

NS公司新型COP8 单片机应用指南

美国国家半导体公司 选编
复旦大学计算机科学系 编译

1.1
COP8

复旦大学出版社

图书在版编目(CIP)数据

NS 公司新型 COP8 单片机应用指南/美国国家半导体公司
选编;复旦大学计算机科学系编译. —上海:复旦大学出版社,
1999.9

ISBN 7-309-02366-8

I. N… I. ①美…②复… II. 单片微型计算机,COP8
IV. TP368.1

中国版本图书馆 CIP 数据核字(1999)第 43252 号

出版发行 复旦大学出版社

上海市国权路 579 号 200433

86-21-65102941(发行部) 86-21-65642892(编辑部)

fupnet@fudanpress.com http://www.fudanpress.com

经销 新华书店上海发行所

印刷 江苏大丰市印刷二厂

开本 787×1092 1/16

印张 12

字数 296 千

版次 1999 年 9 月第一版 1999 年 9 月第一次印刷

印数 1—3 500

定价 20.00 元

如有印装质量问题,请向复旦大学出版社发行部调换。

版权所有 侵权必究

前 言

美国国家半导体公司(NS公司)单片机是世界上发展最早的单片机产品之一,COP8单片机以其设计独特、采用M²CMOS工艺、全静态超低功耗、可靠性高及性能价格比高等优点著称于世。其产品广泛应用于家用电器、智能仪表、工业控制、通信、航空、航天等领域,其销量在世界上一直处于领先地位。

新型COP8单片机具有低价格OTP、超低功耗HALT和IDLE方式、双向三线串行MICROWIRE/PLUS、多功能定时器、WATCHDOG定时器、时钟监控及多输入唤醒等功能。

为了使我国从事单片机应用人员对NS单片机,特别是新型COP8单片机有一个全面了解。在美国国家半导体公司的大力支持下,我们编写了《NS公司新型COP8单片机应用指南》一书,根据应用需要,可以选择应用所需的单片机型号,并可向NS公司及其代理商索取相应单片机的技术手册、用户手册等详细资料。

《NS公司新型COP8单片机应用指南》一书共分七章,第一章概要地介绍了NS公司单片机,第二章介绍了新型COP8单片机系统结构,第三章介绍了新型COP8单片机的I/O功能,第四章为新型COP8单片机指令系统,第五章介绍了COP8各种型号的单片机产品,第六章介绍了COP8单片机的应用方法,第七章介绍了COP8单片机的一些应用实例。本书由复旦大学计算机科学系陈章龙和陈泽文编译,涂时亮审阅,NS公司区竞成先生和徐平波先生给予了大力支持。晨兴电子科技有限公司瞿其刚先生和魏巍先生也给予了帮助。

本书编写力求深入浅出,系统全面,简明扼要。本书可作科研、教学和工程技术人员的设计参考书及技术手册。由于编者水平有限,错误和不妥之处敬请批评指正。

编 者

1999.5

内 容 简 介

NS 单片机是世界上发展最早及应用广泛的单片机之一,具有可靠性好、性能价格比高等优点,广泛应用于家用电器、通信、航天航空、智能仪表、工业控制等领域。本书简明扼要地介绍了 NS 新型 COP8 单片机的系统结构、I/O 功能、产品特点、应用方法及应用实例。使读者能对 NS 单片机有较全面的了解,帮助正确选用 NS 单片机。

本书可作科研、教学和工程技术人员的设计参考书和技术手册。

目 录

第一章 NS 单片机概述	1
§ 1.1 COP8 单片机简介	1
1.1.1 特色型 COP8 单片机简介	3
1.1.2 基本型 COP8 单片机简介	5
1.1.3 新型 COP8 单片机简介	7
1.1.4 闪烁存储器型 COP8 单片机简介	9
§ 1.2 NS 单片机开发	10
1.2.1 iceMASTER 在线仿真器	11
1.2.2 COP8-DM 在线仿真器	12
1.2.3 EPU-COP8 性能评价板	13
1.2.4 Insight PE-COP8 在线仿真器	14
1.2.5 仿真开发软件	14
第二章 新型 COP8 单片机系统结构	16
§ 2.1 新型 COP8 单片机 CPU 结构	16
§ 2.2 新型 COP8 单片机存储器	20
2.2.1 新型 COP8 单片机数据存储器	20
2.2.2 新型 COP8 单片机程序存储器	22
2.2.3 新型 COP8 单片机用户存储区域	23
§ 2.3 新型 COP8 单片机中断及复位系统	23
2.3.1 新型 COP8 单片机中断系统	23
2.3.2 新型 COP8 单片机复位	27
2.3.3 新型 COP8 单片机时钟电路	28
§ 2.4 新型 COP8 单片机省电工作方式	30
2.4.1 新型 COP8 单片机 HALT 工作方式	30
2.4.2 新型 COP8 单片机 IDLE 工作方式	31
第三章 新型 COP8 单片机 I/O 功能	32
§ 3.1 新型 COP8 单片机定时器	32
3.1.1 新型 COP8 单片机空闲 IDLE 定时器 T0	32
3.1.2 新型 COP8 单片机定时器 T1	33
3.1.3 新型 COP8 单片机定时器 T2 和 T3	36
3.1.4 新型 COP8 单片机 WATCHDOG 定时器	37
§ 3.2 新型 COP8 单片机 MICROWIRE/PLUS 微总线	39
3.2.1 新型 COP8 单片机 MICROWIRE/PLUS 微总线简介	39
3.2.2 新型 COP8 单片机 MICROWIRE/PLUS 微总线接口	41
§ 3.3 新型 COP8 单片机 I/O 端口	47
3.3.1 新型 COP8 单片机多功能 I/O 端口	47
3.3.2 新型 COP8 单片机多输入唤醒功能	50

§ 3.4	新型 COP8 单片机 UART	52
3.4.1	新型 COP8 单片机 UART 系统结构	52
3.4.2	新型 COP8 单片机 UART 工作方式	56
§ 3.5	新型 COP8 单片机模拟比较器	57
§ 3.6	新型 COP8 单片机高分辨率 A/D 转换	57
第四章	新型 COP8 单片机指令系统	60
§ 4.1	新型 COP8 单片机寻址方式	60
4.1.1	新型 COP8 单片机操作数寻址	60
4.1.2	新型 COP8 单片机控制转移寻址	61
§ 4.2	新型 COP8 单片机指令系统	61
4.2.1	新型 COP8 单片机指令种类	61
4.2.2	新型 COP8 单片机指令系统	63
§ 4.3	新型 COP8 单片机程序设计举例	68
4.3.1	RAM 清除	68
4.3.2	二进制/BCD 码算术运算	68
4.3.3	二进制码乘法/除法运算	71
第五章	COP8 单片机产品	75
§ 5.1	新型 COP8 单片机产品	75
5.1.1	COP8SA×单片机(Apollo)产品	75
5.1.2	COP8SG×单片机产品	78
5.1.3	COP8ACC7 单片机产品	82
§ 5.2	基本型 COP8 单片机产品	84
5.2.1	基本类单片机产品(一)——COP912C/COP912CH	85
5.2.2	基本类单片机产品(二)——COP620C/COP622C/COP640C/COP642C COP820C/COP822C/COP840C/COP842C COP920C/COP922C/COP940C/COP942C COP680C/COP681C/COP880C/COP881C COP980C/COP981C	86
5.2.3	基本类单片机产品(三)——COP8780/COP8781/COP8782	90
5.2.4	CJ 类单片机产品——COP820CJ/COP822CJ/COP823CJ	90
§ 5.3	特色型 COP8 单片机产品	92
5.3.1	COP8 B 类单片机产品——COP888EB/COP889EB COP688EB/COP689EB	93
5.3.2	COP8 C 类单片机产品——COP684BC/COP884BC	95
5.3.3	COP8 D 类单片机产品——COP888GD/COP988GD	96
5.3.4	COP8 F 类单片机产品——COP988CF/COP984CF COP888CF/COP884CF	98
5.3.5	COP8 G 类单片机产品——COP884CG/COP888CG COP688EG/COP684EG/COP888EG COP884EG/COP988EG/COP984EG COP888GG/COP888HG/COP888KG	99

5.3.6	COP8 H 类单片机产品——COP684FH/COP688FH COP884FH/COP888FH COP984FH/COP988FH	102
5.3.7	COP8 K 类单片机产品——COP984EK/COP884EK COP988EK/COP888EK	104
5.3.8	COP8 L 类单片机产品——COP984CL/COP884CL/COP684CL COP988CL/COP888CL/COP688CL	106
5.3.9	COP8 S 类单片机产品——COP984CS/COP884CS/COP684CS COP988CS/COP888CS/COP688CS	108
5.3.10	COP8 W 类单片机产品——COP888GW	110
第六章	COP8 单片机应用方法	113
§ 6.1	COP8 单片机 LED 直接驱动方法	113
§ 6.2	COP8 单片机 LCD 直接驱动方法	114
§ 6.3	COP8 单片机简易 A/D 转换(V/F)方法	130
§ 6.4	COP8 单片机脉冲调宽 A/D 转换方法	135
§ 6.5	COP8 单片机定时器应用方法	138
6.5.1	COP8 单片机定时器 PWM 应用方法	138
6.5.2	COP8 单片机定时器输入捕获应用方法	144
6.5.3	COP8 单片机定时器外部事件计数器应用方法	146
§ 6.6	COP8 单片机微总线 MICROWIRE/PLUS 应用方法	147
第七章	COP8 单片机的应用实例	148
§ 7.1	COP87A×7 单片机在无绳电话设计中的应用	148
§ 7.2	COP8 单片机在直流步进马达控制设计中的应用	149
§ 7.3	COP8SAC7 单片机在洗衣机中的应用	157
§ 7.4	COP8SAB7 单片机在空调机中的应用	161
§ 7.5	COP8 单片机 MathPak	164
7.5.1	二进制和 BCD 加法及减法	165
7.5.2	乘法运算	166
7.5.3	除法运算	173
7.5.4	十进制(压缩 BCD 码)/二进制数转换	178
参考文献		181

第一章 NS 单片机概述

美国国家半导体公司(National Semiconductor Corporation,简称 NSC 或 NS 公司)是世界上著名的半导体生产厂商,该公司产品系列齐全、性能先进、质量可靠、性能价格比高。

NS 公司生产的单片机系列产品,既有早期生产的 4 位单片机 COP4 系列,也有高性能的 16 位单片机系列 HPC。其 8 位单片机 COP8 系列是 NS 公司的主要单片机产品。

§ 1.1 COP8 单片机简介

COP 是英文 Control Oriented Processor 的缩写,其含义为:面向控制的处理器,即以控制为应用目标的单片机。COP8 单片机品种齐全,应用范围广;根据应用对象不同可以分为 Feature(特色)型、Basic(基本)型和新型(高集成型)三大种类(如图 1-1 所示)。特别是新型(高集成型)COP8 单片机具有更高的性能价格比,本书以新型 COP8 单片机为主来介绍。

表 1-1 是 COP8 单片机系列表。

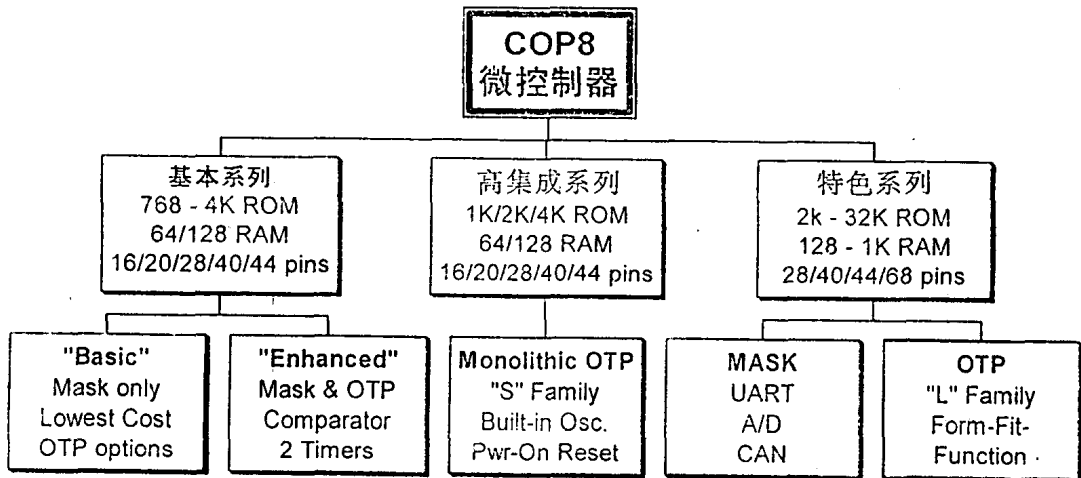


图 1-1 COP8 单片机系列图

表 1-1 COP8 单片机系列表

[illegible]

1.1.1 特色型 COP8 单片机简介

特色型 COP8 单片机是 NS 公司传统的单片机,广泛应用于各个领域。其主要特性如下:

- 高性能价格比,全静态 CMOS 工艺
- 高可靠及高抗干扰性
- 片内 ROM:2K~32K 字节
- 片内 RAM:64~512 字节
- 指令执行时间:1 μ s
- 绝大多数型号单片机都具有 3 个 16 位定时/计数器 T₁,T₂ 和 T₃
- PWM 工作方式
- 外部事件计数器
- 输入捕获工作方式
- IDLE 定时器 T0
- WATCHDOG 定时器
- 时钟监视器
- MICROWIRE/PLUS 微总线
- I/O 端口:23~54 根 I/O
- 软件选择:三态输出、推挽输出、弱上拉输入、高阻抗输入
- I/O 端口 G,L 和 M 具有施密特输入功能
- MIWU(Multi-Input Wake-Up)多路输入唤醒功能
- 中断源:10~14 级
- 工作电压:2.5~6.0V
- 省电工作方式:HALT 和 IDLE 工作方式
- 其他特性:
 - 逐次逼近
 - A/D 转换(B,D 和 F 类单片机)
 - 单积分 A/D 转换(K 类单片机)
 - 模拟比较器(C,G,H 和 K 类单片机)
 - UART 串行接口(B,G 和 H 类单片机)
 - SPI 串行外设接口(B 类单片机)
 - CAN 接口(B,C 类单片机)
 - 减弱 EMI(电磁干扰)电路(B,C,G,H 和 K 类单片机)
 - 高速 PWM(C 类单片机)
 - 硬件乘/除部件(H,W 类单片机)
 - 脉冲发生器(W 类单片机)

特色型 COP8 单片机产品种类很多,其命名方法如图 1-2 所示。

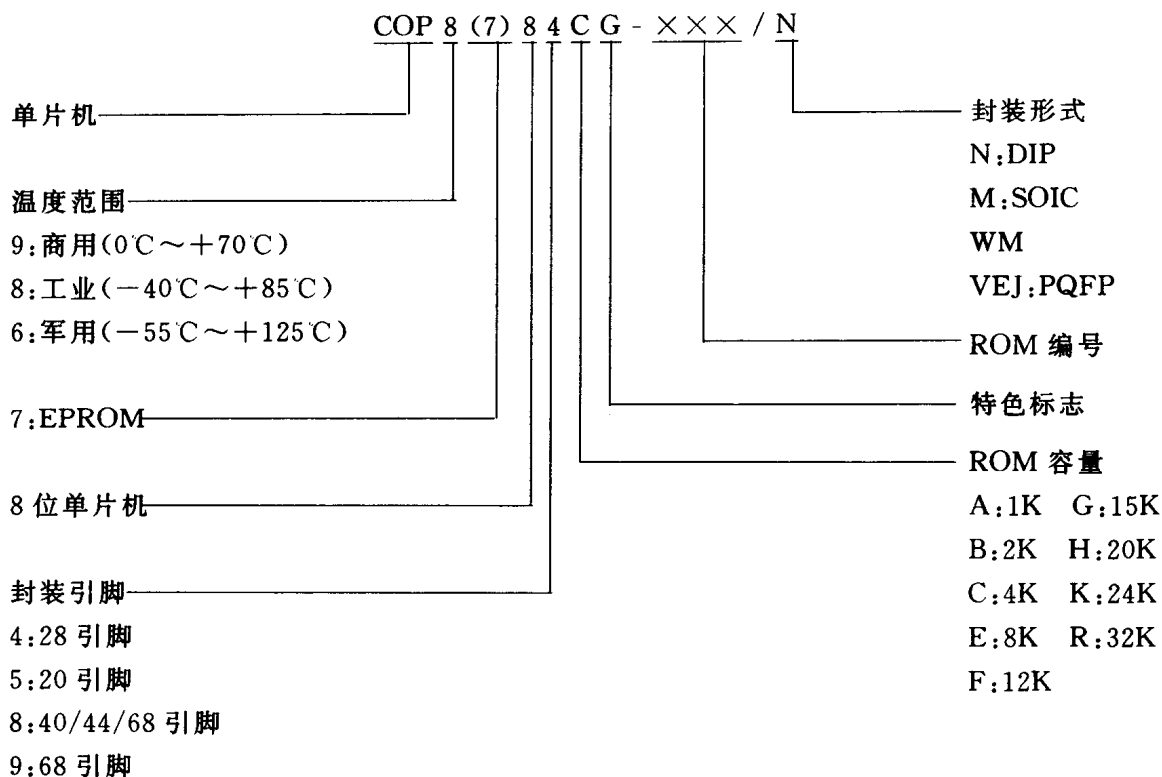


图 1-2 特色型 COP8 单片机命名法

同一种单片机可以有商用(COP98×)、工业用(COP88×)、军用(COP68×)及 OTP 型(L 系列)(COP87L8×)等四种形式。表 1-2 是特色 COP8 单片机产品表。

表 1-2 特色型 COP8 单片机产品表

	商用级 0° ~ +70 C	工业级 -40° ~ +85 C	军用级 -55° ~ +125 C	OTP 型
B 类		888EB 889EB	688EB 689EB	87L88EB 87L89EB
C 类		884BC	684BC	87L84BC
F 类	984CF 988CF	884CF 888CF		87L84CF 87L88CF
L 类	984CL 988CL	884CL 888CL	684CL 688CL	87L84CL 87L88CL
D 类	988GD	888GD		
S 类	984CS 988CS	884CS 888CS	684CS 688CS	87L84EG 87L88EG
G 类	984EG 988EG	888CG 884EG 888EG 888GG 888HG	684EG 888EG	87L88EG 87L84EG 87L88EG 87L88GG 87L88HG
H 类		888KH 888FH		87L88KG 87L88FH
K 类	984EK 988EK	884EK 888EK		87L84EK 87L88EK
W 类		888GW		87L88RW

1.1.2 基本型 COP8 单片机简介

基本(Basic)型 COP8 单片机为了适应家用电器、小型仪器仪表及简易控制系统,简化了特色型 COP8 单片机,同时,根据应用不同,其增强型(Enhanced)增加了一些专用特性(如调整器、个性 E²PROM 等)。基本型 COP8 单片机主要特性如下:

- 低价格、全静态 CMOS 工艺
- 片内 ROM:768~4K 字节
- 片内 RAM:64~128 字节
- 指令执行时间:1 μ s~2.5 μ s
- 一个 16 位定时/计数器
 - 16 位自动重装寄存器的定时器
 - 16 位外部事件计数器
 - 16 位输入捕获功能
- MICROWIRE/PLUS 微总线
- 中断源:3 级
 - 外部中断
 - 定时器中断
 - 软件中断
- I/O 端口:11~35 根 I/O
 - 软件选择:三态输出、推挽输出和弱上拉输入
 - 端口 G 为施密特输入
- 工作电压:2.3~6.0V
- 省电工作方式:HALT 工作方式
- 其他特性:
 - 调制器/定时器(IR 传输的高速 PWM)(CJ 类单片机)
 - WATCHDOG 定时器(CT 类单片机)
 - MIWU 多路输入唤醒功能(CJ 类单片机)
 - 15mA 大电流驱动(CJ 类单片机)
 - 模拟比较器(CJ 类单片机)

基本型 COP8 单片机的命名方法如图 1-3 所示。

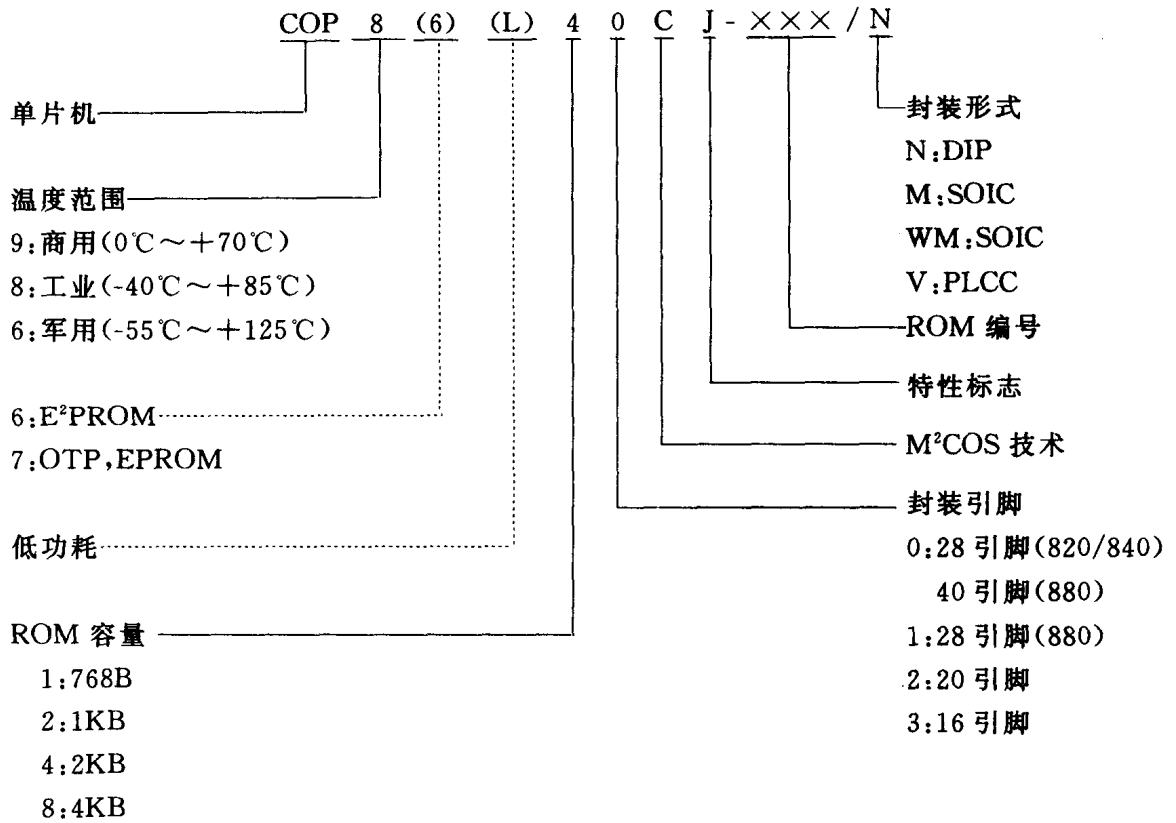


图 1-3 基本型 COP8 单片机命名法

基本型 COP8 单片机分为基本 (C 类) 和增强 (CJ 类) 两类, 同一种产品也有商用级、工业级、军用级和 OTP 型等四种形式, 表 1-3 是基本型 COP8 单片机产品表。

表 1-3 基本型 COP8 单片机产品表

	商用级 0°C ~ +70°C	工业级 -40°C ~ +85°C	军用级 -55°C ~ +125°C	OTP 型
CJ 类	942CJ 940CJ	823CJ 822CJ 820CJ 842CJ 840CJ		87L22CJ 87L20CJ 87L42CJ 87L40CJ
C 类	912C/CH 922C 920C 942C 940C 982C 981C 980C	822C 820C 842C 840C 882C 881C 880C	622C 620C 642C 640C 682C 681C 680C	87L22CK 87L22CJ 87L20CJ 87L42CJ 87L40CJ 87L42CJ 87L40CJ

1.1.3 新型 COP8 单片机简介

新型(高集成型)COP8 单片机(也称为 S 系列)是在特色型的 OTP 单片机进行优化的基础上,增加了片内振荡器电路、上电复位功能及 8 字节用户 EPROM 等。因而,成为目前国际上比较有竞争力的低价格 OTP 型单片机。近来,又推出了相应的掩膜型单片机如图 1-4 所示。新型 COP8 单片机主要特性如下:

- 低价格 OTP
- 片内 EPROM:1K~32K 字节
- 片内 RAM:64~512 字节
- 8 字节用户区域 EPROM
- 指令执行时间:1 μ s
- 16 位定时/计数器:1~3 个
 - PWM 工作方式
 - 外部事件计数器工作方式
 - 输入捕获工作方式

可提供的单片机 OTP/ROM 产品

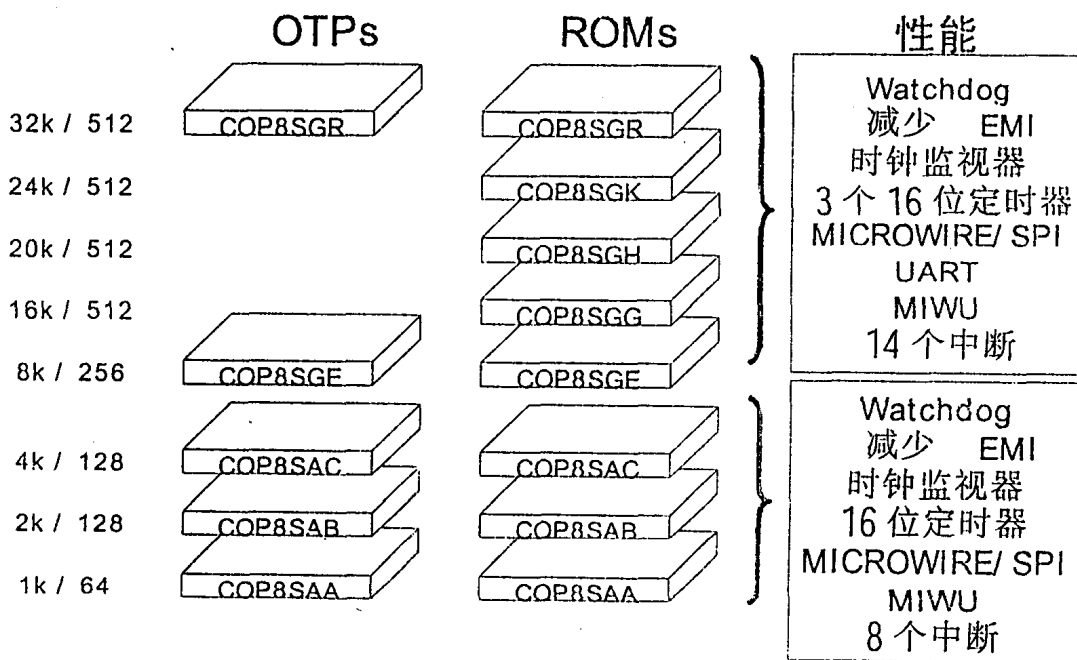


图 1-4 新型 COP8 单片机产品

- 空闲(Idle)定时器 T0
- WATCHDOG 定时器
- MICROWIRE/PLUS 微总线
- I/O 端口:12~40 根 I/O
 - 软件选择:三态输出、推挽输出、弱上拉输入和高阻输入等几种形式
 - 端口 G 和 L 输入具有施密特电路

- 多至 12 根大电充输出引脚
- MIWU 多路输入唤醒逻辑
- 中断源:8~14 个
- 工作电压:2.7~5.5V
- 省电工作方式:HALT 和 IDLE 工作方式
- 片内振荡器电路
- 上电复位功能
- 芯片级封装 CSP(Chip Scale Package)比贴片封装缩小面积 68%
- 其他特性:
 - 全双工 UART(SG 类单片机)
 - 模拟比较器(SG 类单片机)

新型 COP8 单片机命名方法如图 1-5 所示。

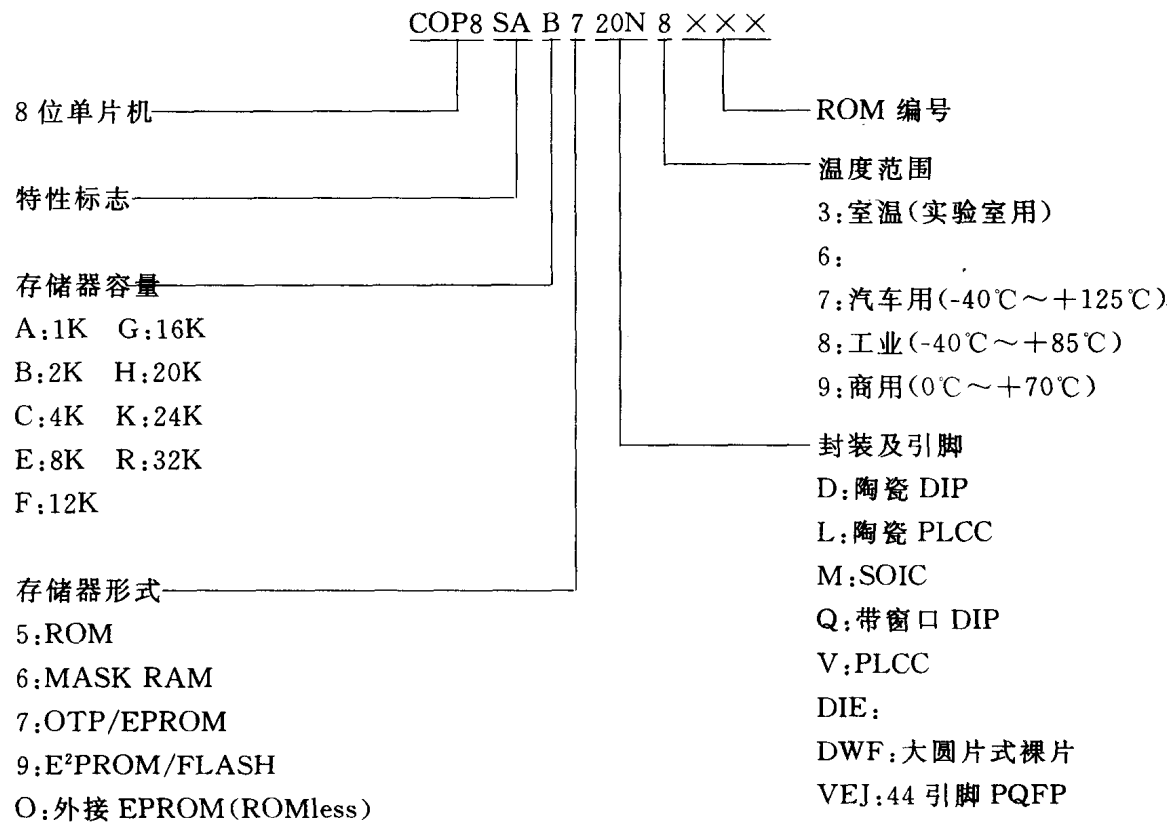


图 1-5 新型 COP8 单片机命名法

新型 COP8 单片机中,SA 类(也称为 Apollo 单片机),和 SG 类单片机也有商用级、工业级、军用级和 OTP 型等四种形式。另外,ACC 类单片机是特性型 COP8 单片机 884EK 的简化,只不过采用了新型 COP8 单片机的命名方式。表 1-4 是新型 COP8 单片机产品表。

表 1-4 新型 COP8 单片机产品表

	商用级 0℃~+70°	工业级 -40℃~+85℃	军用级 -55℃~+125℃	OTP 型
SA 类 Apollo 单片机	8SAA5 8SAB5 8SAC5			8SAA7 8SAB7 8SAC7
SG 类		8SG×5		8SGE7 8SGE7 8SGR7
ACC 类	8ACC5	8ACC5	8ACC5	8ACC7

1.1.4 闪烁存储器型 COP8 单片机简介

随着集成电路工艺的发展,NS 公司推出了闪烁(Flash)存储器型单片机(如图 1-6 所示)。

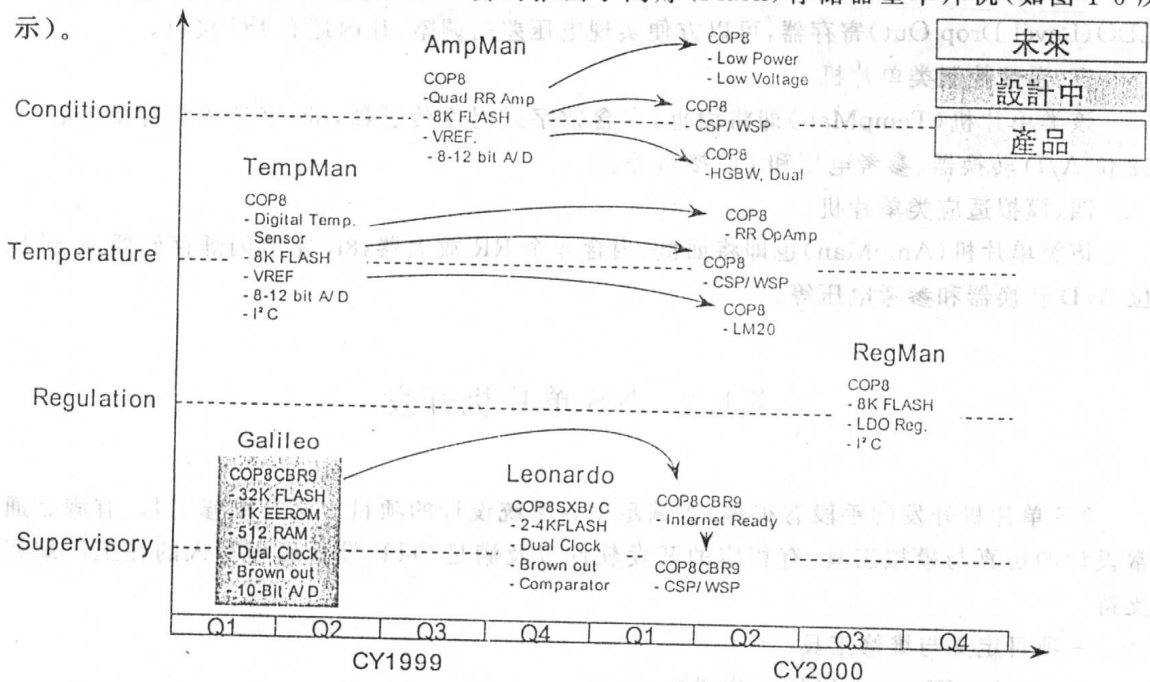


图 1-6 NS 公司闪烁存储器型单片机的发展

如图 1-6 所示,闪烁存储器型 COP8 单片机可以分为 4 类:通用高性能(Supervisory)、电压调整(Regulation)、温度控制(Temperature)和模拟适应(Conditioning)类。

一、通用高性能类单片机

通用高性能类单片机目前有 COP8CBR9(又称为 Galileo)和 COP8S×B/C(又称为 Leonardo)。

其中,COP8CBR9 单片机主要性能如下:

- COP888 核
- 倍速(20MHz)
- 32K 字节闪烁存储器
- 512 字节 RAM
- 1K 字节 EEPROM
- 双时钟

- 三个 16 位定时/计数器
- WATCHDOG 定时器
- 10 位 A/D 转换器
- USART 串行接口
- 时钟监视器
- 多根双向 I/O 引脚
- 44 引脚 PLCC/PQFP 封装和 68 引脚 PLCC 封装

而 COP8S×B/C 单片机性能和 COP8CBR9 单片机类似,但闪烁存储器只有 2K 至 4K 字节,无 10 位 A/D 转换;不过有模拟比较器。

二、电压调整类单片机

该类单片机(RegMan)正在设计之中,片内闪烁存储器为 8K 字节;内含电压差输出 LDO(Level Drop Out)寄存器,可以方便实现电压差的调整;片内还有 I²C 接口。

三、温度控制类单片机

该类单片机(TempMan)即将面世,内含数字式温度传感器,8K 字节闪烁存储器、8 至 12 位 A/D 转换器、参考电压和 I²C 接口等。

四、模拟适应类单片机

该类单片机(AmpMan)也即将面世,内含 4 个 RR 放大器,8K 字节闪烁存储器,8 至 12 位 A/D 转换器和参考电压等。

§ 1.2 NS 单片机开发

NS 单片机开发的手段有很多,有满足项目系统设计的项目定义与维修工具、有满足通常设计的仿真与模拟工具、有相应的开发软件以及满足 OTP 批量编程写入的在工厂编程支持。

* 项目定义与维修工具

- Drive Way-cop(Aisys 公司)

—— Windows 开发平台

—— 自动生成文本化的 C 或汇编源编码模块(包括用户的 I/O 驱动、中断处理及应用需求)

—— 集成化编辑能插入应用特殊编码

—— 配备相应仿真器及开发语言

- WCOP8 IDE(K&K Development 公司)

—— Windows 开发平台

—— 集成化编辑

—— “Project Wizard”提供了开发工具与项目文件的连接

—— Build 及 Make 功能可通过点与单击方法自动调用

—— 配备相应仿真工具及开发语言

* 仿真和模拟工具:

- iceMASTER (Metalink 公司)
 - 全性能在线仿真器
- COP8SA-DM (Metalink 公司)
 - 实惠型在线仿真器
- Insight PE-COP8 (南京万利电子有限公司)
 - 高性能价格比在线仿真器
- COP8SA-EPU (Metalink 公司)
 - 性能评价板 (单板机)
- 指令级模拟器 (Metalink 公司)
 - 非硬件化的模拟开发软件
- * OTP/EPROM 编程器
 - DATA I/O (DATA I/O 公司)
 - ALL07/11 (台湾 HI-LO 公司)
 - SMD (Metalink 公司)
 - BP Microsystem、Ice Technology、Needhams、SMS、Stag、System General 及 Xeltek 公司也提供相应编程器

1.2.1 iceMASTER 在线仿真器

iceMASTER IM-COP8/400 在线仿真器是开发 COP8 单片机的高性能全特性在线仿真器。通过 RS-232C, 在 PC 机支持下进行开发, 如图 1-7 所示。

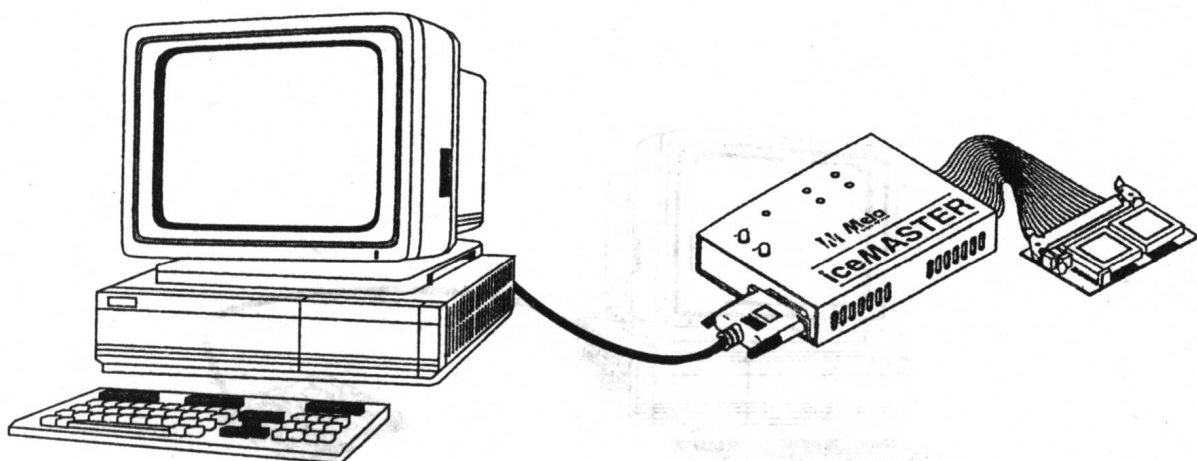


图 1-7 COP8 iceMASTER 开发环境

其特性如下：

- 实时在线仿真，支持 DC——10MHz 时钟运行单片机
- 仿真存储器为 32K 字节，即可映像覆盖 COP8 单片机系列中的片内 ROM/EPROM、RAM 与 I/O
- 跟踪存储器为 4K 帧，每帧包括地址、标号、指令及 8 个外信号信息，能直接显示 C 语言、汇编语言源程序或混合型源程序
- 硬断点为 64K 个
- 符号化调试，可支持汇编及 C 语言编程
- 实时的性能简要分析，选择存储定义
- 可视式显示窗口，能自动更新
- 能支持单步操作，在源程序窗口中可动态显示指令、存储器及 I/O
- 显示屏幕包括：寄存器、片内 RAM 数据、源程序、状态和堆栈等 5 个窗口
- 断点/跟踪触发
- 汇编/反汇编
- 下载文件，能生成 COFF 公共目标文件格式的文件，能生成 COD 二进制文件及 SHF 符号化十六进制文件
- 可支持神经元模糊控制软件 NeuFuz 运行

1.2.2 COP8-DM 在线仿真器

COP8-DM(Debug Module)在线仿真器是实惠型的单片机在线仿真器，既有在线仿真功能，又能进行 OTP/EPROM 编程。通过 RS-232C，在 PC 机支持下进行开发，如图 1-8 所示。

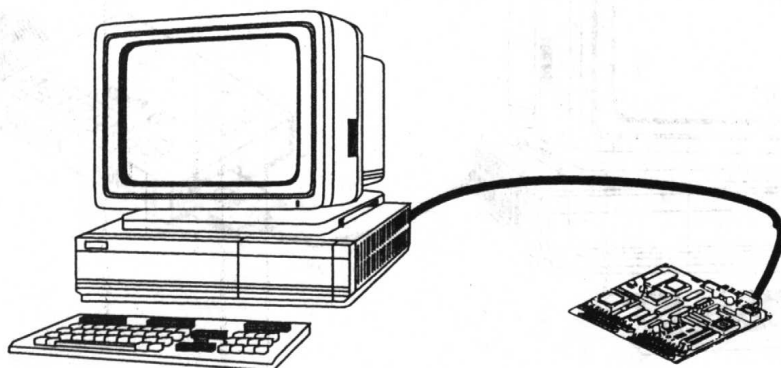


图 1-8 COP8-DM 开发环境

其特性如下：

- 实时在线仿真,支持 DC——10MHz 时钟运行单片机
- 仿真存储器为 32K 字节,即可覆盖 COP8 单片机系列中的片内 ROM/EPROM、RAM 及 I/O
- 跟踪存储器为 100 帧
- 软断点,通过 INTR 指令来实现
- 符号化调试,可支持汇编及 C 语言编程
- 单步操作,能动态显示每条指令、存储器/寄存器的变化
- 汇编/反汇编
- 可支持 SOIC、DIP 与 PLCC 封装的 OTP/EPROM 单片机的编程
- 下载文件,能生成 COFF 公共目标文件格式文件(汇编)及 COD 二进制文件(C 语言)

1.2.3 EPU-COP8 性能评价板

EPU-COP8 性能评价板(单片机)是一个很好的学习工具,并具有 OTP/EPROM 单片机编程功能,但也只能通过 RS-232C,在 PC 机支持下运行。图 1-9 是其开发环境示意图。

