	0	0	0	Switched to "not addressed slave" mode; no recognition of own slave address or general call address
		read data byte or	0	0	0	1	Switched to "not addressed slave" mode; own slave address or general call address will be recognized
		read data byte or	1	0	0	0	Switched to "not addressed slave" mode; no recognition of own slave address or general call address; START condition will be transmitted when the bus becomes free
		read data byte	1	0	0	1	Switched to "not addressed slave" mode; own slave address or general call address will be recognized; START condition will be transmitted when the bus becomes free
90H	Previously addressed with general call address; DATA has been received; ACK	Read data byte or	X	0	0	0	Data byte will be received and "not ACK" will be returned
		read data byte	X	0	0	1	Data byte will be received and ACK will be returned

Status code	Status of the I2C	Application software response					Next action taken by the I2C hardware
		to/from	to i2CCON				
		I2CDAT	sta	sto	si	aa	
	returned						
98H	Previously addressed with general call address; DATA has been received; ACK returned	Read data byte or	0	0	0	0	Switched to “not addressed slave” mode; no recognition of own slave address or general call address
		read data byte or	0	0	0	1	Switched to “not addressed slave” mode; own slave address or general call address will be recognized
		read data byte or	1	0	0	0	Switched to “not addressed slave” mode; no recognition of own slave address or general call address; START condition will be transmitted when the bus becomes free
		read data byte	1	0	0	1	Switched to “not addressed slave” mode; own slave address or general call address will be recognized; START condition will be transmitted when the bus becomes free
A0H	STOP condition or repeated START condition has been received while still addressed	No action or	0	0	0	0	Switched to “not addressed slave” mode; no recognition of own slave address or general call address
	as SLV/REC or	no action or	0	0	0	1	Switched to “not addressed slave” mode; own slave address or general call address will be recognized
	SLV/TRX	no action or	1	0	0	0	Switched to “not addressed slave” mode; no recognition of own slave address or general call address; START condition will be transmitted when the bus becomes free
		no action	1	0	0	1	Switched to “not addressed slave” mode; own slave address or general call address will be recognized; START condition will be transmitted when the bus becomes free

表 3-53 从属传输器模式下的 I2C 状态

Status code	Status of the I2C	Application software response					Next action taken by the I2C hardware
		to/from	to i2CCON				
		I2CDAT	sta	sto	si	aa	
A8H	Own SLA+R has been received; ACK has been returned	Load data byte or	X	0	0	0	Last data byte will be transmitted and ACK will be received
		load data byte	X	0	0	1	Data byte will be transmitted; ACK will be received
B0H	Arbitration lost in SLA+R/W as master; own SLA+R has been received; ACK has been returned	Load data byte or	X	0	0	0	Last data byte will be transmitted and ACK will be received
		load data byte	X	0	0	1	Data byte will be transmitted; ACK will be received
B8H	Data byte has been transmitted; ACK has been received	Load data byte or	X	0	0	0	Last data byte will be transmitted and ACK will be received
		load data byte	X	0	0	1	Data byte will be transmitted; ACK will be received
C0H	Data byte has been transmitted; not ACK has been received	No action or	0	0	0	0	Switched to “not addressed slave” mode; no recognition of own slave address or general call address
		no action or	0	0	0	1	Switched to “not addressed slave” mode; own slave address or general call address will be recognized
		no action or	1	0	0	0	Switched to “not addressed slave” mode; no recognition of own slave address or general call address; START condition will be transmitted when the bus becomes free
		no action	1	0	0	1	Switched to “not addressed slave” mode; own slave address or general call address will be recognized; START condition will be transmitted when the bus becomes free
C8H	Last data byte has been transmitted; ACK has been received	No action or	0	0	0	0	Switched to “not addressed slave” mode; no recognition of own slave address or general call

Status code	Status of the I2C	Application software response					Next action taken by the I2C hardware
		to/from	to i2CCON				
		I2CDAT	sta	sto	si	aa	
							address
		no action or	0	0	0	1	Switched to “not addressed slave” mode; own slave address or general call address will be recognized
		no action or	1	0	0	0	Switched to “not addressed slave” mode; no recognition of own slave address or general call address; START condition will be transmitted when the bus becomes free
		no action	1	0	0	1	Switched to “not addressed slave” mode; own slave address or general call address will be recognized; START condition will be transmitted when the bus becomes free

表 3-54 I2C 状态—混合状态

Status code	Status of the I2C	Application software response					Next action taken by the I2C hardware
		to/from	to i2CCON				
		I2CDAT	sta	sto	si	aa	
F8H	No relevant state information available; si=0	No action	No action				Wait or proceed current transfer
00H	Bus error during MST or selected slave modes	No action	0	1	0	X	Only the internal hardware is affected in the “master” or “addressed slave” modes. In all cases, the bus is released and I^2C is switched to the “not addressed slave” mode. The “sto” flag is reset.

3.10 SPI

3.10.1 概述

AstorII 的 SPI 接口可以实现全双工传输、同步、串行通讯。SPI 模块可以设置为主机或从机，并具有以下特征：

- 全双工传输
- 三线同步传输器
- 主、从模式
- 七个 SPI 主波特率
- 从机时钟率高达 Fclk/4③
- 可编程的极性和相位串行时钟
- 8 位数据传输，最高位（MSB）先传，最低位（LSB）后传
- 4 位从机选择输出

3.10.2 说明

SPI 模块包括 SFR 和 INT 中断控制模块以及 TR 传输模块。

1) 特殊功能寄存器

SPI 模块中有三个特殊功能寄存器。

表 3-55 SPI 单元特殊功能寄存器

Register	Address	Reset value	Description
spcon	E2h	14H	Serial Peripheral Control register
spsta	E1h	00H	Serial Peripheral Status register
spdat	E3h	00H	Serial Peripheral Data register
spssn	E4h	FFH	Serial Peripheral Slave Select register

“spcon”寄存器（参见表 3-42 SPCON 寄存器）用于配置 SPI 模块。可以控制主时钟输出率（“spr0”，“spr1”，“spr2”位），时钟极性(“cpol”)和时钟相位(“cpha”)，可以将 SPI 配置为主或从（“mstr”），允许或禁用“ssn”输入（“ssdis”）以及使能或禁用整个 SPI（“spen”）。

“spsta”寄存器反映 SPI 模块的当前状态。标志“spif”表示传输正在进行或传输已经结束。“wcol”标志表示在“spdat”上发生写冲突，如，“spdat”寄存器经过 SFR 接口写入，此时 SPI 接口有一个传输。“serr”标志表示当 SPI 配置为从机时“ssn”输入在接收序列完成之前被移除。“modf”表示当“ssn”输入与模式实际设置有冲突（如，当“ssn”=0 且 SPI 配置为主机）。

“spdat”寄存器（参见 P62）在传输过程中使用。这个寄存器中的数据可以经由 TR（传输/接收）接口发送，如数据的字节开始从输出引脚移出（“mosio”-主模式下，“misoo”-从属模式下）。同时，另一字节从输入引脚移入（“misoi”-在主模式下，“mosii”-在从属模式下）。传输过程完成后，可以从“spdat”存储器中读取接收到的数据。

2) TR 块

传输/接收功能是由 TR 块实现的。数据以位为单位按照主时钟的上升沿（“cpol”=’0’）或下降沿（“cpol”=‘1’）进行发送，这取决于 SPI 模块的设置。数据按照主时钟的上升沿（“cpol”=’0’）或下降沿（“cpol”=‘1’）进行

③ Fclk 为 8051 的时钟频率

进行接收。如果设置了“cpha”位，第一位（MSB）会在“scko”的第一个有效沿“mosio”/“misoo”发送。如果“cpha”位被清除，则第一位（MSB）会在比“scko”的第一个有效沿的提前半个“scko”信号时期发送。而且，输入数据（”misoi”主，”mosii”从）会在每半个比特传输时进行采样，在相反的时沿处将数据从“mosio”输出引脚移出。

主模式

在主模式下(置 1“spcon”寄存器的“mstr”位)，执行完写入“spdat”寄存器的操作，SPI 主机开始执行传输。“scko”时沿处数据从“mosio”引脚移出。同时，另一数据开始从从机通过 “misoi”引脚移入。

从模式

首先，SPI 模块配置成为从模式（“spcon”寄存器的“mstr=0”）。当然要设置“spen=1”来使能 SPI 模块。

从模式下，SPI 等待在“ssn”的低输入。“ssn”输入必须保持为低，直到整个传输完成。传输的开始取决于 SPCON 寄存器的“cpha”位的状态（参见表 3-42 SPCON 寄存器）。当“cpha”被清零时，从机必须开始在第一个“scki”沿之前输出数据，且用“ssn”的下降沿启动传输。当“cpha”位被置位，从机使用“scki”输入的第一个沿作为传输的开始信号。

3) 中断生成

SPI 提供 SPI 中断输出“intspi”。2 个状态标志可以生成中断请求（参见表 3-56）。

表 3-56 SPI 中断标志

Name	Description
“spif”	When the transmission is finished, this flag is set by hardware.
“modf”	This bit is set when the level on “ssn” input is in conflict with actual mode, i.e. it is ‘0’ in Master mode (if “ssdis” = ‘0’).