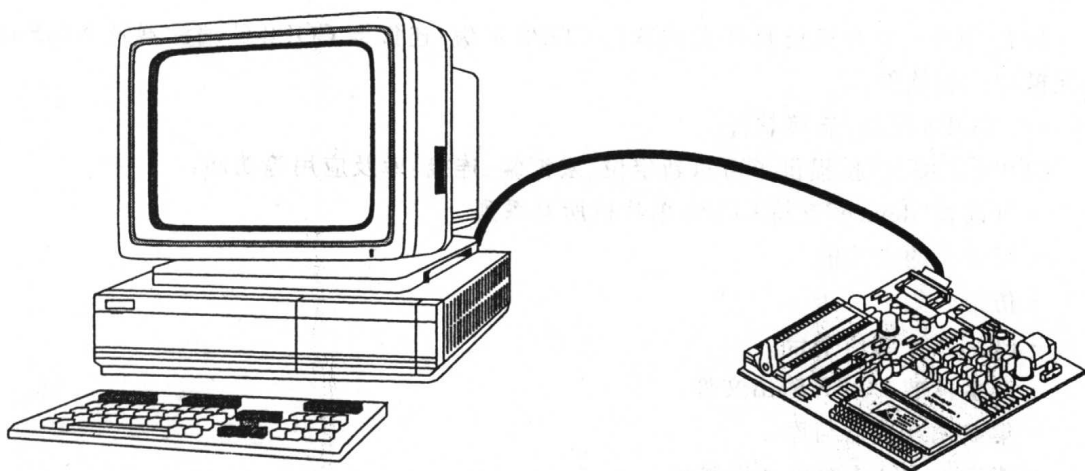


图 1-9 EPU-COP8 开发环境

其特性如下:

- 非实时在线的仿真,仿真控制程序常驻在 PC 机,通过 RS-232C 下载来执行,执行为 20kHz
- 仿真存储器为 32K 字节,即可覆盖 COP8 单片机系列中的片内 ROM/EPROM、RAM 及 I/O
- 单片机中的 WATCHDOG 及定时器不能与指令同步操作
- 跟踪存储器为 100 帧

- 软件断点为 8 个,通过 INTR 指令来实现
- 单片操作,能动态显示每条指令,存储器/寄存器的变化
- 只能支持 40DIP 插座为相应 OTP/EPROM 单片机的编程
- 汇编/反汇编
- 下载文件,能生成 COFF 公共目标文件格式的文件(汇编)及 COD 二进制文件(C 语言)

1.2.4 Insight PE-COP8 在线仿真器

该仿真器由国内研制,有较高的性能价格比,其性能如下:

- 采用“Bondout”的技术,从而保证真正的实时仿真
- 采用先进的 Client/Server 的结构
- 仿真存储器:32K 字节
- 程序地址断点:32K
- 跟踪存储器:8K 帧
- 具有外信号断点
- 支持 C 及 ASM 语言实时源程序调试
- 可仿真 Applo 系列 COP8SA $\times$ /SG $\times$

1.2.5 仿真开发软件

配合 COP8 单片机仿真开发的软件 COP8 汇编/连接器 COP8 C 编译器及 NeuFuz 神经元模糊控制软件。

一、COP8 汇编/连接软件

COP8 汇编/连接提供了可重新定位、宏汇编、连接、库及应用等功能:

- 可设置“device”支持 COP8 单片机所有系列
- 可嵌套的宏功能
- 仿指令集
- PC/DOS 工作平台
- 产生标准 COFF 输出文件
- 集成化连接器与库
- 能产生 ROM 编码文件输出
- DUMPCOFF 应用软件

汇编器 SDK 为 COP8-DEV-IBMA 由 NS 公司提供。

二、COP8 C 编译器

COP8 C 是 ANSI C 的编译器,其特性如下:

- ANSI C 标准,有些限制及扩充
- Byte Craft 连接器,连接目标 C 或汇编源程序
- BITS 数据类型扩充、寄存器说明符参数可直接进行位级定义
- 支持中断子程序
- 专家系统,规则基产生及优化

- 产生程序存储编码
- 全局优化的连接编码
- 符号化调试(配合仿真器)

第二章 新型 COP8 单片机系统结构

COP8 单片机系列是美国国家半导体公司(NS)面向控制的高性能 8 位单片机。本章以新近推出的新型 COP8 单片机作介绍,新型 COP8 单片机是高性能低价格 OTP 型单片机,它的系统结构与典型的 COP8 单片机的系统结构基本相同。图 2-1 是 COP8SA $\times$ 7(又称 Apollo 单片机)的系统结构图。

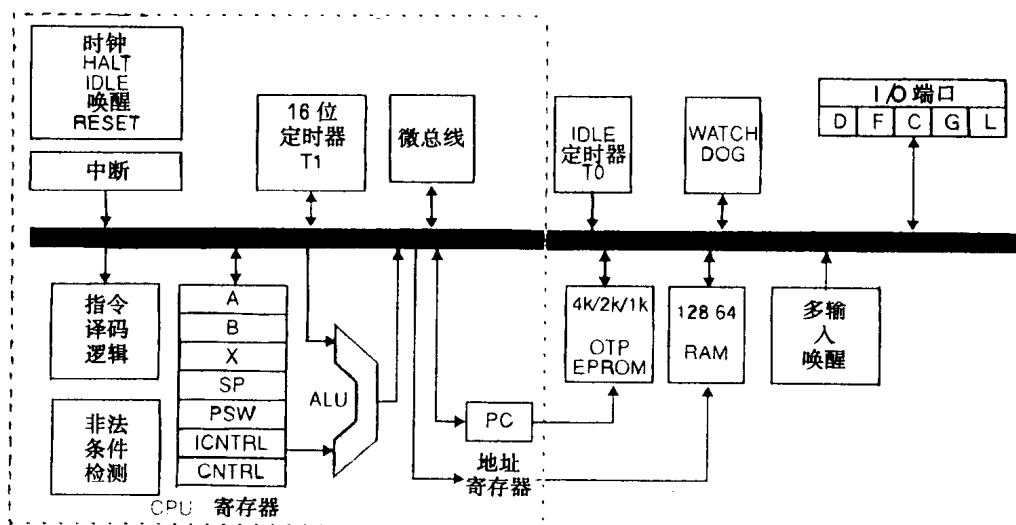


图 2-1 COP7SA $\times$ 7 单片机系统结构图

新型 COP8 单片机是基于改进型的 Harvard 结构。它允许从程序存储器直接访问数据表,这对现代单片机应用十分重要,因为程序存储器通常是 ROM 或 EPROM,而 RAM 作为数据存储器;为了使掉电后数据表不丢失,数据表就要放在 ROM 或 EPROM 之中,此功能就可很好地解决这个问题。另外,在改进型的 Harvard 结构中,指令的访问和数据传送采用了可重叠操作的二级流水线,这样,下一条指令从程序存储器取指时,现行指令正在进行数据传送操作。上述的功能在常规的 Von Neumann 的单地址总线结构是很难实现的。

§ 2.1 新型 COP8 单片机 CPU 结构

新型 COP8 单片机的 CPU 结构图如图 2-2 所示。

新型 COP8 单片机的 CPU 中,除了算术逻辑单元 ALU 及控制逻辑单元外,它还有如下寄存器组成:

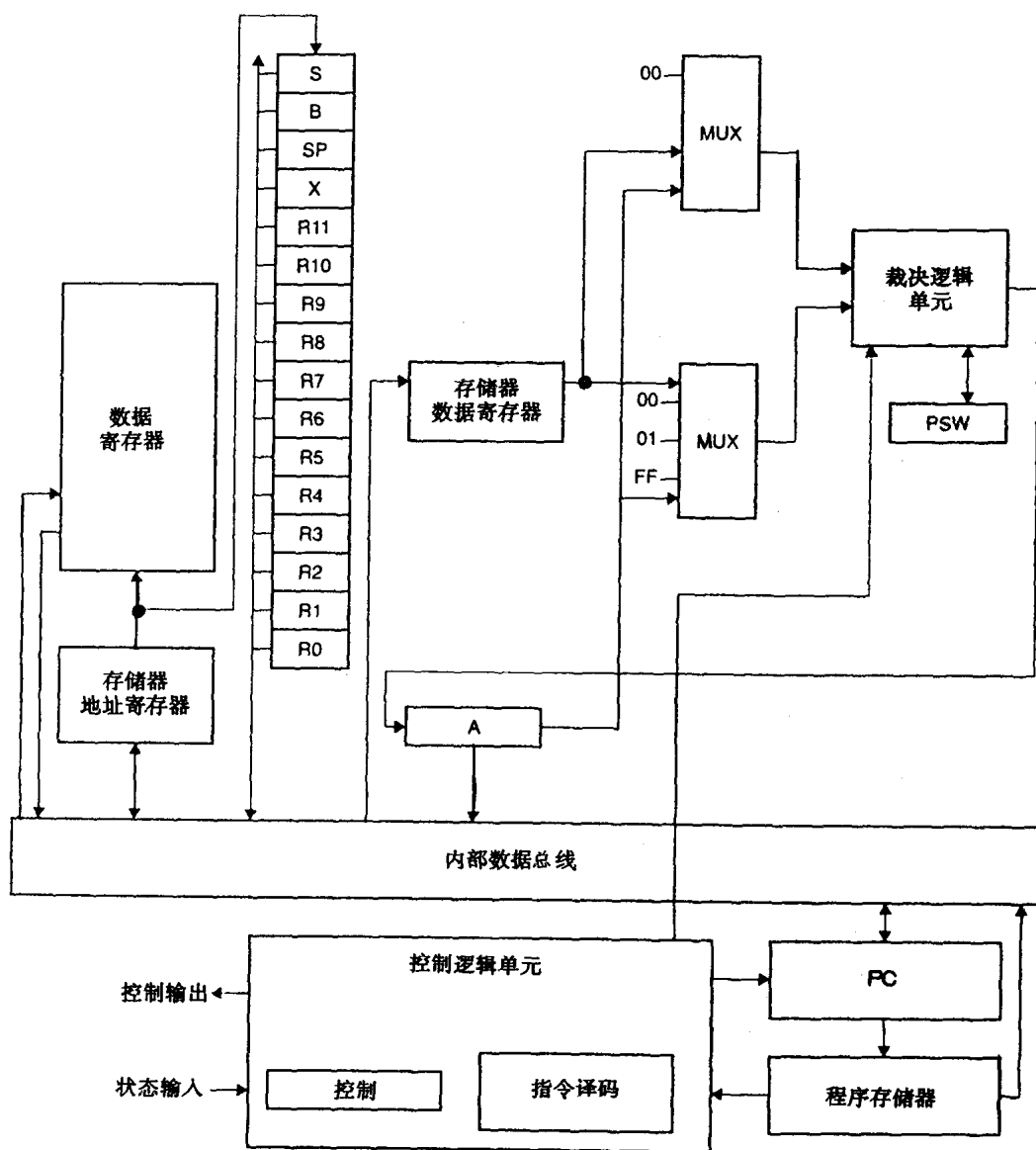


图 2-2 新型 COP8 单片机 CPU 结构图

1. 累加器 A

8 位累加器 A, 常用于算术、逻辑运算和数据交换等操作。

2. 程序计数器 PC

15 位程序计数器 PC 由高 7 位 PU 程序计数器及低 8 位 PL 程序计数器组成。

3. 处理器状态字(PSW)寄存器

PSW(××FFH)

7	6	5	4	3	2	1	0
HC	C	T1PND	T1ENA	EXPND	BUSY	EXEN	GIE

位 0: GIE(Global Interrupt Enable), 单片机中断允许位。

位 1:EXEN(Enable External Interrupt),外部中断允许位。

位 2:BUSY,MICROWIRE/PLUS 微总线移位忙标志位。

位 3:EXPND(EXternal interrupt PeNDing),外部中断挂起。

位 4:T1ENA,T1 定时器溢出或输入捕获时的中断允许。

位 5:T1PNDA,T1 定时器中断挂起标志位(方式 1 的自动装入 RA、方式 2 的 T1 溢出及方式 3 的 T1A 输入捕获)。

位 6:C(Carry),进位标志。

位 7:HC(Half Carry),半进位标志。

所有影响进位标志的指令也都要影响半进位标志。SC(进位位置位)指令和 RIC(进位位复位)指令将分别把进位位置位和清除;除了 SC 和 R/C 指令外,ADC,SUBC,RRC 和 RLC 指令也会影响进位和半进位标志。

4. 堆栈指针 SP

8 位堆栈指针 SP,其堆栈区域设在片内 RAM。初始化时,SP 指向地址 02FH(单片机片内 RAM 为 64 字节)或地址 06FH(单片机片内 RAM 为 128 字节)。

5. RAM 地址指针 B

8 位 RAM 地址指针 B,指向片内 RAM 地址,它具有预先加 1/减 1 功能。

6. 备用 RAM 地址指针 X

8 位备用 RAM 地址指针 X,具有 RAM 地址指针 B 同样的功能,即指向片内 RAM 地址,具有预先加 1/减 1 功能。

B 和 X 都具有变址寄存器的作用。

7. 数据段寄存器 S

当片内 RAM 大于 128 字节,可以通过 8 位数据段寄存器 S 来扩展寻址。

8. 通用数据寄存器 R0~R11

12 个 8 位通用数据寄存器 R0~R11,位于片内 RAM 中,可作通用数据寄存器用。

9. 中断控制寄存器 ICNTRL

ICNTRL(××E8H)

7	6	5	4	3	2	1	0
FLAG1	FLAG2 (LPEN)	PPND (T0PND)	PPEN (T0EN)	MWPND	MWEN	T1PNDB	T1ENB

位 0:T1ENB,定时器 T1B 中断允许。

位 1:T1PNDB,定时器 T1B 中断挂起。

位 2:MWEN,MICROWIRE/PLUS 微总线中断允许。

位 3:MWPND,MICROWIRE/PLUS 微总线中断挂起。

位 4:PPEN,通用外围中断允许(常为 T0EN:定时器 T0(IDLE 定时器)中断允许)。

位 5:PPND,通用外围中断挂起(常为 T0PND:定时器 T0(IDLE 定时器)中断挂起)。

位 6:FLAG2,通用标志(常为 LPEN:I/O 端口 L 中断允许)。

位 7:FLAG1,通用标志(保留)。

10. 控制寄存器 CNTRL

8 位控制寄存器 CNTRL 用来控制 MICROWIRE/PLUS 微总线、外中断和定时器 T1。
CNTRL(××EEH)

7	6	5	4	3	2	1	0
T1C3	T1C2	T1C1	T1C0	MSEL	IEDG	SL1	SL0

位 0, 位 1: SL0&SL1, MICROWIRE/PLUS 微总线时钟分频选择(00=2, 01=4, 1×=8)。

位 2: IEDG, 外中断边沿极性选择(0=上升沿, 1=下降沿)。

位 3: MSEL, 选择 G5 和 G4 为 MICROWIRE/PLUS 微总线的 SK 和 S0 引脚。

位 4: T1C0, 定时器方式 1 和 2 时的定时器 T1 启动/停止控制位; 定时器方式 3 时的 T1 溢出中断标志。

位 5: T1C1, 定时器 T1 方式控制位。

位 6: T1C2, 定时器 T1 方式控制位。

位 7: T1C3, 定时器 T1 方式控制位。

11. EPROM 设置寄存器 ECON

8 位 EPROM 设置寄存器 ECON 可供用户用来配置用户选择时钟、保密、上电复位、WATCHDOG 定时器和 HALT 方式等。

ECON

7	6	5	4	3	2	1	0
X	POR	SECURITY	CKI2	CKI1	WATCDOG	Reserved	HALT

位 0: HALT, 0=HALT 方式允许。

1=HALT 方式不允许。

位 1: 保留。

位 2: WATCDOG, 0=WATCHDOG 定时器允许, G1 引脚作为 WATCHDOG 定时器输出 WDOUT 引脚。1=WATCHDOG 定时器不允许, G1 引脚仍作通用 I/O。

位 3, 位 4: CKI1, CKI2

0, 0: 外 CKI 选择, G7 作为 HALT 再启动和/或通用输入, CKI 为时钟输入。

0, 1: 不用晶振的片内晶振偏置电阻, G7(CKO)为时钟发生器输出至晶振/谐振器。

1, 0: R/C 振荡器选择, G7 作为 HALT 再启动和/或通用输入, CKI 时钟输入, 内部 R/C 器件提供最大 R/C 频率。

1, 1: 采用晶振的片内晶振偏置电阻, G7(CKO)为时钟发生器输出至晶振/谐振器。

位 5: SECURITY, 0=保密不允许, EPROM 允许读和写。

1=保密允许, EPROM 不允许读和写。

位 6: POR, 0=上电复位不允许。

1=上电复位允许。

位 7:X,此为工厂测试。

出厂时,ECON 的初值为 00H(带窗口芯片)或 80H(OTP 型芯片)。

§ 2.2 新型 COP8 单片机存储器

新型 COP8 单片机除了常规 COP8 单片机所具有的数据存储器及程序存储器外,还有 8 字节用户存储区域。新型 COP8 单片机的程序存储器常采用高效的 OTP,并具有很强的保密功能。

2.2.1 新型 COP8 单片机数据存储器

新型 COP8 单片机数据存储器(片内 RAM)一般为 64/128/256 字节。它可以分为通用 RAM、R0~R11 数据寄存器、I/O 端口及控制寄存器,图 2-3 是片内 RAM 存储器映像图,表 2-1 是映像表。

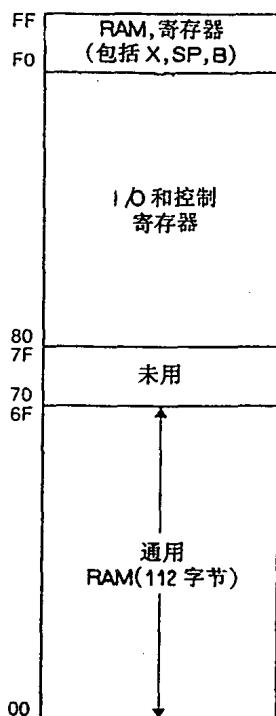


图 2-3 片内 RAM 存储器映像图

00~6F(或 00~2F)是 112 字节(或 64 字节)的片内通用 RAM。70~7F(或 30~7F)是未使用的 RAM 地址空间。

F0~FF 是寄存器区域。F0~FB 是 12 个 8 位通用寄存器 R0~R11,FC~FF 是备用 RAM 指针 X、堆栈指针 SP、RAM 地址指针 B 和数据段寄存器 S。

其余的区域为 I/O 端口及各控制寄存器。

表 2-1 片内 RAM 映像表

地 址	说 明
* 00~6F	片内 RAM(112 字节)
* 70~BF	未使用的 RAM 地址空间
C0	定时器 T2 低位字节
C1	定时器 T2 高位字节
C2	定时器 T2 自动重装寄存器 T2RA 低位字节
C3	定时器 T2 自动重装寄存器 T2RA 高位字节
C4	定时器 T2 自动重装寄存器 T2RB 低位字节
C5	定时器 T2 自动重装寄存器 T2RB 高位字节
C6	定时器 T2 控制寄存器
C7	WATCHDOG 服务寄存器 WDSVR
C8	MIWU 边沿选择寄存器(寄存器:WKEDG)
C9	MIWU 允许寄存器(寄存器:WKEN)
CA	MIWU 挂起寄存器(寄存器:WKPND)
CB	A/D 转换控制寄存器(寄存器:ENAD)
CC	A/D 转换结果寄存器(寄存器:ADRSLT)
CD~CF	保留
D0	口 L 数据寄存器
D1	口 L 配置寄存器
D2	口 L 输入引脚(只读)
D3	为口 L 保留
D4	口 G 数据寄存器
D5	口 G 配置寄存器
D6	口 G 输入引脚(只读)
D7	口 I 输入引脚(只读)
D8	口 C 数据寄存器
D9	口 C 配置寄存器
DA	口 C 输入引脚(只读)
DB	为口 C 保留
DC	口 D 数据寄存器
DD~DF	保留
E0~E5	保留
E6	定时器 T1 自动重装寄存器 T1RB 低位字节
E7	定时器 T1 自动重装寄存器 T1RB 高位字节
E8	ICNTRL 寄存器
E9	微总线移位寄存器
EA	定时器 T1 低位字节
EB	定时器 T1 高位字节
EC	定时器 T1 重装寄存器 T1RA 低位字节
ED	定时器 T1 重装寄存器 T1RA 高位字节
EE	CNTRL 控制寄存器
EF	PSW 寄存器
F0~FB	片内 RAM 映射为寄存器
FC	X 寄存器
FD	SP 寄存器
FE	B 寄存器
FF	保留

* 注:64 字节片内 RAM:00~2F 片内 RAM(48 字节) 30~7F 未使用

单片机的指令系统允许对存储器的任何位进行置位、复位和测试。所有 I/O 端口和寄

存器(除累加器 A 及程序计数器 PC 未映像 in 片内 RAM)也可直接进行置位、复位和测试。
若片内通用 RAM 要大于 128 字节,可以通过数据段寄存器 S 来扩展寻址,如图 2-4 所示。

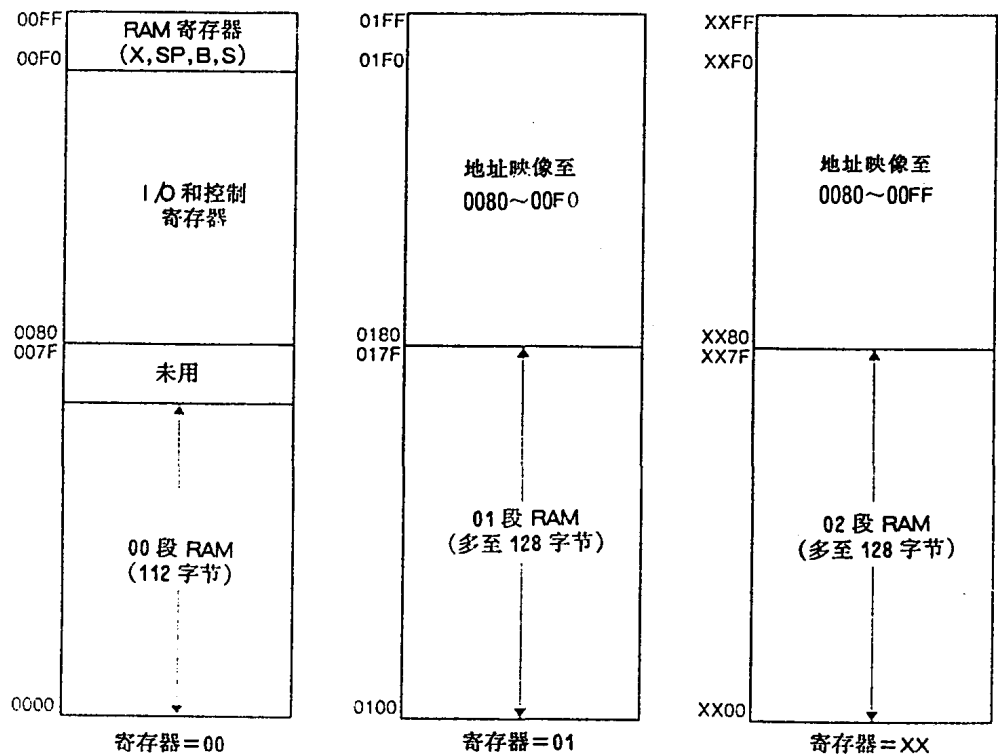


图 2-4 数据段扩展寻址

2.2.2 新型 COP8 单片机程序存储器

COP8 单片机的程序计数器 PC 为 15 位,程序存储器(片内 ROM)可达 32K 字节。新型 COP8 单片机的 SA 类的片内 ROM 可为 4K 字节,SG 类的片内 ROM 可达 32K 字节。
片内 ROM 存放指令或常数(如 LAID 指令数据表、JID 间接转移指令的转移向量和 VIS 向量中断选择指令的中断向量)。

程序存储器是以 256 字节为一页,其中 0 页的 00 单元是 RESET 入口地址,FF 单元是中断入口地址;16 个中断向量 0yE0~0yFF 单元由 VIS 指令来选择页面 y。

新型 COP8 单片机的程序存储器多为一次编程的 EPROM 型的 OTP(SA 类单片机只有 OTP 型单片机)。OTP 型单片机的片内 EPROM 具有安全保密功能,一旦 EPROM 设置寄存器 ECON 的 SECURITY 位(ECON5 位)为“1”时,则片内 EPROM 的内容不允许读出(即读出为全 FF),也不允许写入;同时,此时 ECON 寄存器也不允许写入(而 ECON 寄存器的读出是不受影响的)。因此,建议 ECON 寄存器中的各位可以先设置,而 SECURITY (ECON5)位可以最后设置。

2.2.3 新型 COP8 单片机用户存储区域

新型 COP8 单片机都有 8 个字节用户存储区域,用来存放序号、数据编码、版本号和版权信息等,该 8 个字节存储区域虽也是 EPROM,但是,却位于正常 EPROM 寻址范围之外,无法通过执行软件来存取。

8 字节用户信息和 ECON 寄存器信息可以通过 COP 汇编来定义,作为特殊的 ROM 段类型 CONF 及 ECON,例:

语法:

```
[label:] • sect econ,conf
        • db value ;1 字节组态信息
        • db
        • endsect <user information>;8 字节用户信息
```

假如,系统配置为:允许上电复位、不安全加密、晶振不采用片内偏置、允许 WATCHDOG 和 HALT 工作模式,那末,ECON 寄存器的数值设置为 55。8 字节用户区域存放 my V1.00 信息,则:

```
        • chip8SGR
        • sect econ,conf
        • AB0X55;上电复位、外晶振偏置、WATCHDOG 及 HALT
        • db'my V1.00;8 字节用户信息
        .....
        • end start
```

上述的过程,可以通过 COP8SGR728M7 编程器来设置。

8 字节用户区域是不加密保护的,可以任意读取和写入的。

§ 2.3 新型 COP8 单片机中断及复位系统

2.3.1 新型 COP8 单片机中断系统

一、中断系统结构

新型 COP8 单片机可以处理 16 个中断源(如图 2-5 所示)。除了软件陷阱和 NMI 是非屏蔽中断,其余的为屏蔽中断。屏蔽中断都受系统总中断许可 GIE(Global Interrupt Enable)位控制,GIE 位于 PSW 寄存器的第 0 位。可以通过置位/复位指令,使 GIE 置位/复位(许可/禁止)。图中的中断许可位则分别在 PSW 寄存器(PSW.1(EXEN 外部中断)、PSW.4(T1ENA 的 T1 定时器中断))和中断控制寄存器 ICNTRL(ICNTRL.0(T1ENB 的 T1B 中断)、ICNTRL.2(MWEN 的 MIRCOWIRE/PLUS 微总线中断)、ICNTRL.4(PEN 通用外围中断))中。

COP8 单片机的向量中断及优先级裁决处理是很独特的,COP8 单片机片内无中断优先级裁决逻辑,而是通过 VIS 指令的执行(如图 2-6 所示)来实行优先级裁决及向量中断的转

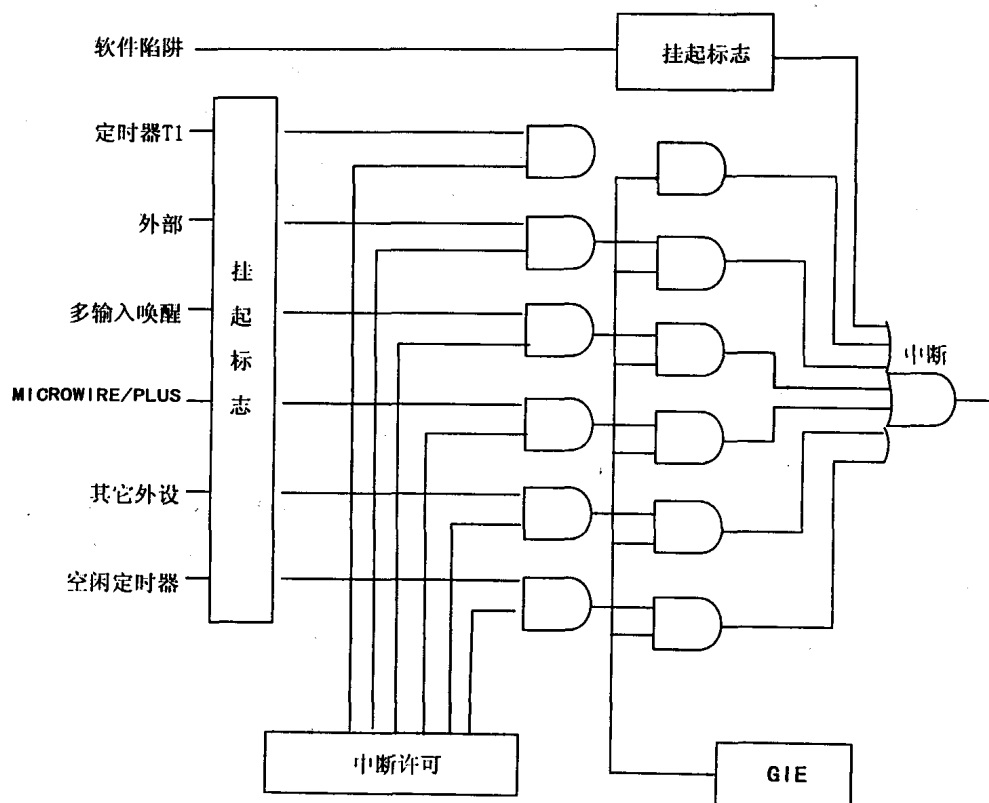


图 2-5 新型 COP8 单片机中断系统结构

移。新型 COP8 单片机向量中断的优先级如表 2-2 所示。

表 2-2 中断向量表

优先级	中断源	描 述	向 量
(1) 高	软件	指令	0yFE~0yFF
(2)	保留	留用	0yFC~0yFD
(3)	外部	G0	0yFA~0yFB
(4)	定时器 T0	下道	0yF8~0yF9
(5)	定时器 T1	T1A/下溢	0yF6~0yF7
(6)	定时器 T1	T1B	0yF4~0yF5
(7)	MICROWIRE/PLUS	忙	0yF2~0yF3
(8)	保留	留用	0yF0~0yF1
(9)	保留	留用	0yEE~0yEF
(10)	保留	留用	0yEC~0yED
(11)	保留	留用	0yEA~0yEB
(12)	保留	留用	0yE8~0yE9
(13)	保留	留用	0yE6~0yE7
(14)	保留	留用	0yE4~0yE5
(15)	端口 L 唤醒	端口 L 边沿	0yE2~0yE3
(16)低	缺省	VIS 指令执行(无其他中断)	0yE0~0yE1

二、中断处理过程

VIS 指令首先查询是否有软件陷阱(最高级中断)。然后,依次查外部中断、定时器、

MICROWIRE/PLUE微总线及端口 L 等中断;若有相应的中断挂起,则转移到相应的向量中断入口。

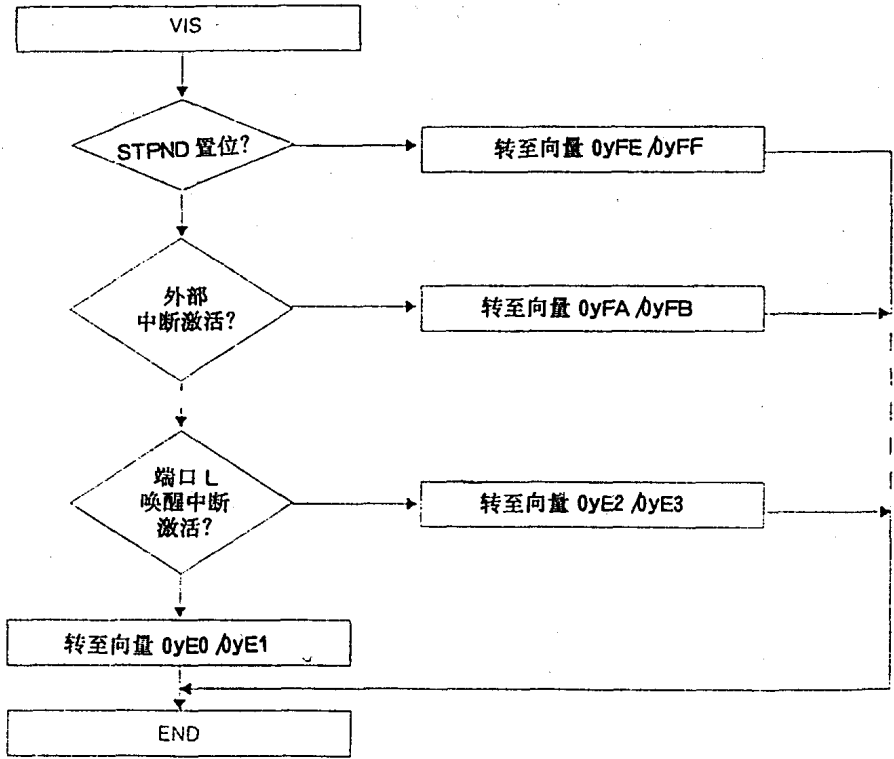


图 2-6 VIS 指令执行流程

整个中断的流程如图 2-7 所示。程序存储器的 0 页 FF 单元为中断入口,内存放一条单

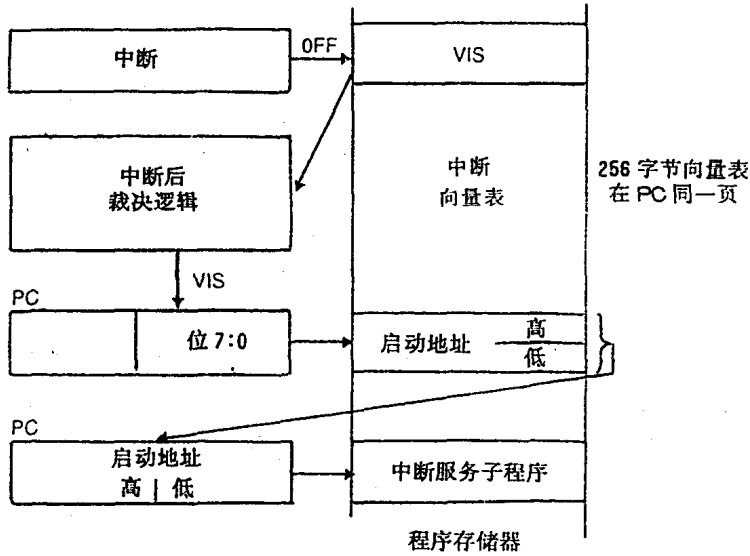


图 2-7 中断处理流程

字节的 VIS 指令码,中断向量表位于 1 页的 01E0~01FF 单元(VIS 指令分配在 0y00~0yDF 及上一页的 FF 单元,那末,中断向量表位于同一页的 0yE0~0yFF 单元)。如图所示,一旦中断产生,转向 0FF 单元;执行 VIS 指令,通过 VIS 指令执行,裁决出相应的中断源,给出向量地址,转向相应中断向量表;在相应的中断向量中找出相应中断服务入口地址;间接转向相应的中断服务入口,执行相应的中断处理。

下面是外部中断编程的举例。把系统总中断许可 GIE 置位,把外部中断许可 EXEN 也置位,设置好外部中断触发边沿的条件。在 0FF 单元设置了 VIS 指令码,在中断向量表的外部中断 01FA,01FB 中填入相应中断服务地址 SERVICE。在 SERVICE 中真正执行相应外部中断处理,先把外部中断挂起位复位,处理完毕后,RETI 返回,且再把 GIE 置位(进入中断处理后,会自动把 GIE 复位,以禁止其他中断响应)。

外部中断编程举例:

```

        PSW                =00EF
        CNTRL               =00EE
        RBIT                0,PORTGC
        RBIT                0,PORTGD      ;G0 引脚高阻
        SBIT                IEDG,CNTRL    ;外中断为下降沿
        SBIT                GIE,PSW       ;置 GIE 位
        SBIT                EXEN,PSW      ;允许外中断
WAIT:    JP                WAIT
        .
        .
        .
        . =0FF              ;外中断使转至 0FF 地址
        VIS                 ;VIS 指令使转至中断向量表
        .
        .
        .
        . =01FA             ;向量表(256 字节,包括 VIS 指令)
        . ADDRW SERVICE
        .
        .
        .
SERVICE:                                ;中断服务子程序
        RBIT,EXPND,PSW           ;复位外中断挂起位
        .
        .
        .
        RETI                     ;返回,置 GIE 位

```

中断挂起位也分别位于 PSW 寄存器 (PSW. 3 (EXPND 外部中断挂起)、PSW. 5 (T1 定时器中断挂起 T1PND) 和中断控制寄存器 ICNTRL (ICNTRL. 1 (定时器 T1B 中断挂起 T1PNDB)、ICNTRL. 3 (MICROWIRE/PLUE 微总线中断挂起 MWPND)、ICNTRL. 5 (通用外围中断挂起 PPND))。

三、软件陷阱中断

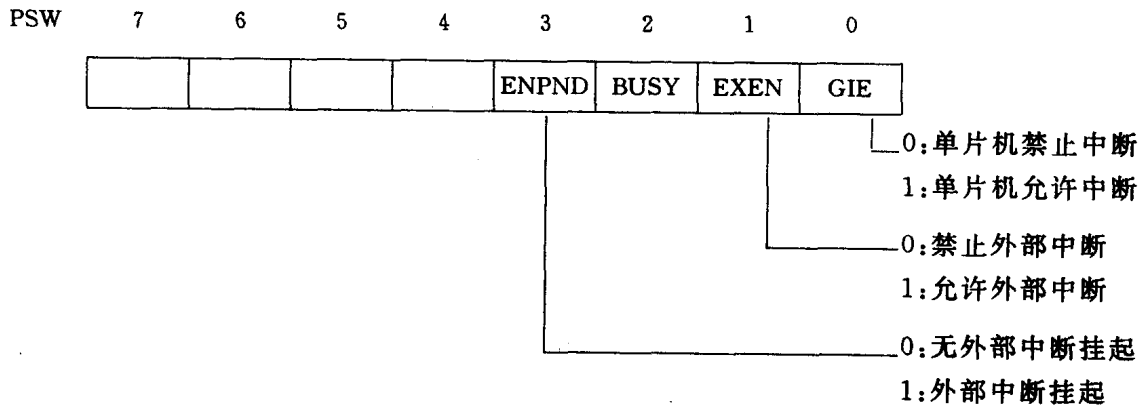
软件陷阱是一种特殊的非屏蔽中断,它出现在当中断指令 INTR (常用于响应中断) 从程序存储器取出并存放在指令寄存器时。它也会出现在程序计数器 PC 所指出地址超出正常的程序存储器空间及堆栈溢出时,以及电源不足或电源闪动时。

软件陷阱中断是优先级最高中断,它不受 GIE 限制,是非屏蔽中断,但却可屏蔽其他中断源。一旦软件陷阱中断挂起 STPND 置位 (软件陷阱中断响应),则禁止其他中断响应。STPND 挂起位可通过 RPND 复位软件陷阱中断挂起指令将其复位。或通过单片机复位来实现。

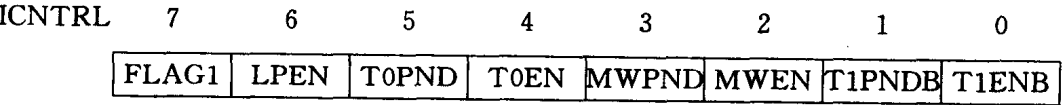
软件陷阱中断表示了一个非常规或未知的出错条件,因此,软件陷阱中断服务程序应重新初始化堆栈指针,执行类似于系统复位的恢复过程 (执行部分复位的功能)。另外,还需执行 RPND 指令把软件陷阱中断挂起 STPND 复位,以允许其他中断的响应。

四、中断有关控制寄存器

1. 处理器状态字 PSW



2. 中断控制寄存器 ICNTRL



详见 § 2.1 节中的对中断控制寄存器 ICNTRL 的说明。

2.3.2 新型 COP8 单片机复位

新型 COP8 单片机复位有外部复位和片内上电复位两种,如图 2-8 所示。

复位后,系统的初始情况如下:

- 程序计数器 PC: 0000
- 堆栈指针 SP: 02FH (64BRAM) / 06FH (128BRAM)
- 处理器状态字 PSW、控制寄存器 CNTRL、中断控制寄存器 ICNTRL 等: 00
- I/O 端口 L, C, G, F: 三态

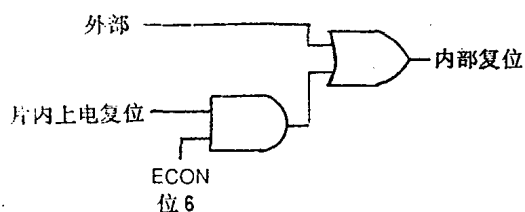


图 2-8 新型 COP8 单片机复位

- I/O 端口 D: 高电平
- 其余的寄存器一般为随机状态(除了有些控制寄存器某些位会置位,如 UART 寄存器的 TBMT 置位)

WATCHDOG 定时器与复位有很密切联系。当 WATCHDOG 服务窗口位和时钟监视位置位,那末,单片机可以通过 WATCHDOG 逻辑及时钟监视器退出复位状态。在复位期间,禁止 WATCHDOG 定时器和时钟监视检测器电路,WATCHDOG 服务窗口置最大值(64Kt_c 时钟周期)。

外部复位电路如图 2-9 所示,RC 时间常数需大于 5 倍电源上升时间或大于 15μs。

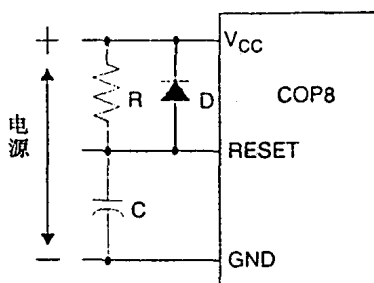


图 2-9 外部复位电路

新型 COP8 单片机都有片内上电复位电路,当 EPROM 设置寄存器 ECON 中的 POR 位(ECON. 6)置位时,允许片内上电复位。采用上电复位,可以把 RESET 引脚接上 Vcc。通过空闲定时器的 4096t_c 延迟,当电源电压到达最小操作电压(10MHz 时 Vcc 为 4.5V、4MHz 时 Vcc 为 2.7V、HALT 模式时 Vcc 为 2.0V),复位操作结束。

2.3.3 新型 COP8 单片机时钟电路

新型 COP8 单片机时钟电路有晶振、RC 和外时钟等几种方式,由 EPROM 设置寄存器 ECON 的第 3,4 位决定,如表 2-3 所示。

表 2-3 时钟电路选择

ECON. 4	ECON. 3	时钟电路选择
0	0	外时钟
1	0	晶振无片内偏置电阻
0	1	RC
1	1	晶振带片内偏置电阻

一、晶振时钟电路

晶振时钟电路有带片内偏置电阻和无片内偏置电阻两种形式,如图 2-10 所示。电阻、电容的配置值及相应时钟频率如表 2-4 所示。

表 2-4 晶振时钟电路配置

R1(kΩ)	R2(MΩ)	C1(pF)	C2(pF)	CKI 频率(MHz)
0	1	30	30	15
0	1	32	32	10
0	1	45	30~36	4
5.6	1	100	100~156	0.455

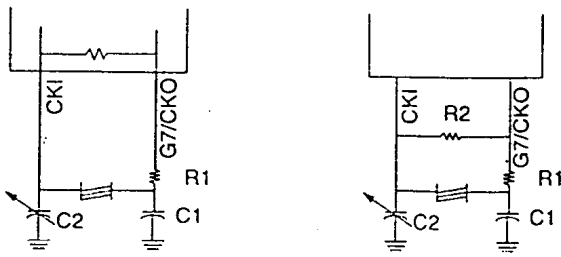


图 2-10 晶振时钟电路

二、外时钟电路

外时钟电路如图 2-11 所示,由 CKI 端引入外时钟,G7/CKO 引脚可作为通用的输入引脚 G7,或作为 HALT 控制。

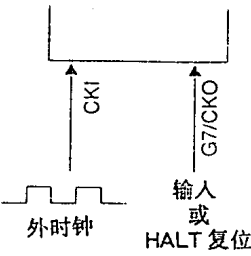


图 2-11 外时钟电路

三、RC 时钟电路

RC 时钟电路 CKI 端可浮空,可采用片内的电阻电容,G7/CKO 引脚可作为通用的输入引脚 G7 或作为 HALT 控制;若时钟频率较低,则可在 CKI 端外接一只电容。电阻电容的取值见表 2-5 所示。

表 2-5 RC 时钟电路配置

外接电容(pF)	时钟频率(MHz)	指令周期(μs)
0	5	2.0
9	4	2.5
52	2	5.0
150	1	10
TBD	32kHz	312.5

§ 2.4 新型 COP8 单片机省电工作方式

新型 COP8 单片机有 HALT 和 IDLE 两种省电工作方式。HALT 工作方式时,单片机停止运行,但是,单片机内寄存器、定时器、I/O 及 RAM 的信息都保持不变,该工作方式的工作电流仅 $4\mu\text{A}/10\mu\text{A}(4\text{MHz}/10\text{MHz})$ 。IDLE 工作方式时,除了片内时钟振荡器及定时器 T0 外,其余的都停止运行,此时,单片机内寄存器、定时器、I/O 及 RAM 也都保持不变,此工作方式的工作电流为 $0.8\text{mA}/1.5\text{mA}(4\text{MHz}/10\text{MHz})$ 。

上述两种省电工作方式,时钟监视器仍可运行。

2.4.1 新型 COP8 单片机 HALT 工作方式

单片机可以通过 G7 位(HALT 标志位)置位,使系统进入 HALT 工作方式。一旦进入 HALT 工作方式,单片机包括时钟、定时器都停止工作,WATCHDOG 逻辑被禁止;但是,时钟监视器仍允许运行,并使 WATCHDOG 的输出引脚 WDOUT 为低电平。

单片机有三种途径退出 HALT 工作方式:

(1) I/O 端口 L 的多路输入唤醒。

(2) 将 G7 引脚上的低电平转为高电平(但不能用于晶振时钟电路,因为 G7/CKO 已成为 CKO 专用输出线)。

(3) 使 RESET 输入低电平,通过系统复位退出 HALT 工作方式。

图 2-12 是 HALT 工作方式的进入和退出示意图。

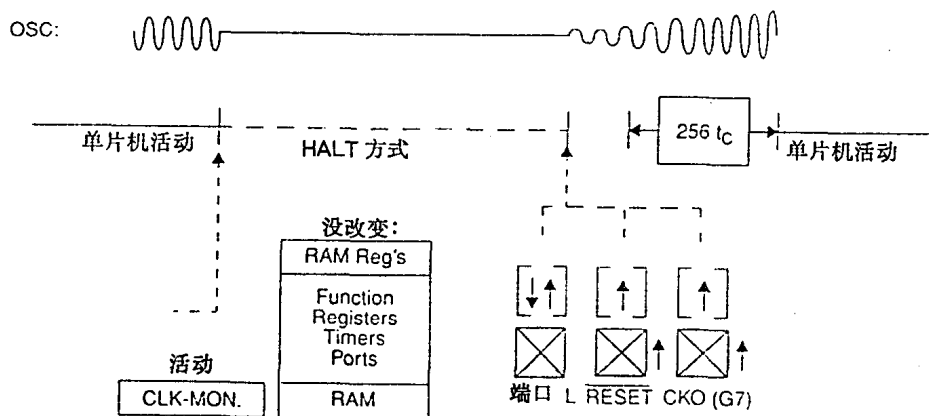


图 2-12 HALT 工作方式示意图

为了确保单片机退出 HALT 工作方式后能正常运行,用空闲定时器加载 FFH,延迟 $256t_c$ 来确保时钟振荡器达到稳定状态,从而使系统进入正常工作方式。

若是 RC 时钟电路,可任意固定延迟。此时,端口 G 的 G7 位映像为 CLKDEY 控制位。把 CLKDEY 位置位,则选用延迟;把 CLKDEY 复位,则不选用延迟。

2.4.2 新型 COP8 单片机 IDLE 工作方式

单片机可以通过 G6 位 (IDLE 标志位) 置位, 使系统进入 IDLE 工作方式。一旦进入 IDLE 工作方式, 单片机除片内的时钟电路, IDLE 定时器和定时器 T0 仍运行外, 其余的都停止工作; WATCHDOG (监视) 逻辑电路与时钟监视器仍处于有效状态。IDLE 工作方式的功耗一般仅为正常工作方式的 30%。

单片机退出 IDLE 工作方式有两种途径:

(1) I/O 端口 L 的多路输入唤醒。

(2) 使 $\overline{\text{RESET}}$ 输入为低电平, 通过系统复位退出 IDLE 工作方式。

图 2-13 是 IDLE 工作方式的进入和退出示意图。

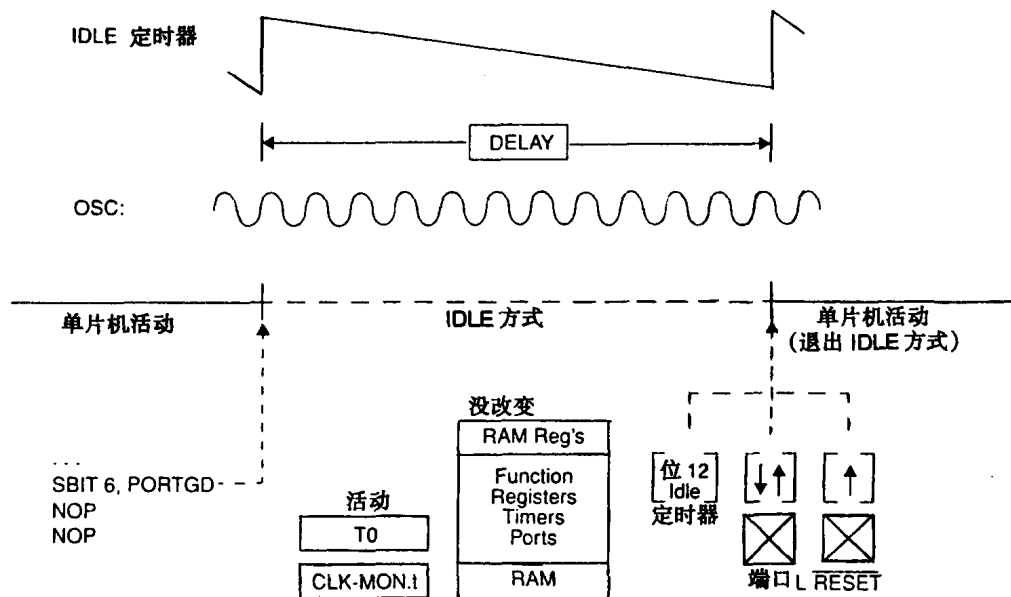


图 2-13 IDLE 工作方式示意图

另外, 通过空闲定时器的第 12 位 (对应 4.096ms, 10MHz, $t_c = 1\mu s$) 触发, 也可使系统从 IDLE 工作方式退出, 进入正常工作方式。空闲定时器第 12 位触发状态锁存在 T0PND 挂起标志上。这样, 用户可通过设置 T0 定时器的中断许可, 来进入 IDLE 工作方式, 此时, 当 T0PND 位置位, 单片机先执行定时器 T0 中断服务程序, 然后, 返回来执行“进入 IDLE 方式”指令之后的指令。另一方面, 用户可通过空闲定时器中断禁止进入 IDLE 工作方式, 此时, 单片机将重新执行紧随“进入 IDLE 方式”指令之后的正常操作指令。

在设置 HALT 工作方式和 IDLE 工作方式指令之后, 建议能随后跟两条 NOP 指令, 这样, 便于 HALT 工作方式和 IDLE 工作方式之后的时钟重新同步。

第三章 新型 COP8 单片机 I/O 功能

新型 COP8 单片机 I/O 功能与 COP8 单片机的一样,具有多功能定时器、WATCHDOG 定时器、MICROWIRE/PLUS 微总线及多输入唤醒的 I/O 端口等。

§ 3.1 新型 COP8 单片机定时器

新型 COP8 单片机中,SA 类单片机具有定时器 T0(空闲定时器)、T1 和 WATCHDOG 定时器,SG 类单片机还多了两个与定时器 T1 功能相同的定时器 T2 与 T3。

3.1.1 新型 COP8 单片机空闲 IDLE 定时器 T0

新型 COP8 单片机定时器 T0 是空闲 IDLE 定时器,它是 16 位计数器,其时钟来自指令周期时钟 t_c (即十分之一的单片机时钟 CKI,若单片机时钟 10MHz,即 $t_c=1\mu s$)。IDLE 定时器是不能读/写的,故该定时器没有映像的 I/O 地址。但是,该定时器状态仍可通过 IDLE 定时器中断或查该中断挂起位来决定。图 3-1 是 IDLE 定时器的系统结构图。

IDLE 定时器有如下功能:

- IDLE 工作方式的退出。
- WATCHDOG 定时器的服务窗口设定。
- 启动/退出 HALT 工作方式的延迟。
- 片内上电复位的时间宽度。

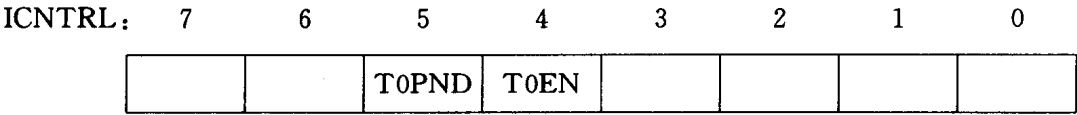
IDLE 工作方式时,其标志位 G6 置起。若 $GIE=1$,T0 定时器(IDLE 定时器)允许中断 $T0EN=1$ (ICNTRL. 4),当定时器计至 $4096t_c$,即产生 T0 中断,把 T0 中断挂起位 $T0PND$ 置 1;从而把 G6(IDLE 标志位)复位,退出 IDLE 工作方式。在有的单片机中,增加了一个 IDLE 定时器控制寄存器 ITMR (IDLE Timer Control Register),并只用了最低 3 位 ITSEL2,ITSEL1,ITSEL0,这样可选择 $4kt_c$, $8kt_c$, $16kt_c$, $32kt_c$ 和 $64kt_c$ 五种中断周期中的一种。

WATCHDOG 定时器服务窗口的下限为 T0 定时器的第 10 位,即 $2kt_c$ (对于 SA 类单片机,下限为 $256t_c$),而其上限可选择第 12 至第 15 位的一种,即 $8kt_c$, $16kt_c$, $32kt_c$ 和 $64kt_c$ 的一种(它们由 WATCHDOG 定时器服务寄存器 WDSRV 的第 6,7 位来选择)。

HALT 工作退出时,为了确保单片机时钟振荡器能达到稳定状态,即需要 $256t_c$ 的延迟,而其延迟也由 IDLE 定时器来提供。

上电复位,从开始至最低工作电压的 V_r 的时间,也可由该定时器来取得。

和 IDLE 定时器有关的控制寄存器是中断控制寄存器 ICNTRL,其控制格式如下:



若有的单片机有 IDLE 定时器控制寄存器 ITMR,其格式如下:

ITMR: 7 6 5 4 3 2 1 0

保留				ITSEL2	ITSEL1	ITSEL0	
				0	0	0	$4096t_c$
				0	0	1	$8192t_c$
				0	1	0	$16384t_c$
				0	1	1	$32768t_c$
				1	×	×	$65536t_c$

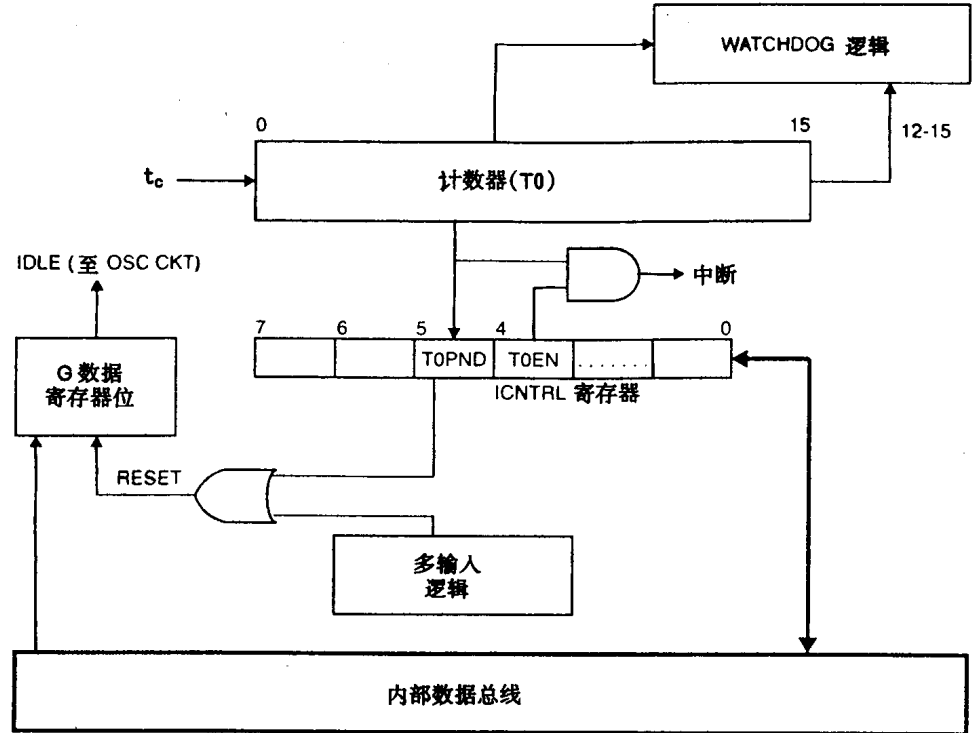


图 3-1 IDLE 定时器系统结构图

3.1.2 新型 COP8 单片机定时器 T1

定时器 T1 模块由一个 16 位定时/计数器和两个 16 位自动装入/捕获寄存器 R1A, R1B 组成,两个 I/O 引脚 T1A, T1B。定时器 T1 有 PWM、外部事件计数及输入捕获等三种工作方式。

一、PWM 工作方式

该工作方式是由两个自动重装寄存器来控制 PWM 的开通与关闭时间。其系统结构图如图 3-2 所示。

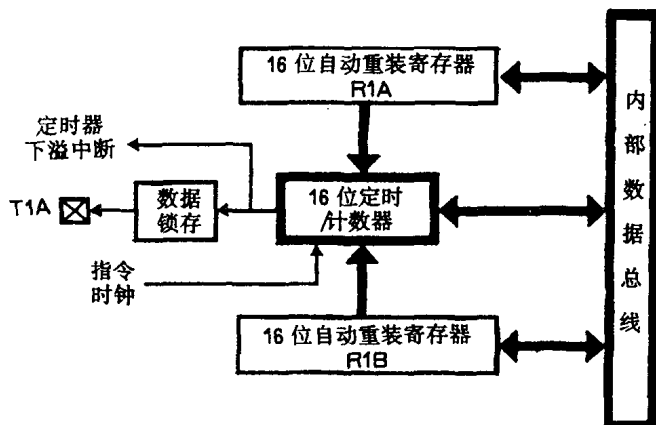


图 3-2 PWM 方式的系统结构图

定时器 T1 以 t_c 时钟频率递减计数，一旦定时器下溢，就用寄存器 R1A 和 R1B 交替地重装定时/计数器。通过置位相应的中断挂起标志 T1PNDA 和 T1PNDB，以中断方式来实现 PWM 的编程。

二、外部事件计数器方式

该工作方式由 T1A 引脚输入外部事件信号，并由 T1 定时器控制器来选择输入信号的跳边。定时/计数器溢出，将置起 T1PNDA 的中断挂起位，从而产生相应中断。图 3-3 是外部事件计数器方式的示意图。

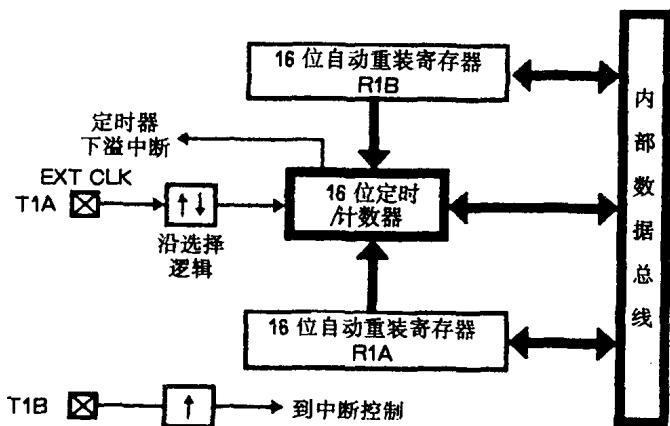


图 3-3 外部事件计数器方式

此工作方式，T1B 引脚可作为某一独立正跳边沿的中断输入端使用。

三、输入捕获方式

该工作方式，16 个定时/计数器以 t_c 时钟频率运行，即为该工作方式的实时时钟；寄存器 R1A 和 R1B 为两个捕获寄存器。由定时器 T1 控制寄存器来决定 T1A 和 T1B 输入事件的逻辑（即 E 跳边或负跳边），一旦捕获到相应的输入事件，把定时/计数器的当时值装入相

应的捕获寄存器 R1A 或 R1B,并置相应的中断挂起位 T1PNDA 或 T1PNDB。但要注意在中断处理时,要区别是输入捕获产生的中断,还是定时/计数器溢出中断。图 3-4 是输入捕获方式示意图。

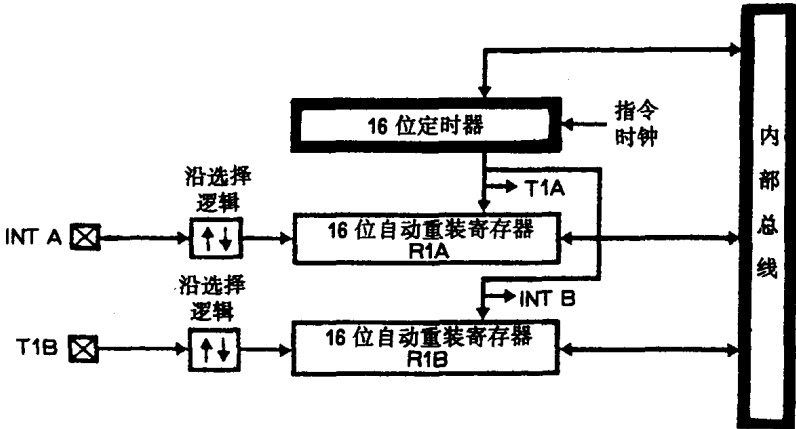
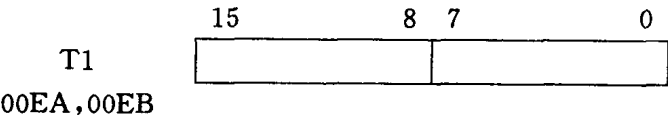


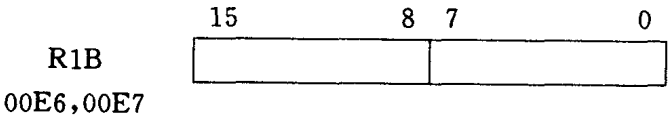
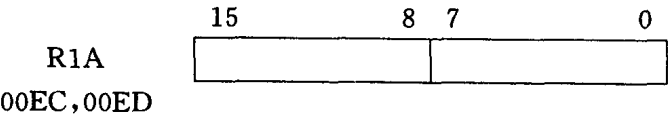
图 3-4 输入捕获方式示意图

四、定时器 T1 的控制寄存器

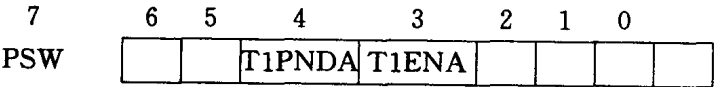
1. 16 位定时/计数器 T1



2. 16 位自动装入/捕获寄存器 R1A 及 R1B



3. 处理器状态字 PSW

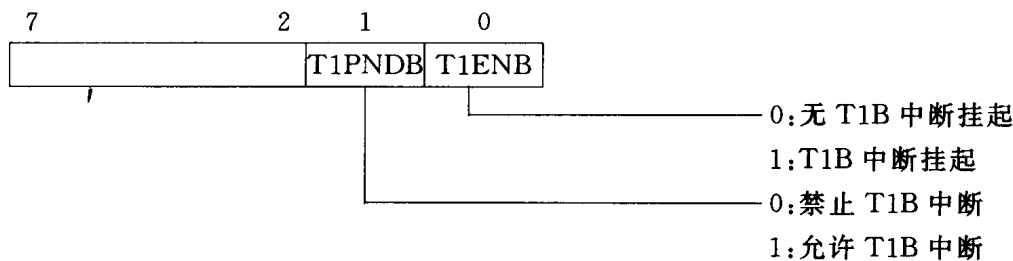


0:无 T1A 中断挂起
1:T1A 中断挂起

0:禁止 T1A 中断
1:允许 T1A 中断

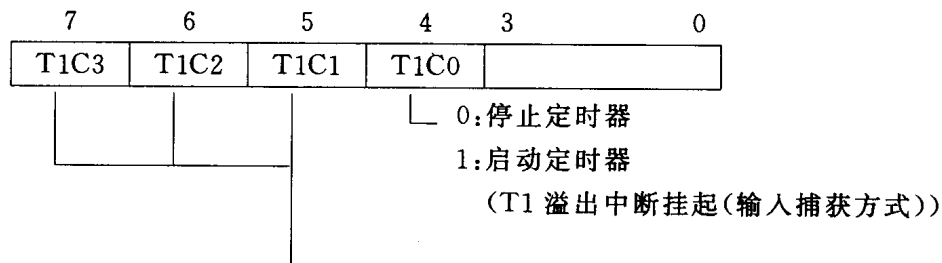
4. 中断控制寄存器 ICNTRL

ICNTRL



5. 控制寄存器 CNTRL

CNTRL



见表 3-1 说明

表 3-1 定时器 T1 工作方式表

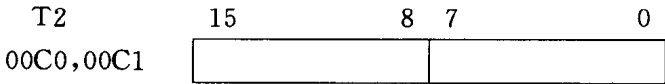
T1C3	T1C2	T1C1	定时器方式	中断源 A	中断源 B	定时器正在计数
0	0	0	方式 2(外部事件计数器)	定时器下溢	TxB 正沿	TxA 正沿
0	0	1	方式 2(外部事件计数器)	定时器下溢	TxB 正沿	TxA 负沿
1	0	1	方式 1(PWM) TxA 触发	自动重装 RA	自动重装 RB	t_c
1	0	0	方式 1(PWM) 非 TxA 触发	自动重装 RA	自动重装 RB	t_c
0	1	0	方式 3(捕获) 捕获: TxA 正沿 TxB 正沿	TxA 正沿或 定时器下溢	TxB 正沿	t_c
1	1	0	方式 3(捕获) 捕获: TxA 正沿 TxB 负沿	TxA 正沿或 定时器下溢	TxB 负沿	t_c
0	1	1	方式 3(捕获) 捕获: TxA 负沿 TxB 正沿	TxB 负沿或 定时器下溢	TxB 正沿	t_c
1	1	1	方式 3(捕获) 捕获: TxA 负沿 TxB 负沿	TxA 负沿或 定时器下溢	TxB 负沿	t_c

3.1.3 新型 COP8 单片机定时器 T2 和 T3

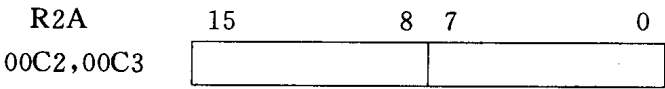
有的单片机还有定时器 T2 和 T3,它们的功能与定时器 T1 功能完全相同。

一、定时器 T2

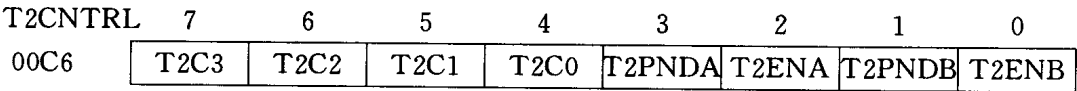
1. 16 位定时/计数器 T2



2. 16 位自动装入/捕获寄存器 R2A 及 R2B



3. 定时器 T2 控制寄存器 T2CNTRL



4. 定时器 T2 引脚

T2A:I/O 端口 L4

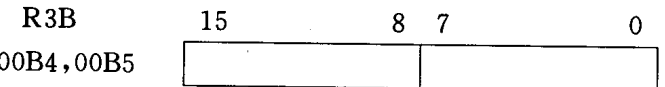
T2B:I/O 端口 L5

二、定时器 T3

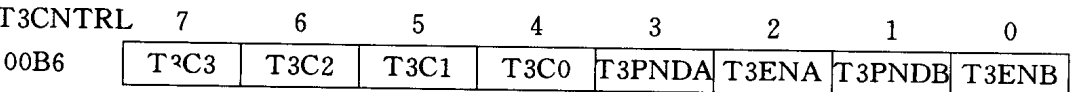
1. 16 位定时/计数器 T3



2. 16 位自动装入/捕获寄存器 R3A 及 R3B



3. 定时器 T3 控制寄存器 T3CNTRL



4. 定时器 T3 引脚

T3A:I/O 端口 L6

T3B:I/O 端口 L7

3. 1. 4 新型 COP8 单片机 WATCHDOG 定时器

COP8 单片机有一个 WATCHDOG 定时器和时钟监视器。WATCHDOG 定时器用来检测用户程序是否进入死循环而产生程序失控。时钟监视器用来检测有无时钟或 CKI 时钟频率是否小于规定时钟频率。

如图 3-5 所示, WATCHDOG 定时器有 WDUPPER 和 WDLLOWER 两个独立服务窗,

辑电路将下拉 WDOUT 引脚为低电平,并维持 $16t_c \sim 32t_c$ 指令周期。当 WDOUT 为高电平后,WATCHDOG 服务窗口重新启动。一旦时钟监视器检测到时钟频率未达到规定频率,G1 引脚也会输出低电平。

WATCHDOG 定时器还受 EPROM 设置寄存器 ECON 的控制,即:

ECON. 2=0, WATCHDOG 定时器允许,G1 引脚作 WDOUT 引脚

ECON. 2=1, WATCHDOG 定时器不允许,G1 作能用 I/O 引脚

§ 3.2 新型 COP8 单片机 MICROWIRE/PLUS 微总线

3.2.1 新型 COP8 单片机 MICROWIRE/PLUS 微总线简介

MICROWIRE/PLUS 微总线是与 SPI 兼容的串行同步通信接口。允许连接 A/D 转换器、显示驱动器、E²PROM 和其他单片机。MICROWIRE/PLUS 微总线内有一个 8 位移位寄存器 SIO、串行数据输入 SI、串行数据输出 SO 和串行移位时钟 SK。图 3-6 是 MICROWIRE/PLUS 微总线系统结构图。

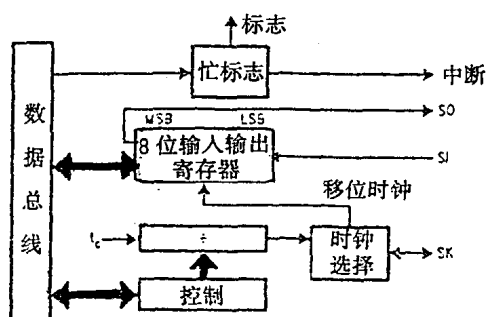


图 3-6 MICROWIRE/PLUS 微总线系统结构图

移位时钟可选用内部时钟或外部时钟,由控制寄存器 CNTRL 中的 SL0,SL1(CNTRL. 0,CNTRL. 1)两位来选择。采用内部时钟源的 MICROWIRE/PLUS 微总线接口称为主控运行方式,采用外部时钟源的 MICROWIRE/PLUS 微总线接口称为从属运行方式。

通过设置处理器状态字 PSW 中的 BUSY(PSW. 2)位可使 MICROWIRE/PLUS 微总线开始移位数据。当 8 位数据移位结束后,BUSY 位复位。对于小于 8 位数据的移位,用户可用软件复位 BUSY 位。

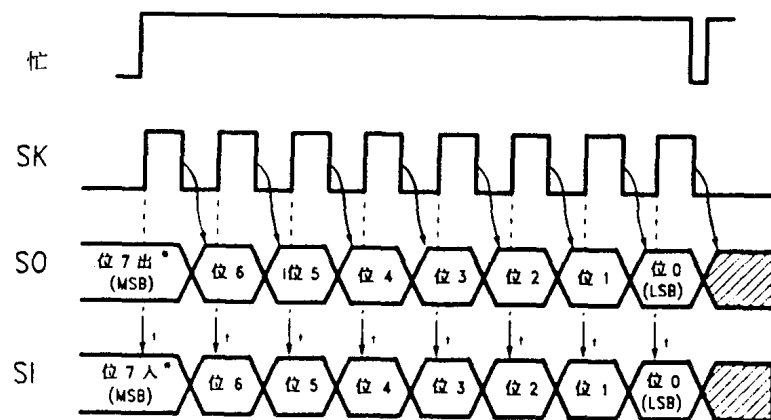
为了使 I/O 端口具有 SI,SO 和 SK 的功能。控制寄存器 CNTRL 中的 MSEL(CNTRL. 3)位必须置位。然后对 I/O 端口 G 的配置寄存器相应位进行设置,表 3-2 列出了相应设置的功能。

表 3-2 微总线的 I/O 端口 G 的配置

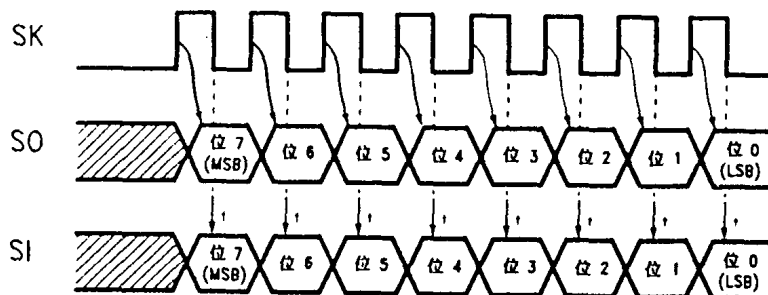
G4(SO) 配置位	G5(SK) 配置位	G4 功能	G5 功能	G6 功能	操作
1	1	SO	内部. SK	SI	微总线主控
0	1	三态	内部. SK	SI	微总线主控
1	0	SO	外部. SK	SI	微总线从属
0	0	三态	外部. SK	SI	微总线从属

系统一旦进入从属运行方式,用户必须立即设置 BUSY 标志位,这样将保证由主控运行方式传送的所有数据能正确移值。在 8 个时钟脉冲之后,BUSY 标志位将清除,移位时钟停止。此过程可重复进行。

单片机既可以正常 SK 时钟方式也可以交替相位时钟方式,从 SIO 寄存器移入和移出数据。此两种工作方式,SK 空闲的电平可以是高电平或低电平,由 I/O 端口 G5 位来选择。在正常 SK 时钟方式中,数据在 SK 时钟的上升沿时移入,在 SK 时钟的下降沿移出。而在交替相位时钟方式中,数据则在 SK 时钟的下降沿时移入,在 SK 时钟的上升沿移出。I/O 端口 G6 则来选择 SK 时钟的边沿。图 3-7 是 SK 时钟方式与交替相位时钟方式的时序图。



(a) SK 时钟方式



(b) 交替相位时钟方式

图 3-7 MICROWIRE/PLUS 微总线时序图

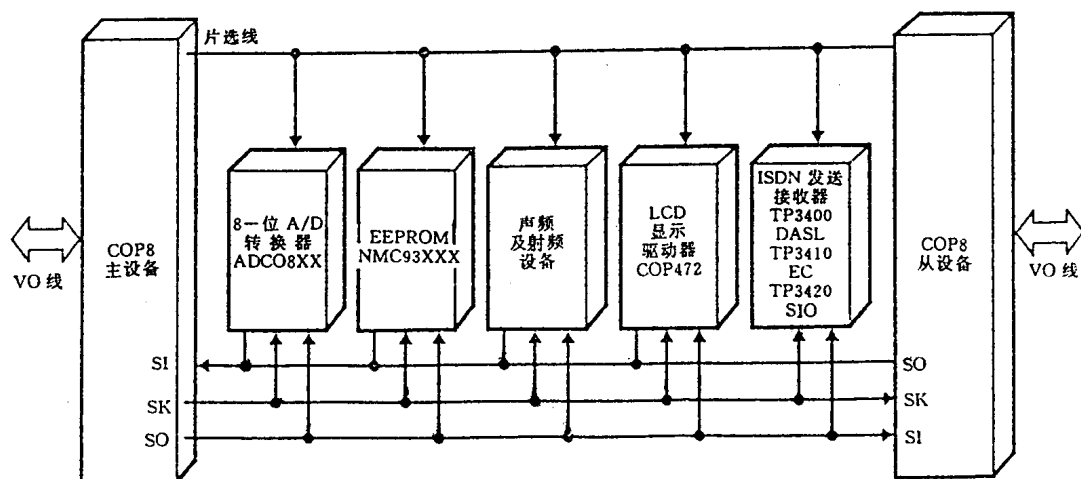


图 3-8 微总线外扩示意图

表 3-3 微总线外围设备

器件	说明
A/D 转换器和比较器	
ADC0811	11 通道 8 位多路 A/D 转换器
ADC0819	19 通道 8 位多路 A/D 转换器
ADC0831	1 通道 8 位多路 A/D 转换器
ADC0838	8 通道 8 位多路 A/D 转换器
ADC0832	2 通道 8 位多路 A/D 转换器
ADC0833	4 通道 8 位多路 A/D 转换器
ADC0834	4 通道 8 位多路 A/D 转换器
ADC0852	具有 8 位参考除法器的多路比较器
ADC0854	具有 8 位参考除法器的多路比较器
显示器驱动器	
COP472-3	3×12 多路可扩展 LCD 显示驱动器
MM5450	35 输出 LED 显示驱动器
MM5451	34 输出 LED 显示驱动器
MM5483	31 段 LCD 显示驱动器
MM5484	16 段 LED 显示驱动器
MM5486	33 输出 LED 显示驱动器
MM58241	32 输入 LED 显示驱动器
MM58341	32 输出高电压显示驱动器
MM58342	20 输出高电压显示驱动器
存储器设备	
NM93C06	16×16 CMOS EEPROM
NM93C13	16×16CMOS EEPROM
NM93C14	64×16CMOS EEPROM
NM93C46	64×16CMOS EEPROM
NM93CS06	具有写保护的 16×16CMOS EEPROM
NM93CS46	具有写保护的 64×16CMOS EEPROM
NM93CS56	具有写保护的 128×160CMOS EEPROM
NM93C56	128×16CMOS EEPROM
NM93CS66	具有写保护的 256×15CMOS EEPROM
NM93C66	256×16COMS EEPROM

(续表)

器件	说明
音频和无线设备	
LMC1982	双输入立体声音量/音调
LMC1983	3 输入立体声音量/音调
LMC1992	4 输入立体声音量/音调
LMC835	7 波段图形均衡器
LMC1971/72/73	1/2/3 通道 μ 端口 68/78/76dB 音频衰减

图 3-9 是通过微总线外扩 E²PROM 的例子。并列出了相应的源程序。

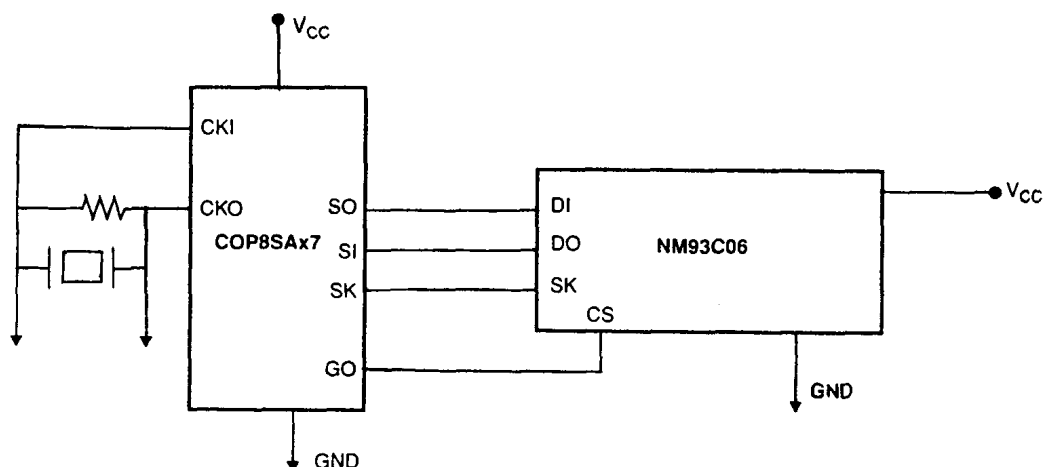


图 3-9 外接 E²PROM 接口图

所有命令、数据入/出都是在 SK 时钟上升沿进行的，片选 $\overline{CS}$ 的低电平维护时间不小于 10ms。下面是 COP8A×7 单片机与 256 位 E²PROM 的 NM93C06 的接口源程序：

```
.INCLD COP888. INC
```

```
;
```

；本程序以子程序形式提供 NMC9306 EEPROM 的擦除、使能、禁止、读和写功能。

```
;
```

```
;
```

```
.INCLD COP888. INC
```

```
.SECT NMC, RAM
```

```
NMCMEM: .DSB 5
```

```
SNDBUF = NMCMEM
```

```
RDATL = NMCMEM + 1
```

```
RDATH = NMCMEM + 2
```

```
WDATL = NMCMEM + 3
```

```
WDATH = NMCMEM + 4
```

```
ADRESS: .DSB 1
```

FLAGS: .DSB 1

.SECT CNT,REG

DLYH: .DSB 1

DLYL: .DSB 1

;

;COP888CG 和 COP494(256 位 EEPROM)的接口包含 4 行:

;G0(芯片选择行),G4(串行输出 SO),G5(串行时钟 SK)和 G6(串行 SI)

;

;初始化

;

LD PORTGC, #031 ;设置 G0、G4、G5 为输出

LD PORTGD, #00 ;初始化 G 数据寄存器为 0

LD CNTROL, #08 ;使能 MSEL,选择 MW 的 $2t_c$ 速率

LD B, #PSW

LD X, #SIOR

;

;本程序擦除由“ADDRESS”单元中的地址指向的存储器单元。

;"ADDRESS"的低 4 位字节包含 COP494 寄存器地址,高 4 位应设为 0。

;

ERASE: LD A,ADDRESS

OR A, #0C0

X A,SNDBUF

LD FLAGS, #0

JSR INT

RET

;本程序使能 COP494 编程。编程使能(EWEN)后必须进行编码程。

;

EWEN: LD SNDBUF, #030

LD FLAGS, #0

JSR INIT

RET

;

;本程序使能 COP494 编程。

;

EWDS: LD SNDBUF, #0

LD FLAGS, #0

JSR INIT

```

        RET
;
;本程序擦除所有 COP494 寄存器。
;
ERAL:   LD    SNDBUF, #020
        LD    FLAGS, #0
        JSR   INIT
        RET
;
;本程序读 COP494 寄存器内容。COP494 寄存器被指定在单元“ADDRESS”的低 4 位字节。
;高 4 位字节应置为 0。COP494 寄存器的 16 位字节存储在 RDATL 和 RDATH 中。
;
READ:   LD    A, ADDRESS
        OR    A, #080
        X     A, SNDBUF
        LD    FLAGS, #1
        JSR   INIT
        RET
;
;本程序写存储在 WDATL 和 WDATH 中的 16 位值到 COP494 寄存器,
;该寄存器的地址包含在单元“ADDRESS”的低 4 位字节中,
;地址单元的高 4 位字节应置为 0。
;
WRITE:  LD    A, ADDRESS
        OR    A, #040
        X     A, SNDBUF
        LD    FLAGS, #3
        JSR   INIT
        RET
;
;本程序送起始位和命令字节。它也对标志单元内容译码并决定写、
;读或返回到调用程序。
;
INIT:   SBIT   0, PORTGD           ;设置芯片选择为高
        LD     SIOR, #001         ;传送起始位到 SIOR
        SBIT   BUSY, [B]          ;送出起始位
PUNT1:  IFBIT   BUSY, [B]
        JP     PUNT1

```

	LD	A,SNDBUF	
	X	A,[X]	; 传送起始位到 SIOR
	SBIT	BUSY,[B]	; 送出命令字节
PUNT2:	IFBIT	BUSY,[B]	
	JP	PUNT2	
	IFBIT	0,FLAGS	; 进一步处理?
	JP	NOTDON	; 是
	RBIT	0,PORTGD	; 否,重置 CS 并返回
	RET		
NOTDON:	IFBIT	1,FLAGS	; 读或写?
	JP	WR494	; 跳至写程序
	LD	SIOR,#000	; 否,读 COP494
	SBIT	BUSY,PSW	
	RBIT	BUSY,[B]	
	SBIT	BUSY,[B]	
PUNT3:	IFBIT	BUSY,[B]	
	JP	PUNT3	
	X	A,[X]	
	SBIT	BUSY,[B]	
	X	A,RDATH	
PUNT4:	IFBIT	BUST[B]	
	JP	PUNT4	
	LD	A,[X]	
	X	A,RDATC	
	RBIT	0,PORTGD	
	RET		
WR494;	LD	A,WDATH	
	X	A,[X]	
	SBIT	BUSY,[B]	
PUNT5:	IFBIT	BUSY,[B]	
	JP	PUNT5	
	LD	A,WDATL	
	X	A,[X]	
	SBIT	BUSY,[B]	
PUNT6:	IFBIT	BUSY,[B]	
	JP	PUNT6	

```

        RBIT    0,PORTGD
        JSR     TOUT
        RET
;
;该程序为写产生延迟
;
TOUT:    LD      DLYH,#00A
WAIT:    LD      DLYL,#0FF
WAIT1:   DRSZ   DLYL
        JP      WAIT1
        DRSZ   DLYH
        JP      WAIT
        RET
.END