当标志“spif”和“modf”被清除时，中断请求禁用。

共用相同向量的 SPI 中断作为外部中断 2。“intspi”输出信号在进入中断控制器（ISR）之前，与外部中断 2 的边缘标志进行或计算。为确定此中断，标志“modf”和“spif”需要在中断子程序中检测。由于外部中断 2 标志定位到服务子程序之后会自动被清除，只有 “modf”和“spif” 标志的状态带有关于实际中断源的信息。

3.11 UART

HME-M1 有两个串口（UARTs）：UART0 和 UART1。

3.11.1 UART0

UART0 提供灵活的全双工传输同步/异步的接收器/传输器。它有 4 种操作模式（1 个同步 3 个异步）。UART0 在接收端有数据缓冲，如，接收寄存器在第二次传输完成后接收新数据，而前面接收的数据在接收寄存器不会损坏。UART0 与标准 8051 串口完全兼容。

1) UART0 操作模式

UART0 支持 4 种操作模式。

模式 0

模式 0 时，UART0 作为同步传输/接收器运行。“txd0”输出移位时钟。“txd0o”输出数据“txd0i”输入数据。8 位数据传输从 LSB 开始。波特率定为主时钟频的 1/12。当置 1“s0con”寄存器的标志“ren0”时启动接收器，并清除“ri0”标志。

将数据写入“s0buf”寄存器可以启动传输。

模式 1

模式 1 时，UART0 作为 8 位异步传输/接收器运行，波特率可编程。寄存器”adcon“上的“bd”决定用 Timer1 的溢出标志，或者是用“s0relh”、“s0rell”的波特率生成器来指定波特率。此外，如果使用“pcon”寄存器的“smod”位，可以使波特率提高一倍。

写入“s0buf”寄存器可以启动传送。引脚“txd0”输出数据。传送的第一位是启动位（通常为 0），然后是 8 位的数据，最后是一个停止位（通常为 1）。

引脚“rxd0i”输入数据。接收开始时，UART0 与检测到引脚“rxd0i”的下降沿同步。接收完成后数据存放在“s0buf”寄存器中，停止位从“s0con”寄存器的“rb80”位获得。接收过程中，“s0buf”和“rb80”的值保持不变直到完成。

模式 2

模式 2 时，UART0 作为 9 位异步传输/接收器运行，波特率为 Fclkper/32 或 Fclkper/64 的波特率，这取决于“pcon”寄存器的“smod”比特的设定。

写入“s0buf”寄存器可以启动传送。引脚“txd0”输出数据。传输的第一位是起始位（通常为 0），然后是 9 位的数据，第 9 位的数据来自“tb80”，最后传输的是一个停止位（通常为 1）。

引脚“rxd0i”输入数据。当接收开始时，UART0 与检测到引脚“rxd0i”的下降沿同步。接收完成后数据存放在“s0buf”寄存器中，第 9 位可以从“s0con”寄存器的“rb80”位获得。在接收过程中，“s0buf”和“rb80”的值保持不变直到完成。

模式 3

模式 2 和模式 3 唯一的不同在于模式 3 中，用内部波特率生成器或是定时器 1 来指定波特率。

2) UART0 多处理器通信

UART0 在模式 2 或模式 3 下接收数据的特性可以用于多处理器通信。

当“s0con”寄存器的“sm20”位被设为 1，且第 9 个接收位（“s0con”寄存器的“rb80”比特）是 1 时，接收中断生成。否则，不会因接收而生成中断。

要在多处理器通信中利用这一特点，需置 1 从处理器的“sm20”。主处理器传送 8 位从地址和第 9 位，第 9 位为 1，引起对所有从处理器中断的接收。从处理器软件将接收的字节与自己的网络地址进行对比。如果匹配，从处理器地址清除自己的“sm20”标志，继续接收主处理器传来的其他信息，其中的第 9 位始终为 0。其他从处理器保持“sm20”值为 1，这样它们可以忽略主处理器传来的剩余信息。

3.11.2 UART1

UART1 提供灵活的全双工传输异步的接收器/传输器。它可操作 2 个模式。Serial1 在接收端有数据缓冲，如，它可以在第二次传输完成后接收新数据，先前接收的数据不损坏。

1) UART1 运行模式

模式 A

模式 A 时，UART1 作为 9 位异步传输/接收器运行，波特率可编程。用“s1relh”、“s1rell”的波特率生成器来同步输入输出传输。UART1 的波特率不能使用 “pcon”寄存器的“smod”位进行修改。

写入“s1buf”寄存器可以启动传送。引脚“txd1”输出数据。传送的第一位是启动位（通常为 0），然后是 9 位数据，其中第 9 位来自“s1con”寄存器的“tb81”，再传送一个停止位（通常为 1）。

引脚“rxd1i”输入数据。接收开始时，UART1 与检测到引脚“rxd1i”的下降沿同步。接收完成后数据存放在“s1buf”寄存器中，第 9 位可以从“s1con”寄存器的“rb81”位获得。在接收过程中，“s1buf”和“rb81”的值保持不变直到完成。

模式 B

模式 B 时，UART1 作为 8 位异步传输/接收器运行，波特率可编程。用“s1relh”、“s1rell”的波特率生成器来同步输入输出传输。UART1 的波特率不能使用 “pcon”寄存器的“smod”位进行修改。

写入“s1buf”寄存器可以启动传送。引脚“txd1”输出数据。传送的第一位是启动位（通常为 0），然后是 8 位数据，再传送一个停止位（通常为 1）。

引脚“rxd1i”输入数据。当接收器启动时，UART1 与下降沿探测的引脚“rxd1i”同步。在“s0buf”寄存器完成接收后可获得输入数据，停止位可以从“s1con”寄存器的“rb81”引脚获得。在接收过程中，“s1buf”和“rb81”的只保持不变直到完成。

2) UART1 多处理器通信

UART1 的模式 A 的特性可以用于多处理器通信。

当“s1con”寄存器的“sm21” 位被设为 1，且第 9 个接收位（“s1con”寄存器的“rb80”）是 1 时，接收中断生成。否则，不会因接收而生成中断。

要在多处理器通信中利用这一特点，需要设置从处理器的“sm21”为 1。主处理器传送 8 位从地址和第 9 位，第 9 位为 1，引起接收所有从处理器的中断。从处理器软件将接收的字节与自己的网络地址进行对比。如果匹配，从处理器地址清除自己的“sm21”标志，继续接收从主处理器传来的其他信息，其中的第 9 位始终为 0。其他从处理器保持“sm21”值为 1，这样它们可以忽略主处理器传来的剩余信息。

3.12 P 端口

3.12.1 概述

HME-M1 中，有 4 个 8 位寄存器“p0”,“p1”,“p2”,“p3”（参见 8) ）可以作为特殊功能寄存器进行访问，并执

行写入或读-改-写指令。

某些端口引脚与其他功能引脚多路复用。细节请参见下节。

3.12.2 端口引脚的多路复用

某些功能引脚，例如：外部中断 1，UART0，UART1，定时器 0~2,和比较-捕获单元，会与端口 1 和端口 3 共享引脚。详见下表描述。

表 3-57 端口引脚多重通道

Name	Type	Polarity Bus size	Alternate Port	Description
External interrupts inputs				
int0	I	Low/Fall	port3i[2]	External interrupt 0
int1	I	Low/Fall	port3i[3]	External interrupt 1
int2	I	Fall/Rise	port1i[4]	External interrupt 2
int3	I	Fall/Rise	port1i[0]	External interrupt 3
int4	I	Rise	port1i[1]	External interrupt 4
int5	I	Rise	port1i[2]	External interrupt 5
int6	I	Rise	port1i[3]	External interrupt 6
int7	I	Rise	port1i[6]	External interrupt 7
Serial 0 interface				
rxd0i	I	1	port3i[0]	Serial 0 receive data
rxd0o	O	1	port3o[0]	Serial 0 transmit data
txd0	O	1	port3o[1]	Serial 0 transmit data or receive clock in mode 0
Serial 1 interface				
rxd1	I	1	port1i[0]	Serial 1 receive data
txd1	O	1	port1o[1]	Serial 1 transmit data
Timers inputs				
t0	I	Fall	port3i[4]	Timer 0 external input
t1	I	Fall	port3i[5]	Timer 1 external input
t2	I	Fall	port1i[7]	Timer 2 external input
t2ex	I	Fall	port1i[5]	Timer 2 capture trigger
Compare – Capture Unit				
cc(0)	I	Rise/Fall	port1i[0]	Compare/Capture 0 input
cc(1)	I	Rise	port1i[1]	Compare/Capture 1 input
cc(2)	I	Rise	port1i[2]	Compare/Capture 2 input
cc(3)	I	Rise	port1i[3]	Compare/Capture 3 input
Ccubus[0]	O	1	port1o[0]	Compare/Capture 0 Output
Ccubus[1]	O	1	port1o[1]	Compare/Capture 1 Output
Ccubus[2]	O	1	port1o[2]	Compare/Capture 2 Output
Ccubus[3]	O	1	port1o[3]	Compare/Capture 3 Output

3.13 电源管理

3.13.1 节电方式

8051 中采取了功耗管理机制，并附有 2 种节电方式。节电方式受 PCON 寄存器控制（请参见 32) “电源控制寄存器—PCON”）。两种节电方式：STOP 方式和 IDLE 方式。

STOP 方式

此方式下，“clkperen”和“clkcpuen”信号无效，门控时钟停止，所有受“clkcpu”和“clkper”时钟控制的同步电路停止工作。

IDLE 方式

此方式下仅 CPU 停止运行，即控制单元、ALU、程序/数据存储接口和 RAM/SFR 接口都停止工作。“clkcpuen”无效，“clkperen”仍处于有效状态。

3.13.2 退出节电方式

STOP 方式

此方式下“clkperen“和“clkcpuen”信号无效，门控时钟停止。由于没有外设处于工作的状态，中断控制器无法产生任何中断请求。因此只能通过外部中断 0 或 1 产生中断来结束 STOP 方式。