```

§ 3.3 新型 COP8 单片机 I/O 端口

新型(高集成型)COP8 单片机有多个多功能 I/O 端口,特别是 I/O 端口 L 还具有多输入唤醒的功能。

3.3.1 新型 COP8 单片机多功能 I/O 端口

新型 COP8 单片机多功能 I/O 端口具有双向 I/O、单输出和单输入三种形式,如图 3-10 所示。

一、双向 I/O 端口

COP8 单片机双向 I/O 端口有 I/O 端口 C、F、G 和 L 等。对于每一个双向 I/O 端口一般有数据寄存器、配置寄存器和 I/O 引脚等三个存储器映像 I/O 地址。

双向 I/O 端口的端口设置如表 3-4 所示,图 3-11 是输入方式和输出方式的配置示意图。

表 3-4 I/O 端口设置

	配置寄存器	数据寄存器
G7	CLKDLY	HALT
G6	交替 SK	IDLE

1. I/O 端口 C

I/O 端口 C 是通用双向 I/O 端口。新型 COP8 单片机中,40 引脚封装的单片机,有 C0~C3 I/O 引脚;44 引脚封装单片机,有 C0~C7 I/O 引脚。

I/O 端口 C 的三个存储器映像 I/O 地址:

- I/O 端口 C 数据寄存器:D8

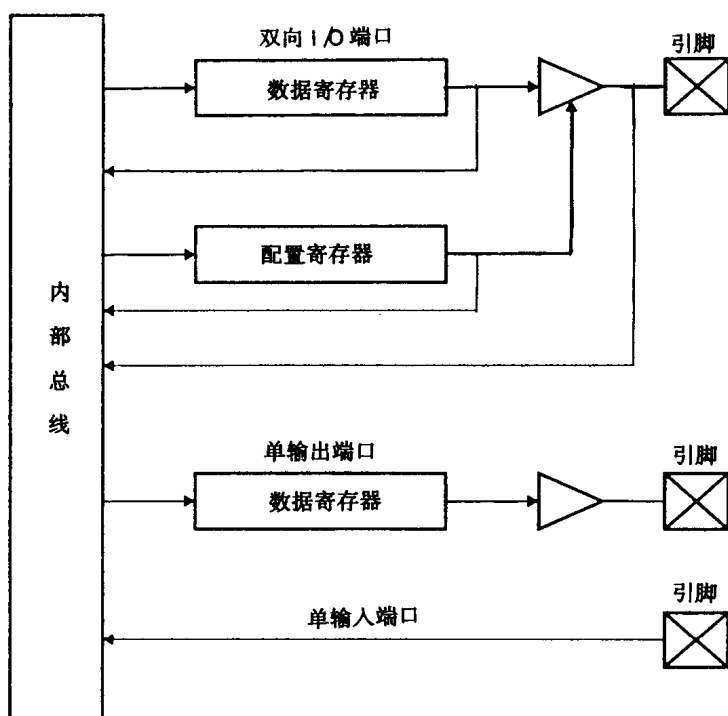


图 3-10 多功能 I/O 端口形式

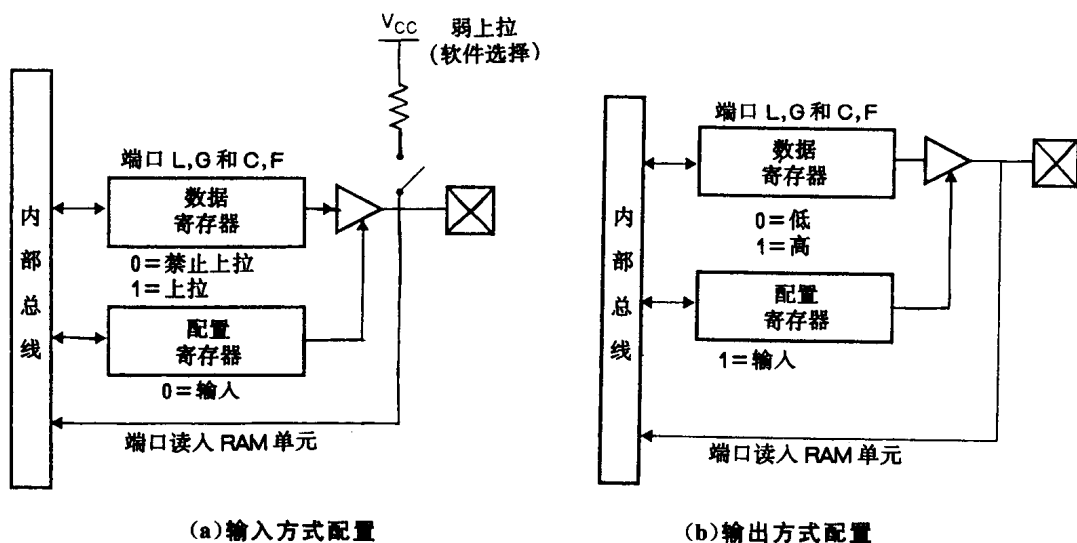


图 3-11 I/O 端口配置图

- I/O 端口 C 配置寄存器:D9
 - I/O 端口输入引脚:DA(只读)
2. I/O 端口 F

I/O 端口 F 也是双向通用 I/O 端口，在 SG 类单片机中，它还可以作比较器的输入、输出引脚。28 引脚单片机有 F0~F3 引脚，40 引脚以上单片机有 F0~F7 引脚。

I/O 端口 F 的三个存储器映像 I/O 地址:

- I/O 端口 F 数据寄存器:94
- I/O 端口 F 配置寄存器:95
- I/O 端口 F 输入引脚:96(只读)

另外,在 SG 类单片机,I/O 端口下还可以模拟比较器输入、输出引脚:

- F1:COMP1-IN(比较器 1 负输入)
- F2:COMP1+IN(比较器 1 正输入)
- F3:COMP1OUT(比较器 1 输出)
- F4:COMP2-IN(比较器 2 负输入)
- F5:COMP2+IN(比较器 2 正输入)
- F6:COMP2OUT(比较器 2 输出)

3. I/O 端口 G

I/O 端口 G 也是双向可通用 I/O 端口,并且输入引脚带有施密特电路。在所有新型 COP8 单片机中,都有 G0~G7 I/O 引脚。I/O 端口 G 的引脚往往作定时器 T1、WATCH-DOG 定时器、中断、时钟和 MICROWIRE/PLUS 微总线等其他 I/O 的引脚:

- G0/INT:外中断输入
- G1/WD:WATCHDOG 的 WDOOUT 引脚
- G2/T1B:定时器 T1 捕获输入
- G3/T1A:定时器 T1 I/O 引脚
- G4/SO:微总线串行数据输出
- G5/SK:微总线串行时钟
- G6/SI:微总线串行数据输入
- G7/CKO:振荡器 CKO 输出

I/O 端口 G 的三个存储器映像 I/O 地址:

- I/O 端口 G 数据寄存器:D4
- I/O 端口 G 配置寄存器:D5
- I/O 端口 G 输入引脚:D6(只读)

另外,数据寄存器与配置寄存器的 G6,G7 位还可作为相应的标志位,如表 3-5 所示。

表 3-5 G6,G7 的标志位

	配置寄存器	数据寄存器
G6	交替相位 SK	IDLE
G7	CLKLY	HALT

4. I/O 端口 L

I/O 端口 L 也是双向 I/O 端口,并有输入唤醒的功能。除了 16 引脚单片机只有 L0~L3 引脚,其余封装的单片机都有 L0~L7 引脚。

I/O 端口 L 的三个存储器映像 I/O 地址:

- I/O 端口 L 数据寄存器:D0
- I/O 端口 L 配置寄存器:D1
- I/O 端口 L 输入引脚:D2(只读)

二、输入端口

I/O 端口 I 是专门输入 I/O 端口,一般往往作为 A/D 转换的输入,或模拟比较器输入引脚。新型 COP8 单片机中 SA 类及 SR 类单片机都无 I/O 端口。

三、输出端口

I/O 端口 D 是专门输出 I/O 端口。28 引脚单片机有 D0~D3 I/O 引脚,40 引脚以上单片机有 D0~D7 I/O 引脚。

I/O 端口 D 只有一个存储器映像 I/O 地址:

- I/O 端口 D;DC

3.3.2 新型 COP8 单片机多输入唤醒功能

多输入唤醒 MIWU (Multi-Input Wakeup) 可用于使单片机退出 HALT 工作方式或 IDLE 工作方式;另外,多输入唤醒也可提供 8 个触发边沿可选的外中断。I/O 端口 L 具有多输入唤醒的功能。图 3-12 是新型 COP8 单片机的多输入唤醒的系统结构图。

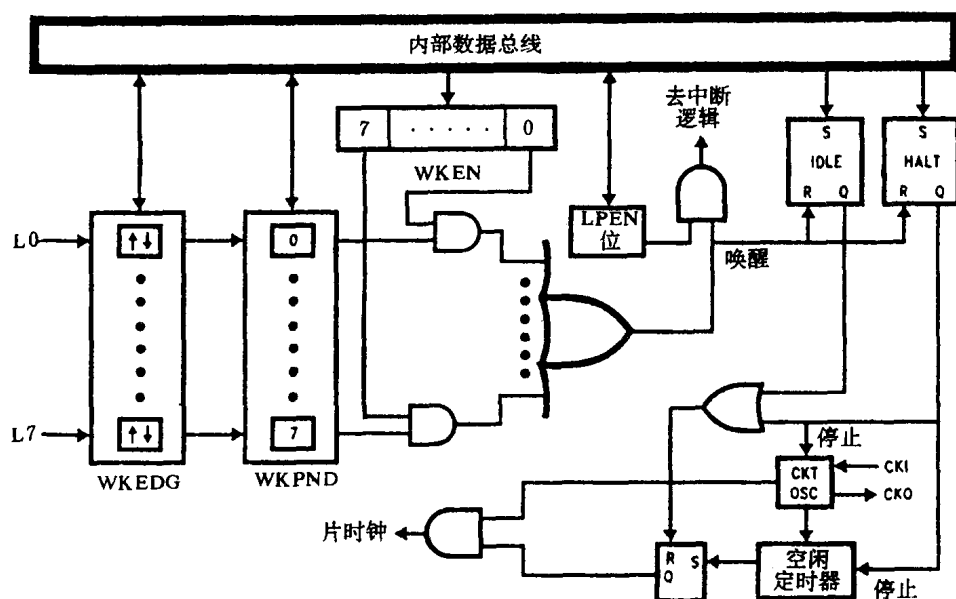


图 3-12 多输入唤醒的系统结构图

多输入唤醒逻辑中有三个可寻址的寄存器:MIWU 边沿选择寄存器 WKEDG (MIWU Edge Select Register)、MIWU 许可寄存器 WKEN (MIWU Enable Register) 和 MIWU 挂起寄存器 WKPND (MIWU Pending Register):

- MIWU 边沿选择寄存器 WKEDG

WKEDG	7	0
C8		

0: 正跳边沿

1: 负跳边沿

• MIWU 许可寄存器 WKEN



0: 禁止 MIWU

1: 允许 MIWU

• MIWU 挂起寄存器 WKPND



0: 无 MIWU 挂起

1: MIWU 挂起

I/O 端口 L 的多输入端 L0~L7 中有正跳边(或负跳变)产生(由 WKEDG 寄存器来选择),记录在 WKPND 寄存器中,在 WKEN 寄存器的相应位允许下,产生唤醒的逻辑。该唤醒逻辑可复位 IDLE 或 HALT 触发器(即退出 IDLE 或 HALT 工作方式);同时,也可产生中断逻辑请求。

为了避免由于输入边沿改变而引起伪唤醒。因此,若要改变输入边沿需:

- 首先复位,相关的 WKEN 寄存器位。
- 然后,改变 WKEDG 寄存器中相应的触发边沿选择。
- 再把相关的 WKPND 寄存器的位清零。
- 最后,重新允许相应的 WKEN 寄存器位。

例:把 I/O 端口 L5 由正跳边沿改为负跳边沿触发,而 L5 原已作为输入端,程序如下:

```

RBIT 5,WKEN
SBIT 5,WKEDG
RBIT 5,WKPND
SBIT 5,WKEN
    
```

I/O 端口 L 实际上也为用户提供了 8 个外中断,这些中断都指向同一个中断服务程序。此时,WKEDG 寄存器可作为外中断输入边沿的选择,WKPND 寄存器可作为外中断挂起寄存器,WKEN 寄存器作为中断许可寄存器。

当系统中断 GIE 开放中断,中断控制寄存器 ICNTL 中 LPEN 位(ICNTL. 6)置位(即允许 MIWU 中断),系统就允许 I/O 端口 L 外输入中断。

§ 3.4 新型 COP8 单片机 UART

3.4.1 新型 COP8 单片机 UART 系统结构

COP8SG 类单片机有一个全双工 UART, 它由发送、接收移位寄存器和发送缓冲寄存器 TBUF(Transmit Buffer Register)、接收缓冲寄存器 RBUF(Receiver Buffer Register)、UART 控制/状态寄存器 ENU(UART Control and Status Register)、UART 接收控制/状态寄存器 ENUR(UART Receiver Control and Status Register)、UART 中断/时钟脉冲源寄存器 ENUI(UART Interrupt and Clock Source Register)、预分频器选择寄存器 PSR(Prescaler Select Register)以及波特率寄存器 BAUD 等 7 个可寻址寄存器构成。如图 3-13 所示。

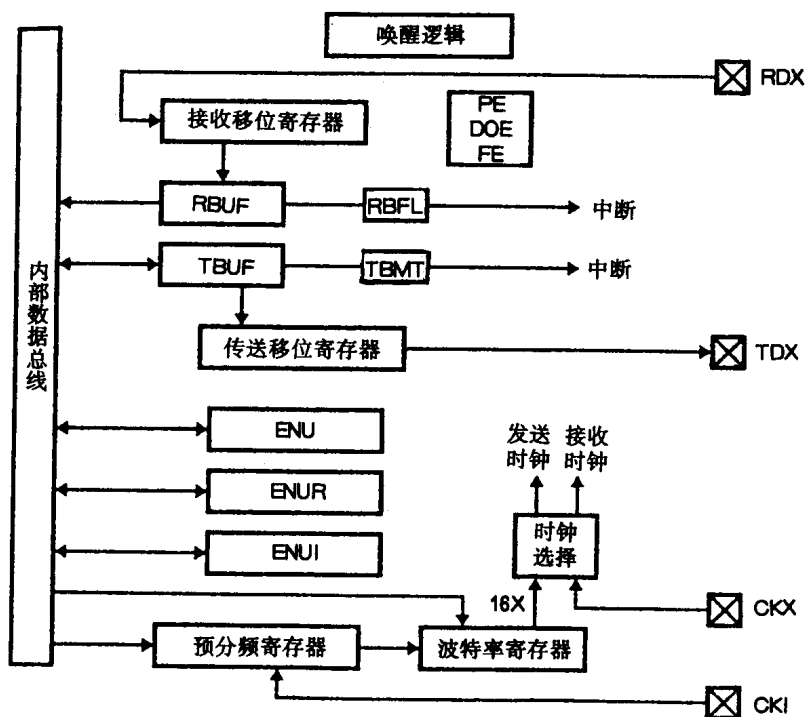


图 3-13 全双工 UART 系统结构图

UART 控制/状态寄存器 ENU 用来确定数据帧长度、奇/偶校验、发送第 9 位值及发送/接收标志, 见图 3-14。

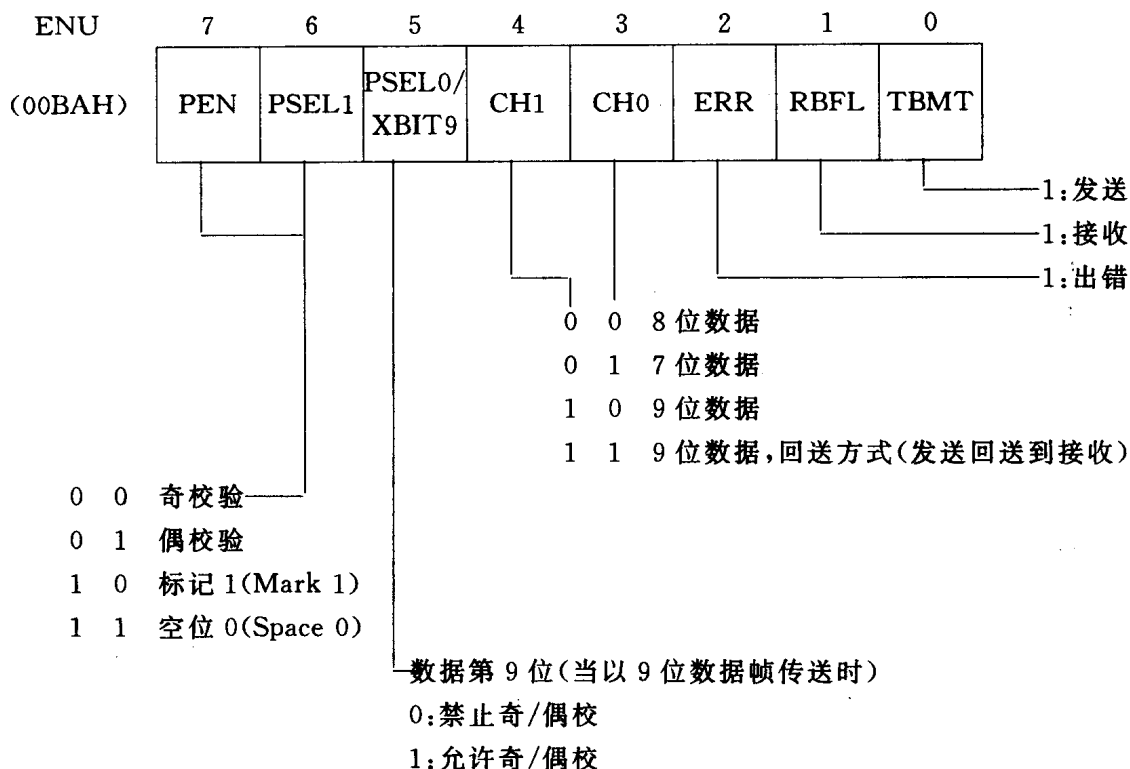


图 3-14 UART 控制/状态寄存器 ENU

UART 接收控制/状态寄存器 ENUR 在 UART 接收数据时,检验接收的数据是否帧超限、奇/偶错和帧标志,以及保存接收数据的第 9 位、允许/禁止 UART 的 ATTENTION 工作方式和由 RCVG 与 XMTG 位提供附加的接收/发送状态信息等,见图 3-15。

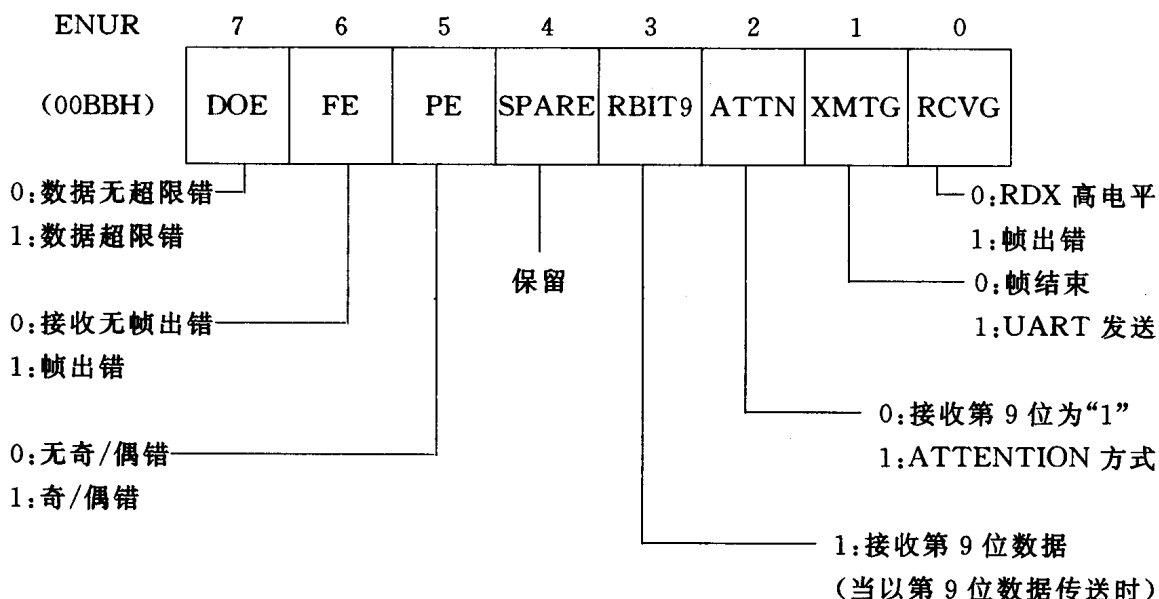


图 3-15 UART 接收控制/状态寄存器 ENUR

中断/时钟脉冲源寄存器 ENUI 用来确定内部/外部时钟源,选择停止位个数和允许/禁止发送与接收中断;另外,还用来确定 UART 的同步/异步方式,见图 3-16。

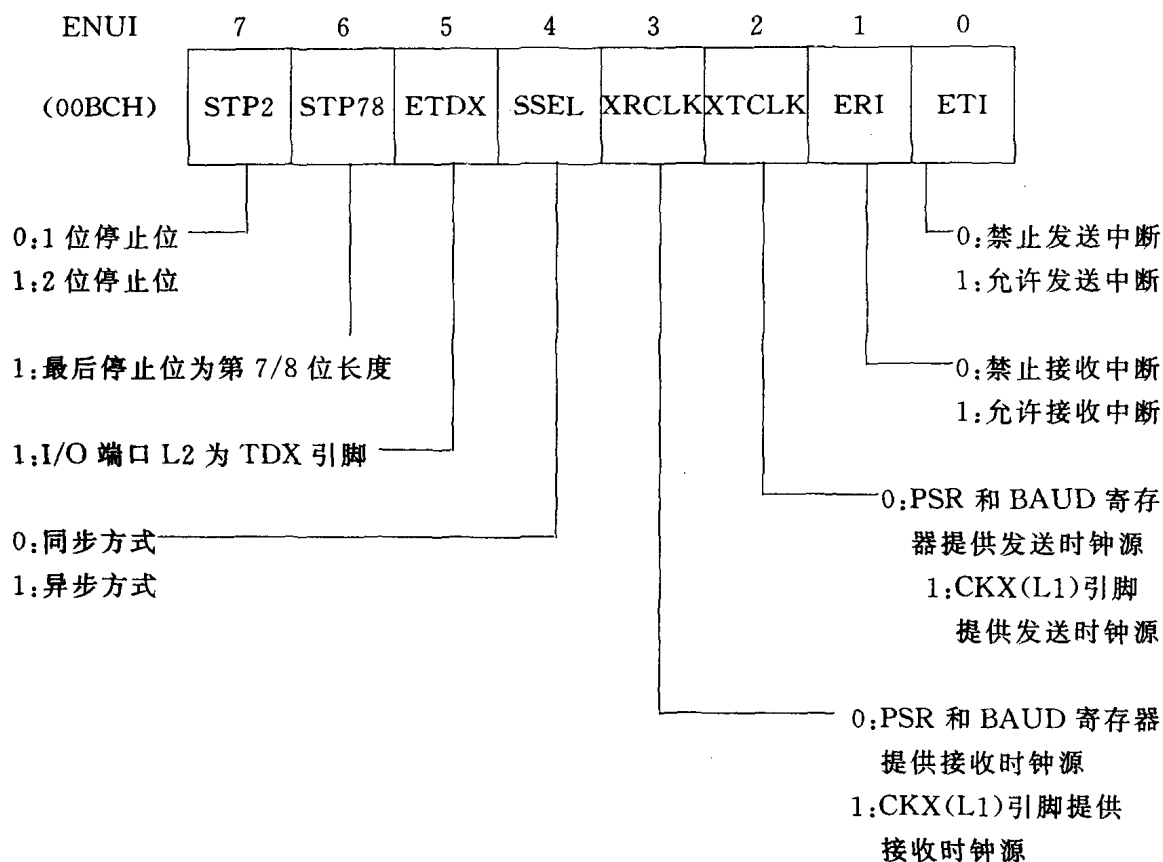


图 3-16 中断/时钟脉冲源寄存器 ENUI

预分频选择寄存器 PSR 和波特率寄存器 BAUD 用来选择发送/接收内部时钟源,如图 3-16 所示,内部时钟经 1~16(增量为 0.5)的预分频和一个 11 位二进制计数器的二级分频:

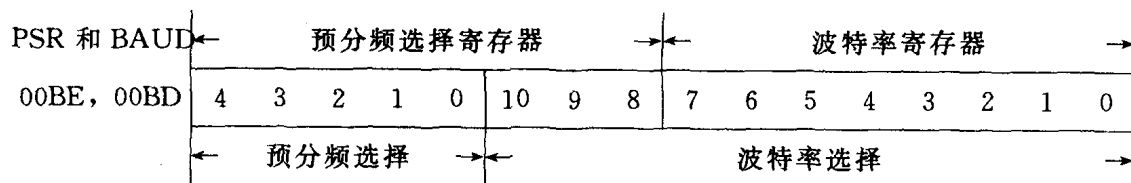


表 3-6 给出了 5 位预分频器选择与预分频因子之间的对应关系。一般可以使第一级分频(5 位预分频)的频率为 1.8432MHz;然后,通过第二级分频(11 位二进制计数器)达到 110~38400 各种波特率(见表 3-7 所示)。

表 3-6 预分频因子表

预分频器选择	预分频器因子
00000	无时钟
00001	1
00010	1.5
00011	2
00100	2.5
00101	3
00110	3.5
00111	4
01000	4.5
01001	5
01010	5.5
01011	6
01100	6.5
01101	7
01110	7.5
01111	8
10000	8.5
10001	9
10010	9.5
10011	10
10100	10.5
10101	11
10110	11.5
10111	12
11000	12.5
11001	13
11010	13.5
11011	14
11100	14.5
11101	15
11110	15.5
11111	16

表 3-7 波特率分频器(1.8432MHz 预分频器输出)

波特率	波特率分频器 -(N-1)
110(110.03)	1046
134.5(134.58)	855
150	767
300	383
600	191
1200	95
1800	63
2400	47
3600	31
4800	23
7200	15
9600	11
19200	5
38400	2

例如,在异步方式时,CKI 为 4.608MHz。预分频器因子选为:

$$4.608/1.8432=2.5$$

查表 3-6 来确定 5 位预分频选择值 00100。

若所需波特率为 19200,查表 3-7 可得:

$$N-1=5$$

$$N=6$$

在异步方式时,输入至 UART 的频率已先 16 分频,故:

$$\text{波特率}=1.8432\text{MHz}/(16\times 6)=19200$$

即:

$$BR=Fc/(16\times N\times P)$$

BP:波特率

Fc:CKI 频率

N:波特率因子

P:预分频器分频因子

注:在同步方式时,由 2 取代 16 分频因子。

另外,UART 发送缓冲器 TBUF 可寻址地址:B8

UART 接收缓冲器 RBUF 可寻址地址:B9

3.4.2 新型 COP8 单片机 UART 工作方式

UART 有两种工作方式:异步方式和同步方式。

异步方式(ENUI 寄存器的位 0,SSEL=0)。当发送移位寄存器 TSFT 在 TDX 引脚移出当前数据时,程序可把要发送的下一字节装载至发送缓冲器 TBUF;发送移位寄存器 TSFT 完成移送数据后,发送缓冲器 TBUF 内容送至发送移位寄存器 TSFT,并置发送缓冲器空标志 TBMT(ENUI.0 位);当新的数据装载发送缓冲器 TBUF,UART 就自动把 TBMT 位复位,并置位(ENUR.1 位);在每帧结束时(停止位),复位 XMTG。

若 RDX 引脚确认到有效起始位后,就对该帧的其余各位数据串行移入到接收移位寄存器 RSFT;接收到一帧完整数据后,就送至接收缓冲器 RBUF,并置接收缓冲器满标志 RBFL(ENUI.1 位);单片机从接收缓冲器 RBUF 读取数据时,自动复位 RBFL。当帧有错时,RCVG 位(ENUR.0 位)置起;一旦 RDX 引脚为高电平,RCVG 位就为“0”。

同步方式(ENUI 寄存器位 0,SSEL=1),数据传输与时钟同步,在同步时钟上升沿发送数据,在同步时钟下降沿接收数据。

当选择 CKX 引脚作外时钟输入时,通过 TDX/RDX 引脚的外时钟同步地完成数据发送/接收。

在内时钟时,CKX 引脚产生同步时钟输出,由内部波特率发生器产生同步时钟,完成数据发送/接收。

当接收缓冲器满或发送缓冲器空时,UART 产生中断。这两个中断有各自的中断向量 0×EC~0×EFH。由 ENUI 寄存器的 ETI(ENUI,0 位)控制发送中断禁止/许可,ERI(ENUI.1 位)控制接收中断禁止/许可。

对 TBMT(ENUI.0 位)和 ETI(ENUI.0 位)置位,来自发送的中断置为挂起状态,并一

直保持挂起状态。中断撤除时,程序应把 ETI 位清除,或写发送缓冲器 TBUF(即把 TBMT 位清除)。

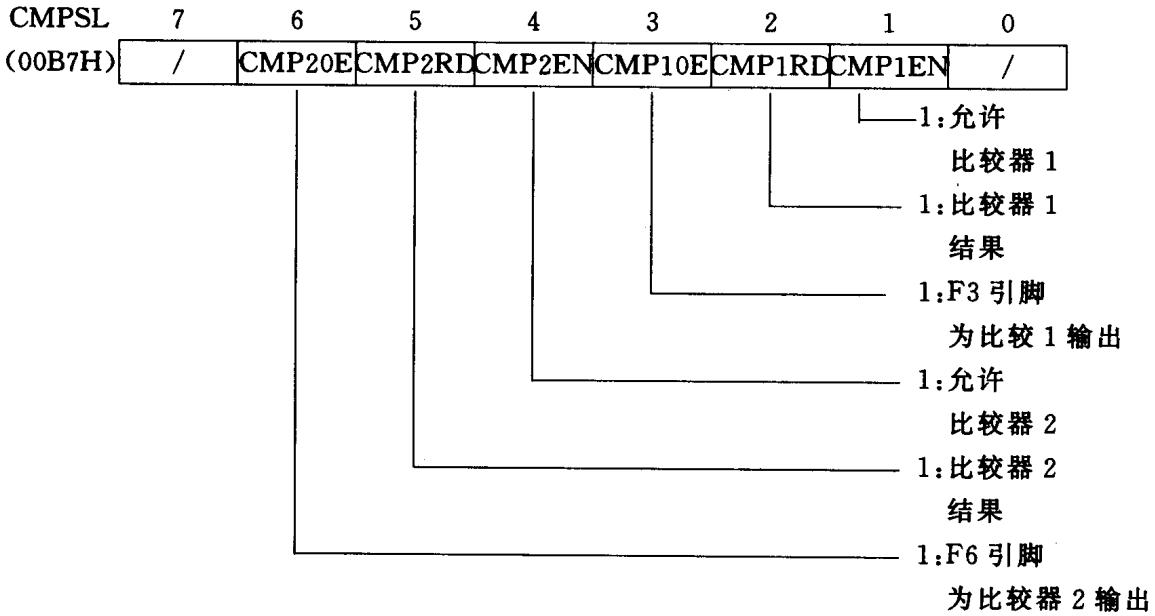
对 RBMF(ENU. 1 位)和 ERI(ENUI. 1 位)置位,来自收接的中断置为挂起状态,并一直保持。中断撤除时,程序应把 ERI 位清除,或读接收缓冲器 RBUF(即把 RBFL 位清除)。

§ 3.5 新型 COP8 单片机模拟比较器

COP8SG 单片机有两个差动比较器,每一个比较器有一对输入(正向和反向输入),一个输出。I/O 端口下的 F1~F6 引脚用于比较器:

- F1: 比较器 1 负输入
- F2: 比较器 1 正输入
- F3: 比较器 1 输出
- F4: 比较器 2 负输入
- F5: 比较器 2 正输入
- F6: 比较器 2 输出

模拟比较器有一个比较器选择寄存器 CMPSL(Compare Select Register)用来使能比较器,读比较器内部输出,使能比较器输出引脚,见图 3-17。



* 两个没有用的位可作为软件标志位。

图 3-17 比较器选择寄存器 CMPSL

§ 3.6 新型 COP8 单片机高分辨率 A/D 转换

COP8ACC 和 COP888EK 单片机含有 4~6 通道的 16 位单积分 A/D 转换。该模拟模块

由 16 位定时器 T2B、比较器、基准电源 $V_{cc}/2$ 、 $20\mu A$ 恒流源及比较器选择寄存器 CMPSL (Comparator Select Register)等组成,如图 3-18 所示。

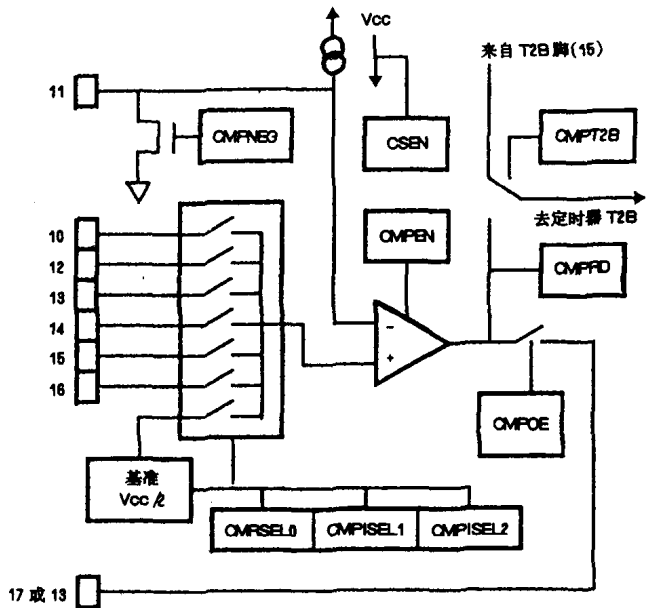


图 3-18 单积分 A/D 转换框图

比较器选择寄存器 CMPSL 用来选择模拟输入通道,控制比较器、定时器 T2B 及模拟比较输出,见图 3-19。

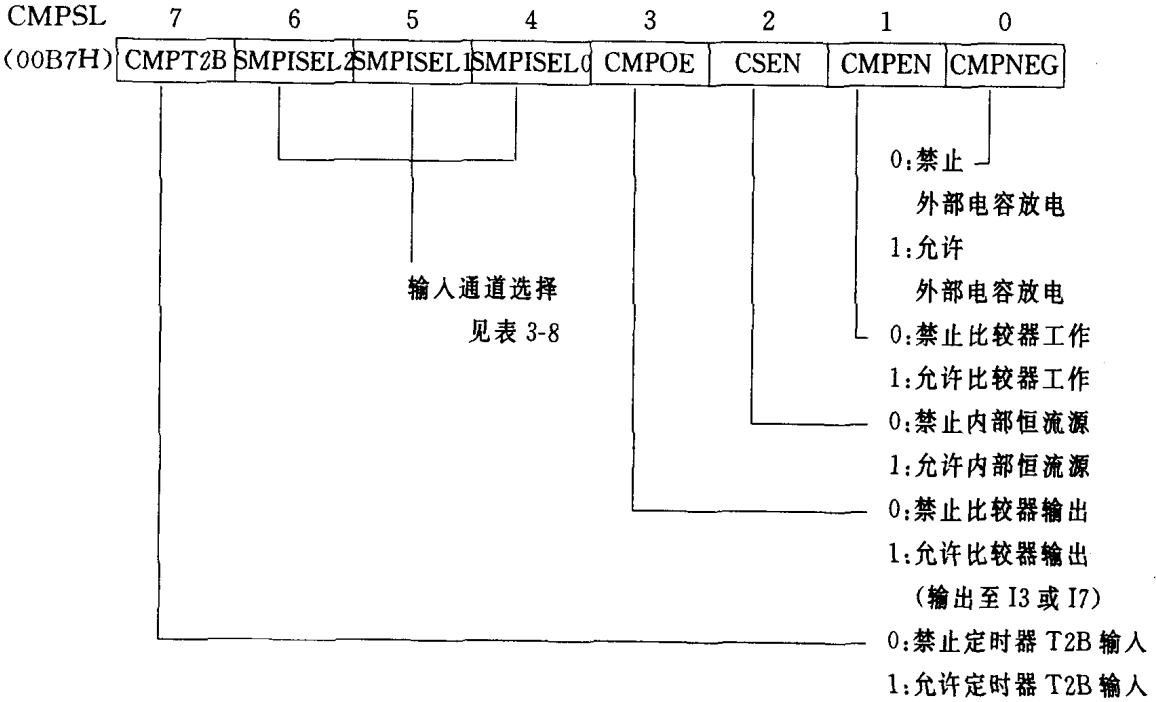


图 3-19 比较器选择寄存器 CMPSL

表 3-8 输入通道的选择

控制位			比较器输入源		比较器输出
CMPSEL2	CMPSEL1	CMPSEL0	负输入	正输入	
0	0	0	I1	I2	I3
0	0	1	I1	I2	I7
0	1	0	I1	I3	I7
0	1	1	I1	I0	I7
1	0	0	I1	I4	I7
1	0	1	I1	I5	I7
1	1	0	I1	I6	I7
1	1	1	I1	Vcc/2 参考电压	I7

复位时,清除比较器选择寄存器 CMPSL(比较器被禁止)。为了节省功耗,在单片机进入 HALT/IDLE 工作方式之前,程序应先禁止比较器。禁止比较器将断开恒流源和 Vcc/2 基准电压,断开比较器与定时器 T2B 输入,断开比较器输出与 I3 或 I7 的连接,以及断开外部电容放电。

第四章 新型 COP8 单片机指令系统

COP8 单片机指令系统简易有效,有以下特点:

- 大多数为单字节指令。
- 一些主要单字节指令执行为单指令周期。
- 许多单字节指令具有多功能,如 DRSZ。
- 三种存储器映像指针:2 个为寄存器间接寻址,1 个为软件堆栈。
- 16 个存储器映像寄存器 R0~R15。
- 可以对存储器、寄存器及 I/O 端口数据进行位测试、置位及复位。
- 寄存器间接寻址的取数和交换指令具有加 1/减 1 功能。
- 有许多单一指令,如 DRSZ,IFBNE,DCOR,RETSK,VIS 和 RRC 指令,这样既可减少程序编码,并可提高执行速度。

§ 4.1 新型 COP8 单片机寻址方式

COP8 单片机有 10 种寻址方式,6 种为操作数寻址,4 种为控制转移寻址。

4.1.1 新型 COP8 单片机操作数寻址

新型 COP8 单片机具有直接寻址、B(×)寄存器间接寻址带加 1/减 1 的 B(×)寄存器间接寻址、立即数寻址、短立即数寻址及从程序存储器间接寻址等 6 种寻址方式:

1. 直接寻址

指令中包括了直接指向数据存储器操作数的 8 位地址,如:

LD A,05;(0005)→A

2. B(×)寄存器间接寻址

操作数由 B 寄存器或 X 寄存器给出存储器的地址,B(X)寄存器实际上是数据存储器的指针,如:

X A,[B];((B))→A,且(A)→(B)

3. 加 1/减 1 的 B(×)寄存器间接寻址

在上一种寻址方式基础上,执行指令后,B(×)寄存器自动加 1/减 1。如:

X A,[B+];((B))→A,且(A)→(B),且(B)+1→B

4. 立即数寻址

指令中包括作为操作数的 8 位立即数,如:

LD A,#05;05→A

5. 短立即数寻址

指令中包括作为操作数的 4 位立即数,如:

LD B, #7;7→B

6. 间接寻址

该寻址方式,从程序存储器访问数据操作数,此时累加器 A 为程序计数器 PC 低 8 位地址。如:

LAID;ROM(PC.A)→A

4.1.2 新型 COP8 单片机控制转移寻址

新型 COP8 单片机具有相对寻址、绝对寻址、长绝对寻址及间接寻址等 4 种寻址方式。

1. 相对转移寻址

该寻址方式用于 JP 指令,指令字段加到程序计数器 PC 产生下一条指令地址,相对转移的范围为 -31~+32(注意:JP+1 指令可由 NOP 指令来取代),如:

JP 0A;(PC)+0AH→PC

2. 绝对转移寻址

该寻址方式用于 JMP 与 JSR 指令,12 位的指令字段取代 PC 的低 12 位,即允许跳转到当前 4K 字节程序存储器段的任何单元。如:

JMP 0125;0125H→PC0~11

3. 长绝对转移寻址

该寻址方式用于 JMPL 与 JSRL 指令,15 位的指令字段全部取代 PC 的 15 位,即允许跳转到整个 32K 字节程序存储器任何单元。如:

JMPL 03625;03625H→PC

4. 间接转移寻址

该寻址方式用于 JID 指令,转移的低 8 位地址可以存放在程序存储器的表格中获取,此时,低 8 位地址由累加器间接给出,如:

JID;((A))→PL

VIS 指令也是间接转移寻址方式,它是一种中断向量的间接转移,即:(VU)→PU,(VL)→PL。

§ 4.2 新型 COP8 单片机指令系统

4.2.1 新型 COP8 单片机指令种类

新型 COP8 单片机共有算术类、转跳类、取数/交换类、逻辑类、累加器位操作类、堆栈控制类、存储器位操作类、条件控制类和无操作类等几类指令。

1. 算术类指令

算术类指令有:

- 加法:ADD
- 进位加:ADC

- 减法:SUB
- 借位减:SUBC
- 加 1:INC
- 减 1:DEC
- 十进制调整:DCOR
- 累加器清除:CLR
- 进位位置位:SC
- 进位位复位:RC

2. 转移控制类指令

转移控制类指令有:

- 相对转移:JP
- 绝对转移:JMP
- 长绝对转移:JMPL
- 间接转移:JID
- 子程序转移:JSR
- 长子程序转移:JSRL
- 子程序返回:RET
- 子程序返回跳转:RETSK
- 中断返回:RETI
- 软件陷阱中断:INTR
- 向量中断选择:VIS

3. 取数及交换类指令

取数及交换类指令有:

- 取数:LD
- 累加器间接取数:LAID
- 交换:X

4. 逻辑类指令

逻辑类指令有:

- 逻辑与:AND
- 逻辑或:OR
- 异或:XOR

5. 累加器位操作类指令

累加器位操作类指令有:

- 经进位位循环右移:RRC
- 经进位位循环左移:RLC
- 累加器半字节交换:SWAP

6. 堆栈控制类指令

堆栈控制类指令有:

- 进栈:PUSH

- 出栈:POP

7. 存储器位操作类指令

存储器位操作类指令有:

- 位置位:SBIT
- 位复位:RBIT
- 挂起位复位:RPND

8. 条件控制类指令

条件控制类指令有:

- 若相等:IFEQ
- 若不相等:IFNE
- 若大于则:IFGT
- 若进位:IFC
- 若无进位:IFNC
- 若位置位:IFBIT
- 若 B 不相等:IFBNE
- 若逻辑与立即数为零:ANDSZ
- 寄存器减 1,若为零跳:DRSZ

9. 无操作类指令

无操作类指令只有:

- 无操作:NOP

4.2.2 新型 COP8 单片机指令系统

表 4-1 是 COP8 单片机指令系统表,表 4-2 是操作码表。

表 4-1 COP8 单片机指令系统表

指 令		功 能	寄存器操作
ADD	A, Meml	加	$A \leftarrow A + \text{Meml}$
ADC	A, Meml	带进位加	$A \leftarrow A + \text{Meml} + C, C \leftarrow \text{进位}, C \leftarrow \text{半进位}$
SUBC	A, Meml	带进位减	$A \leftarrow A - \text{Meml} + C, C \rightarrow \text{进位}, C \leftarrow \text{半进位}$
AND	A, Meml	逻辑与	$A \leftarrow A \text{“与”} \text{Meml}$
ANDSZ	A, Imm	逻辑与立即数, 如果为零跳跃	如果 $(A \text{ 逻辑与立即数}) = 0$, 跳跃下一指令
OR	A, Meml	逻辑或	$A \leftarrow A \text{“或”} \text{Meml}$
XOR	A, Meml	逻辑异或	$A \leftarrow A \text{“异或”} \text{Meml}$
IFEQ	MD, Imm	如果相等	比较 MD 和 Imm, 如 $MD = \text{Imm}$, 继续执行下一条指令
IFEQ	A, Meml	如果相等	比较 A 和 Meml, 如 $A = \text{Meml}$, 继续执行下一条指令
IFNE	A, Meml	如果不相等	比较 A 和 Meml, 如 $A \neq \text{Meml}$, 继续执行下一条指令
IFGT	A, Meml	如果大于	比较 A 和 Meml, 如 $A > \text{Meml}$, 继续执行下一条指令
IFBNE	#	如 B 不等	如果 B 的低 4 位不等于 Imm, 继续执行下一条指令
DRSZ	Reg	寄存器减 1, 如为 0 跳跃	$\text{Reg} \leftarrow \text{Reg} - 1$, 如 $\text{Reg} = 0$, 则跳跃
SBIT	#, Meml	某位置位	指令中的立即数所表示的位号的一位置“1”(位号#为 0~7)
RBIT	#, Meml	某位复位	指令中的立即数所表示的位号的一位复位(位号#为 0~7)
IFBIT	#, Meml	测试某位	如该位为 1(真), 则执行一下条指令
RPND		复位挂起标志	复位软件中断挂起标志
X	A, Meml	A 与存储器内容交换	$A \leftrightarrow \text{Meml}$
X	X, [X]	A 与存储器[X]交换	$A \leftrightarrow [X]$
LD	A, Meml	存储器内容写入 A	$A \leftarrow \text{Meml}$
LD	A, [X]	存储器[X]写入 A	$A \leftarrow [X]$
LD	B, Imm	立即数写入 B	$B \leftarrow \text{Imm}$
LD	Meml,	立即数写入存储器	$\text{Meml} \leftarrow \text{Imm}$
LD	Imm	立即数写入寄存器存储器	$\text{Reg} \leftarrow \text{Imm}$
X	A, (B $\pm$ 1)	A 与(B)交换后, B $\pm$ 1	$A \leftrightarrow (B) (B \leftarrow B \pm 1)$
X	A, (X $\pm$ 1)	A 与(X)交换后, X $\pm$ 1	$A \leftrightarrow (X) (X \leftarrow X \pm 1)$
LD	A, (B $\pm$ 1)	(B)写入 A 后, B $\pm$ 1	$A \leftarrow (B) (B \leftarrow B \pm 1)$
LD	A, (X $\pm$ 1)	(X)写入 A 后, X $\pm$ 1	$A \leftarrow (X) (X \leftarrow X \pm 1)$
LD	(B $\pm$ 1), Imm	立即数写入, B $\pm$ 1	$(B) \leftarrow \text{Imm} (B \leftarrow B \pm 1)$

(续表)

指 令		功 能	寄存器操作
CLR	A	清 A	$A \leftarrow 0$
INC	A	A 增 1	$A \leftarrow A+1$
DEC	A	A 减 1	$A \leftarrow A-1$
LAND		ROM 间接写入 A	$A \leftarrow \text{ROM}(\text{PU}, A)$
DCOR	A	对 A 作十进制校正	$A \leftarrow A$ 的 BCD 校正, 此指令接在 DC 或 SUB 之后
RRC	A	通过 C 右移	$C \leftrightarrow A7 \leftrightarrow \dots \leftrightarrow A0 \leftrightarrow C$
RLC	A	通过 C 左移	$C \leftarrow A7 \leftarrow \dots \leftarrow A0 \leftarrow C$
SWAP	A	A 半字节互换	$A7 \dots A4 \leftrightarrow A3 \dots A0$
SC		C 置位	$C \leftarrow 1, \text{HC} \leftarrow 1$
RC		C 复位	$C \leftarrow 0, \text{HC} \leftarrow 0$
IFC		测试 C 如为 1(真), 继续	如果 C 为真, 继续执行下一行指令
IFNC		测试 C 如不为 1(真), 继续	如果 C 非真, 继续执行下一行指令
POP	A	弹出堆栈到 A	$\text{SP} \leftarrow \text{SP}+1, A \leftarrow [\text{SP}]$
PUSH	A	压入 A 到堆栈	$[\text{SP}] \leftarrow A, \text{SP} \leftarrow \text{SP}-1$
VIS		指向中断服务程序矢量	$\text{PU} \leftarrow [\text{VU}], \text{PL} \leftarrow [\text{VL}]$
JMPL	Addr.	无条件长跳	$\text{PC} \leftarrow ii (ii=15 \text{ 位}, 0K \sim 32K)$
JMP	Addr.	无条件跳转	$\text{PC}9 \dots \text{PC}0 \leftarrow i (i=12 \text{ 位})$
JP	Disp.	相对短跳	$\text{PC} \leftarrow \text{PC}+r, r \text{ 为 } -31 \sim +32, \text{ 但 } r \neq 1$
JSRL	Addr.	长跳至子程序	$[\text{SP}] \leftarrow \text{PL}, (\text{SP}-1) \leftarrow \text{PU}, \text{SP}-2, \text{PC} \leftarrow ii$
JSR	Addr.	跳至子程序	$[\text{SP}] \leftarrow \text{PL}, (\text{SP}-1) \leftarrow \text{PU}, \text{SP}-2, \text{PC}9 \dots \text{PC}0 \leftarrow i$
JID		间接跳转	$\text{PL} \leftarrow \text{ROM}(\text{PU}, A)$
RET		从子程序返回	$\text{SP}+2, \text{PL} \leftarrow [\text{SP}], \text{PU} \leftarrow [\text{SP}-1]$
RETSK		返回且跳跃	$\text{SP}+2, \text{PL} \leftarrow [\text{SP}], \text{PU} \leftarrow [\text{SP}-1]$
RETI		从中断返回	$\text{SP}+2, \text{PL} \leftarrow [\text{SP}], \text{PU} \leftarrow [\text{SP}-1], \text{GIE} \leftarrow 1$
INTR		产生一次中断	$[\text{SP}] \leftarrow \text{PL}, (\text{SP}-1) \leftarrow \text{PU}, \text{SP}-2, \text{PC} \leftarrow \text{OFF}$
NOP		空操作	$\text{PC} \leftarrow \text{PC}+1$

表 4-2 操作码表

沿 X 轴方向为高半字节,沿 Y 轴方向为低半字节。

F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
JP-15	JP-31	LD 0F0, #i	DRSZ 0F0	RRCA	RC	ADC A, #1	ADC A, [B]	IFBIT 0, [B]	ANDSZ A, #i	LD B, 0F	IFBNE 0	JSR 0000-00FF	JMP 0000-00FF	JP+17	INTR 0
JP-14	JP-30	LD 0F1, #i	DRSZ 0F1	*	SC	SUBC A, #i	SUB A, [B]	IFBIT 1, [B]	*	LD B, 0E	IFBNE 1	JSR 0100-01FF	10100-01FF	JP+18	JP+2 1
JP-13	JP-29	LD 0F2, #i	DRSZ 0F2	XA, [X+]	XA, [B+]	IFEQ A, #i	IFEQ A, [B]	IFBIT 2, [B]	*	LD B, 0D	IFBNE 2	JSR 0200-02FF	JMP 0200-02FF	JP+19	JP+3 2
JP-12	JP-28	LD 0F3, #i	DRSZ 0F3	XA, [X-]	XA, [B-]	IFGT A, #i	IFGT A, [B]	IFBIT 3, [B]	*	LD B, 0C	IFBNE 3	JSR 0300-03FF	JMP 0300-03FF	JP+20	JP+4 3
JP-11	JP-27	LD 0F4, #i	DRSZ 0F4	VIS	LAID	ADD A, #i	ADD A, [B]	IFBIT 4, [B]	CLRA	LD B, 0B	IFBNE 4	JSR 0400-04FF	JMP 0400-04FF	JP+21	JP+5 4
JP-10	JP-26	LD 0F5, #i	DRSZ 0F5	RPND	LID	ADD A, #i	AND A, [B]	IFBIT 5, [B]	SWAPA	LD B, 0A	IFBNE 5	JSR 0500-05FF	JMP 0500-05FF	JP+22	JP+6 5
JP-9	JP-25	LD 0F6, #i	DRSZ 0F6	XA, [X]	XA, [B]	XOR A, #i	XOR A, [B]	IFBIT 6, [B]	DCORA	LD B, 9	IFBNE 6	JSR 0600-06FF	JMP 0600-06FF	JP+23	JP+7 6
JP-8	JP-24	LD 0F7, #i	DRSZ 0F7	*	*	OR A, #i	OR A, [B]	IFBIT 7, [B]	PUSHA	LD B, 8	IFBNE 7	JSR 0700-07FF	JMP 0700-07FF	JP+24	JP+8 7
JP-7	JP-23	LD 0F8, #i	DRSZ 0F8	NOP	RLCA	LD A, #i	IFC	SBIT 0, [B]	RBIT 0, [B]	LD B, 7	IFBNE 8	JSR 0800-08FF	JMP 0800-08FF	JP+25	JP+9 8
JP-6	JP-22	LD 0F9, #i	DRSZ 0F9	IFNE A, [B]	IFEQ Md, #i	IFNE A, #i	IFNC	SBIT 1, [B]	RBIT 1, [B]	LD B, 6	IFBNE 9	JSR 0900-09FF	JMP 0900-09FF	JP+26	JP+10 9
JP-5	JP-21	LD 0FA, #i	DRSZ 0FA	LD A, [X+]	LD A, [B+]	LD, [B+] #i	INCA	SBIT 2, [B]	RBIT 2, [B]	LD B, 5	IFBNE 0A	JSR 0A00-0AFF	JMP 0A00-0AFF	JP+27	JP+11 A
JP-4	JP-20	LD 0FB, #i	DRSZ 0FB	LD A, [X-]	LD A, [B-]	LD, [B-] #i	DECA	SBIT 3, [B]	RBIT 3, [B]	LD B, 4	IFBNE 0D	JSR 0B00-0BFF	JMP 0B00-0BFF	JP+28	JP+12 B
JP-3	JP-19	LD 0FC, #i	DRSZ 0FC	LD Md, #i	JMPL	X A, Md	POPA	SBIT 4, [B]	RBIT 4, [B]	LD B, 3	IFBNE 0C	JSR 0C00-0CFF	JMP 0C00-0CFF	JP+29	JP+13 C
JP-2	JP-18	LD 0FD, #i	DRSZ 0FD	DIR	JSRL	LD A, Md	RETSK	SBIT 5, [B]	RBIT 5, [B]	LD B, 2	IFBNE 0D	JSR 0D00-0DFF	JMP 0D00-0DFF	JP+30	JP+14 D
JP-1	JP-17	LD 0FE, #i	DRSZ 0FE	LD A, [X]	LD A, [B]	LD, [B] #i	RET	SBIT 6, [B]	RBIT 6, [B]	LD B, 1	IFBNE 0E	JSR 0E00-0EFF	JMP 0E00-0EFF	JP+31	JP+15 E
JP-0	JP-16	LD 0FF, #i	DRSZ 0FF	*	*	LD, B #i	RETI	SBIT 7, [B]	RBIT 7, [B]	LD B, 0	IFBNE 0F	JSR 0F00-0FFF	JMP 0F00-0FFF	JP+32	JP+16 F

其中: i 为立即数, Md 为直接寻址存储器单元, * 为未用的操作码。注: 操作码 60H 也是 IFBIT #i, A 的操作码。

COP8 单片机大多数指令为单字节指令,而大多数单字节指令执行时间为一个指令周期,表 4-3 至 4-6 是相应指令的字节/周期数表。

表 4-3 算术操作类指令字节/周期数

指令	[B]	直接寻址	立即寻址
ADD	1/1	3/4	2/2
ADC	1/1	3/4	2/2
SUBC	1/1	3/4	2/2
AND	1/1	3/4	2/2
OR	1/1	3/4	2/2
XOR	1/1	3/4	2/2
IFEQ	1/1	3/4	2/2
IFNE	1/1	3/4	2/2
IFGT	1/1	3/4	2/2
IFBNE	1/1		
DRSZ		1/3	
SBIT	1/1	3/4	
RBIT	1/1	3/4	
IFBIT	1/1	3/4	
RPND	1/1		

表 4-4 使用累加器 A 与进位 C 的指令字节/周期数表

指令	字节/周期
CLRA	1/1
INCA	1/1
DECA	1/1
LAI D	1/3
DCOR	1/1
RRCA	1/1
RLCA	1/1
SWAPA	1/1
SC	1/1
RC	1/1
IFC	1/1
IFNC	1/1
PUSHA	1/3
POPA	1/3
ANDSZ	2/2

表 4-5 控制转移指令字节/周期数

指令	字节/周期
JMPL	3/4
JMP	2/3
JP	1/3
JSRL	3/5
JSR	2/5
JID	1/3
VIS	1/5
RET	1/5
RETSK	1/5
RETI	1/5
INTR	1/7
NOP	1/1

表 4-6 存储器数据传送指令字节/周期数

指 令	寄存器间接寻址		直接寻址	立即寻址	寄存器间接寻址, 自动加 1 和减 1	
	[B]	[X]			[B+, B-]	[X+, X-]
XA,*	1/1	1/3	2/3		1/2	1/3
LD A,*	1/1	1/3	2/3	2/2	1/2	1/3
LD B,Imm				1/1 ^b		
LD B,Imm				2/2 ^c		
LD Mem1,Imm	2/2		3/3		2/2	
LD Reg,Imm			2/3			
IFEQ MD,Imm			3/3			

a. 由 B 或 X 或直接寻址的存储器单元

b. 如果 B<16

c. 如果 B>15

§ 4.3 新型 COP8 单片机程序设计举例

在 COP8 单片机 Data book 中 COP8 应用部分的 AN-579 COP800MathPak 中有加、减、乘、除各种算术运算,以及二进制和 BCD 运算及转换的例子。本节只举一些简单例子作介绍。

4.3.1 RAM 清除

RAM 清除的程序较简单,用 X 寄存器(0FCH)作计数器,B 寄存器作 RAM 指针,一般 128 字节 RAM,其真正片内 RAM 除 R0~R11 为 00~6FH 的 112 字节。另外,还要清除 0F0~0FBH 的 12 个通用寄存器及 X 寄存器外。故 RAM 需分两部分清除。

RAM 清除程序:

```
CLRAM: LD    0FC, #070    ;X 指针为计数器
        LD    B, #0       ;B 指针初始化
CLRAM2: LD    [B+], #0    ;写入 0,且 B 指针加 1
        DRSZ  0FC        ;计数器减 1
        JP    CLRAM2     ;做一半部 RAM 存储单元已清除,则跳
        LD    B, #0F0     ;指针 B 指向高一半部 RAM 存储地址
CLRAM3: LD    [B+], #0    ;清“0”
        IFBNE #0D        ;若 B=0FD,则跳
        JP    CLRAM3     ;若 B=0FD,则跳
        LD    B, #0       ;B 初始化为 0
```

4.3.2 二进制/BCD 码算术运算

二进制码加/减法可以利用 ADC/SUBC 指令来实现,下面是 4 字节的加/减法程序,两个操作数分别存放在 016~019H 单元及 024~027H 单元之中。通过 IFBNC #12 指令来判

断 4 字节加/减法结束(此外,B 寄存器为 028H,其低 4 位的值为 012H)。而 BCD 码加/减运算可以在二进制码加/减法基础上,增加 ADD A, #066 指令(BCD 减法不需要)及十进制调整指令 DCOR 即可。

二进制加法程序:

BINARY ADDITION

```

        LD    X, #16          ;无前导零,表示十进制
        LD    B, #24
        RC
LOOP:   LD    A, [X+]
        ADC   A, [B]
        X     A, [B+]
        IFBNE #12
        JP    LOOP
        IFC
        JP    OVFLOW          ;若有进位 C,则溢出

```

二进制减法程序:

BINARY SUBTRACTION

```

        LD    X, #010         ;前导零,表示十六进制
        LD    B, #018
        SC
LOOP:   LD    A, [X+]
        SUBC  A, [B]
        X     A, [B+]
        IFBNE #12
        JP    LOOP
        IFNC
        JP    NEGRSLT         ;若无进位位 C(借位),结果为负

```

BCD 码加法程序:

BCD ADDITION

```

        LD    X, #010         ;前导零,表示十六进制
        LD    B, #018
        RC
LOOP:   LD    A, [X+]
        ADD   A, #066          ;加十六进制 66
        ADC   A, [B]
        DCOR  A                ;十进制调整

```

```

X      A,[B+]
IFBNE  #12
JP     LOOP
IFC
JP     OVFLOW      ;若有进位位 C,则溢出

```

BCD 码减法程序:

BCD SUBTRACTION

```

LD      B, #16
LD      X, #24
SC
LOOP:   LD      A,[X+]
X      A,[B]
SUBC    A,[B]
DCOR    A
X      A,[B+]
IFBNE   #4
JP      LOOP
IFNC
JP      NEGRSLT    ;若无进位位(借位),结果负

```

下面是一个混合算术运算例子,在程序存储器中有 5 个连续字节的数据表的和(2 字节的 TOT)减去 SUM(也是 2 个字节)。假如,数据表位于 0401 单元,SUM 与 TOT 分别位于 1,0 存储单元及 3,2 存储单元。

混合算术运算程序:

```

MATH-   .SECT      MATH, RAM
MEM:    .DSB4                      ;常量说明
SUMLO =  MATHMEM
SUMHI =  MATHMEM+1
TOTLO =  MATHMEM+2
TOTHI =  MATHMEM+3
.SECT   CODE, ROM, ABS=0401
                                           ;ROM 表
.BYTE   102
.BYTE   41
.BYTE   31
.BYTE   26
.BYTE   5

```

			;执行加法和减法
ARITH1:	LD	X, #5	;设置 ROM 表指针
	LD	B, #SUMLO	;设置 SUM 指针
LOOP:	RC		;进位位复位
	LD	A, X	
	LAI	D	;从 ROM 读 DATA
	ADC	A, [B]	
	X	A, [B+]	
	CLR	A	
	ADC	A, [B]	
	X	A, [B-]	
	DRSZ	X	
	JP	LOOP	
	SC		
	LD	B, #2	;指向 TOTLO
LUP:	LD	A, [X+]	
	X	A, [B]	
	SUBC	A, [B]	
	X	A, [B+]	
	IFBNE	#4	
	JP	LUP	
	RET		

4.3.3 二进制码乘法/除法运算

下面 16×16 的双字节乘法,从低位端开始乘法为基础,首先清除双字节乘积的高位端,接着向右移动一位双字节乘积。从低位端移出的位代表乘数的低位。若该位为“1”,则被乘数加到乘积的高位端。移位次数等于乘数的位数加1(附加额外一次移位是合理必要的)。

16×16 乘法程序:

MULTIPLY(16×16)子程序

被乘数:[1,0]乘数:[3,2]

积:[5,4,3,2]

	.SECT	MEMCNT,REG
CNTR:	.DSB	1
	.SECT	CODE,ROM
MULT:	LD	CNTR, #17
	LD	B, #4

```

        LD      [B+], #0
        LD      [B], #0
        LD      X, #0
        RC
MLOOP:  LD      A, [B]
        RRC     A
        X       A, [B-]
        LD      A, [B]
        RRC     A
        X       A, [B-]
        LD      A, [B]
        RRC     A
        X       A, [B-]
        LD      A, [B]
        RRC     A
        X       A, [B]
        LD      B, #5
        IFNC
        JP      TEST
        RC
        LD      B, #4
        LD      A, [X+]
        ADC     A, [B]
        X       A, [B+]
        LD      A, [X-]
        ADC     A, [B]
        X       A, [B]
TEST:   DRSZ    CNTR
        JP      MLOOP
        RET

```

下面是 $16 \div 16$ 除法, 16 位商随着 16 位余数而产生。被除数左移至已清零的 16 位测试窗口, 该窗口除数进行测试——减法。若不产生高端借法, 则进行真正减法, 结果存回测试窗口; 同时, 商为 1 插入被除数窗口低端, 以表示真正减法。然后整个被除数和测试窗口左移 1 位(商紧跟着被除数)。

程序的 LSHFT 部分, 4 次左移(LD, ADC 和 X 指令)而没有用循环程序, 为的是可提高执行速度。

16 $\div$ 16 除法程序:

DIVIDE(16÷16)子程序

被除数 [3,2]

除数 [1,0]

商 [3,2]

余数 [5,4]

```

                .SECT    MEMCNT,REG
CNTR:           .DSB 1
                .SECT    CODE,ROM
DIV:            LD       CNTR, #16
                LD       B, #5
                LD       [B-], #0
                LD       [B], #0
                LD       X, #4
LSHFT:          RC
                LD       B, #2
                LD       A, [B]
                ADC      A, [B]
                X        A, [B+]
                LD       A, [B]
                ADC      A, [B]
                X        A, [B+]
                LD       A, [B]
                ADC      A, [B]
                X        A, [B+]
                LD       A, [B]
                ADC      A, [B]
                X        A, [B+]
TSUBT:          SC
                LD       B, #0
                LD       A, [X+]
                SUBC     A, [B]
                LD       B, #1
                LD       A, [X-]
                SUBC     A, [B]
                IFNC
                JP       TEST
SUBT:           LD       B, #0

```



```
LD      A,[X]
SUBC    A,[B]
X        A,[X+]
LD      B,#1
LD      A,[X]
SUBC    A,[B]
X        A,[X-]
LD      B,#2
SBIT    0,[B]
TEST:   DRSZ    CNTR
        JMP     LSHFT
        RET
```

第五章 COP8 单片机产品

COP8 单片机可以分为基本型(BASIC Family)、特色型(Feature Family)和新型(高集成型)等几大类。

§ 5.1 新型 COP8 单片机产品

新型(高集成型)COP8 单片机有 SA 和 SG 两大类。新型 COP8 单片机采用新型的半导体工艺,除了与原有的 COP8 单片机的功能完全兼容外,还具有:

- 低价格 OTP
- 片内振荡器电路
- 上电复位功能
- 8 字节用户存储区域(EPROM)

另外,COP8ACC×单片机,它也采用了新型 COP8 单片机的命名型号,实际上是 COP884EK 单片机的简易型,并具有低价格 OTP 的特点。因而,也放在本节进行介绍。

5.1.1 COP8SA×单片机(Apollo)产品

COP8SA×类单片机,又称为 Apollo 单片机。其最大特点是价格便宜、使用方便;不但有 OTP 型(因为其价格低,已接近于 MaskROM 型单片机的价格),也有更低价格的掩膜型单片机。COP8SA×单片机的主要特性如下:

- 低价格 OTP
- 与 COP8 单片机相同的 CPU 核
- 片内 RAM:64~128 字节
- 片内 EPROM:1K~4K 字节
- 8 字节用户区域(EPROM)
- 16 位定时/计数器 T1
 - PWM 工作方式
 - 外部事件计数器工作方式
 - 输入捕获工作方式
- 空闲(IDLE)定时器 T0
- WATCHDOG 定时器
- MICROWIRE/PLUS 微总线(与 SPI 兼容)
- I/O 端口:12~40 引脚
 - 软件选择:三态输出、推挽输出、弱上拉输入和高阻输入等几种形式

- I/O 端口 G 和 L 输入具有施密特电路
- 多至 12 根大电流输出引脚
- 多输入唤醒逻辑
- 中断源: 8 个
- 工作电压: 2.7~5.5V
- 工作电流: 6mA(10MHz), HALT 工作方式 $< 4\mu\text{A}$
- 节电工作方式: HALT 工作方式和 IDLE 工作方式
- 片内振荡器电路
- 上电复位功能

图 5-1 是 COP8SAX7 单片机的系统结构图。图 5-2 是 COP8SA $\times$ 单片机的引脚图。

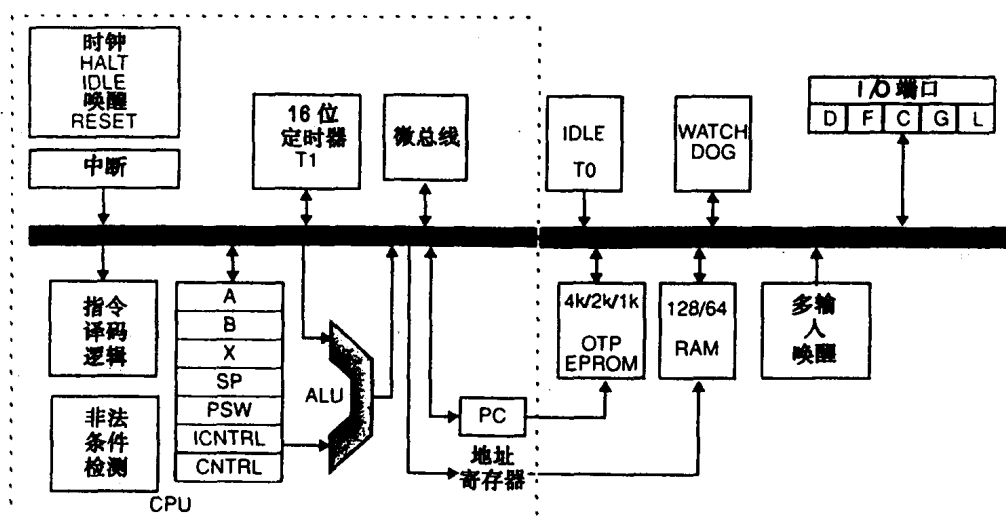


图 5-1 COP8SA $\times$ 7 单片机系统结构图

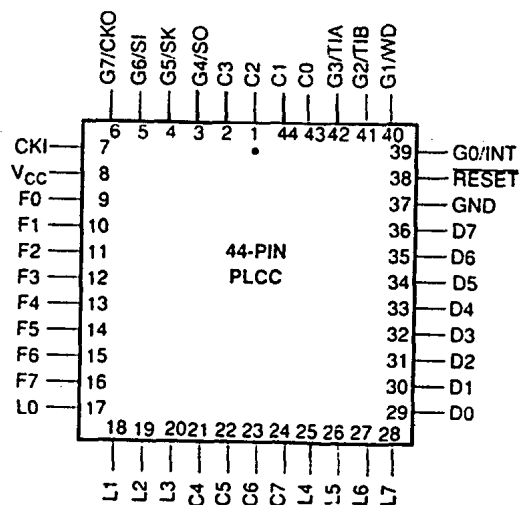
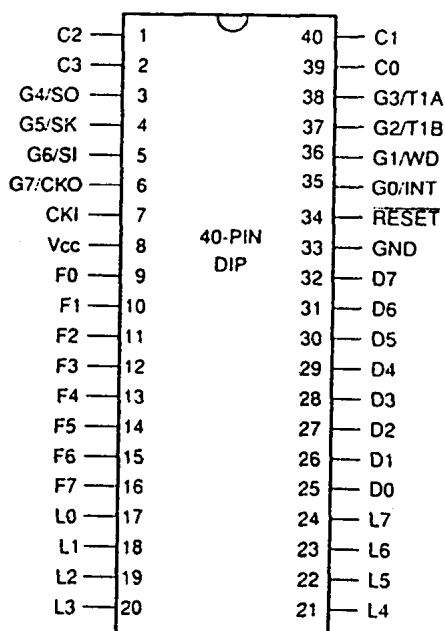
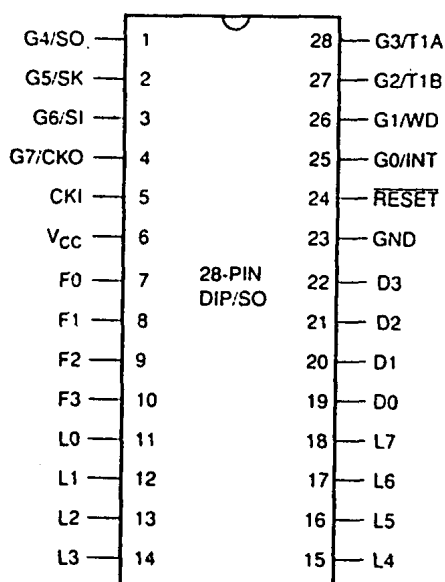
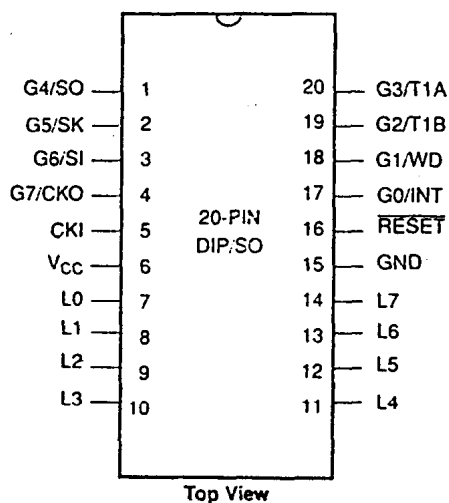
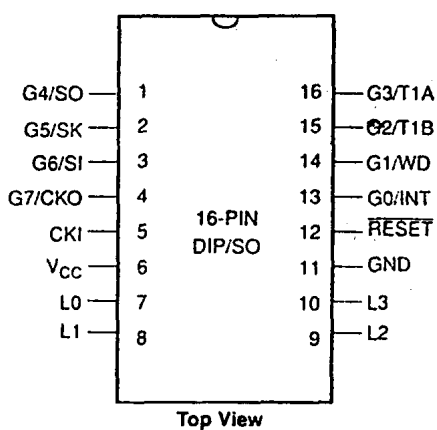


图 5-2 COP8SA 单片机引脚图

表 5-1 是 COP8SA 类单片机分类表,表 5-2 是 COP8SA 类单片机产品表。

表 5-1 COP8SA×单片机

单片机	EPROM	RAM	封装和 I/O	
			封装形式	I/O 引脚
COP8SAC7	4K	128	20DIP/SO	16
			28DIP/SO	24
			40DIP	36
			44PLCC	40
COP8SAB7	2K	128	20DIP/SO	16
			28DIP/SO	24
COP8SAA7	1K	64	16DIP/SO	12
			20DIP/SO	16
			28DIP/SO	24

表 5-2 COP8SA×类单片机产品

	1K EPROM		2K EPROM		4K EPROM		4K EPROM	
温度	型号	封装	型号	封装	型号	封装	带窗口 型号	封装
0℃至+70℃	COP8SAA716M9	16M						
	COP8SAA720M9	20M	COP8SAB720M9	20M	COP8SAC720M9	20M		
	COP8SAA728M9	28M	COP8SAB728M9	28M	COP8SAC728M9	28M		
	COP8SAA716N9	16N						
	COP8SAA720N9	20N	COP8SAB720N9	20N	COP8SAC720N9	20N	COP8SAC720Q3	20Q
	COP8SAA728N9	28N	COP8SAB728N9	28N	COP8SAC728N9	28N	COP8SAC728Q3	28Q
					COP8SAC740N9	40N	COP8SAC740Q3	40Q
-40℃至+85℃					COP8SAC744V9	44V	COP8SAC744J3	44J
	COP8SAA716M8	16M						
	COP8SAA720M8	20M	COP8SAB720M8	20M	COP8SAC720M8	20M		
	COP8SAA728M8	28M	COP8SAB728M8	28M	COP8SAC728M8	28M		
	COP8SAA716N8	16N						
	COP8SAA720N8	20N	COP8SAB720N8	20N	COP8SAC720N8	20N		
	COP8SAA728N8	28N	COP8SAB728N8	28N	COP8SAC728N8	28N		
-40℃至+125℃					COP8SAC740N8	40N		
					COP8SAC744V8	44V		
					COP8SAC720M7	20M		
					COP8SAC728M7	28M		
					COP8SAC720N7	20N		
					COP8SAC728N7	28N		
					COP8SAC740N7	40N		
					COP8SAC744V7	44V		

5.1.2 COP8SG×单片机产品

新型 COP8SG×类单片机是 COP8 单片机系列的 G 型单片机的低价格 OTP 化产品，并提供掩膜服务。其既有原有 COP8 单片机中的 G 型单片机的功能,又有新型 COP8 单片机的特点。COP8SG×单片机的主要特性如下：

- 低价格 OTP
- 与 COP8 单片机相同的 CPU 核
- 片内 RAM:256 字节/512 字节(COP8SGE7/COP8SGR7)
- 片内 EPROM:8K 字节/32K 字节(COP8SGE7/COP8SGR7)
- 8 字节用户区域(EPROM)
- 三个 16 位定时/计时器 T1~T3
 - PWM 工作方式
 - 外部事件计数器工作方式
 - 输入捕获工作方式
- 空闲(IDLE)定时器 T0
- WATCHDOG 定时器
- MICROWIRE/PLUS 微总线(与 SPI 兼容)
- I/O 端口:24~40 引脚
 - 软件选择:三态输出、推挽输出、弱上拉输入和高阻输入等几种形式
 - I/O 端口 G 和 L 输入具有施密特电路
 - 8 根大电流输出
- 多输入唤醒逻辑
- 全双工 UART
- 两个模拟比较器
- 中断源:14 个
- 工作电压:2.7~5.5V
- 节电工作方式:HALT 工作方式和 IDLE 工作方式
- 片内振荡器电路
- 上电复位功能

图 5-3 是 COP8SGE7/SGR7 单片机系统结构图,图 5-4 是 COP8SGE7/SGR7 单片机的引脚图。

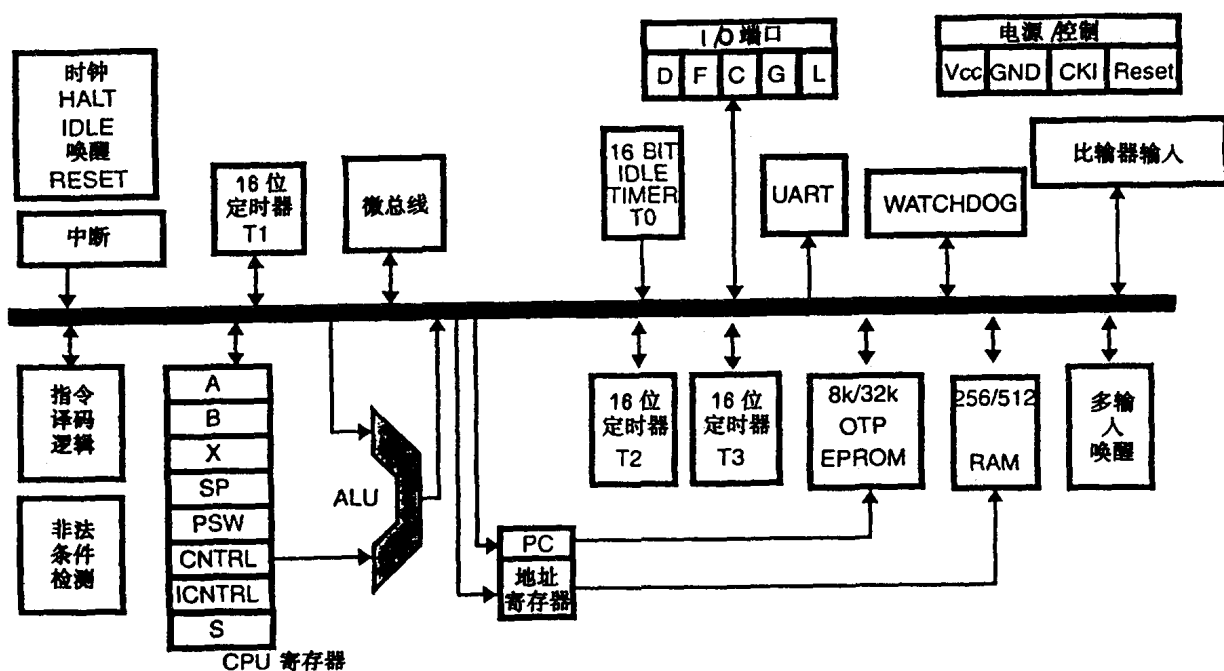


图 5-3 COP8SGE7/SGR7 单片机系统结构图

COP8SG 类单片机目前只有 COP8SGE7 和 COP8SGR7 两种,它们的功能实际上可与原有的 COP888EG 和 COP888RG 单片机相当。COP8SGE7/COP8SGR7 单片机有四种引脚封装形式,表 5-3 列出了这几种引脚封装形式。

表 5-3 COP8SGE7/COP8SGR7 单片机引脚功能

端口	I/O	功能	28 引脚	40 引脚	44 引脚 PLCC	44 引脚 PQFP
L0	I/O	MIWU	11	17	17	11
L1	I/O	MIWU 或 CKX	12	18	18	12
L2	I/O	MIWU 或 TDX	13	19	19	13
L3	I/O	MIWU 或 RDX	14	20	20	14
L4	I/O	MIWU 或 T2A	15	21	25	15
L5	I/O	MIWU 或 T2B	16	22	26	16
L6	I/O	MIWU 或 T3A	17	23	27	17
L7	I/O	MIWU 或 T3B	18	24	28	18
G0	I/O	INT	25	35	39	3
G1	I/O	WDOUT	26	36	40	34
G2	I/O	T1B	27	37	41	3
G3	I/O	T1A	28	38	42	3
G4	I/O	SO	1	3	3	41
G5	I/O	SK	2	4	4	42
G6	I	SI	3	5	5	43
G7	I	CKO	4	6	6	44
D0	O		19	25	29	23
D1	O		20	26	30	24
D2	O		21	27	31	25
D3	O		22	28	32	26
D4	O			29	33	27
D5	O			30	34	28
D6	O			31	35	29
D7	O			32	36	30

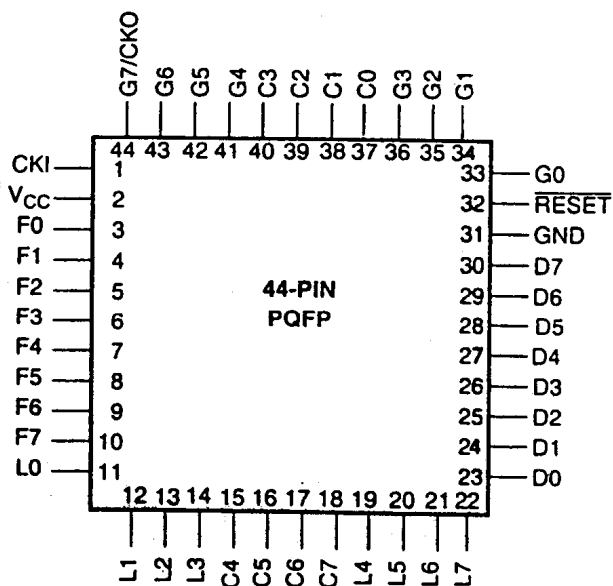
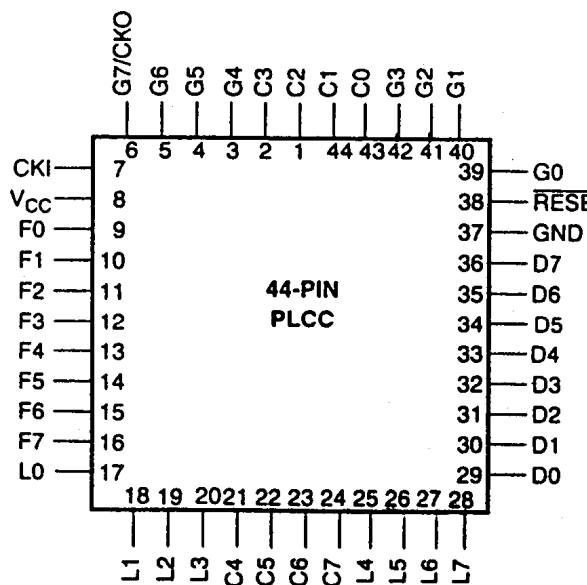
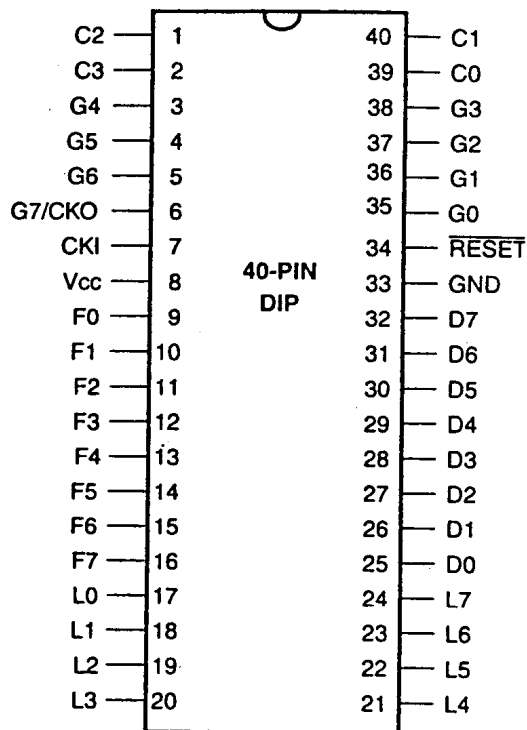
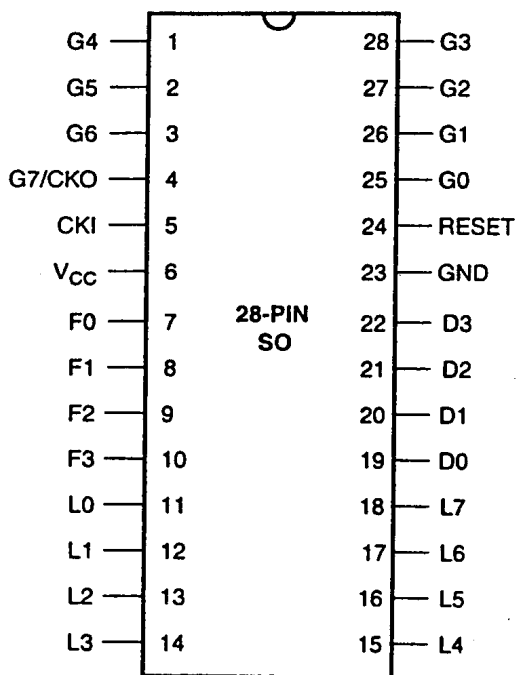