IDLE 方式

此方式下只有 CPU 停止工作。中断控制器和其他外设都正常工作，中断也可以正常生成。因此任何中断都可以结束 IDLE 方式。

3.14 MSS 系统控制

MSS 通过对扩展的 SFR 寄存器的操作可以控制 HME-M1 的一些特殊功能，例如，ISC、GCLK 控制、电源管理以及 I/O 控制等。相应的 SFR 详见 3.4 扩展特殊功能寄存器。

3.14.1 MSS 系统时钟电源管理

1) PLL 操作

MISCCON.3:PLLPWD 为 1 时关闭 PLL 的电源，为 0 时接通 PLL 的电源。必须设置 MISCCON.4: CLKCPUEN、MISCCON.5:PLLO0EN、MISCCON.6:PLLO1EN、MISCCON.7: PLLO2EN 任何一位为 1 才能使 MISCCON.3:PLLPWD 设置有效。

读 MISCCON.2: PLLLOCK 位可以查看 PLL 时钟是否锁定，0 为没有锁定，1 为锁定。

2) clkcpu、clkout0、clkout1、clkout2 操作

当在 PLL wizard 中设置 clkcpu 为动态切换模式时，CLKCPUCON 的 0 位选择 MSS 8051 的时钟源，0 来自 clkx，1 来自 plloutx，其中 clkx 和 plloutx 是在 PLL wizard 中设定的。

设置 CLKCPUCON (F9)[2:0]改变 MSS 的 clkcpu 的分频值，最多为基频的 128 分频，基频是在 PLL wizard 中设定的；

改变 MISCCON.4: CLKCPUEN 的值使 CLKCPUEN 从 0 变化到 1 完成 clkcpu 的动态切换；

clkout0、clkout1、clkout2 的动态切换与 clkcpu 的动态切换操作相同，只是对应的 SFR 不同。

3) Fabric 的 gclk 门控操作

8051 有电源管理功能，利用此功能可以智能的门控 Fabric 的 gclk 时钟，最大限度的降低 HME-M1 的功耗。

用 clkcpuen 或 clkperen 作为 Fabric 的 gclk 时钟的门控信号，利用 8051 的电源管理功能就可以在需要的时候关断 Fabric 的 gclk 时钟。当 8051 cpu 处在 STOP 模式时，clkcpuen 和 clkperen 都为低，当 8051 cpu 处在 IDLE 模式时，clkcpuen 为低，clkperen 为高。8051 的电源管理详见 3.13 电源管理。

3.14.2 MSS SPI 操作

必须设置 I2CSPISEL（AB）的 0 位为 1 才能使 MSS 的 SPI 接口连到固定 I/O 上，否则 SPI 接口无法使用。

把 SPI FLASH 的端口与 MSS 的 SPI interface 通过 Fabric 相连后，HME-M1 就具备了 ISP 特征。下面以使用内嵌 SPI FLASH 为例说明。

FPGA 设计：例化 spi_interface 并连接到 MSS 的 8051SPI 接口上

Firmware 设计：

- 设置 I2CSPISEL（AB）的 0 位为 1
- 调用 SPI 驱动函数操作 SPI FLASH

3.14.3 MSS I²C I/O 设置

I2C 的 SCL 和 SDA 连到了固定的 I/O 引脚上，详见 6.2 引脚列表。用 MSS 的 I2C 接口连接片外的 I2C 器件必须设置 I2CSPISEL（AB）的 1 位为 1，使 SCL 和 SDA 对应的 I/O 为 I2C 属性。

3.14.4 MSS 在系统重配置

HME-M1 的配置 Image 由 FPGA 配置数据和 MSS 的程序代码组成，FPGA 配置数据大小基本上一样，MSS 的代码随程序的的大小变化。HME-M1 的配置 Image 是按 block 存放在 SPI FLASH 里。Sector 是 Image 存放的最小单位，大小为 0x10000。一个 Image 可占用多个 block。图 3-10 描述了多个 Image 存放在 SPI FLASH 的映射关系，其中一个 Image 小于 1 个 block 的大小。在 Fuxi 中，可利用 Configuration

packer 把多个 Image 组合成.mcf 文件，再用 Dwonload 把.mcf 里面的多个 Image 文件一次下载到 SPI FALSH 里，在输出信息里可以看到每个 Image 的开始地址。利用 ISC 特性，HME-M1 就可以利用 SPI FLASH 的空间换取 HME-M1 的容量，也就是说某些应用，HME-M1 可以实现远大于 1K FPGA LE 的逻辑。

ISC 步骤如下：

- 向 ISCADDR3~ISCADDR0 4 个 SFR 写入想要配置的 Image 在 SPI FLASH 的起始地址
- 向 MISCCON（0xF8）的第 0 位 ISCEN 写入 1

这两步后，HME-M1 从虚线指向的地址读取 Image2 的数据，进行自我配置。在重配置过程中 MSS 是处在复位状态，ISCADDR3~ISCADDR0 以及 ISCEN 都自动清 0。

注意：HME-M1 上电复位后，HME-M1 仍然是默认读取 Image1 的数据进行配置。

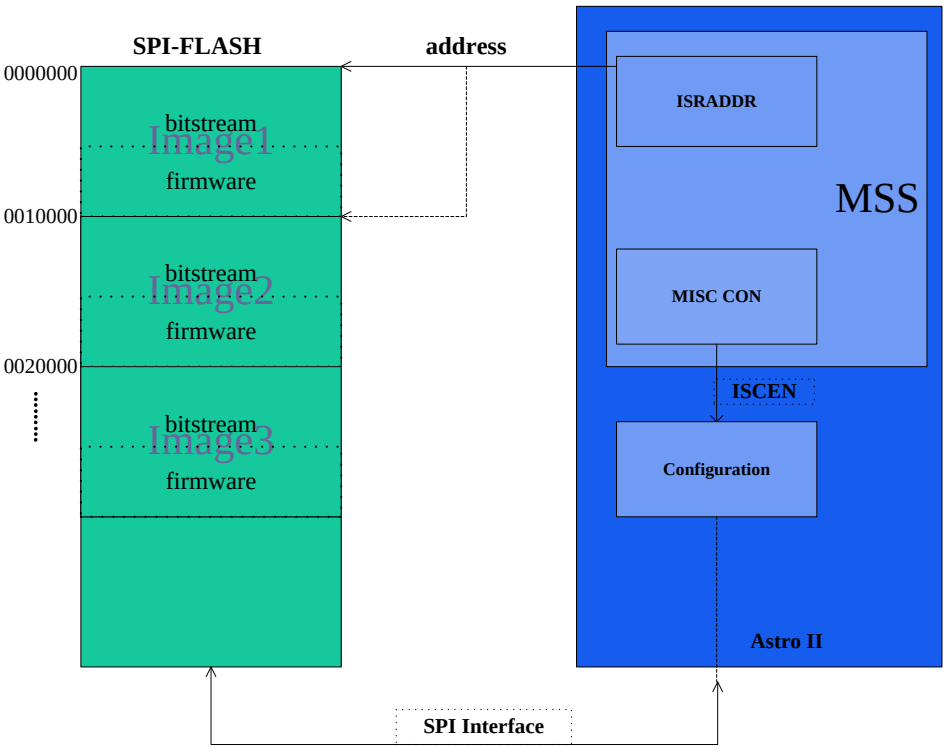


图 3-10 ISC

3.14.5 MSS 在系统更新

利用 ISC 和 ISP 特性就可以实现 HME-M1 的在系统更新。

- 利用 ISP 特性把新的 Image 数据编程到 SPI-FLASH 对应的 block
- 利用 ISC 特性用更新后的 Image 重新配置 HME-M1。

3.15 8051 实例化

使用 HME-M1 进行设计时，必须在 RTL 代码中对 8051 宏模块进行实例化。

3.15.1 8051 宏模块端口列表

表 3-58 8051 端口列表

Name	Type	Width	Description
clkcpu	I	1	MSS 8051 clock④
Fabric interface			
clkcpuen	O	1	is low when cpu is in STOP and IDLE mode.
clkperen	O	1	is low when cpu is in IDLE mode
reset	I	1	MSS reset，Low active
ro	O	1	MSS core reset output
swd	I	1	High level on this pin during reset starts the watchdog Timer immediately after reset is released
SPI interface			
scki	I	1	Serial clock input
scko	O	1	Serial clock output
scktri	O	1	Serial clock tri-state enable
ssn	I	1	Slave select input
misoi	I	1	“Master input / slave output” input pin
misoo	O	1	“Master input / slave output” output pin
misotri	O	1	“Master input / slave output” tri-state enable
mosii	I	1	“Master output / slave input” input pin
mosio	O	1	“Master output / slave input” output pin
mositri	O	1	“Master output / slave input” tri-state enable
spssn	O	4	Slave select output register
General IO			
port0i	I	8	8-bit input port
port0o	O	8	8-bit output port
port1i	I	8	8-bit input port, combine with int2-7, ccu, t2, rxd1
port1o	O	8	8-bit output port, combine with ccu, txd1
port2i	I	8	8-bit input port
port2o	O	8	8-bit output port
port3i	I	8	8-bit input port, combine with int0-1, rxd0, t0, t1
port3o	O	8	8-bit output port, combine with txd0, rxd0o
EMIF interface			
clkemif	I	1	EMIF interface clock
memack	I	1	memory acknowledge
memdatai	I	8	Memory data input
memdatao	O	8	Memory data output
memaddr	O	23	Memory address

④硬件内部已经连到全局时钟，需要在 PLL wizard 中选择时钟，不需要设计连接。

memwr	O	1	Memory write enable
memrd	O	1	Memory read enable
DPRAM 4K interface			
clkb	I	1	dpram 4k b port clock
cenb	I	1	dpram 4k b port chip select,low active
wenb	I	1	dpram 4k b port write enable, low active
ab	I	12	dpram 4k b port address
db	I	8	dpram 4k b port write data
qb	O	8	dpram 4k b port read output data

4 配置

4.1 配置模式

HME-M1 有三种配置模式：JTAG，AS，PS 模式。引脚 MSEL 来设置 HME-M1 的配置模式，如下所示。

注意：HME-M1 内置的 FLASH 芯片，只有 AS 和 JTAG 模式，没有 MSEL 引脚。

表 4-1 配置模式

Mode select pin MSEL	Mode	Description
0	AS	Active Serial mode. The chip will be configured automatically. Configuration data is stored in the SPI flash.
1	PS	Chip acts as slave. External microcontroller feeds configuration data into the chip.
0/1	JTAG	JTAG-based configuration. This mode takes high privilege over AS and PS modes

4.1.1 AS 模式

在 AS 配置模式下，HME-M1 上电复位或引脚 nCONFIG 复位后自动从 SPI flash 0 地址读取配置数据，配置 FPGA 和 MSS 的内嵌 RAM。

图 4-1 描述了不带 FLASH 的 HME-M1 在 AS 模式下的连接图。

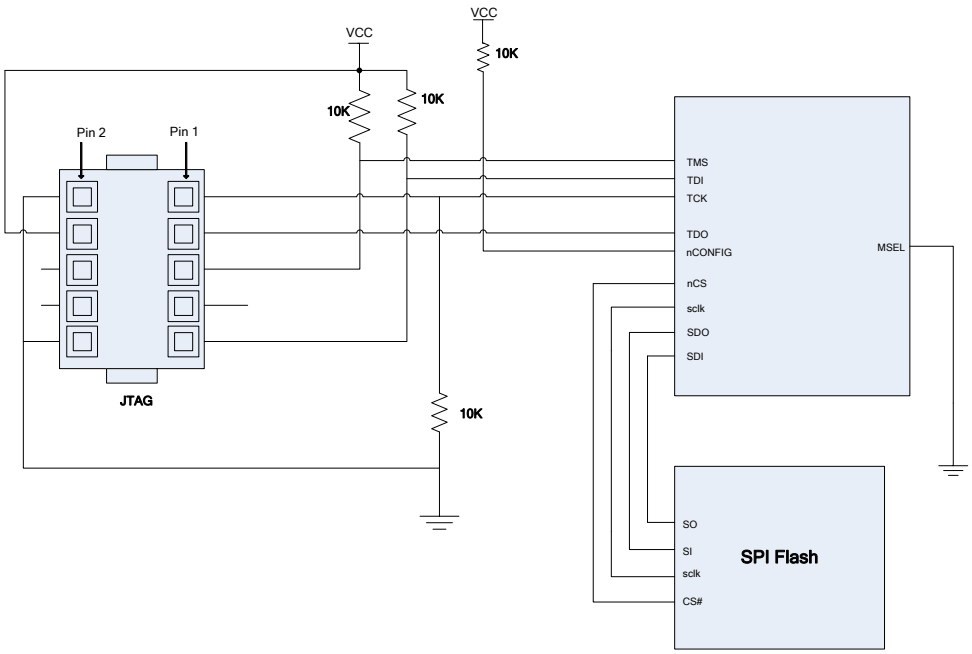


图 4-1 无内嵌 Flash 的 AS 配置

图 4-2 描述了带 FLASH 的 HME-M1 在 AS 模式下的连接图。

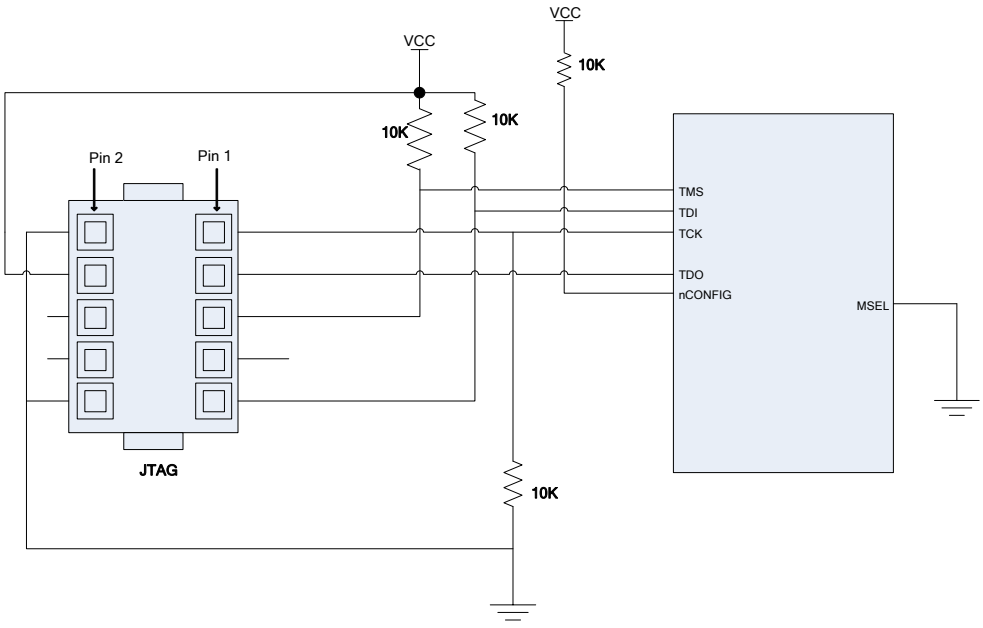


图 4-2 内嵌 Flash 的 AS 配置

4.1.2 PS 模式

在 PS 模式下，HME-M1 作为从设备被动接收从外部主控制器发来的配置数据。SPI Master 不能从 HME-M1 读出配置数据。

图 4-3 描述了 HME-M1 在 PS 模式下的连接图。

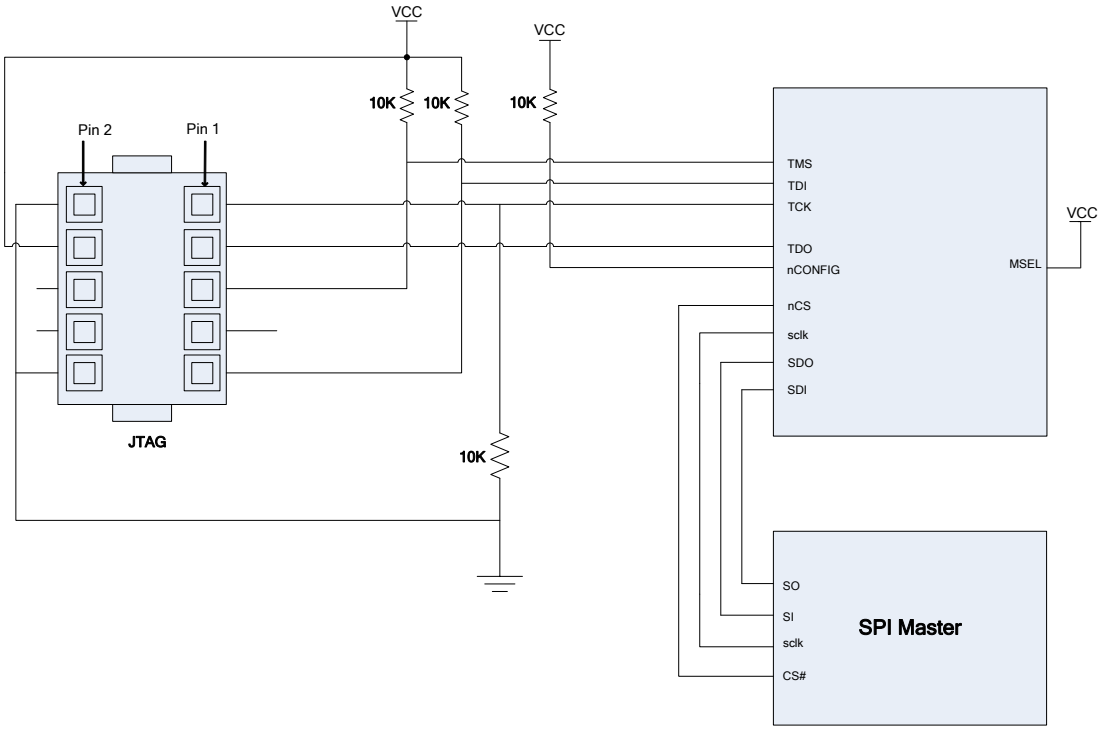


图 4-3 PS 配置

4.1.3 JTAG 模式

JTAG 模式下，主机通过 HME-M1 的 JTAG 端口配置和调试 FPGA 和 MSS。JTAG 端口比其他配置模式有更高优先权，可以在任何配置模式下优先访问配置和调试。

4.2 非易失性存储器

用于配置 HME-M1 的 SPI-Flash，无论是内嵌还是片外的在配置完成后，都可以在用户模式下操作。

使用内嵌的 SPI-Flash 存储器时，需要在 Fuxi wizard 中调用 spi interface。内嵌的 SPI-Flash 的 datasheet 见 GD25QXX_Rev1.0.pdf。

表 4-2 SPI Interface 端口

Port name	Type	Description
sclk	Input	spi flash input clock
sdo	output	spi flash serial output data
cson	Input	spi flash chip select，low active
sdi	Input	spi flash serial input data

使用外挂的 SPI-Flash 存储器时，需要如图 4-1 所示连接，并且要在用户设计中把 SPI 对应的 I/O 配置成用户 I/O，见 3.14.2。

4.3 ISC

ISC(In System Configuration)可以使 MSS 对 HME-M1 进行静态或动态的重新配置。只有在 AS 模式下，才可实现 ISC。

静态重配置就是 HME-M1 在运行过程中，MSS 程序向相应 ISC 的 SFR 里写入的地址和命令就可以使 HME-M1 重新从 SPI FLASH 指定的地址读取对应的 Image 重新配置 HME-M1。