图 5-4 COP8SGE7/SGR7 单片机引脚图

(续表)

端口	I/O	功能	28 引脚	40 引脚	44 引脚 PLCC	44 引脚 PQFP
F0	I/O	COMP1IN- COMP1IN+ COMP1OUT COMP2IN- COMP2IN+ COMP2OUT	7	9	9	3
F1	I/O		8	10	10	4
F2	I/O		9	11	11	5
F3	I/O		10	12	12	6
F4	I/O			13	13	7
F5	I/O			14	14	8
F6	I/O			15	15	9
F7	I/O			16	16	10
C0	I/O			39	43	37
C1	I/O			40	44	38
C2	I/O			1	1	39
C3	I/O			2	2	40
C4	I/O			21	21	15
C5	I/O				22	16
C6	I/O				23	17
C7	I/O				24	18
VCC	I		6	8	8	2
GND			23	33	37	31
CKI			5	7	7	1
RESET			24	34	38	32

5.1.3 COP8SACC7 单片机产品

COP8SACC7 单片机是原有 COP884EK 单片机的 OTP 化简易型。其 CPU 核仍相同；片内 RAM 从 256 字节减为 128 字节，片内 EPROM 从 8K 字节减为 4K 字节；16 位定时/计数器从 3 个减为 1 个，中断源从 12 个减至 9 个，高分辨率单积分式 16 位 A/D 转换有的从 6 个通道减为 4 个。这样，可以使 OTP 型的 COP8ACC7 单片机的价格大大降低。COP8ACC7 单片机主要特性如下：

- 与 COP8 单片机相同的 CPU 核
- 片内 RAM:128 字节
- 片内 EPROM:4K 字节
- 16 位定时/计数器 T1
 - PWM 工作方式
 - 外部事件计数器工作方式
 - 输入捕获工作方式
- 空闲(IDLE)定时器 T0
- WATCHDOG 定时器
- MICROWIRE/PLUS 微总线(与 SPI 兼容)
- I/O 端口：
 - 软件选择：三态输出、推挽输出、弱上拉输入和高阻输入等几种形式
 - I/O 端口 G 和 L 输入具有施密特电路
 - 8 根大电流输出引脚
- 多输入唤醒逻辑
- 16 位 A/D 转换:4~6 通道
- 中断源:9 个

• 工作电压:2.7~5.5V

• 节电工作方式:HALT 工作方式和 IDLE 工作方式

图 5-5 是 COP8ACC7 单片机系统结构图,图 5-6 是相应单片机的引脚图。

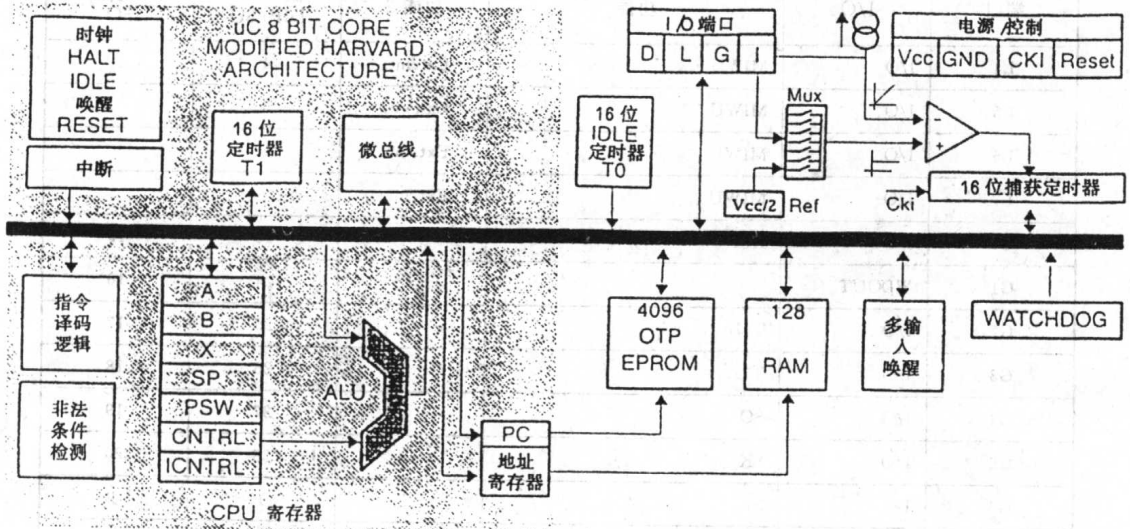


图 5-5 COP8ACC7 单片机系统结构图

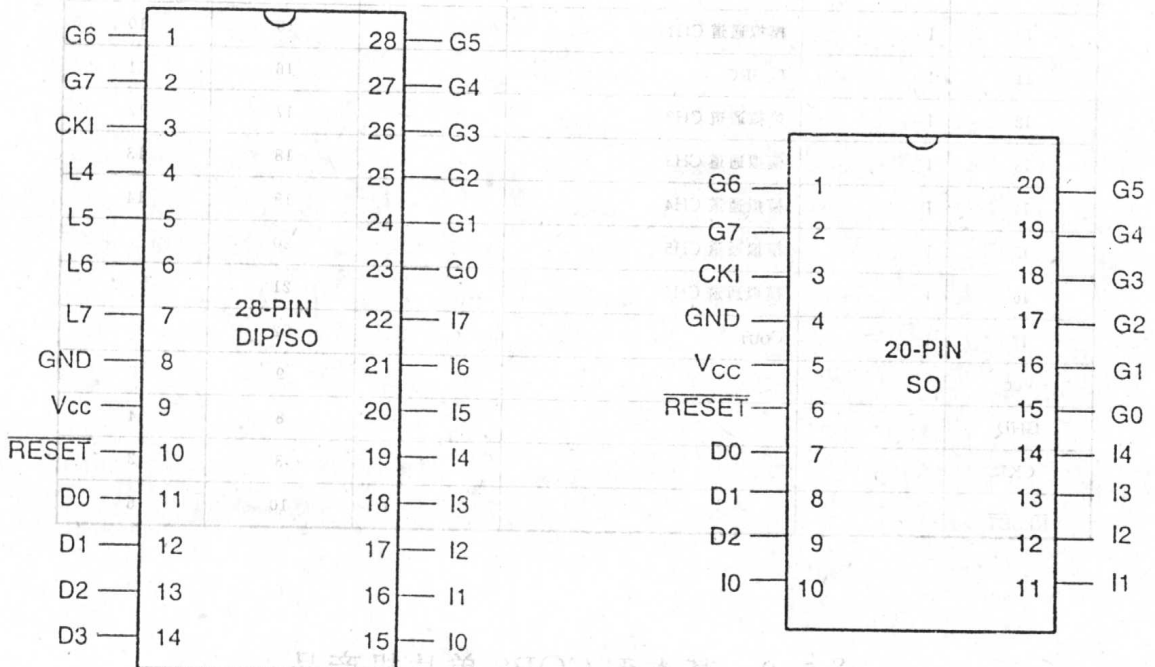


图 5-6 COP8ACC7 单片机引脚图

COP8ACC7 单片机有相应的 MaskROM 型单片机 COP8ACC5。COP8ACC7 单片机目前只有 20 与 28 引脚两种封装,表 5-4 列出了相应引脚封装形式。

表 5-4 COP8ACC7 单片机引脚功能

端口	I/O	功能	功能 2	28 引脚 DIP/SO	20 引脚 SO
L4	I/O	MIWU	Ext. int.	4	
L5	I/O	MIWU	Ext. int.	5	
L6	I/O	MIWU	Ext. int.	6	
L7	I/O	MIWU	Ext. int.	7	
G0	I/O	INT		23	15
G1	WDOUT			24	16
G2	I/O	T1B		25	17
G3	I/O	T1A		26	18
G4	I/O	SO		27	19
G5	I/O	SK		28	20
G6	I	SI		1	1
G7	I/CKO	HALT 再启动		2	2
D0	O			11	7
D1	O			12	8
D2	O			13	9
D3	O			14	
I0	I	模拟通道 CH1		15	10
I1	I	I\~SRC		16	11
I2	I	模拟通道 CH2		17	12
I3	I	模拟通道 CH3		18	13
I4	I	模拟通道 CH4		19	14
I5	I	模拟通道 CH5		20	
I6	I	模拟通道 CH6		21	
I7	I	COUT		22	
Vcc				9	5
GND				8	4
CKI				3	3
RESET				10	6

§ 5.2 基本型 COP8 单片机产品

基本(Basic)型 COP8 单片机除了具有传统的特性型 COP8 单片机的基本功能外,只是

简化了一些结构,降低了价格,并还增加了一些专用的特性(如调整器、模拟比较器等)。

基本型 COP8 单片机大致上可分为 C 类和 CJ 类。其中, COP912 单片机其价格十分低廉,具有很大的竞争力。

5.2.1 基本类单片机产品(一)——COP912C/COP912CH

COP912/COP912CH 是十分便宜的单片机,其片内 RAM 和 ROM 都很小,定时器 1 个及中断源 3 个。COP912 的工作电压为 2.3~4.0V,指令执行时间为 $2\mu\text{s}$; COP912CH 的工作电压为 4.0~5.5V,指令执行时间为 $2.5\mu\text{s}$ 。图 5-7 是 COP912 单片机系统结构图。

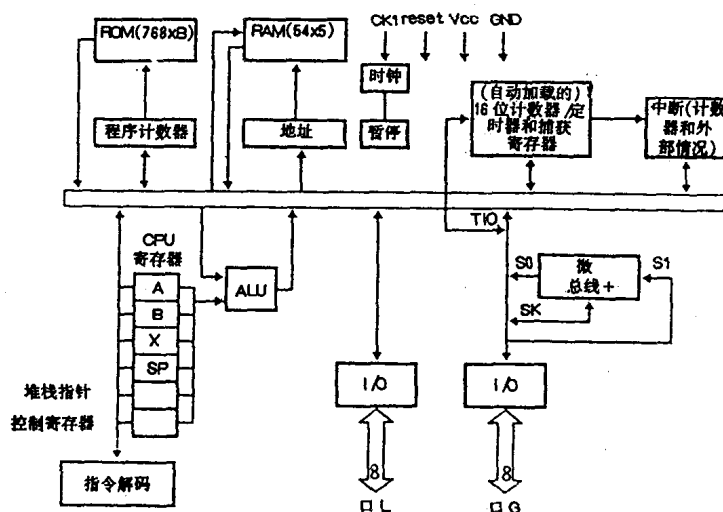


图 5-7 COP912 单片机系统结构图

主要特性:

- 低价格 8 位单片机
- 指令执行时间: $2\mu\text{s}$ (COP912)
 $2.5\mu\text{s}$ (COP912CH)
- 片内 RAM: 64 字节
- 片内 ROM: 768 字节
- 一个 16 位定时/计数器
 - 16 位自动重装寄存器的定时器
 - 16 位外部事件计数器
 - 16 位输入捕获功能
- 三级中断源
 - 外部中断
 - 定时器中断或捕获中断
 - 软件中断
- I/O 端口: 15 根引脚
 - 软件选择: 三态输出、推挽输出和弱上拉输入
 - 端口 G 为施密特触发器输入

- MICROWIRE/PLUS 微总线接口
- HALT 工作方式

COP912/COP912CH 应用：

- 电子键和开关
- 遥控器
- 定时器
- 报警器
- 小型工业控制装置
- 低成本的从属控制器
- 数字温度计
- 小型家用电器
- 玩具和游戏机

COP912C/COP912CH 单片机引脚如图 5-8 所示。

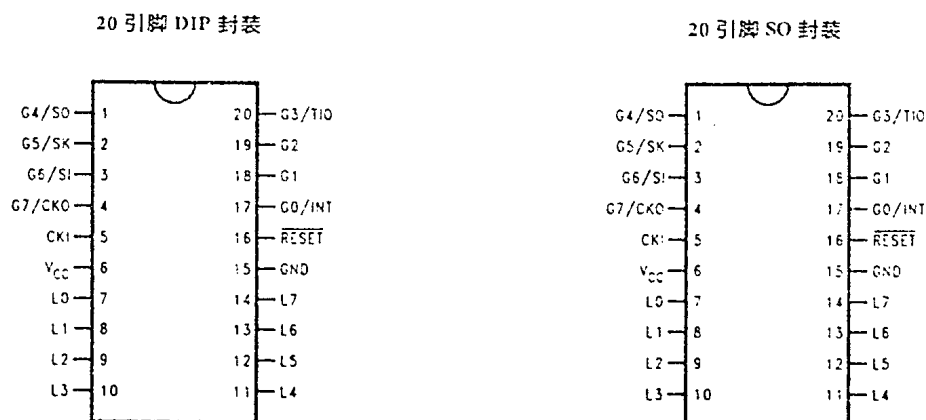


图 5-8 COP912C/COP912CH 单片机引脚图

5.2.2 基本类单片机产品(二)——COP620C/COP622C/COP640C/COP642C COP820C/COP822C/COP840C/COP842C COP920C/COP922C/COP940C/COP942C COP680C/COP681C/COP880C/COP881C COP980C/COP981C

COP620C/COP622C/COP640C/COP642C/COP820C/COP822C/COP840C/
COP842C/COP920C/COP922C/COP940C/COP942C/COP680C/COP681C/COP880C/
COP881C/COP980C/COP981C 单片机也是基本类 COP 单片机,它们与上述 COP912C 单
片机的特性基本相同,只是片内 RAM 和 ROM 增大了。图 5-9 是它们系统结构图。

主要特性：

- 低价格 8 位单片机

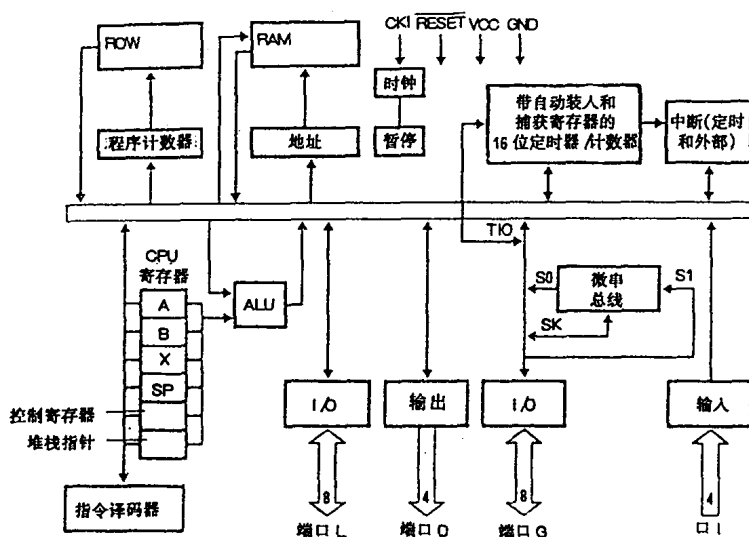


图 5-9 基本类单片机系统结构图

- 指令执行时间： $1\mu\text{s}$ （时钟为 10MHz）
- 片内 RAM：64~128 字节
- 片内 ROM：1K~4K 字节
- 一个 16 位定时/计数器
 - 16 位自动重载寄存器
 - 16 位外部事件计数器
 - 16 位输入捕获功能
- 多级中断源
 - 复位总清
 - 外部中断
 - 定时器中断或捕获中断
 - 软件中断
- I/O 端口：15~35 根引脚
 - 软件选择：三态输出、推挽输出和弱上拉输入
 - I/O 端口 G 为施密特输入
- MICROWIRE/PLUS 微总线
- HALT 工作方式（工作电流： $1\mu\text{A}$ ）

图 5-10 是 COP620C/622C/640C/642C/820C/822C/840C/842C/920C/922C/940C/942C 单片机的引脚图。

图 5-11 是 COP680C/681C/880C/881C/980C/P81C 单片机引脚图。

双列直插式封装

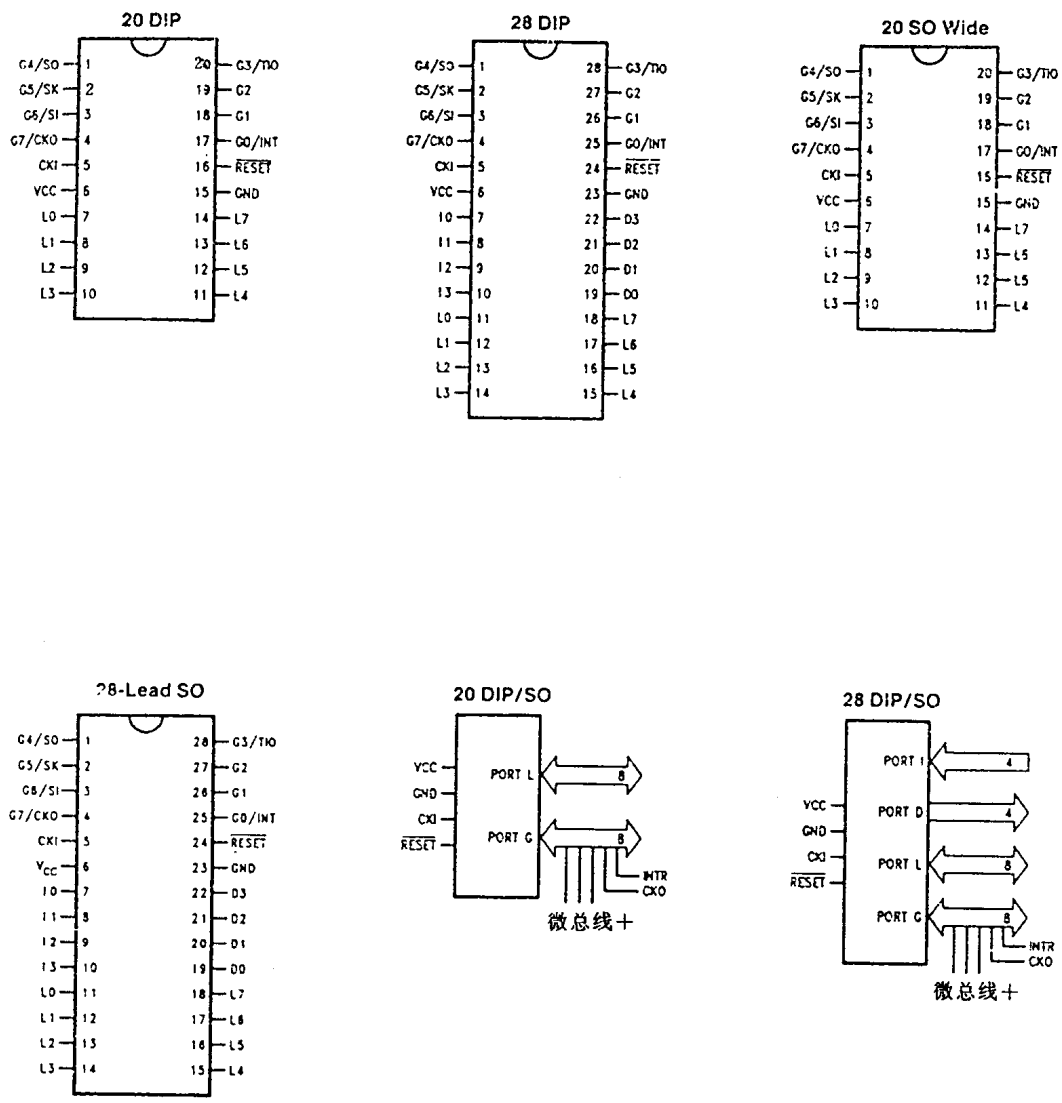
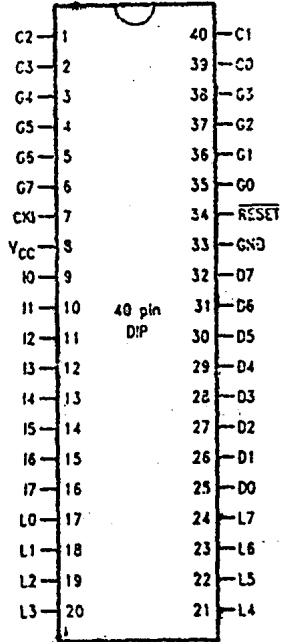
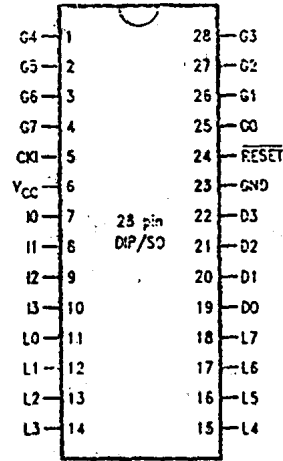


图 5-10 COP62×C/64×C/82×C/84×C/92×C/94×C 单片机引脚图

双列直插封装



DIP 封装(N)和 28 引脚宽 SO 封装(WM)



PLCC 封装

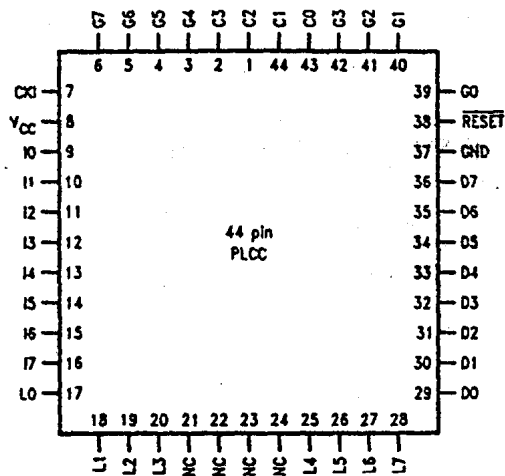


图 5-11 COP68×C/88×C/98×C 单片机引脚图

5.2.3 基本类单片机产品(三)——COP8780/COP8781/COP8782

COP8780/COP8781/COP8782 单片机是 EPROM 或 OTP 型基本类单片机,其性能同上述单片机相同,只是片内 ROM 为 EPROM 或 OTP 型,且有加密功能;故可作为上述基本类单片机的仿真单片机。

主要特性:

- 低价格 EPROM 或 OTP 单片机
- 片内 EPROM 或 OTP:4K 字节
- 片内 RAM:128 或 64 字节(用户可进行配置)
- EPROM 加密功能
- 其余的性能与基本类单片机相同

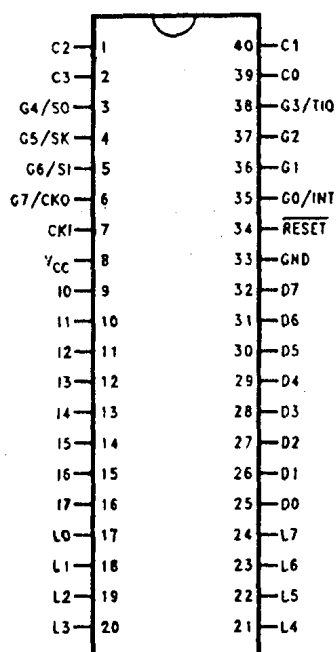
图 5-12 是 COP8780C/COP8781C/COP8782C 单片机的引脚图。

5.2.4 CJ 类单片机产品——COP820CJ/COP822CJ/COP823CJ

COP820CJ/COP822CJ/COP823CJ 单片机具有调制器、高速 PWM、掉电保护及多路唤醒的独特功能,此外,还具有基本类单片机的一些通用功能,图 5-13 是系统结构图。

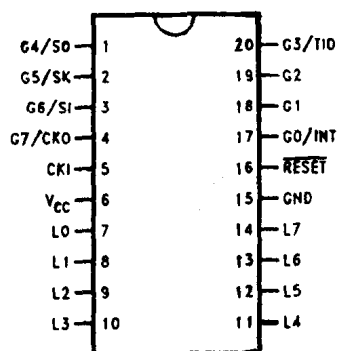
主要特性:

- 价格低廉 8 位单片机
- 片内 ROM:1K 字节
- 片内 RAM:64 字节
- 指令执行时间:1 μ s
- 一个 16 位定时/计数器
 - 16 位自动重装寄存器的定时器
 - 16 位外部事件计数器
 - 16 位输入捕获功能
- WATCHDOG 定时器
- 模拟比较器
- 调制器/定时器(IR 传输的高速 PWM 定时器)
- I/O 端口:11~23 根
 - 多输入唤醒功能(I/O 端口 L)
 - 4 根具有 15mA 大电流驱动
 - I/O 端口 G 和 L 具有施密特输入
- 多级中断源
 - 复位总清
 - 外部中断
 - 定时器中断或捕获中断
 - 软件中断
- MICROWIRE/PLUS 微总线



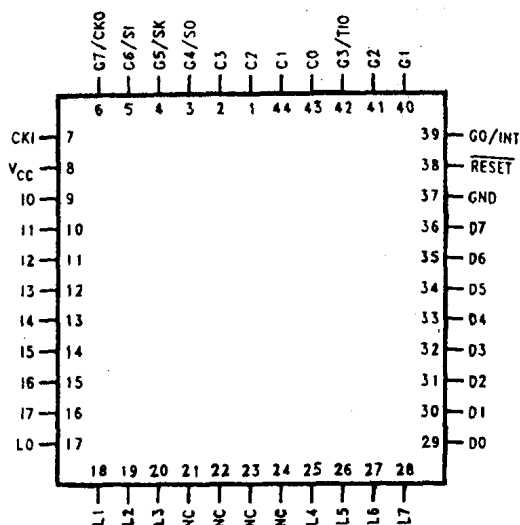
顶视

COP8780CN, COP8780CJ



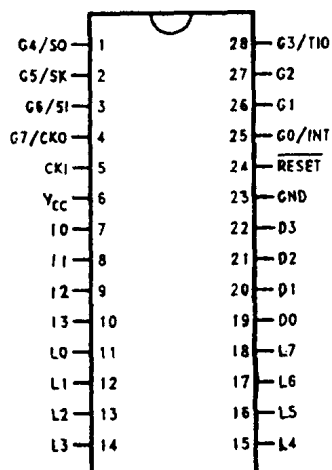
顶视

COP8782CN, COP8782CWM,
COP8782CJ



顶视

COP8780CV, COP8780CEL



顶视

COP8781CN, COP8781CWM
COP8781CJ

图 5-12 COP878×单片机引脚图

- 掉电保护
- HALT 工作方式
- 工作电压: 2.5~6V

图 5-14 是 COP820CJ/COP822CJ/COP823CJ 单片机引脚图。

其仿真用的 EPROM 型单片机为 COP87L22CJN/COP87L20CJN 及 COP87LCJM²/

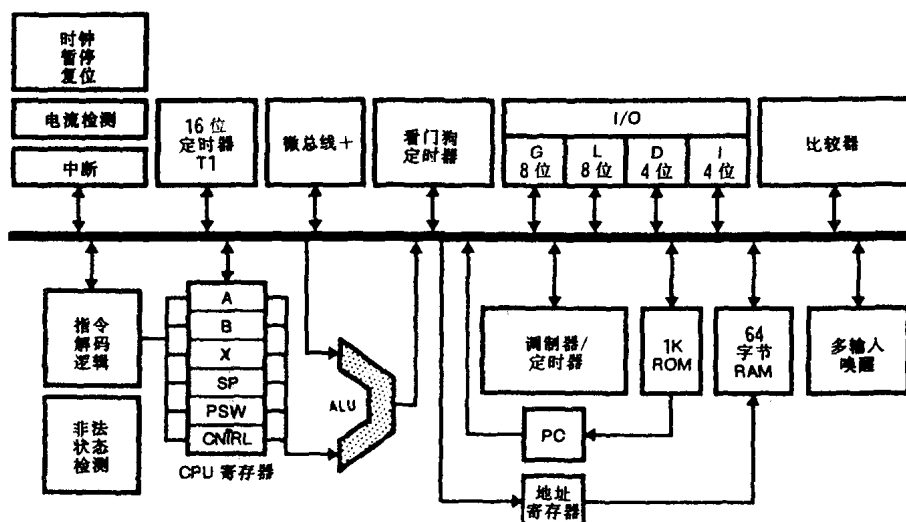


图 5-13 COP82×CJ 单片机系统结构图

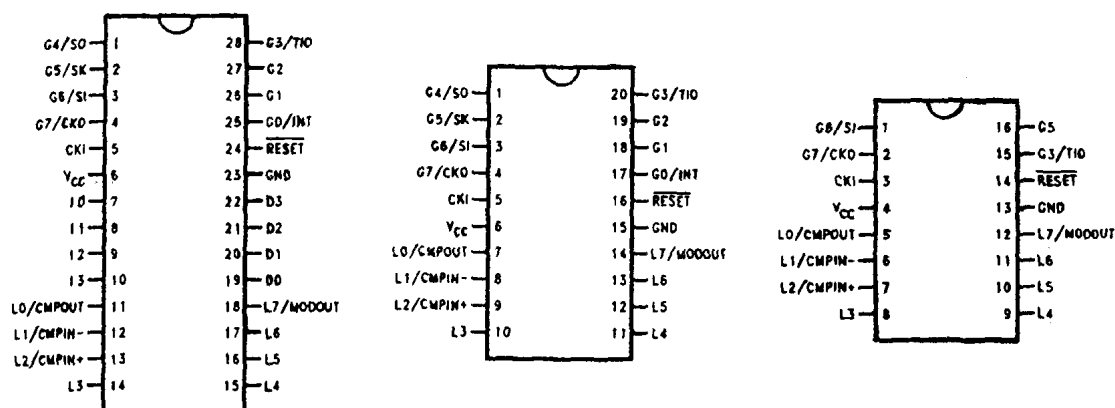


图 5-14 COP82×CJ 单片机引脚图

COP87L20CJM².

§ 5.3 特色型 COP8 单片机产品

特色 (Feature) 型 COP8 单片机是 NS 公司传统的单片机。有许多种产品:

- B类单片机: A/D 转换、CAN 及 UART 接口单片机
- C类单片机: CAN、高速 PWM、模拟比较器单片机
- D类单片机: A/D 转换、减弱 EMI 及大容量单片机

- F类单片机:D类单片机简化型
- G类单片机:UART及模拟比较器单片机
- H类单片机:乘/除部件、UART及大容量单片机
- K类单片机:单积分A/D和减弱EMI单片机
- L类单片机:特色性的基本型单片机
- S类单片机:G类单片机简化型
- W类单片机:乘/除部件、脉冲发生器及UART单片机

5.3.1 COP8 B类单片机产品——COP888EB/COP889EB COP688EB/COP689EB

COP8 B类单片机的特点是:具有1个CAN(Control Area Network)、8通道8位A/D转换器、1个全双工的UART和SPI接口等功能。此外,还具有COP8单片机的基本功能。图5-15是COP888EB单片机的系统结构图。

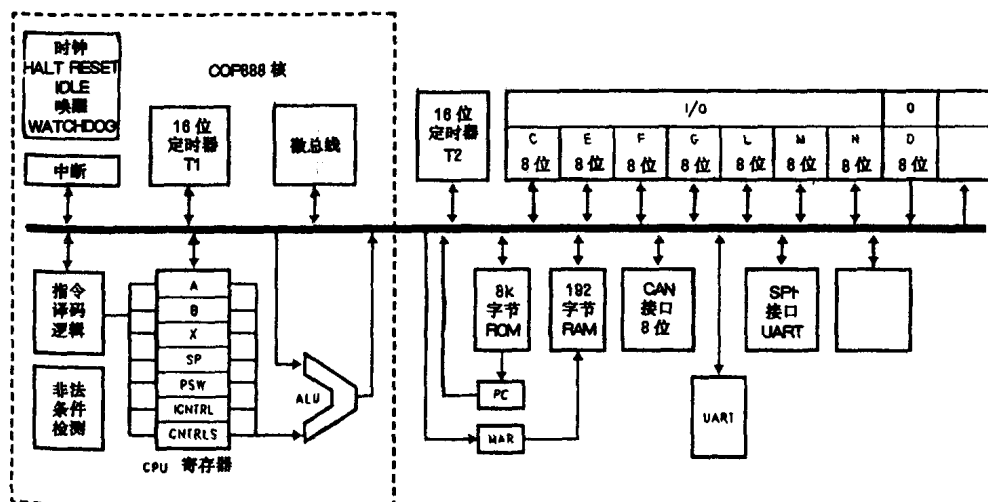


图 5-15 COP888EB 单片机系统结构图

主要特性:

- 片内RAM:192字节
- 片内ROM:8K字节
- 指令执行时间:1 μ s
- 两个16位定时/计数器T1和T2
 - PWM工作方式
 - 外部事件计数器
 - 输入捕获工作方式
- IDLE定时器T0
- WATCHDOG定时器和时钟监视器
- CAN接口
- SPI兼容的主/从式接口

- MICROWIRE/PLUS 微总线
- I/O 端口: 31~54 根 I/O
 - 软件选择: 三态输出、推挽输出、弱上拉输入、高阻抗输入
 - I/O 端口 G, L 和 M 具有施密特输入功能
 - 多输入唤醒 MIWU 功能
- 14 级中断源
 - 外部中断
 - IDLE 定时器中断
 - 4 个定时器中断 (T1 和 T2)
 - 微总线和 SPI 中断
 - 多输入唤醒中断
 - 软件陷阱中断
 - 3 个 CAN 中断
 - 2 个 UART 中断
- 工作电压: 4.5~5.5V
- 省电工作方式: HALT 和 IDLE 工作方式

图 5-16 是 COP8 B 类单片机的封装引脚图。其 OTP 型单片机为 COP87L88EB 和 COP87L89EB。

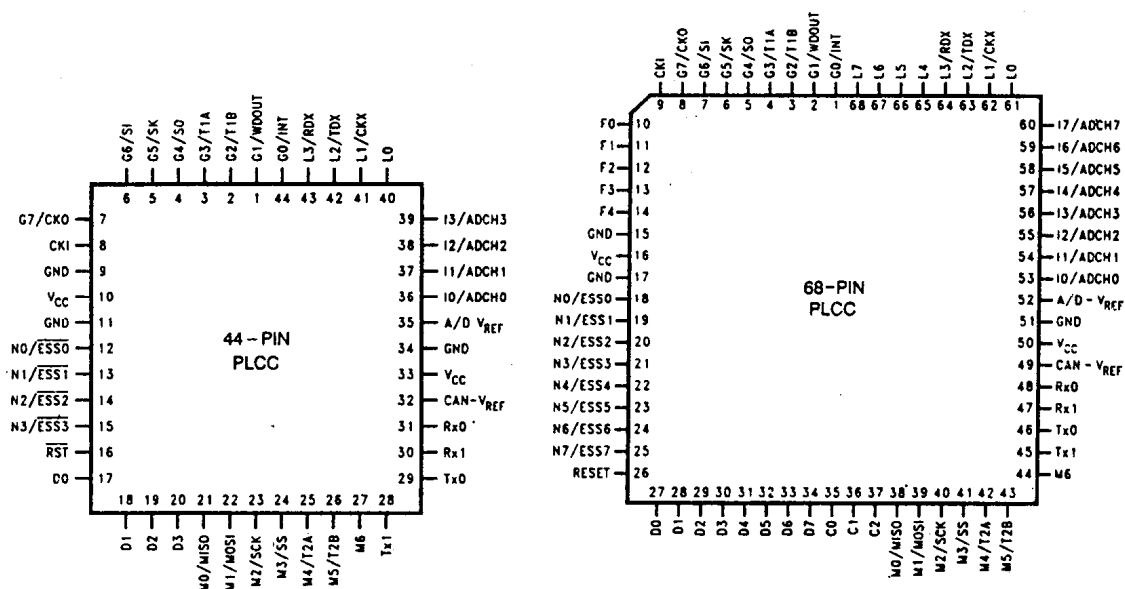


图 5-16 COP8 B 类单片机引脚图

5.3.2 COP8 C 类单片机产品——COP684BC/COP884BC

COP684BC/COP884BC 单片机的特性是:具有 1 个 CAN、马达控制定时器(8 位高速 PWM)、2 个模拟比较器、减弱 EMI 电磁干扰电路。此外,还具有 COP8 单片机的功能(但无 WATCHDOG 定时器)。特别适合于工业控制及马达控制。图 5-17 是 COP684BC/COP884BC 单片机系统结构图。

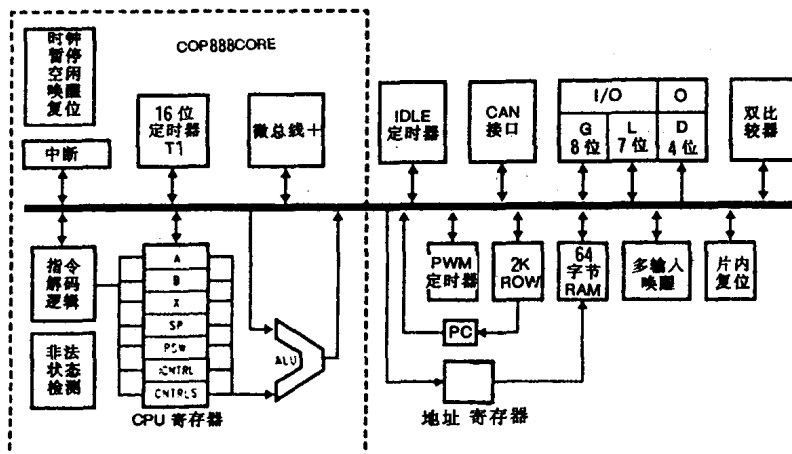


图 5-17 COP684BC/COP884BC 单片机系统结构图

主要特性:

- 片内 RAM:64 字节
- 片内 ROM:2K 字节
- 指令执行时间:1 μ s
- 一个 16 位定时/计数器 T1
 - PWM 工作方式
 - 外部事件计数器
 - 输入捕获工作方式
- IDLE 定时器 T0
- 8 位高速 PWM/频率监视定时器
- 两个模拟比较器
- MICROWIRE/PLUS 微总线
- CAN 接口
- I/O 端口:18 根 I/O
 - 软件选择:三态输出、推挽输出、弱上拉输入、高阻抗输入
 - I/O 端口 G 和 L 具有施密特输入
 - 多输入唤醒 MIWU 功能
- 11 级中断源
 - 外部中断
 - IDLE 定时器 T0 中断

- 2 个定时器 T1 中断
- 微总线中断
- 多输入唤醒中断
- 软件陷阱中断
- 高速 PWM 中断
- 3 个 CAN 中断
- 片内复位电路
- 减弱 EMI(电磁干扰)电路
- 工作电压:4.5~5.5V
- 省电工作方式:HALT 及 IDLE 工作方式

图 5-18 是 COP684BC 及 COP884BC 单片机的封装引脚图。其 OTP 型单片机为 COP87L84BC。

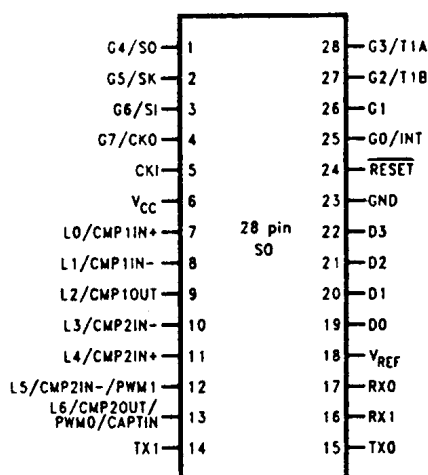


图 5-18 COP684BC/884BC 单片机引脚图

5.3.3 COP8 D 类单片机产品——COP888GD/COP988GD

COP8 D 类单片机的特点是:具有 8 通道 8 位 A/D 转换,减弱 EMI(电磁干扰)电路及较大片内 RAM 与片内 ROM。COP8 单片机系列中,B,F 和 D 类单片机都具有 8 通道 8 位 A/D 转换。D 类单片机与 B 类单片机相比,无全双工 UART、SPI 接口及 CAN 接口;但是,片内 RAM 与片内 ROM 比 B 类单片机大,16 位定时/计数器也增至 3 个。与 F 类单片机相比,功能比较接近,只不过片内 RAM 与片内 ROM 比较大、具有减弱 EMI(电磁干扰)电路及多了一个 16 位定时/计数器 T3。图 5-19 是 COP888GD 单片机系统结构图。

主要特性:

- 片内 RAM:256 字节
- 片内 ROM:16K 字节
- 指令执行时间:1 μ s
- PWM 工作方式

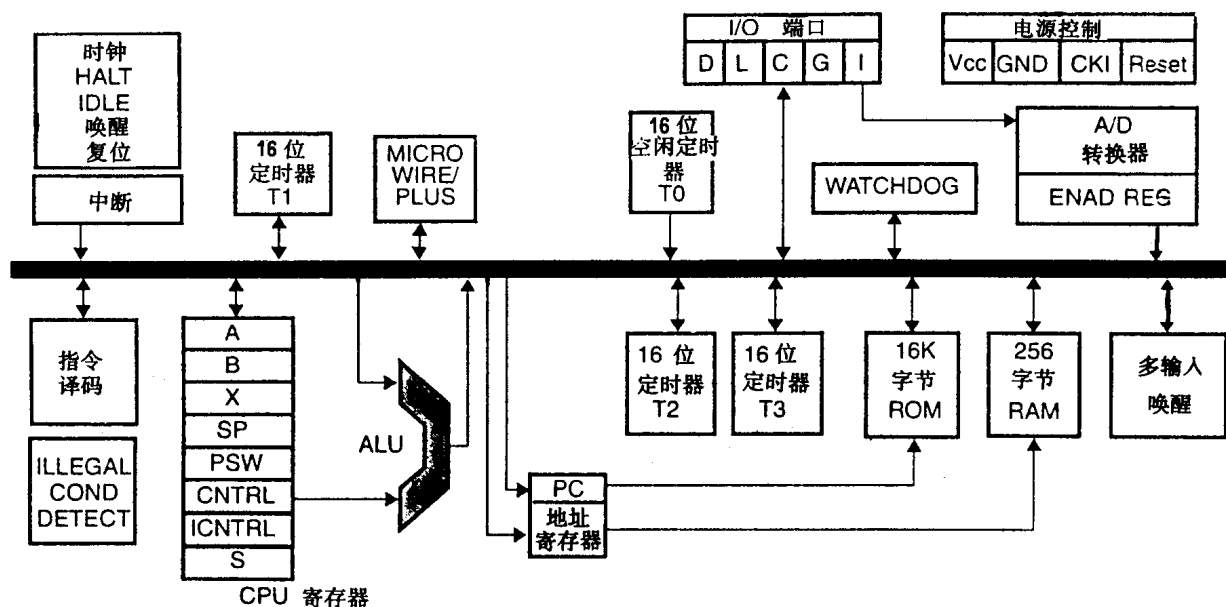


图 5-19 COP888GD 单片机系统结构图

- 外部事件计数器
- 输入捕获工作方式
- IDLE 定时器 T0
- WATCHDOG 定时器和时钟监视器
- MICROWIRE/PLUS 微总线
- 8 通道 8 位 A/D 转换
- I/O 端口:40 根 I/O
 - 软件选择:三态输出、推挽输出、弱上拉输入、高阻抗输入
 - I/O 端口 G 和 L 具有施密特输入
 - 多输入唤醒 MIWU 功能
- 12 级中断源
 - 外部中断
 - IDLE 定时器 T0 中断
 - 6 个定时/计数器中断 T1,T2 和 T3
 - 微总线中断
 - 多输入唤醒中断
 - 软件陷阱中断
 - 系统设置 VIS 中断
- 减弱 EMI(电磁干扰)电路
- 工作电压:2.5~5.5V
- 省电工作方式 HALT 及 IDLE 工作方式