动态重配置就是 HME-M1 在运行过程中，MSS 程序从外界读取 Image（通过串口或其他接口），通过 SPI 接口写到相应的 Image 空间里，更新 Image，MSS 程序再向相应 ISC SFR 里写入地址和命令就可以用新的 Image 配置 HME-M1。具体操作详见 3.14 MSS 系统控制。

4.4 安全

HME-M1 安全机制：

- 配置数据加密
- 基于密钥 flash 访问控制
- 配置 memory 保护

保护外部 flash 数据
保护内部 flash 数据
保护 FPGA 配置

4.4.1 配置数据加密

配置数据文件可加密。使用 LSFR 算法给配置数据加密，用户可以自定义 128 位的密钥。

4.4.2 FLASH 的操作控制

JTAG 操作保护使能后，在 AS 模式下 JTAG 不能读取 SPI FLASH 的数据，也不能操作 FPGA 和调试 MSS 的 8051 程序。1 位的 JTAG 操作保护使能标志和配置数据一起编程到 FLASH 里。

JTAG 操作保护使能后，只有 BE 指令（芯片擦除）和 RDSR 指令（读取状态寄存器）有效，其他指令都会被 HME-M1 内部的安全机制过滤。在 FLASH 中实施块擦除后，禁止标志被去除，JTAG 可以正常操作 FLASH。

JTAG 启动保护使能后，HME-M1 禁止 JTAG 对 HME-M1 的配置 Memory 操作。也就是说 JTAG 不能配置 HME-M1，也不能在线调试 MSS 8051。这个状态下的 HME-M1 就是用户自定义的 ASIC，只有使用擦除指令，才能重新配置 Astor Ⅱ。

HME-M1 的这些安全特性不仅能够保护用户的 FPGA 设计也能保护用户的固件。

注意：上述功能需要在 Fuxi 中设置才能实现。

4.5 HME-M1 上电复位

HME-M1 有内部 POR 电路用于监控 VCCINT 和 VCCIO 电压电平。上电后，POR 使芯片一直保持复位状

态，直到 VCCINT 和 VCCIO 都达到触发电平。当 HME-M1 进入用户方式后，POR 电路只监控 VCCINT 是否掉电，不再监控 VCCIO。POR 同时也受外部 nCONFIG 引脚控制，低则复位芯片。

POR 电路特点如下：

- 加电监控，触发点：
  - VCCINT： 0.75V-1.08V
  - VCCIO ： 1.8V-2.97V
- 断电监控，触发点：
  - VCCINT : 0.65V-0.9V
- 对 VCCINT 的低电压监控

延时：典型值是 4.9ms，范围是 3.3ms~7.4ms

5 电气特性

5.1 LVCMOS33 D.C. 规格

表 5-1 LVCMOS33 D.C. 特性

Parameter	Symbol	Condition	Min	Typ	Max	Units
Supply Voltage	VCCIO	-	2.97	3.3	3.63	V
Input Voltage Low	VIL	-	-0.3	-	0.8	V
Input Voltage High	VIH	-	2.0	-	5.5	V
Output Voltage Low	VOL	Iol = 8mA	-	-	0.4	V
Output Voltage High	VOH	Ioh= 8mA	2.4	-	-	V

5.2 LVTTL33 D.C. 规格

表 5-2 LVTTL33 D.C. 特性

Parameter	Symbol	Condition	Min	Typ	Max	Units
Supply Voltage	VCCIO	-	2.97	3.3	3.63	V
Input Voltage Low	VIL	-	-0.3	-	0.8	V
Input Voltage High	VIH	-	2.0	-	5.5	V
Output Voltage Low	VOL	Iol = 8mA	-	-	0.4	V
Output Voltage High	VOH	Ioh= 8mA	2.4	-	-	V

提示：

- **上面表格中的缩写描述如下：**
  VCCIO — 为单端输入和输出驱动提供电压
  Iol — VOL 检测到的情况下输出电流状态
  Ioh — VOH 检测到的情况下输出电流状态
  VIL — 低逻辑电平的输入电压
  VIH — 高逻辑电平的输入电压
  VOL — 低逻辑电平的输出电压
  VOH — 高逻辑电平的输出电压
- **绝对最大额定参数为压力等级。使用或超过上表中的参数对设备进行操作会造成不可恢复的损坏。**

注意：上表中的数值基于测试条件为：温度=25℃，VCCINT=1.2V， VCCIO=3.3V 。

6 引脚及封装

6.1 引脚定义

表 6-1 电源引脚

Name	Type	Description
VCCIO	Power	Digital supply to the FPGA fabric I/O bank，typical is 3.3V
VCCINT	Power	Digital supply to the FPGA core,typical is 1.2V
VCCA_PLL	Power	Analog supply to the PLL,typical is 1.2V
GND	Ground	Digital ground
GNDA_PLL	Ground	Analog ground to PLL

表 6-2 JTAG 引脚

Name	Type	Description
TMS	input	JTAG input pin.
TDI	input	JTAG input pin.
TCK	input	JTAG input pin.
TDO	output	JTAG output pin.

表 6-3 SPI 引脚

Name	Type	Description
SCLK	inout	Input (PS⑤ mode) output (AS mode) In passive serial configuration mode, SCLK is a clock input used to clock configuration data from external device source into device. In active serial configuration mode, SCLK is a clock output from device.
SDI	input	Dedicated configuration data input pin.
SDO	output	Active serial data output from the CLD device. In passive serial configuration, this pin becomes a user I/O pin.
nCSO	output	Chip select output to enable/disable a serial configuration device. This output is used during active serial configuration mode. In passive serial configuration mode this pin become user I/O.

所有 SPI 引脚若不作为 SPI 功能，可以用作普通 I/O。

⑤ 带 FLASH 的封装没有 MSEL 和 PS 模式

表 6-4 配置相关引脚表

Name	Type	Description
nCONFIG	input	Chip global reset input. Active low
CONF_DONE	output	Configuration status. configuration success: output high level configuration fail or not configured: output low level This pin can be used as general user I/O after configuration.
MSEL⑤	input	Dedicated mode select control pins for the configuration mode for the device: 0: Active Serial (Active serial Configuration), 1: Passive Serial This pin can be used as general user I/O after configuration.

表 6-5 用户引脚列表

Pin Name	Pin Type	Description
XIN	input	Crystal Input. This pin is for the oscillator circuit for a crystal or ceramic resonator. If overdriven by an external CMOS clock, this becomes the system clock input. This pin is better to connect to GND if it is not used.
XOUT	output	Crystal Output. This pin is the excitation driver for a crystal or ceramic resonator.
SCL	inout	I^2C clock This pin can be used as general user I/O if it is not used as I^2C SCL.
SDA	inout	I^2C data This pin can be used as general user I/O if it is not used as I^2C SDA.
CLKx	input	Dedicated global clock input This pin can be used as general user I/O if it is not used as CLK input.
IOxxx	inout	user I/O

6.2 引脚列表

目前，HME-M1 系列有 4 款产品：M1C01N0L144(AS2E5F1KL144)、M1C01N3L144 (AS2E5F1KAL144)、M1C01N3T100(AS2E5F1KAT100)、M1C01N0T100(AS2E5F1KT100)；分 2 种封装：LQFP-144 和 TQFP-100。M1C01N0L144(AS2E5F1KL144)和 M1C01N3L144(AS2E5F1KAL144)在电源和 I/O 上兼容；M1C01N3T100 (AS2E5F1KAT100)、M1C01N0T100 (AS2E5F1KT100)在电源和 I/O 上兼容。详细描述请参见表 8-1。

6.2.1 TQFP-100 封装引脚列表

表 6-6 TQFP-100 引脚列表

No.	M1C01N0T100 (AS2E5F1KT100)	M1C01N3T100 (AS2E5F1KAT100)	No.	M1C01N0T100 (AS2E5F1KT100)	M1C01N3T100 (AS2E5F1KAT100)
1	IO1	IO1	38	IO22	IO22
2	IO2	IO2	39	VCCINT	VCCINT
3	IO3	IO3	40	IO23	IO23
4	IO4	IO4	41	IO24	IO24
5	IO5	IO5	42	IO25	IO25
6	IO6	IO6	43	IO26	IO26
7	SDO /IO7	IO7	44	IO27	IO27
8	SCLK/IO8	IO8	45	VCCIO	VCCIO
9	VCCIO	VCCIO	46	GND	GND
10	GND	GND	47	IO28	IO28
11	GND	GND	48	IO29	IO29
12	CLK0 /IO9	CLK0 /IO9	49	IO30	IO30
13	VCCINT	VCCINT	50	IO31	IO31
14	CLK1 /IO10	CLK1 /IO10	51	IO32	IO32
15	IO11	IO11	52	IO33	IO33
16	IO12	IO12	53	IO34	IO34
17	XOUT	XOUT	54	IO35	IO35
18	XIN	XIN	55	IO36	IO36
19	IO13	IO13	56	IO37	IO37
20	IO14/SCL	IO14/SCL	57	IO38	IO38
21	IO15/SDA	IO15/SDA	58	CSON /IO39	IO39
22	TMS	TMS	59	VCCIO	VCCIO
23	TDI	TDI	60	GND	GND
24	TCK	TCK	61	SDI /IO40	IO40
25	TDO	TDO	62	CLK2/IO41	CLK2/IO41
26	VCCA_PLL	VCCA_PLL	63	VCCINT	VCCINT
27	GNDA_PLL	GNDA_PLL	64	CLK3/IO42	CLK3/IO42
28	CONF_DONE/IO16	CONF_DONE/I O16	65	GND	GND
29	IO17	IO17	66	IO43	IO43
30	nCONFIG	nCONFIG	67	IO44	IO44
31	VCCIO	VCCIO	68	IO45	IO45
32	GND	GND	69	IO46	IO46
33	MSEL/ I O18	IO18	70	IO47	IO47
34	IO19	IO19	71	IO48	IO48
35	IO20	IO20	72	IO49	IO49
36	IO21	IO21	73	IO50	IO50
37	GND	GND	74	IO51	IO51

No.	M1C01N0T100 (AS2E5F1KT100)	M1C01N3T100 (AS2E5F1KAT100)
75	IO52	IO52
76	IO53	IO53
77	IO54	IO54
78	IO55	IO55
79	GND	GND
80	VCCIO	VCCIO
81	IO56	IO56
82	IO57	IO57
83	IO58	IO58
84	IO59	IO59
85	IO60	IO60
86	IO61	IO61
87	IO62	IO62

No.	M1C01N0T100 (AS2E5F1KT100)	M1C01N3T100 (AS2E5F1KAT100)
88	VCCINT	VCCINT
89	IO63	IO63
90	GND	GND
91	IO64	IO64
92	IO65	IO65
93	GND	GND
94	VCCIO	VCCIO
95	IO66	IO66
96	IO67	IO67
97	IO68	IO68
98	IO69	IO69
99	IO70	IO70
100	IO71	IO71

6.2.2 LQFP-144 封装引脚列表

表 6-7 LQFP-144 引脚列表

No.	M1C01N0L144 (AS2E5F1KL144)	M1C01N3L144 (AS2E5F1KAL144)
1	IO1	IO1
2	IO2	IO2
3	IO3	IO3
4	IO4	IO4
5	IO5	IO5
6	IO6	IO6
7	SDO/ IO7	IO7
8	SCLK/ IO8	IO8
9	VCCIO	VCCIO
10	GND	GND
11	IO9	IO9
12	IO10	IO10
13	IO11	IO11
14	IO12	IO12
15	IO13	IO13
16	IO14	IO14
17	GND	GND
18	CLK0/IO15	CLK0/IO15
19	VCCINT	VCCINT
20	CLK1/IO16	CLK1/IO16
21	IO17	IO17

No.	M1C01N0L144 (AS2E5F1KL144)	M1C01N3L144 (AS2E5F1KAL144)
22	IO18	IO18
23	IO19	IO19
24	IO20	IO20
25	VCCIO	VCCIO
26	GND	GND
27	XOUT	XOUT
28	XIN	XIN
29	IO21	IO21
30	IO22/SCL	IO22/SCL
31	IO23/SDA	IO23/SDA
32	IO24	IO24
33	TMS	TMS
34	TDI	TDI
35	TCK	TCK
36	TDO	TDO
37	VCCA_PLL	VCCA_PLL
38	GNDA_PLL	GNDA_PLL
39	CONF_DONE/ IO25	CONF_DONE/IO25
40	IO26	IO26
41	IO27	IO27
42	IO28	IO28

No.	M1C01N0L144 (AS2E5F1KL144)	M1C01N3L144 (AS2E5F1KAL144)
43	IO29	IO29
44	IO30	IO30
45	nCONFIG	nCONFIG
46	VCCIO	VCCIO
47	GND	GND
48	MSEL/ IO31	IO31
49	IO32	IO32
50	IO33	IO33
51	IO34	IO34
52	IO35	IO35
53	IO36	IO36
54	GND	GND
55	IO37	IO37
56	VCCINT	VCCINT
57	IO38	IO38
58	IO39	IO39
59	IO40	IO40
60	IO41	IO41
61	IO42	IO42
62	IO43	IO43
63	IO44	IO44
64	VCCIO	VCCIO
65	GND	GND
66	IO45	IO45
67	IO46	IO46
68	IO47	IO47
69	IO48	IO48
70	IO49	IO49
71	IO50	IO50
72	IO51	IO51
73	IO52	IO52
74	IO53	IO53
75	IO54	IO54
76	IO55	IO55
77	IO56	IO56
78	IO57	IO57
79	IO58	IO58
80	IO59	IO59
81	SCON/ IO60	IO60
82	VCCIO	VCCIO
83	GND	GND

No.	M1C01N0L144 (AS2E5F1KL144)	M1C01N3L144 (AS2E5F1KAL144)
84	SDI /IO61	IO61
85	IO62	IO62
86	IO63	IO63
87	IO64	IO64
88	IO65	IO65
89	CLK2/IO66	CLK2/IO66
90	VCCINT	VCCINT
91	CLK3/IO67	CLK3/IO67
92	GND	GND
93	IO68	IO68
94	IO69	IO69
95	IO70	IO70
96	IO71	IO71
97	IO72	IO72
98	IO73	IO73
99	Gnd	Gnd
100	VCCIO	VCCIO
101	IO74	IO74
102	IO75	IO75
103	IO76	IO76
104	IO77	IO77
105	IO78	IO78
106	IO79	IO79
107	IO80	IO80
108	IO81	IO81
109	IO82	IO82
110	IO83	IO83
111	IO84	IO84
112	IO85	IO85
113	IO86	IO86
114	IO87	IO87
115	GND	GND
116	VCCIO	VCCIO
117	IO88	IO88
118	IO89	IO89
119	IO90	IO90
120	IO91	IO91
121	IO92	IO92
122	IO93	IO93
123	IO94	IO94
124	IO95	IO95

No.	M1C01N0L144 (AS2E5F1KL144)	M1C01N3L144 (AS2E5F1KAL144)
125	IO96	IO96
126	VCCINT	VCCINT
127	IO97	IO97
128	GND	GND
129	IO98	IO98
130	IO99	IO99
131	IO100	IO100
132	IO101	IO101
133	IO102	IO102
134	IO103	IO103

No.	M1C01N0L144 (AS2E5F1KL144)	M1C01N3L144 (AS2E5F1KAL144)
135	GND	GND
136	VCCIO	VCCIO
137	IO104	IO104
138	IO105	IO105
139	IO106	IO106
140	IO107	IO107
141	IO108	IO108
142	IO109	IO109
143	IO110	IO110
144	IO111	IO111

6.2.3 QFN-68 封装引脚列表

表 6-8 QFN-68 引脚列表

No.	QFN68
1	IO1
2	IO2
3	IO3
4	IO4
5	IO5
6	IO6
7	VCCIO0
8	IO7
9	VCCINT
10	IO8
11	VCCIO1
12	XOUT
13	XIN
14	TMS
15	TDI
16	TCK
17	TDO
18	VCCA_PLL
19	GNDA_PLL
20	IO9
21	IO10
22	IO11
23	VCCIO2
24	IO12
25	IO13

No.	QFN68
26	IO14
27	IO15
28	VCCINT
29	IO16
30	IO17
31	IO18
32	VCCIO3
33	IO19
34	IO20
35	IO21
36	IO22
37	IO23
38	VCCIO4
39	IO24
40	IO25
41	IO26
42	VCCINT
43	IO27
44	IO28
45	IO29
46	IO30
47	VCCIO5
48	IO31
49	IO32
50	IO33

No.	QFN68
51	IO34
52	IO35
53	IO36
54	IO37
55	VCCIO6
56	IO38
57	IO39
58	IO40
59	IO41

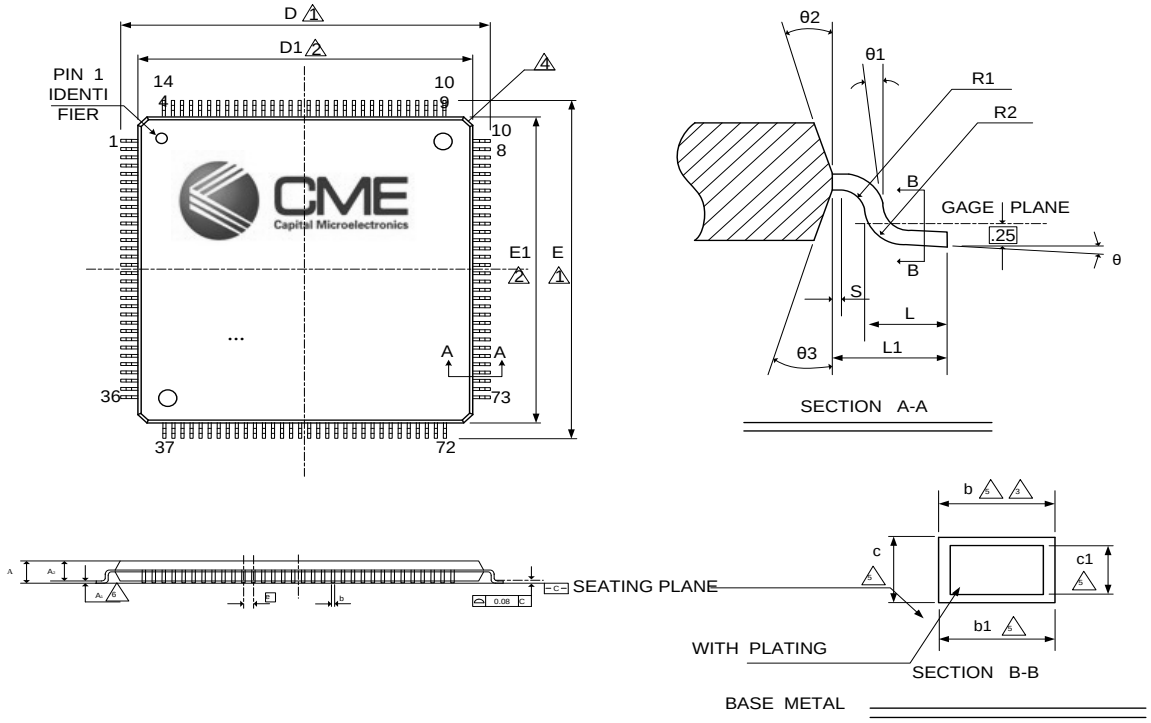
No.	QFN68
60	VCCINT
61	IO42
62	IO43
63	IO44
64	VCCIO7
65	IO45
66	IO46
67	IO47
68	IO48