图 5-20 是 COP888GD 与 COP988GD 单片机封装引脚图。其 OTP 型单片机为 COP87L88RD。

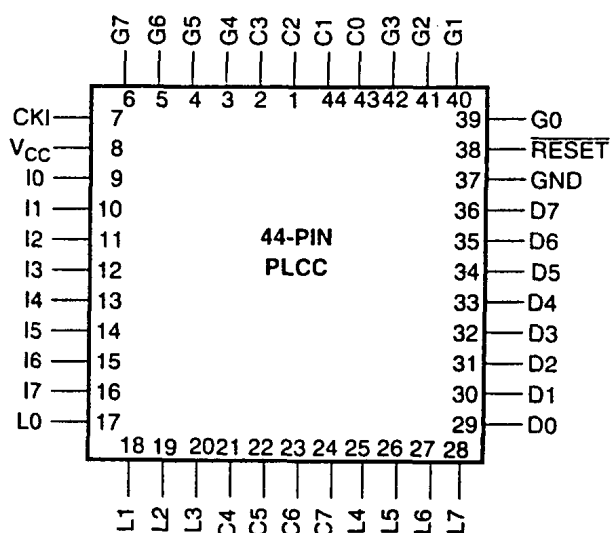


图 5-20 COP888GD/988GD 单片机引脚图

5.3.4 COP8 F 类单片机——COP988CF/COP984CF COP888CF/COP884CF

COP8 F 类单片机的特点是:具有 8 通道 8 位 A/D 转换。与上述的 D 类单片机相比,片内 RAM 与片内 ROM 比较小,16 位定时/计数器为 2 个,也无减弱 EMI(电磁干扰)电路。此外,具有 COP8 单片机的基本特点。图 5-21 是 COP888CF 单片机系统结构图。

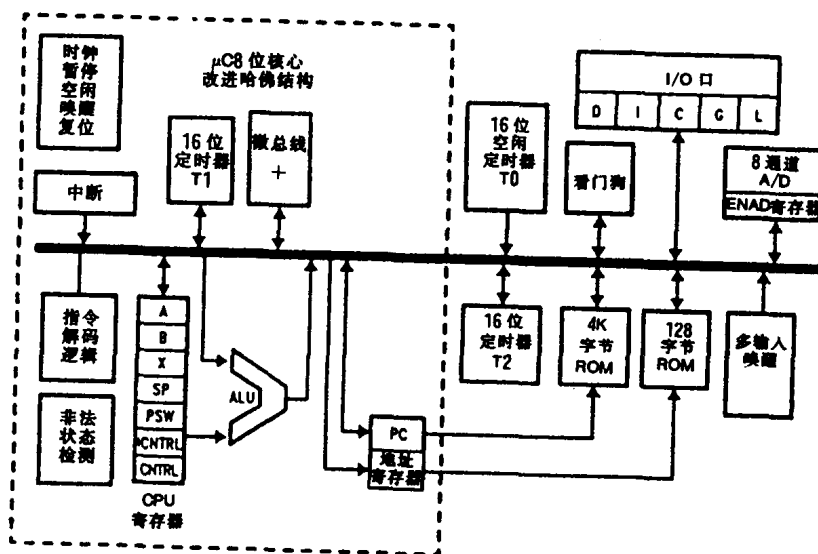


图 5-21 COP888CF 单片机系统结构图

主要特性:

- 片内 RAM:128 字节
- 片内 ROM:4K 字节
- 指令执行时间:1μs

- 两个 16 位定时/计数器 T1 和 T2
 - PWM 工作方式
 - 外部事件计数器
 - 输入捕获工作方式
- IDLE 定时器 T0
- WATHCHDOG 定时器和时钟监视器逻辑
- MICROWIRE/PLUS 微总线
- 8 通道 8 位逐步逼近 A/D 转换
- I/O 端口:23~37 根
 - 软件选择:三态输出、推挽输出、弱上拉输入、高阻抗输入
 - I/O 端口 G 和 L 具有施密特输入
 - 多输入唤醒 MIWU 功能
- 10 级中断源
 - 外部中断
 - IDLE 定时器 T0 中断
 - 4 个定时器 T1 和 T2 中断
 - 微总线中断
 - 多路输入唤醒中断
 - 软件陷阱中断
 - 系统设置 VIS 中断
- 工作电压:2.5~6V
- 省电工作方式:HALT 和 IDLE 工作方式

图 5-22 是 COP884CF/888CF 单片机封装引脚图,其 OTP 型单片机为 COP87L84CF/COP87L88CF 单片机。

5.3.5 COP8 G 类单片机产品——COP884CG/COP888CG

COP688EG/COP684EG/COP888EG

COP884EG/COP988EG/COP984EG

COP888GG/COP888HG/COP888KG

COP8 G 类单片机的明显特点是:具有全双工 UART 和两个模拟比较器。此外,16 位定时/计数器有 3 个,工作电压为 2.5~6V。图 5-23 是 COP888EG 单片机的系统结构图。

主要特性:

- 片内 RAM:192 字节(CG)
 - 256 字节(EG)
 - 512 字节(GG/HG/KG)
- 片内 ROM:4K 字节(CG)
 - 8K 字节(EG)
 - 16K/20K/24K 字节(GG/HG/KG)
- 指令执行时间:1 μ s

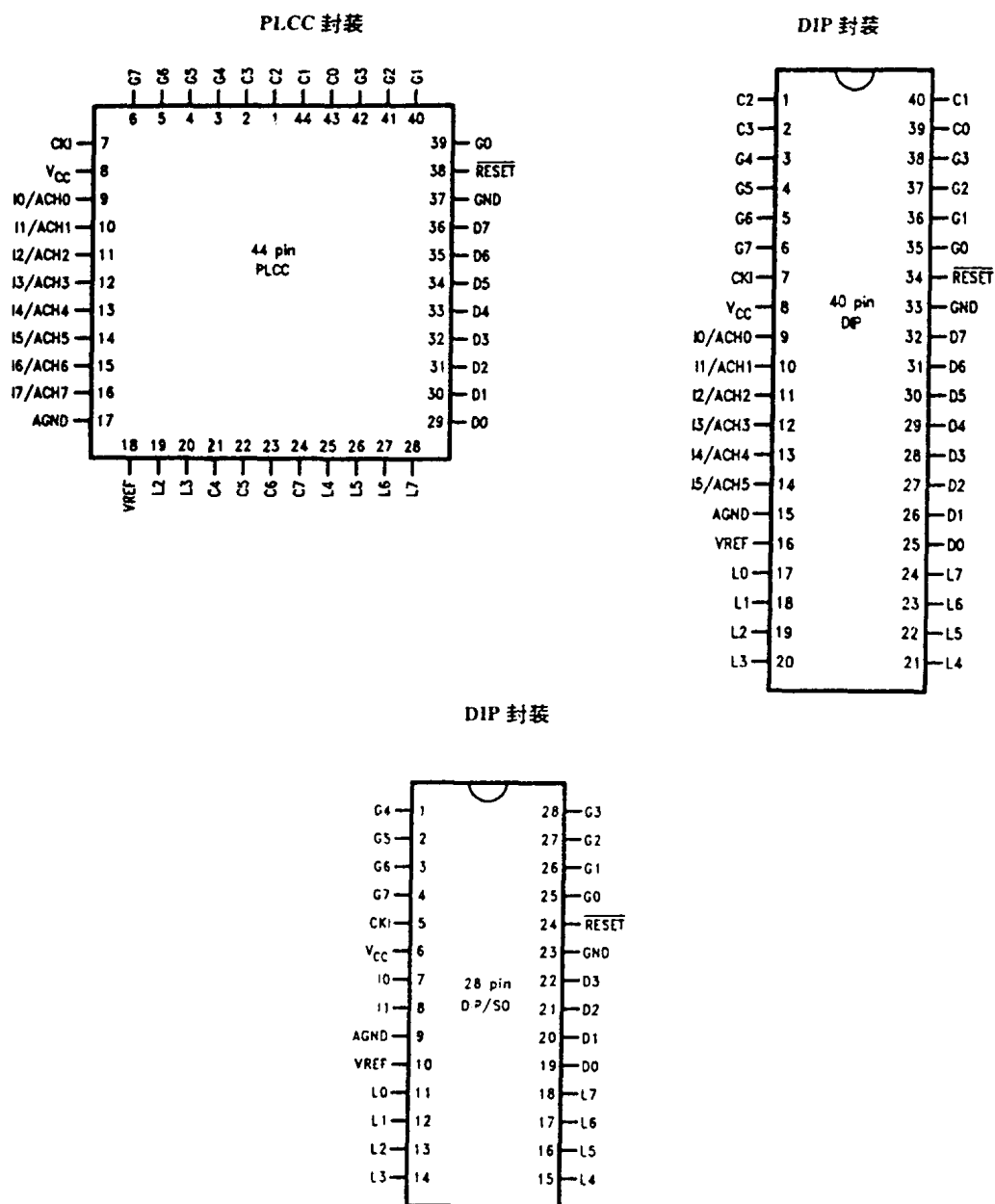


图 5-22 COP884CF/COP888CF 单片机引脚图

- 3 个 16 位定时/计数器 T1, T2 和 T3
 - PWM 工作方式
 - 外部事件计数器
 - 输入捕获工作方式
- IDLE 定时器 T0
- WATCHDOG 定时器和时钟监视器逻辑
- 全双工 UART
- 两个模拟比较器
- MICROWIRE/PLUE 微总线

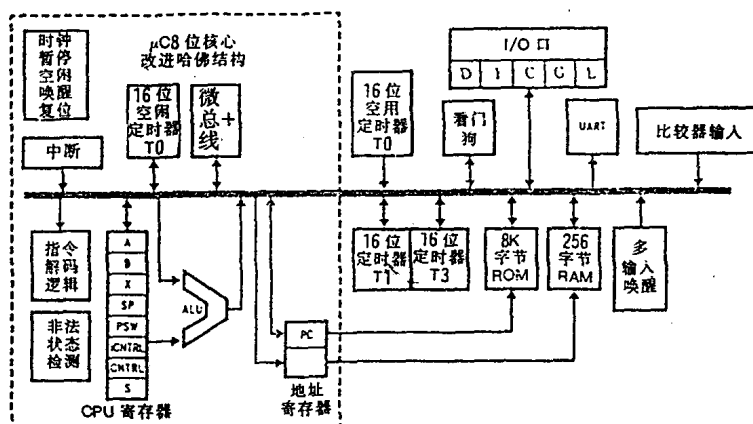


图 5-23 COP888EG 单片机系统结构图

- I/O 端口:23~39 极
 - 软件选择:三态输出、推挽输出、弱上拉输入、高阻抗输入
 - I/O 端口 G 和 L 具有施密特输入功能
 - 多输入唤醒 MIWU 功能
- 14 级中断源
 - 外部中断
 - IDLE 定时器 T0 中断
 - 6 个定时器 T1,T2 和 T3 中断
 - 微总线中断
 - 多输入唤醒中断
 - 两个 UART 中断
 - 软件陷阱中断
 - 系统设置 VIS 中断
- 工作电压:2.5~6V
- 省电工作方式:HALT 和 IDLE 工作方式

图 5-24 是 EG 单片机的封装引脚图,OTP 型单片机有 COP87L84EG/COP87L88EG, COP87L88GG,COP87L88HG 和 COP87L88KG 单片机。

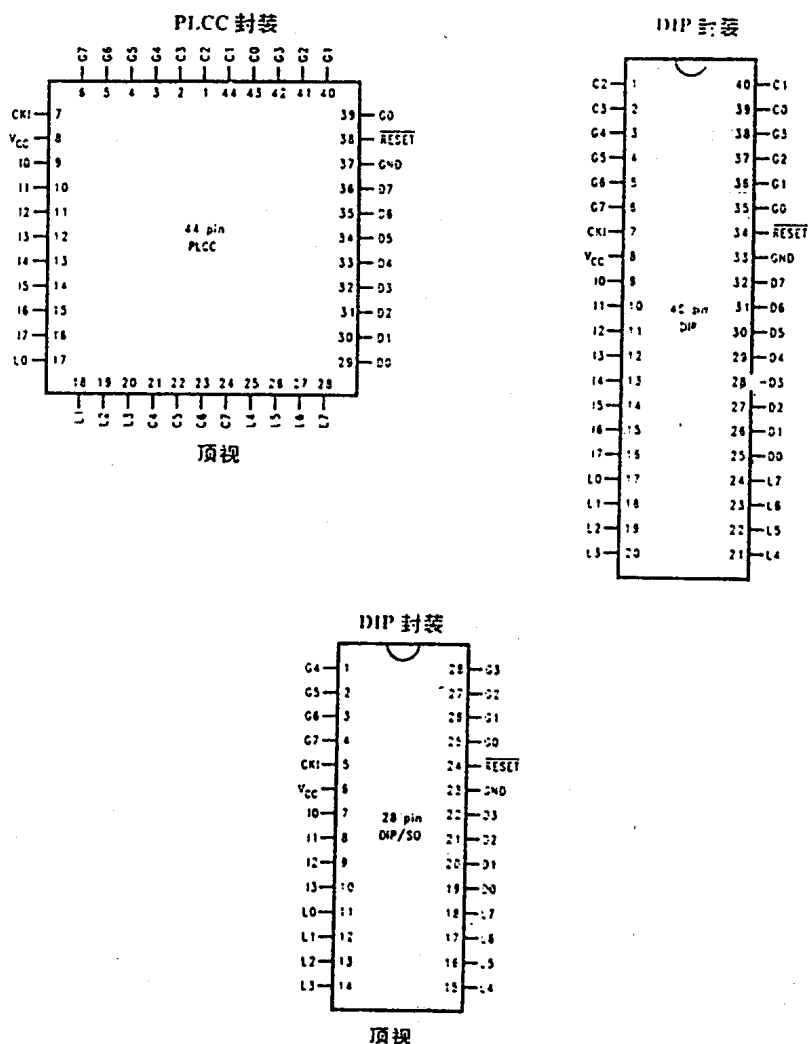


图 5-24 COP8EG 单片机引脚图

5.3.6 COP8 H 类单片机产品——COP684FH/COP688FH COP884FH/COP888FH COP984FH/COP988FH

COP8 H 类单片机的特点是:具有乘/除法硬件部件,1 个全双工 UART 及 3 个 16 位定时/计数器等功能,以及大容量片内存储器(片内 RAM:512B,片内 ROM:12KB)。此外,还具有 COP8 单片机的基本功能。图 5-25 是 COP888FH 单片机的系统结构图。

主要特性:

- 片内 RAM:512 字节
- 片内 ROM:12K 字节
- 执行时间:1 μ s
- 3 个 16 位定时/计数器 T1, T2 和 T3
 - PWM 工作方式
 - 外部事件计数器
 - 输入捕获工作方式

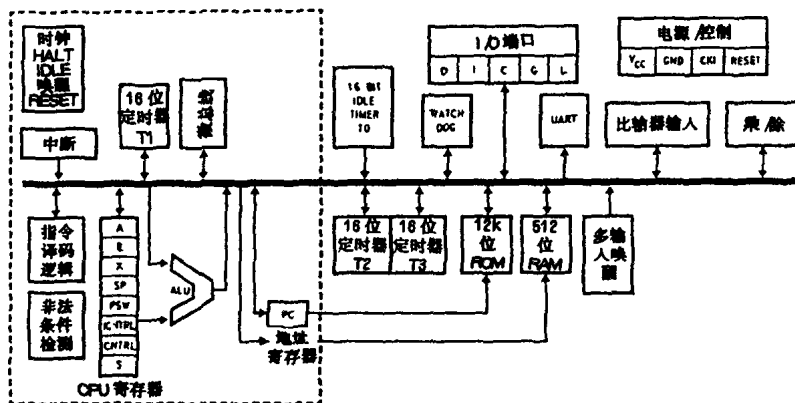


图 5-25 COP888FH 单片机系统结构图

- IDLE 定时器 T0
- WATCHDOG 定时器和时钟监视器逻辑
- MICROWIRE/PLUS 微总线
- 1 个双全工 UART
- 乘/除法硬件部件
- I/O 端口: 24~36 根 I/O
 - 软件选择: 三态输出、推挽输出、弱上拉输入、高阻抗输入
 - I/O 端口 G 和 L 具有施密特输入
 - 多输入唤醒 MIWU 功能
- 9 级中断源
 - 外部中断
 - IDLE 定时器 T0 中断
 - 6 个定时器 T1, T2 和 T3 中断
 - 微总线中断
 - 多输入唤醒中断
 - 两个 UART 中断
 - 软件陷阱中断
 - 系统设置 VIS 中断
- 工作电压: 2.5~5.5V
- 省电工作方式: HALT 和 IDLE 工作方式

图 5-26 是 COP8 H 类单片机封装引脚图。其 OTP 型单片机为 COP87L88FH。

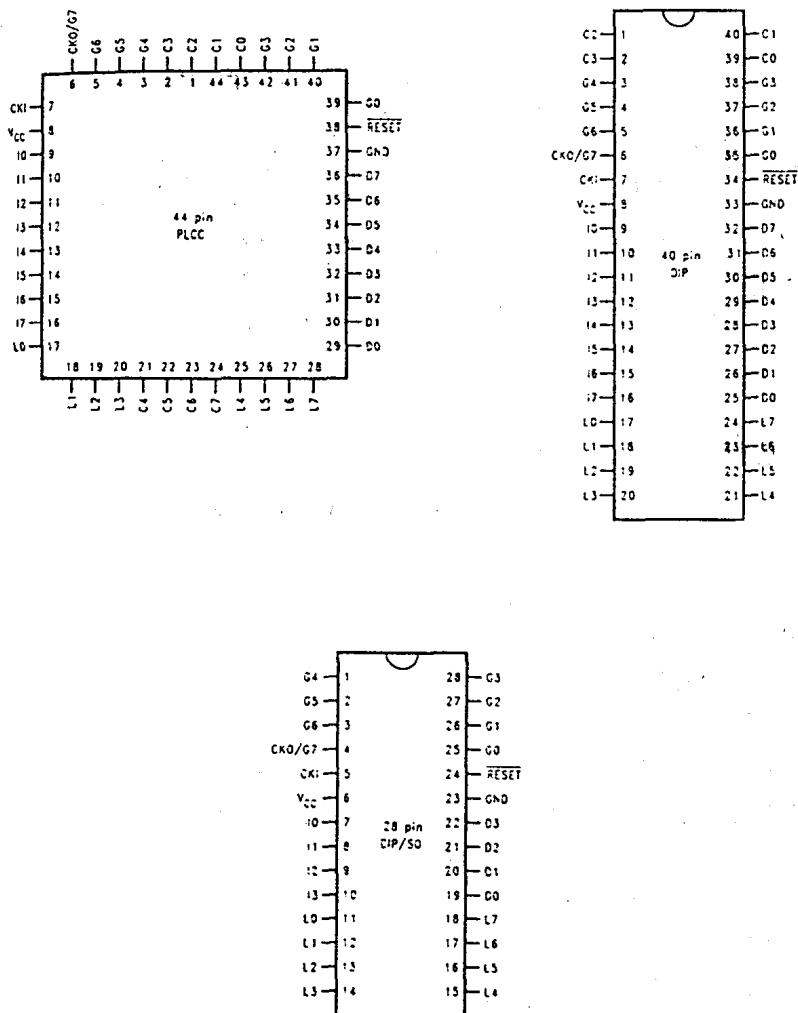


图 5-26 COP8 H 类单片机引脚图

5.3.7 COP8 K 类单片机产品——COP984EK/COP884EK COP988EK/COP888EK

COP8 K 类单片机的最大特点是有 6 个通道 16 位单积分 A/D 转换和减弱 EMI(电磁干扰)等。在该模拟功能模块中,有多路输入开关,模拟比较器、16 位计数器、恒流源和 $V_{CC}/2$ 参考电压等部件。此外,还具有 3 个 16 位定时/计数器、MICROWIRE/PLUS 微总线、多路输入唤醒、WATCHDOG 定时器、IDLE 定时器等 COP8 单片机功能。图 5-27 是 COP888EK 单片机系统结构图。

主要特性:

- 片内 RAM:256 字节
- 片内 ROM:8K 字节
- 指令执行时间:1 μ s
- 3 个 16 位定时/计数器 T1,T2 和 T3

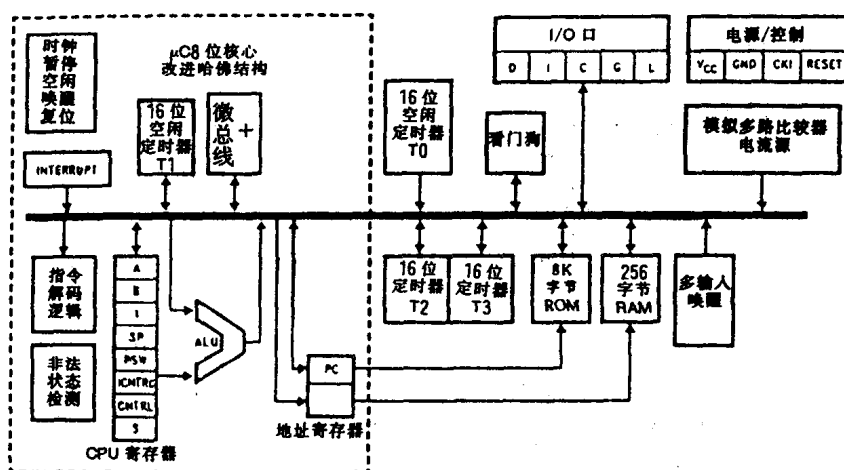


图 5-27 COP888EK 单片机系统结构图

- PWM 工作方式
- 外部事件计数器
- 输入捕获工作方式
- IDLE 定时器 T0
- WATCHDOG 定时器和时钟监视器逻辑
- 6 通道 16 位单积分 A/D 转换器
 - 多路输入开关的模拟比较器
 - 恒流源和 $V_{cc}/2$ 参考电压
- 减弱 EMI(电磁干扰)电路
- I/O 端口:23~39 根
 - 软件选择:三态输出、推挽输出、弱上拉输入、高阻抗输入
 - I/O 端口 G 和 L 具有施密特电路
 - 多路输入唤醒 MIWU 功能
- MICROWIRE/PLUS 微总线
- 12 级中断源
 - 外部中断
 - IDLE 定时器 T0 中断
 - 6 个定时器 T1,T2 和 T3 中断
 - 微总线中断
 - 多路输入唤醒中断
 - 软件陷阱中断
 - 系统设置 VIS 中断
- 工作电压:2.5~6V
- 省电工作方式:HALT 和 IDLE 工作方式

图 5-28 是 COP8 K 类单片机封装引脚图,其 OTP 单片机为 COP87L84EK/COP87L88EK。

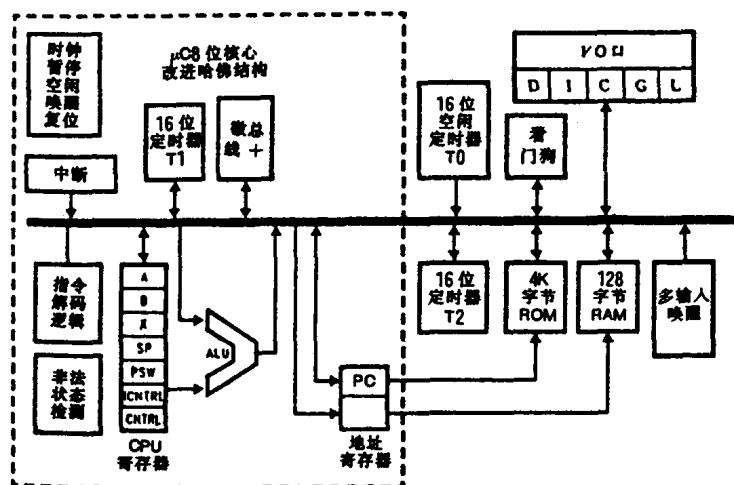


图 5-29 COP888CL 单片机系统结构图

- 两个 16 位定时/计数器 T1 和 T2
 - PWM 工作方式
 - 外部事件计数器
 - 输入捕获工作方式
- IDLE 定时器 T0
- WATCHDOG 定时器和时钟监视器逻辑
- MICRO WIRE/PLUS 微总线
- I/O 端口:23~39 根
 - 软件选择:三态输出、推挽输出、弱上拉输入、高阻抗输入
 - I/O 端口 G 和 L 具有施密特电路
 - 多路输入唤醒 MIWU 功能
- 10 级中断源
 - 外部中断
 - IDLE 定时器 T0 中断
 - 4 个定时器 T1 和 T2 中断
 - 微总线中断
 - 多路输入唤醒中断
 - 软件陷阱中断
 - 系统设置 VIS 中断
- 工作电压:2.5~6.0V
- 省电工作方式:HALT 和 IDLE 工作方式

图 5-30 是 COP8 L 类单片机封装引脚图,其 OTP 型单片机为 COP87L84CL 和 COP87L88CL。

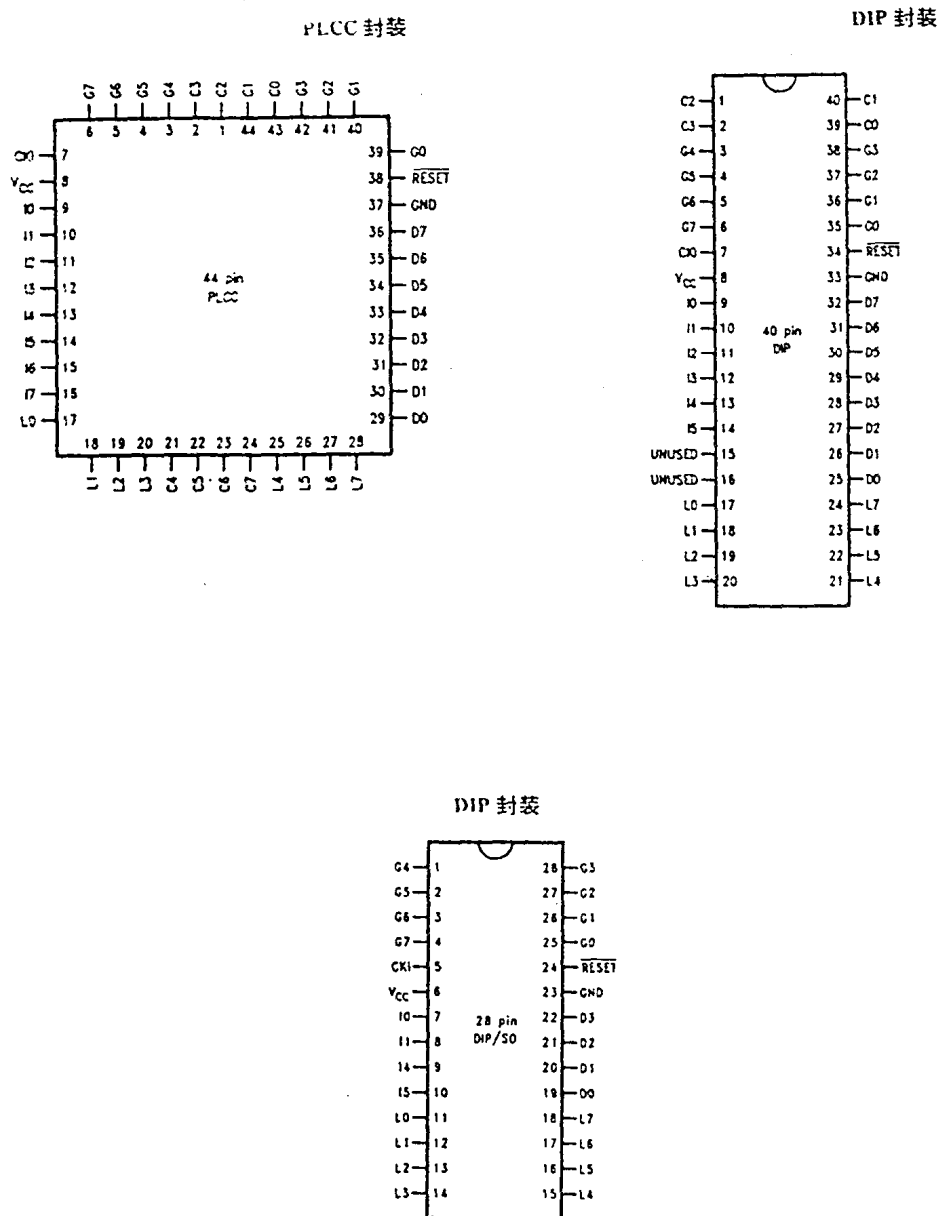


图 5-30 COP8 L 类单片机引脚图

5.3.9 COP8 S 类单片机产品——COP984CS/COP884CS/COP684CS COP988CS/COP888CS/COP688CS

COP8 S 类单片机的功能与 COP8 G 类单片机基本相同,适当给以简化,16 位定时/计数器减至 1 个,模拟比较器也只有 1 个及无减弱 EMI(电磁干扰)等功能。图 5-31 是 COP888CS 单片机的系统结构图。

主要特性:

- 片内 RAM:192 字节
- 片内 ROM:4K 字节

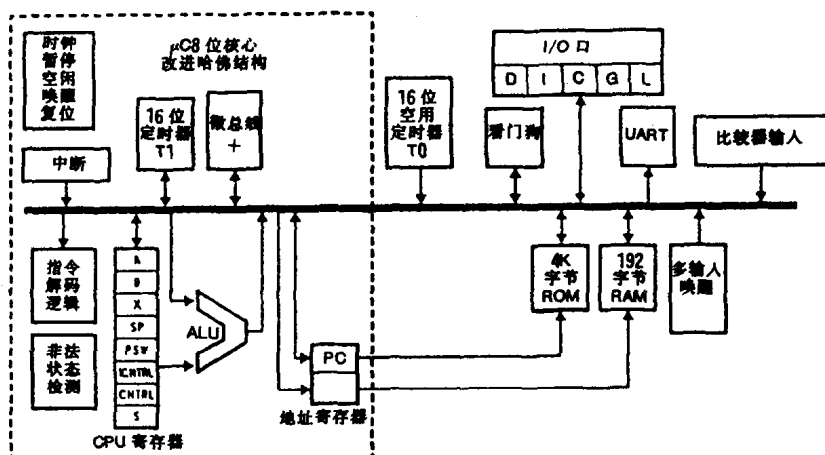


图 5-31 COP888CS 单片机系统结构图

- 指令执行时间: $1\mu\text{s}$
- 一个 16 位定时/计数器 T1
 - PWM 工作方式
 - 外部事件计数器
 - 输入捕获工作方式
- IDLE 定时器 T0
- WATCHDOG 定时器和时钟监视器逻辑
- 全双工 UART
- 一个模拟比较器
- MICROWIRE/PLUS 微总线
- I/O 端口: 23~39 根
 - 软件选择: 三态输出、推挽输出、弱上拉输入、高阻抗输入
 - I/O 端口 G 和 L 具有施密特输入功能
 - 多路输入唤醒 MIWU 功能
- 10 级中断源
 - 外部中断
 - IDLE 定时器 T0 中断
 - 2 个 16 位定时/计数器 T1 中断
 - 微总线中断
 - 多路输入唤醒中断
 - 2 个 UART 中断
 - 软件陷阱中断
 - 系统设置 VIS 中断
- 工作电压: 2.5~6.0V
- 省电工作方式: HALT 和 IDLE 工作方式

图 5-32 是 COP8 S 类单片机封装引脚图,其 OTP 型单片机则采用了 G 类单片机的 OTP 型单片机 COP87L84EG 和 COP87L88EG。

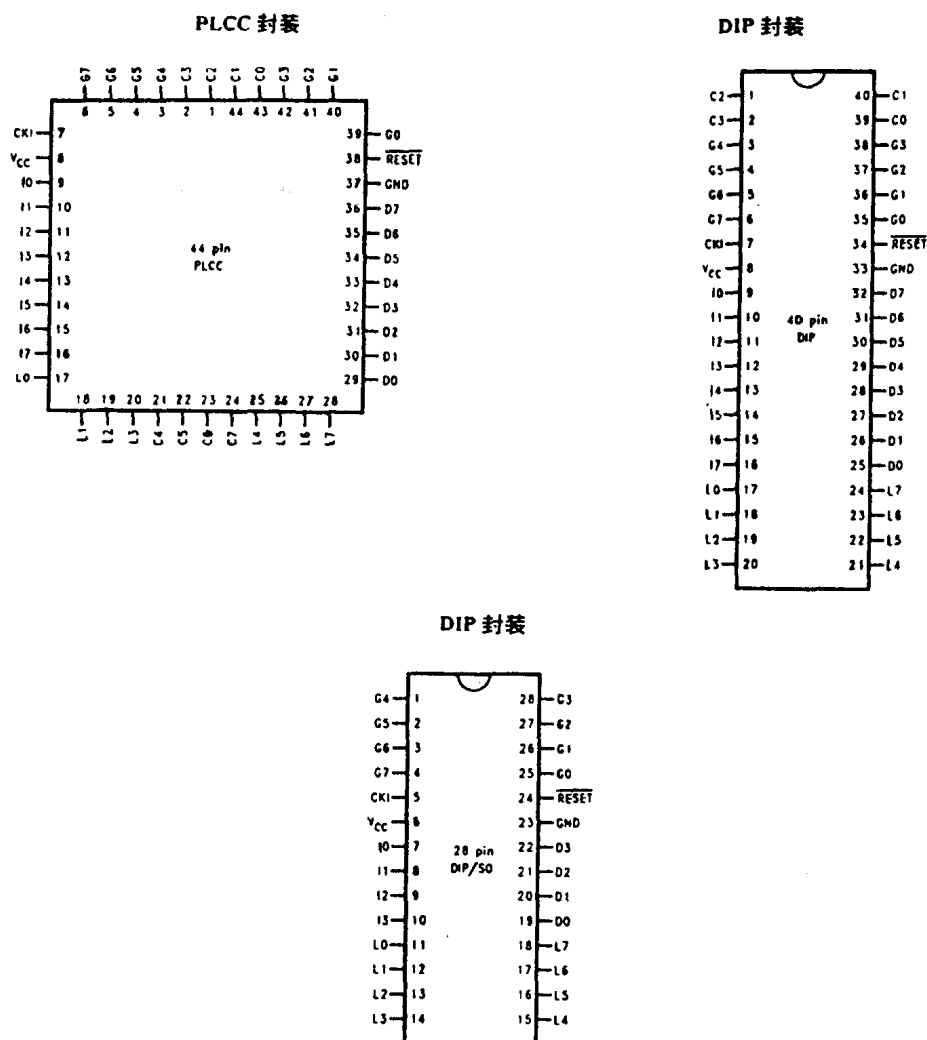


图 5-32 COP8 S 类单片机引脚图

5.3.10 COP8 W 类单片机产品——COP888GW

COP888GW 单片机是功能很强的单片机,它增加了硬件乘/除部件,4 个 16 位脉冲串发生器、2 个 16 位输入捕获模块和减弱 EMI(电磁干扰)等。此外,还具有 2 个 16 位定时/计数器、全双工 UART、微总线及多路输入唤醒等功能,但无 WATCHDOG 定时器。图 5-33 是 COP888GW 单片机系统结构图。

主要特性:

- 片内 RAM:512 字节
- 片内 ROM:16K 字节
- 指令执行时间:1 μ s
- 2 个 16 位定时/计数器 T1 和 T2

—PWM 工作方式

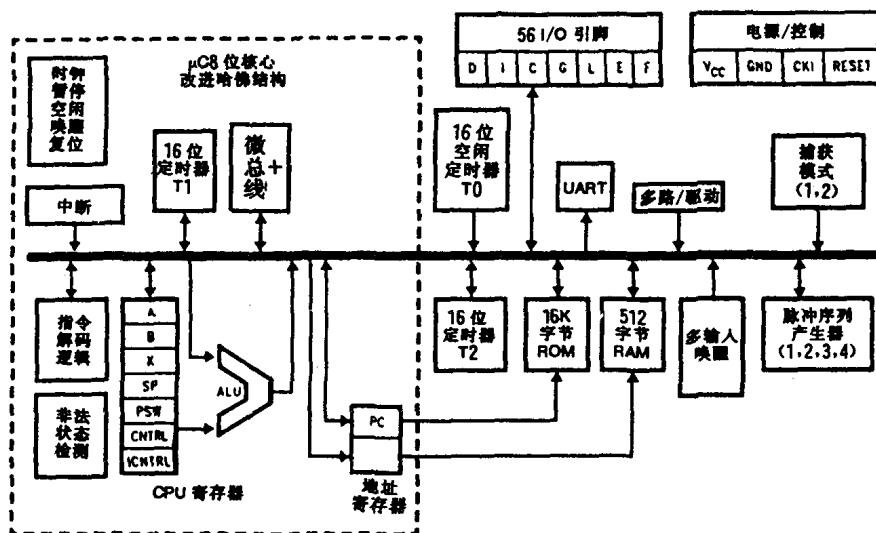


图 5-33 COP888GW 单片机系统结构图

- 外部事件计数器
- 输入捕获工作方式
- 4 个 16 位脉冲串发生器(带 16 位预分频器)
- 2 个 16 位捕获模块(带 8 位预分频器)
- IDLE 定时器 T0
- 硬件乘/除部件
- MICROWIRE/PLUS 微总线
- 减弱 EMI(电磁干扰)电路
- I/O 端口:56 根
 - 软件选择:三态输出、推挽输出、弱上拉输入、高阻抗输入
 - I/O 端口 G 和 L 具有施密特输入
 - 多路输入唤醒 MIWU 功能
- 14 级中断源
 - 外部中断
 - IDLE 定时器 T0 中断
 - 4 个定时器 T1 和 T2 中断
 - 微总线中断
 - 多路输入唤醒中断
 - 2 个 UART 中断
 - 输入捕获定时器中断
 - 脉冲串发生器中断(4 个共享 1 个中断)
 - 软件陷阱中断
 - 系统设置 VIS 中断
- 工作电压:2.5~6.0V

• 省电工作方式:HALT 和 IDLE 工作方式

图 5-34 是 COP888GW 单片机封装引脚图,其 OTP 型单片机是 COP87L88RW。

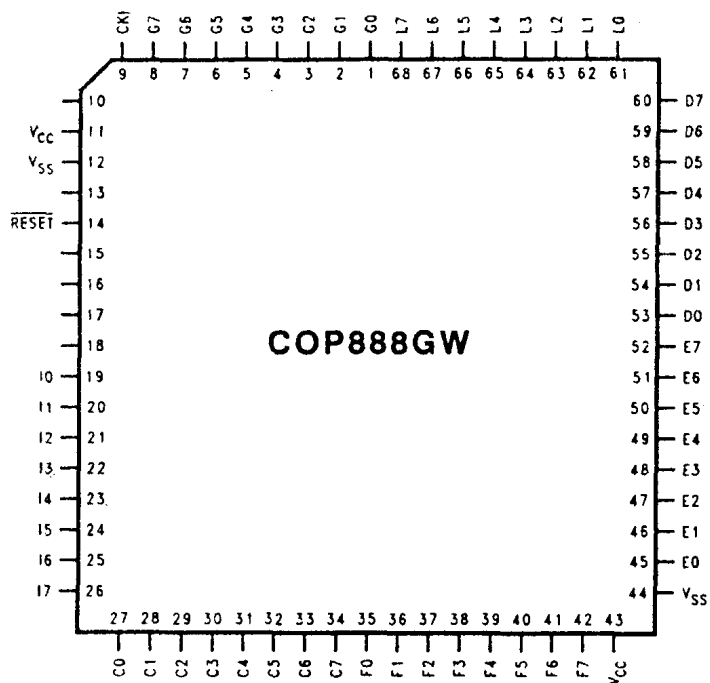


图 5-34 COP888GW 单片机引脚图

第六章 COP8 单片机应用方法

COP8 单片机功能很强,可以直接应用在各种不同应用场合;即使需外接一些少量 I/O 及存储器,也可以通过具有特色的 MICROWIRE/PLUS 微总线来有效地实现。本章所介绍的一些应用方法可以适合 COP8 单片机各种系列,许多例子都是用新型单片机来实施的。

§ 6.1 COP8 单片机 LED 直接驱动方法

新型单片机 COP8SA $\times$ 7 中的端口 D 与端口 L 中的 L3~L0 以及基本型 COP820CJ 单片机中的端口 D 与端口 L 中的 L4~L7 I/O 引脚具有大电流驱动能力。因此,20/28 引脚单片机可以直接驱动 4 位 LED(32 段 LED),40/44 引脚单片机则能直接驱动 8 位 LED(64 段 LED)。

图 6-1 是用 COP8SA $\times$ 7 单片机驱动 4 位 LED 的例子。采用动态扫描的方法,扫描定时可以用定时器 T1 中断来实现,也可以用 RC 外时钟中断或电源 50Hz 频率来实现。