注意：

- 引脚属性在前的为默认值，使用别的引脚属性需要配置。例如引脚 SDO /IO7，SDO 是默认值，若用作 IO，需要重配置 IO 属性。
- 内带 Flash 的 HME-M1 没有 MSEL，此类器件只有 AS 模式，没有 PS 模式。
- I^2C 的 SCL 和 SDA 只能从固定的引脚出来，该引脚默认为普通 IO，要用作 I^2C，详见 3.14.3。

6.3 封装信息

6.3.1 LQFP144 封装规格

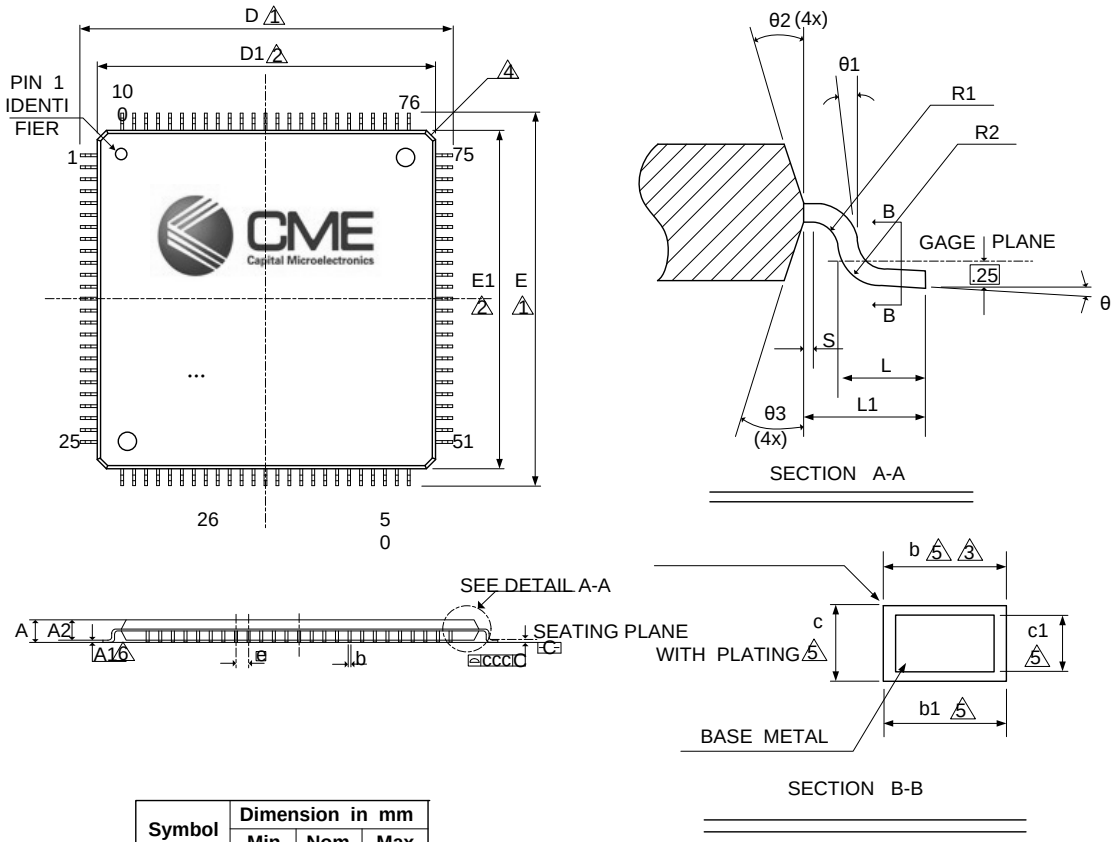


Symbol	Dimension in mm		
	Min	Nom	Max
A	——	——	1.60
A1	0.05	——	——
A2	1.35	1.40	1.45
b	0.17	0.22	0.27
b1	0.20	REF	
c	0.12	——	0.20
c1	0.13	REF	
D	21.85	22.00	22.15
D1	19.90	20.00	20.10
E	21.85	22.00	22.15
E1	19.90	20.00	20.10
e	0.50	BSC	
L	0.45	0.60	0.75
L1	1.00	REF	
R	0.15	REF	
R1	0.15	REF	
S	0.19	REF	
θ	0°	3.5°	7°
θ1	7°	REF	
θ2	12°	REF	
θ3	12°	REF	

1. TO BE DETERMINED AT SEATING PLANE -C-
2. DIMENSION D1 AND E1 DO NOT INCLUDE MOLD PROTRUSION D1 AND E1 ARE MAXIMUM PLASTIC BODY SIZE DIMENSION INCLUDING MOLD MISMATCH.
3. DIMENSION b DOES NOT INCLUDE DAMBAR PROTRUSION DAMBAR CAN NOT BE LOCATED ON THE LOWER RADIUS OR THE FOOT.
4. EXACT SHAPE OF EACH CORNER IS OPTIONAL.
5. THESE DIMENSIONS APPLY TO THE FLAT SECTION OF THE LEAD BETWEEN 0.10 mm AND 0.25 mm FROM THE LEAD TIP.
6. A1 IS DEFINED AS THE DISTANCE FROM THE SEATING PLANE
7. CONTROLLING DIMENSION : MILLIMETER. TO THE LOWEST POINT OF THE PACKAGE BODY.
8. REFERENCE DOCUMENT : JEDEC MS－026 , BFB

图 6-1 TQFP-144 封装

6.3.2 TQFP100 封装规格

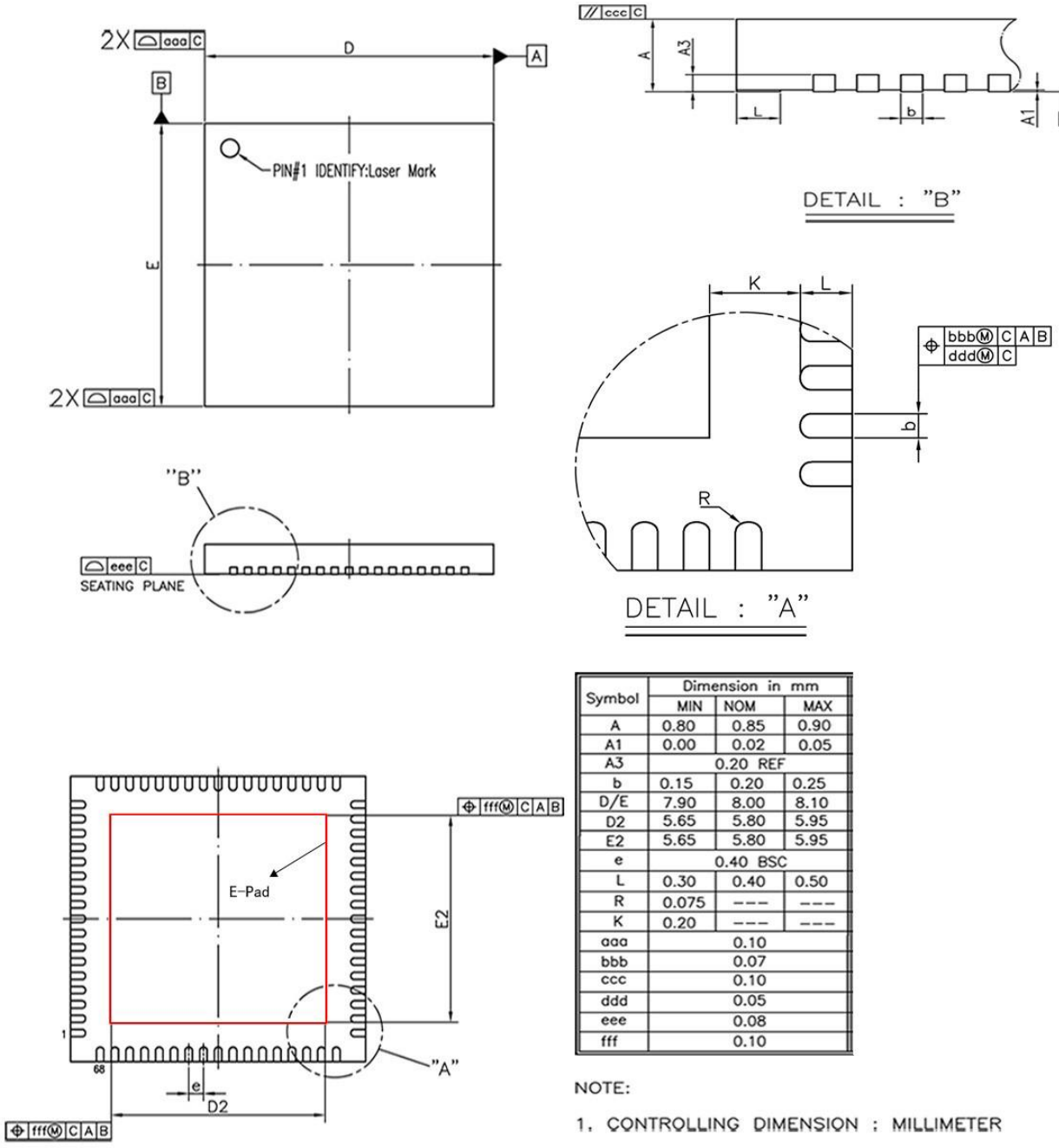


Symbol	Dimension in mm		
	Min	Nom	Max
A	——	——	1.20
A1	0.05	——	0.15
A2	0.95	1.00	1.05
b	0.17	0.22	0.27
b1	0.17	0.20	0.23
c	0.09	——	0.20
c1	0.09	——	0.16
D	16.00 BSC		
D1	14.00 BSC		
E	16.00 BSC		
E1	14.00 BSC		
e	0.50 BSC		
L	0.45	0.60	0.75
L1	0.15 REF		
R	0.08	——	——
R1	0.08	——	0.20
S	0.20	——	——
θ	0°	3.5°	7°
θ1	0°	——	——
θ2	11°	12°	13°
θ3	11°	12°	13°
ccc	0.08		

1. TO BE DETERMINED AT SEATING PLANE –C–
   DIMENSION D1 AND E1 DO NOT INCLUDE MOLD PROTRUSION
   D1 AND E1 ARE MAXIMUM PLASTIC BODY SIZE DIMENSION
2. INCLUDING MOLD MISMATCH.
   DIMENSION b DOES NOT INCLUDE DAMBAR PROTRUSION
3. DAMBAR CAN NOT BE LOCATED ON THE LOWER RADIUS OR THEFOOT.
4. EXACT SHAPE OF EACH CORNER IS OPTIONAL.
   THESE DIMENSIONS APPLY TO THE FLAT SECTION OF THE LEAD
5. BETWEEN 0.10 mm AND 0.25 mm FROM THE LEAD TIP.
6. A1 IS DEFINED AS THE DISTANCE FROM THE SEATING PLANE
7. CONTROLLING DIMENSION : MILLIMETER.
   TO THE LOWEST POINT OF THE PACKAGE BODY.
8. REFERENCE DOCUMENT : JEDEC MS－026 , BFB
9. SPECIAL CHARACTERISTICS C CLASS : ccc

图 6-2 LQFP-100 封装

6.3.3 QFN68 封装规格



Symbol	Dimension in mm		
	MIN	NOM	MAX
A	0.80	0.85	0.90
A1	0.00	0.02	0.05
A3		0.20 REF	
b	0.15	0.20	0.25
D/E	7.90	8.00	8.10
D2	5.65	5.80	5.95
E2	5.65	5.80	5.95
e		0.40 BSC	
L	0.30	0.40	0.50
R	0.075	---	---
K	0.20	---	---
aaa		0.10	
bbb		0.07	
ccc		0.10	
ddd		0.05	
eee		0.08	
fff		0.10	

NOTE:

1. CONTROLLING DIMENSION : MILLIMETER

图 6-3 QFN68 封装

7 开发工具

Hercules Microelectronics 的开发工具可以使开发者在 HME-M1 芯片上简单方便地实现两种设计：FPGA 设计和嵌入式 software 设计。

Hercules Microelectronics 的 Fuxi Integrated Design Environment (IDE) 支持全部 Hercules Microelectronics 公司的芯片，能够实现 synthesis、mapper、placement、routing、bitgen 、simulation 等全流程的 FPGA 设计。支持第三方 EDA 工具：Leondaro Spectrum 和 Modelsim。

Fuxi 内嵌 Keil-C 的 Hercules Microelectronics 的 AGDI 接口，嵌入式软件开发者可以在 Keil-C 编译和在线调试 8051firmware。HME-M1 的 MSS 具有片上调试（OCDS）性能能 帮助用户轻松调试 8051 程序。更多信息，请参见“**HME-M1 8051 调试用户手册**”。

请购买 Keil-C 软件工具，应用于 8051 编程、编译、调试等。更多信息，请参照 http://www.keil.com/c51。

目前，不支持其它第三方开发环境。请将您的意见和需求发送给我们 support@hercules-micro.com。

MSS 的 OCDS 调试接口与 FPGA 共用同一个 JTAG 接口。Hercules Microelectronics 提供的下载电缆和软件工具自动识别并操作 MSS 的 OCDS 或 FPGA 的 JTAG，但两者不能同时使用。

8 商务指南

8.1 HME-M1 系列产品列表

目前，HME-M1 系列有 8 个产品。若有其他需求请联系 hercules-micro。详细描述请参见表 8-1。

表 8-1 HME-M1 系列产品列表

Index	Ordering number	Packaging	Description
1	M1C01N0L144I7(AS2E5F1KL144I7)	LQFP-144	LQFP-144 without internal flash
2	M1C01N0L144C7(AS2E5F1KL144C7)	LQFP-144	LQFP-144 without internal flash
3	M1C01N3L144I7(AS2E5F1KAL144I7)	LQFP-144	LQFP-144 with internal flash
4	M1C01N3L144C7(AS2E5F1KAL144C7)	LQFP-144	LQFP-144 with internal flash
5	M1C01N0T100I7(AS2E5F1KT100I7)	TQFP-100	TQFP-100 without internal flash
6	M1C01N0T100C7(AS2E5F1KT100C7)	TQFP-100	TQFP-100 without internal flash
7	M1C01N3T100I7(AS2E5F1KAT100I7	TQFP-100	TQFP-100 with internal flash
8	M1C01N3T100C7(AS2E5F1KT100C7)	TQFP-100	TQFP-100 with internal flash
9	M1C01N3Q68I7	QFN-68	QFN-68 with internal flash
10	M1C01N3Q68C7	QFN-68	QFN-68 with internal flash

8.2 产品命名规则

所有产品型号都遵守如下的定义方式

Vendor	Product Series	Device Type	LUT Density	NVM Density	Package Type	Temperature Range	Speed Grade
HME-	M1	C	06	N3	L144	C	7

图 8-1 产品命名规则

产品系列

- M1　　衡山系列

器件类型

- C　　FPGA + SRAM + MCU

逻辑密度

- 01　　1K LUTs

非挥发性存储器

- N0 Without internal SPI-flash
- N3 With 4Mb internal SPI-flash

封装类型

- T Thin Quad Flat Pack (TQFP)
- L Low profile quad flat package (LQFP)
- Q Quad Flat No-leadPackage (QFN)

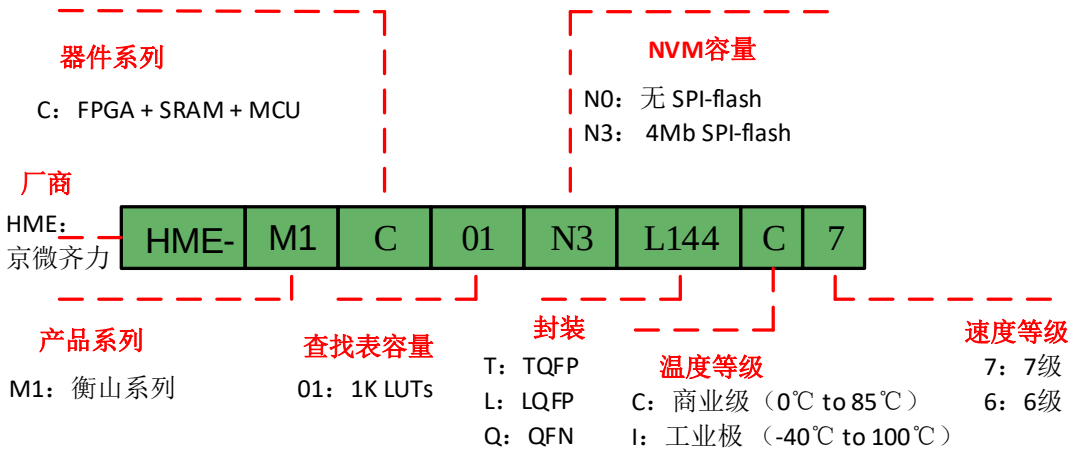
温度范围

- C 商业级 (0℃ to 85℃)
- I 工业级 (-40℃ to 100℃)

速度等级

- # Speed (7 for speed 7, 6 for speed 6, …)

器件型号实例：



9 缩写

缩写	全名
ALU	**A**rithmetic-**L**ogic **U**nit
AS	**A**ctive **S**erial
CCU	**C**ompare **C**apture **U**nit
CMS	**C**ontrol **M**ux **S**witch
CPU	**C**ontrol **P**rocessor **U**nit
DMS	**D**own **M**ux **S**witch
DPRAM	**D**ual **P**ort **R**AM
EMB	**E**mbedded **M**emory **B**lock
IAP	**I**n **A**pplication **P**rogramming
I2C	**I**nter-**IC** – a serial interface designed by Philips Semiconductors
ISC	**I**n **S**ystem **C**onfiguration
ISP	**I**n **S**ystem **P**rogramming
ISR	**I**nterrupt **S**ervice **R**outine unit
LSB	**L**east **S**ignificant **B**it
MDU	**M**ultiplication-**D**ivision **U**nit
MOVC	**M**ove **P**rogram **M**emory
MOVX	**M**ove **E**xternal **M**emory
MSB	**M**ost **S**ignificant **B**it
MSS	**M**icrocontroller **S**ubsystem
OCDS	**O**n-**C**hip **D**ebug **S**upport
OCI	**O**n-**C**hip **I**nstrumentations
PMU	**P**ower **M**anagement **U**nit
PS	**P**assive **S**erial
RTC	**R**eal **T**ime **C**lock
SFR	**S**pecial **F**unction **R**egister
SPI	**S**erial **P**eripheral **I**nterface
UMS	**U**p **M**ux **S**witch