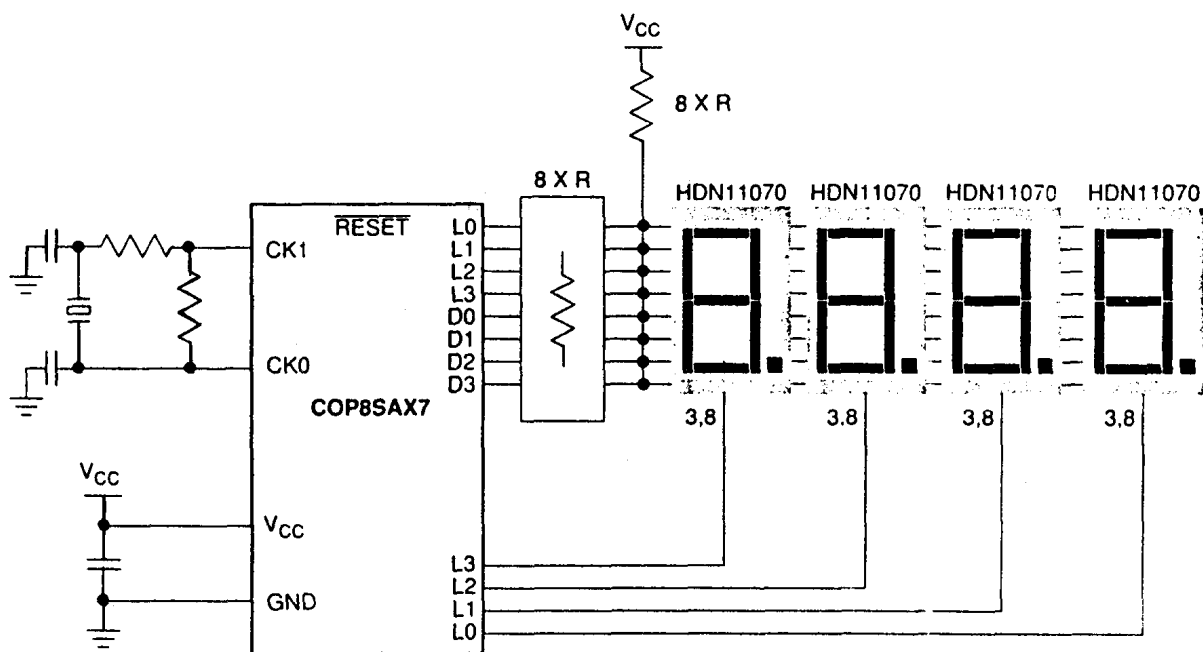


图 6-1 COP8SA $\times$ 7 单片机驱动 LED 图

动态扫描显示,LED 的亮度往往会受影响,因此,上述的显示方式,最好不要超过 3 路(即最多可驱动 24 段 LED)。若各路驱动增加三极管驱动,则可以大大提高显示亮度,而且,

驱动 I/O 引脚也不需大电流输出,可以采用一般 I/O 引脚来驱动,如图 6-2 所示。

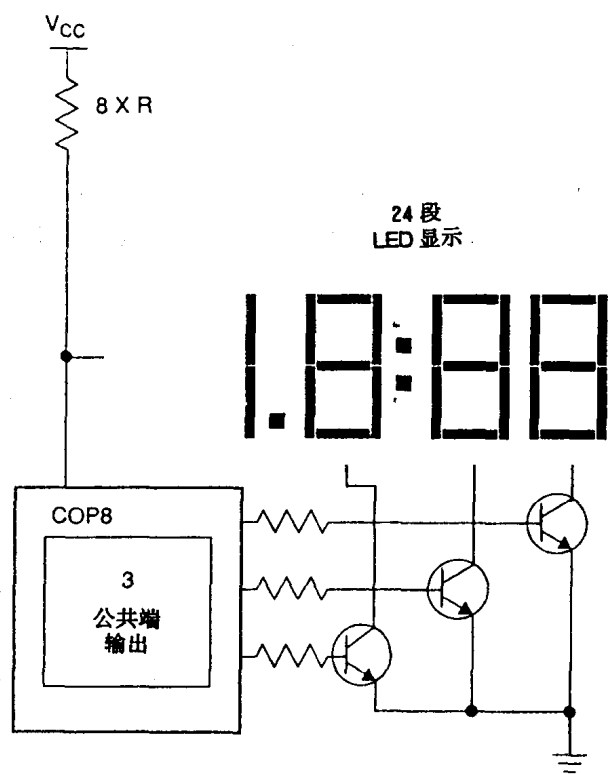


图 6-2 亮度增加 LED 驱动示意图

单片机的相应控制流程图如图 6-3 所示。

§ 6.2 COP8 单片机 LCD 直接驱动方法

在许多应用中,液晶 LCD 显示是通过 LCD 驱动器来实现的。常用的方法是单片机连接特定的 LCD 驱动器来驱动相应的 LCD;也有的把 LCD 驱动器集成在单片机内。对于一些显示段数不多的简易的 LCD,则可以用单片机直接驱动,其方法类似于 LED 直接驱动。

LCD 的多路显示是由后板电极(公共段底板)COM 的数量来决定的。显示的像元段则为公共端 COM 与段 SEG 的乘积。对于 28 引脚封装单片机,其可作为输出的 I/O 引脚为 18 根。

若显示多路 N 为两路(两个 COM 端),输出引脚 M 为 18,可驱动显示像元段 S 为:
 $S = (M - N) \times N = (18 - 2) \times 2 = 32$
 若显示多路 N 为三路(3 个 COM 段),输出引脚 M 仍为 18,则可驱动显示像元段 S 为:
 $S = (M - N) \times N = (18 - 3) \times 3 = 45$

在一个 LCD 刷新周期 t_x 内($1/t_x = f_x$,典型值为 30Hz 至 60Hz),必须产生三个不同电平: V_{OP} , $0.5V_{OP}$ 和 $0V$ 。LCD 显示段两端的“off”电压不等于 $0V$,“on”电压也不等于 V_{OP} ,不仅是 V_{OP} 电压的一部分。“on”到“off”的有效值电压比率(分辨率)是由多路传输率和所涉及

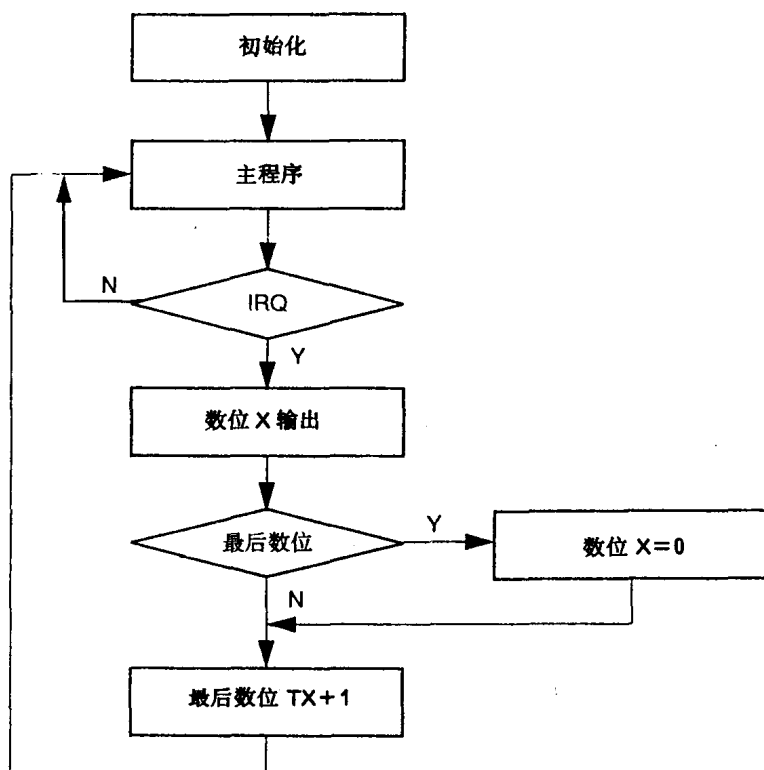


图 6-3 LED 驱动流程图

的电压电平决定的。最理想的分辨率为 V_{ON} 与 V_{OFF} 的比率最大的分辨率。这种分辨率能够使有效状态和无效状态之间的电压差最大：

$$(V_{OP}/V_{OFF})_{\max} = \text{SQR}((\text{SQR}(N)+1)/(\text{SQR}(N)-1))$$

根据上式，两路传输（2 个 COM 端）LCD 最大分辨率为 2.41，三路传输（3 个 COM 端）LCD 最大分辨率为 1.93。为了使用方便，大多数 LCD 驱动器使用了等差分压（0V, 0.5 V_{OP} , V_{OP} ）。为了取得三个等差电压只需要外加 4 个等值电阻，构成两个电压分压器（两路传输 LCD），如图 6-4 所示。

单片机每个 I/O 引脚都可单独设置为：高阻、“I”和“O”三个状态，所以上述过程很容易。

一般情况下，每平方厘米 LCD 有效区域耗用电流为 3~4 μA （ $V_{OFF}=4.5\text{V}$ ，刷新率为 60Hz）。若刷新率较大时，LCD 耗用电流会突然增加，故刷新率不大于 60Hz。

LCD 显示驱动时序如图 6-5 和图 6-6 所示，一个 LCD 刷新周期 t_x 分成了四个相等的时间相位 t_a, t_b, t_c 和 t_d ，在分解过程中，为断开或接通特定段，必须更新段终端和后板，就 50Hz 刷新频率（ $t_x=20\text{ms}$ ）而言， t_a, t_b, t_c 和 t_d 为 5ms。

通过单片机中定时器的 PWM 自动加载方式，可以满足时序要求，通过设置 CNTRL 寄存器中的 TURN 位标志，使定时器在每个指令周期减 1。当定时器溢出时，置位 TPND 标志，从而自动加载寄存器中的值自动地加载至定时器，也可以通过定时器溢出来产生一个中断。

图 6-5 给出了用于两个 COM(a)和段电极(b)的电压波形。这是开通段 A 和断开段 B

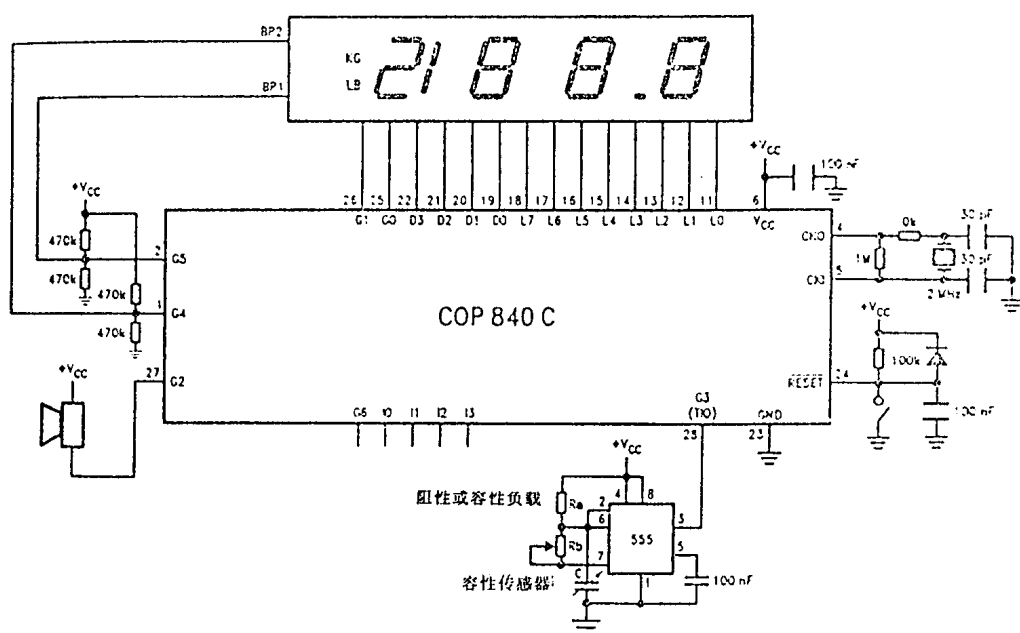


图 6-4 单片机 LCD 显示驱动电路图

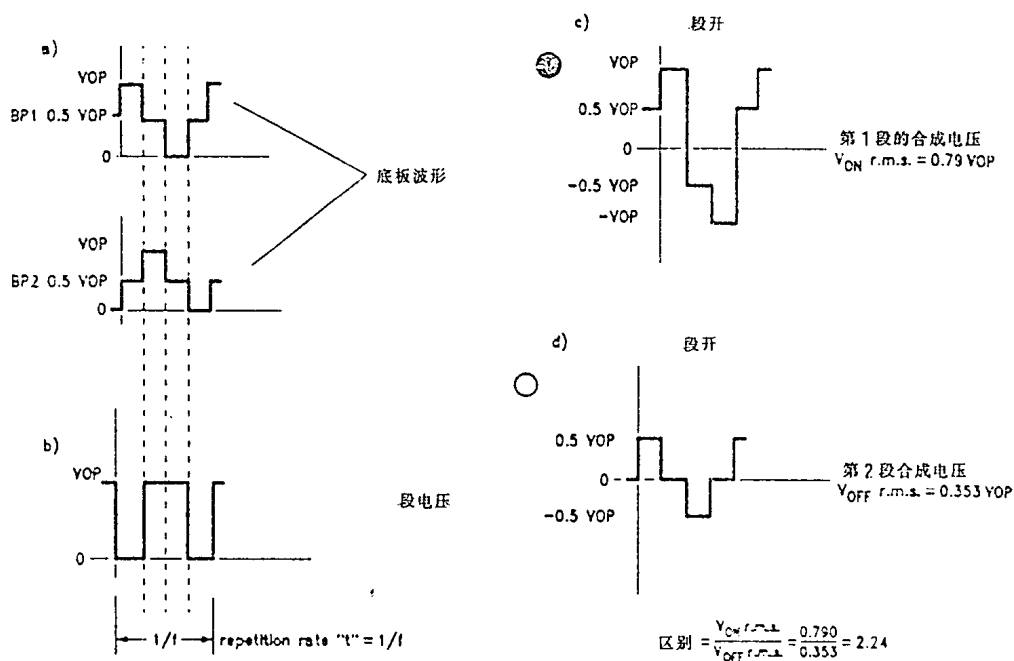


图 6-5 LCD 波形图

所要求的。段(c和d)上的结果电压要通过从BP1中(段A)减去波形(a)和从BP2(段B)减去波形(b)获得。

图 6-6 也给出了 4 个不同的波形,以满足与同一 COM 端(关-关、关-开、开-关、开-开)连接的两个段的所有可能组合。

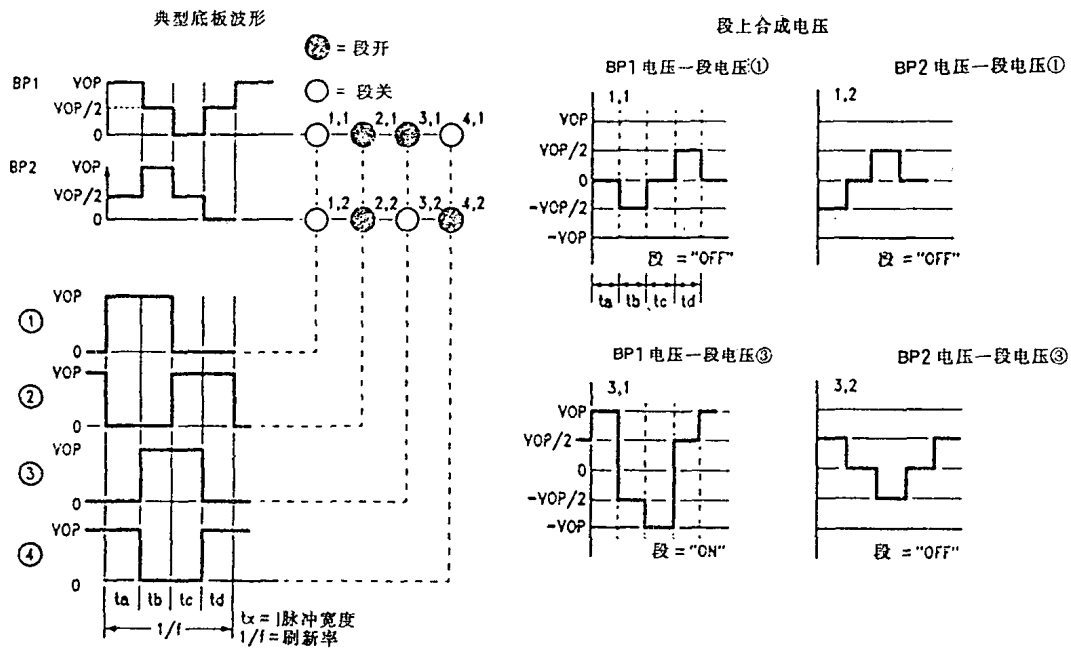


图 6-6 1:2 多路传输 LCD 驱动波形图

图 6-7 是定制两路传输 LCD 内部段和公共端(COM)连接图。

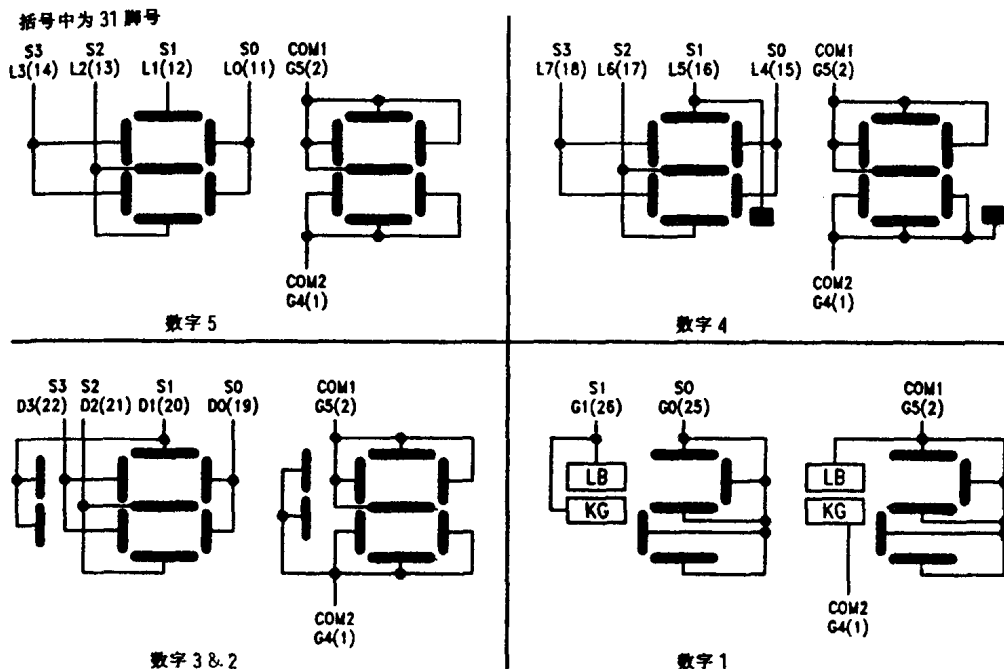


图 6-7 定制两路传输 LCD 显示器结构图

为了便于获得显示数据, LCD 驱动程序 DISPL 把 16 位二进制数转换成 24 位的 BCD 码。LCD 驱动子程序本身由主程序和 7 个子程序组成。子程序 SEG0~SEG3 用来分别为时

间相位 ta, tb, tc 和 td 从 ROM 查表中取 LCD 段数据。子程序 SEGOUT 是写每个时间相应的段数据到相应的输出端口。写一个数据要花 $50\mu\text{s}$, 这样总刷新时间为 20ms (50Hz)。子程序 TTPND 循环检查 CNTRL 寄存器中的 TPND 挂起标志, 若该标志被置位, 子程序 TTPND 就将其复位并返回调用程序, 这样就实现了在下一时间位的数据段和 COM 端数据写到输出端口延迟 5ms 。DISPD 子程序通过设置后板和段连接到“0”, 断开 LCD。

相应的程序如下:

;被定制的双向多路 LCD 查找表
.=X'200

;在 ROM 地址 200 处开始查找表

;时间相位 Ta 7 段数据

. BYTE	004	; "0" 和 ".0"
. BYTE	00E	; "1" 和 ".1"
. BYTE	008	; "2" 和 ".2"
. BYTE	008	; "3" 和 ".3"
. BYTE	002	; "4" 和 ".4"
. BYTE	001	; "5" 和 ".5"
. BYTE	001	; "6" 和 ".6"
. BYTE	00C	; "7" 和 ".7"
. BYTE	000	; "8" 和 ".8"
. BYTE	000	; "9" 和 ".9"
. BYTE	00F	; " " 和 "."

;特殊段时间相位 Ta

. BYTE	001	; "LB"
. BYTE	000	; "LB2"
. BYTE	003	; "KG"
. BYTE	002	; "KG2"

. =. +1

;时间相位 Tb7 段数据

. BYTE	002	; "0"
. BYTE	00E	; "1"
. BYTE	003	; "2"
. BYTE	00A	; "3"
. BYTE	00E	; "4"
. BYTE	00A	; "5"
. BYTE	002	; "6"
. BYTE	00E	; "7"
. BYTE	002	; "8"
. BYTE	00A	; "9"

```

        .BYTE      00F      ;" "
        .BYTE      000      ;". 0"
        .BYTE      00C      ;". 1"
        .BYTE      001      ;". 2"
        .BYTE      008      ;". 3"
        .BYTE      00C      ;". 4"
        .BYTE      008      ;". 5"
        .BYTE      000      ;". 6"
        .BYTE      00C      ;". 7"
        .BYTE      000      ;". 8"
        .BYTE      008      ;". 9"
        .BYTE      00D      ;". "

        .LOCAL

TTPND:
        LD          B, #PSW

$ LOOP:
        IFBIT       #TPND, [B]
        JP          $END
        JP          $LOOP

$ END:
        RBIT       #TRND, [B]
        LD          B, #PORTGD
        RET

        .LOCAL

        . = . + 1
;时间相位 Tc 7 段数据
        .BYTE      00B      ;"0"和". 0"
        .BYTE      001      ;"1"和". 1"
        .BYTE      007      ;"2"和". 2"
        .BYTE      007      ;"3"和". 3"
        .BYTE      00D      ;"4"和". 4"
        .BYTE      00E      ;"5"和". 5"
        .BYTE      00E      ;"6"和". 6"
        .BYTE      003      ;"7"和". 7"
        .BYTE      00F      ;"8"和". 8"
        .BYTE      00F      ;"9"和". 9"

```

. BYTE	000	; " " 和 ". 0"
COPY;		
		; 拷贝由 B 和 B+1 指向的
		; 2 字节到由 X 和
		; X+1 指向的 RAM
LD	A,[B+]	
X	A,[X+]	
LD	A,[B+]	
X	A,[X+]	
RET		
. LOCAL		
; 时间相位 Td 7 段数据		
. BYTE	00D	; " 0"
. BYTE	001	; " 1"
. BYTE	00C	; " 2"
. BYTE	005	; " 3"
. BYTE	001	; " 4"
. BYTE	005	; " 5"
. BYTE	00D	; " 6"
. BYTE	001	; " 7"
. BYTE	00D	; " 8"
. BYTE	005	; " 9"
. BYTE	000	; ". "
. BYTE	00F	; ". 0"
. BYTE	003	; ". 1"
. BYTE	00E	; ". 2"
. BYTE	007	; ". 3"
. BYTE	003	; ". 4"
. BYTE	007	; ". 5"
. BYTE	00F	; ". 6"
. BYTE	003	; ". 7"
. BYTE	00F	; ". 8"
. BYTE	007	; ". 9"
. BYTE	002	; ". "
; 特殊段时间相位 Tb		
. BYTE	003	; "LB"

```
. BYTE      003      ;"LB 2"
. BYTE      001      ;"KG"
. BYTE      001      ;"KG 2"
```

;特殊段时间相位 Tc

```
. BYTE      002      ;"LB"
. BYTE      003      ;"LB 2"
. BYTE      001      ;"KG"
. BYTE      000      ;"KG 2"
```

;特殊段时间相位 Td

```
. BYTE      000      ;"LB"
. BYTE      000      ;"LB 2"
. BYTE      002      ;"KG"
. BYTE      002      ;"KG 2"
```

. END

;显示:

;输入参数:COUNT2=RAM 寄存器,它包含了以秒显示的时间。

;例如 COUNT2=1->显示时间为 1s

;被定制的双向多路 LCD 驱动程序

;LCD

;程序转换存储在 RAM 单元的 BCD 数据

;BCDLO,BCDHI 到 LCD 的输出数据存储在

;MWBUF0=口 L 数据

;MWBUF1=口 D 数据

;MWBUF2=口 G 数据(仅对 G0,G1,其他位保持不变)

;

;子程序包括:

;SEG0:为时间相位 TA 获得 LCD 段数据。

;SEG1:为时间相位 TB 获得 LCD 段数据。

;SEG2:为时间相位 TC 获得 LCD 段数据。

;SEG3:为时间相位 TD 获得 LCD 段数据。

;DISPD:关断显示并配置口 G,L 和 D

;TTPND:检查定时器挂起标志(更新比率发生器)

;SEGOUT:输出 LCD 段和底板数据

;子程序 SEG0...SEG1 必须直接接在查找表之后,因为使用了 LAID 指令。


```

        .LOCAL
SEG0:
        LD      B, #0FF1          ;指向偏移 1 寄存器
        LD      [B+], #000
        LD      [B+], #000
        LD      A, #00B

$ TWO:
        IFBIT    #05,BCDHI        ;重量 ≥ 200 磅?
        INCA     ;是,显示数字 5("2")

$ POUND:
        IFBIT    #POUND,FLAG
        JP      $ LPORT
        ADD     A, #002

$ LPORT:
        X        A,[B]
        LD      A, #BCDLO
        LD      A, #MWBUF0
        LD      A,[X]
        ADD     A, #00F          ;取消数字 1 的位
        AND     A, 0FF2
        LAID     ;获得数字 1 的数据
        X        A,[B]          ;数字 1 的数据
                                ;存入 MWBUF0
        LD      A,[X+]
        AND     A, #0F0          ;取消数字 1 的位
        SWAP    A
        ADD     A, 0FF1          ;总是显示小数点
        LAID     ;获得数字 1 的数据
        SWAP    A
        OR      A,[B]          ;数字 1 和
        X        A,[B+]        ;数字 2 的数据存入 MWBUF1

$ DPORT:
        LD      A,[X]
        IFBIT    #04,BCDHI
        JP      $ ADD1
        AND     A, #00F
        ADD     A, 0FF2          ;显示无引导零
        JP      $ GET

```

```

$ADD1:
    AND    A,#00F
    ADD    A,0FF1           ;显示“1”(数字 4)

$GET:
    LAID                    ;获得数字 3 的数据
    X      A,[B+]           ;数字 3 的数据
                                ;存入 MWBUF1

$GPORT:
    LD      A,0FF3
    LAID                    ;获得数字 5("2")和特殊段数据
    OR      A,#0FC          ;置位 2...7 为 1
    X      A,[B]            ;数据存入 MWBUF2 中
    RET

SEG1:
    LD      B,#0FF1
    LD      [B+],#01B
    LD      [B+],#010
    LD      A,#056
    JP      $TWO

SEG2:
    LD      B,#0FF1
    LD      [B+],#030
    LD      [B+],#030
    LD      A,#05A
    JP      $TWO

SEG3:
    LD      B,#0FF1
    LD      [B+],#04B
    LD      [B+],#040
    LD      A,#05E
    JP      $TWO
    .LOCAL

DISPL:
    IFBIT   #POUND,FLAG
    JP      MULT2
    JP      LDT

MULT2:
                                ;以磅为单位计算重量
    LD      B,#BUF12LO        ;(公斤乘以 2.2 的乘积)

```

	LD	[B+], #22	
	LD	X, #STALO	
	JSR	MULBI168	
	LD	B, #BUF12LO	
	JSR	COPY	
	LD	STAH1+1, #00	
	LD	DIV0, #10	
	JSR	DIVBI248	
LDT:			
	JSR	BINBCD16	;转换二进制到BCD重量
	LD	COUNT, #50	;重复显示循环50次
			; (=1s 显示时间)
	LD	B, #TMRLO	
	LD	[B+], #0E8	送1000(03E8H)到定时器
	LD	[B+], #003	; (=50Hz LCD, 更新 tc=5μs)
	LD	[B+], #0E8	;送1000到自动寄存器
	LD	[B+], #0E8	;
	LD	[B+], #003	;
	LD	[B+], #090	;CNTRL寄存器:“具有自动重装
			;定时器”方式,启动定时器
	LD	[B+], #010	;PSW寄存器:复位TPND标志
DISP1:			
	JSR	SEG0	;为刷新时间相位Ta获得
			;7段数据
	JSR	TTPND	;测试定时器挂起标志
TP0:			;底板刷新Ta
	SBIT	#BP1, [B]	
	LD	A, [B+]	;指向G配置寄存器
	RBIT	#BP2, [B]	
	SBIT	#BP1, [B]	
	LD	A, [B-]	;指向G数据寄存器
	RBIT	#BP2, [B]	
	JSR	SEGOUT	;段数据输出
	JSR	SEG1	;为Tb获得7段数据
	JSR	TTPND	
TP1:			
	SBIT	#BP2, [B]	
	LD	A, [B+]	;指向G配置寄存器

	RBIT	#BP1,[B]	
	SBIT	#BP2,[B]	
	LD	A,[B-]	;指向 G 数据寄存器
	RBIT	#BP1,[B]	
	JSR	SEGOUT	
	JSR	SEG2	;为 Tc 获得 7 段数据
	JSR	TTPND	
TP2:			
	RBIT	#BP1,[B]	
	LD	A,[B+]	;指向 G 配置寄存器
	RBIT	#BP2,[B]	
	SBIT	#BP1,[B]	
	LD	A,[B-]	;指向 G 数据寄存器
	RBIT	#BP2,[B]	
	JSR	SEGOUT	
	JSR	SEG3	
	JSR	TTPND	
TP3:			
	RBIT	#BP1,[B]	
	RBIT	#BP2,[B]	
	LD	A,[B+]	
	RBIT	#BP1,[B]	
	SBIT	#BP2,[B]	
	JSR	SEGOUT	
	DRSZ	COUNT	
	JP	DISP1	
	LD	COUNT, #50	
	DRSZ	COUNT2	;10s 结束?
	JP	DISP1	;否,显示重量
	JSR	DISTD	
	RET		;是,程序完成
DISPD:			;关闭显示
	LD	B, #PORTLD	
	LD	[B+], #000	;输出 0 到口 L
	LD	[B+], #0FF	;口 L=输出口
	LD	B, #PORTGD	
	LD	[B+], #000	;输出 0 到 G 输出

LD	[B+], #037	;G0...G2,G4,G5=输出
LD	PORTD, #000	;输出 0 到 D 口
RET		

SEGOUT:

LD	B, #MWBUF0	
LD	A, [B+]	;指向 MWBUF1
X	A, PORTLD	;输出 MWBUF1 中的 7 段数据 ;到口 L
LD	A, [B+]	;指向 MWBUF2
X	A, PORTD	;输出 MWBUF1 到口 D
LD	X, #PORTGK	
LD	A, [X]	
AND	A, [B]	;将 MWBUF2 与 PORTGD 相加 ;使位 2...7 不变
X	A, [B]	;结果(A')存入 MWBUF2, ;原始 MWBUF2 值送入 A
AND	A, #003	;将 007 与原始 MWBUF2(A") ;相加,置位 0,1 为正确值
OR	A, [B]	;A' 逻辑或"A",恢复 ;原始 G2...G7 位
X	A, [X]	;输出结果到口 G
RET		

;16 位二进制到 BCD 转换

;存储器分配如下:

;BINLO:RAM 地址二进制低字节

;BCDLO:RAM 地址 BCD 低字节

;COUNT:RAM 地址移位计数器(0F0...0FB,0FF)

;BCDLO,BCDLO+1 和 BCDLO+2 中的 BCD 数

;

;存储器地址 M(BINLO+1) M(BINLO)

;数据 二进制 HB 二进制低字节

;

;存储器地址 M(BCDLO+2) M(BCDLO+1) M(BCDLO)

;数据 BCD HB BCD BCD 低位

BINLO=STALO

.LOCAL

```

$BCDT = (BCDLO + 3) & 0F
$BINT = (BINLO + 2) & 0F
BINBCD:
    LD          COUNT, #16      ;左移次数送入控制寄存器
    LD          B, #BCDLO       ;装载 BCD 数最低字节地址
$CBCD:
    LD          [B+], #00       ;清 BCD RAM 寄存器
    IFBNE       # $BCDT
    JP          $CBCD
$LSH:
    LD          B, #BINLO       ;左移二进制数
    RC
$LSHFT:
    LD          A, [B]
    ADC         A, [B]          ;如果 MSB 置位,置位进位位
    X           A, [B+]
    IFBNE       # $BINT
    JP          $LSHFT
    LD          B, #BCDLO
$BCDADD:
    LD          A, [B]
    ADD         A, #066         ;加校正因子
    ADC         A, [B]          ;左移 BCD 数
                                ; (BCD = 2 * * 二进制
                                ; 位重量(二进制))
    DCOR        A              ;十进制校正加法
    X           A, [B+]
    IFBNE       # $BCDT
    JP          $BCDADD
    DRSZ        COUNT         ;移位计数器减 1
    JP          $LSH
    RET
    .LOCAL

```

;二进制除法,24 位除以 8 位($Q=Y/Z$)
;YL:低字节 RAM 地址被除数
;ZL:低字节 RAM 地址除数
;CNTR:RAM 地址移位计数器(0F0...0FB,0FF)
;商在 RAM 单元 YL...YL+2

;余数在 YL+3
 ;商都为 1(如果被 0 除),余数
 ;则包含 YL

;存储器分配如下:

```

;
;M(YH+1)    M(YH)    M(YL+1)    M(YL)
;0          Y(高字节)Y          Y(低字节)
;-----

```

;M(ZL)

;Z

;

;在 $t_c=1\mu s$ 执行时,程序需 1.21ms

```

      ZL          =DIV0
      YL          =STALO
      CNTR        =COUNT
      .LOCAL
      $ YH        =YL+2
      $ BTY       =($ YH&00F)+2;“IFBNE”指令的参数

```

DIVBI248:

```

      LD          CNTR,#018      ;初始化移位计数器
      LD          B,$ YH+1      ;为 24 计数
      LD          [B],#000      ;送 0 到 M(YH+1)
      LD          X,$ YH+1

```

\$ LSHFT:

```

      LD          B,$ YL        ;左移被除数
      RC

```

\$ LUP:

```

      LD          A,[B]
      ADC         A,[B]
      X           A,[B+]
      IFBNE      #$ BTY
      JP         $ LUP
      LD          B,$ ZL
      IFC
      JP         $ SUBT

```

\$ TSUBT:

;减并测试

```

SC                                     ;从 M(YH+1,YH+2)中减去 Z
LD      A,[X]
SUBC    A,[B]
IFNC
JP      $TEST
$SUBC:
LD      A,[X]                                     ;从 M(YH+1,YH+2)中减去 Z
SUBC    A,[B]
X        A,[X]
LD      B,#YL
SBIT    #0,[B]
$TEST:
DRSZ    CNTR                                     ;24 次移位执行?
JP      $LSHFT                                   ;否,左移被除数
RET
.LOCAL

```

;二进制乘法,16 位值(X1)

;乘以 8 位值(X2); $M=X1 * X2$

;X1L:RAM 地址 X1 低字节

;X2L:RAM 地址 X2

;COUNT RAM 地址移位计数器

;M 存储在 RAM 地址 X2L...X2L+2 处

;存储器分配如下:

;存储器	M(X2L+2)	M(X2L+1)	M(X2L)
;数据	0	0	X2

;-----

;存储器	M(X1L+1)	M(X1L)
------	----------	--------

;数据	XI(H. B)	X1(低字节)
-----	----------	---------

;tc=1μs 时的程序执行时间为 240μs

;

;

.LOCAL

MULBI168;

	LD	COUNT, #9	;预置移位计数器
	LD	[B+], #00	;用“0”预置 X2L+1 和 X2L+2
	LD	[B], #00	
	RC		
\$ LOOP:			
	LD	A, [B]	;右移
	RRCA		
	X	A, [B-]	
	LD	A, [B]	
	RRCA		
	X	A, [B-]	
	LD	A, [B]	
	RRCA		
	X	A, [B+]	
	LD	A, [B+]	;B 指针加 1
	IFNC		;X2 的符号位置位?
	JP	\$ TEST	;否,测试移位计数器
	RC		;是,复位进位位
	LD	A, [B-]	;指向结果的第 2 最高字节
	LD	A, [X+]	;执行加权加法
	ADC	A, [B]	
	X	A, [B+]	
	LD	A, [X-]	
	ADC	A, [B]	
	X	A, [B]	
\$ TEST:			
	DRSZ	COUNT	;8 次右移执行否?
	JP	\$ LOOP	;否,移位
	RET		;是,乘法结束
	. LOCAL		
	. END		

§ 6.3 COP8 单片机简易 A/D 转换(V/F)方法

采用 V/F 方式的 A/D 转换可以实现低成本的 A/D 转换,广泛应用于数字电子秤的 A/D 转换。V/F 转换可以采用专用的 V/F 转换器集成电路 LM331/LM331A。也可以采用更简单的方法。

上节的图 6-4 已给出数字电子秤中的配置。电阻传感器或电阻传感器与重量相关的电阻或电容变化由 555 定时器转换成频率变化,即重量(也可以是温度等模拟量)的变化由电阻(或电容)产生相应变化电压,再由 555 定时器构成的 V/F 转换器转换成频率,最后由单片机中 16 位定时器进行频率测量,把频率变化最终变换成数字,从而实现了简易的 A/D 转换。

由 555 定时器构成的 V/F 转换器,其输出频率为:

$$f=1.44/((R_a+2R_b)\times C)$$

输出高电平时间为:

$$t_1=0.693\times(R_a+R_b)\times C$$

输出低电平时间为:

$$t_2=0.693\times R_b\times C$$

单片机的 16 位定时器设置为输入捕获工作方式,其定时/计数器的输入端 G3(TIO)作输入捕获端,每个正跳沿(也可选负跳沿)作为输入捕获事件。一旦捕获到,即把此时定时/计数器的值送至自动加载/捕获寄存器。在 PSW 寄存器中把定时器挂起标志 TPND 置位,表明捕获事件已发生,若允许定时器中断,则可产生相应中断。

定时器预设置为 FFFFH,并通过 TRUN 位置位来使定时器开始运行,然后,用软件检查循环中的 TPND 标志来实现。当首次设置 TPND 标志时,在 STAHI,STALO 存储单元中存储此时捕获寄存器的值作为起始值;通过软件来复位 TPND 挂起标志,计数另外 255 个正跳沿(等于 255 个脉冲),再把此时捕获寄存器的值(结束值)存储在 RAM 单元 ENDHI,ENDLO 之中。同时,可测量到最短时间周期,也就是存储捕获寄存器值所需指令数(最坏情况是 18 个指令周期, $t_c=1\mu s$ 时,最大频率为 55.5kHz)。

初始值减去结束值为结果,存储在 RAM 单元 STAHI,STALO 中,再用该结果来计算频率时间周期:

$$T=(\text{起始值}-\text{结束值})\times t_c/N$$

N:测量到脉冲数 255

数字电子秤的频率测量、重量计算是在 LCD 显示器关闭时进行的,并用 10 秒时间来显示重量,在 10 秒之后,LCD 显示器被再次关闭,并进入 HALT 工作方式,以节省用电($I_{DD}<10mA$)。通过单片机复位按钮,初始化数字秤,重新开始称重过程。

V/F 转换的 A/D 转换程序如下:

;口、配置和控制寄存器

PORTLD	=0D0	;L 口数据寄存器
PORTLC	=0D1	;L 口配置
PORTLP	=0D2	;L 口输入寄存器
PORTGD	=0D4	;G 口数据寄存器
PORTGC	=0D5	;G 口配置
PORTGP	=0D6	;G 口输入寄存器
PORTD	=0DC	;D 口(输出)
PORTI	=0D7	;I 口(输入)

SIOR	=0D9	;写移位寄存器
TMRLO	=0EA	;定时寄存器
TMRHI	=0EB	;定时器高字节
TAULO	=0EC	;定时器自动装载寄存器的低字节
TAUHI	=0ED	;定时器自动装载寄存器的高字节
CNTRL	=0EE	;控制寄存器
PSW	=0EF	;PSW 寄存器
.FORM		

;常量说明

;---控制寄存器位---		
S0	=00	;微总线时钟被位 0 分频
S1	=01	;微总线时钟被位 1 分频
IEDG	=02	;外部中断沿 ;极性选择(0=上升沿, ;1=下降沿)
MSEL	=03	;允许微总线功能 ;---SO 和 SK---
TRUN	=04	;启动/停止定时器/计数器 ;(1=启动,0=停止)
TEDG	=05	;定时器输入沿极性选择 ;(0=上升沿,1=下降沿)
CSEL	=06	;选择捕获方式 ;选择定时器方式
TSEL	=07	; ;---PSW 寄存器---
GIE	=00	;总中断允许
EIN	=01	;外部中断允许
BUSY	=02	;微总线移位忙
IPND	=03	;外部中断挂起
ENTI	=04	;定时器中断允许
TPND	=05	;定时器中断挂起
C	=06	;进位标志
HC	=07	;半进位标志
; * * * RAM 定义 * * *		
BCDLO	=000	;用 BCD 低字节计算

BCDHI	=001	;重量 ;用 BCD 高字节计算
MWBUF0	=003	;重量 ;LCD7 段数据显示
MWBUF1	=004	;口 L
MWBUF2	=005	;口 D
OFF1	=006	;口 G
OFF2	=007	;7 段编码表的偏移寄存器
OFF3	=008	
STALO	=009	;起始值,低字节
STAH1	=00A	;起始值,高字节
ENDLO	=00B	;结束值,低字节
ENDHI	=00C	;结束值,高字节
DIV0	=00D	;DINBI248 程序的除数
;022..02F 保留为 COP820 的堆栈		
;062..06F 保留为 COP840 的堆栈		
; * * * * * 寄存器定义 * * * * *		
COUNT	=0F0	
COUNT2	=0F1	
COUNT3	=0F2	
FIAG	=0FF	;标志寄存器
; * * * * * 位定义标志寄存器 * * * * *		
POUND	=04	;POUND=1,显示磅段 ;POUND=0,显示公斤段
; * * * * * 口 G 位定义 * * * * *		
BP1	=05	;底板 1
BP2	=04	;底板 2
;255 脉冲时间,使用定时器输入捕获方式		
FMEAS:		
		;周期时间=
		;(起始值-结束值)*tc/255
		;起始值与结束值的差
		;存入 ENDLO 和 ENDHI 中
LD	COUNT,#000	;装入脉冲计数器(255 脉冲)
LD	X,#TAULO	;指向自动装载寄存器低字节
LD	B,#TMRLO	;预置定时器
LD	[B+],#0FF	;FFFFH 送入寄存器

	LD	[B], # 0FF	
	LD	B, # CNTRL	
	LD	[B+], # 0D0	; CNTRL 寄存器: 定时器捕获
			; 方式, TIO 正沿触发,
			; 启动定时器
	RBIT	# TPND, [B]	; 预置定时器挂起标志
L1:			
	IFBIT	# TPND, [B]	
	JP	SSTORE	
	JP	L1	
SSTORE:			; 存储起始值
	RBIT	# TPND, [B]	
	LD	[X+]	; 装载定时器捕获寄存器的
			; 低字节
	X	A, STALO	; 存入 RAM
	LD	A, [X-]	; 装载高字节捕获,
			; 指向低字节捕获
	X	A, STAHI	; 存入 RAM
	LD	B, # PSW	
L256:			
	IFBIT	# TPND, [B]	
	JP	DCOU	
	JP	L256	
DCOU:			
	RBIT	# TPND, [B]	; 预置定时器挂起标志
	DRSZ	COUNT	; 脉冲计数器减 1
			; 计数器=0?
	JP	L256	; 否, 循环直至测量到 255 个脉冲
ESTORE:			; 存储结束值
	LD	CNTRL, # 00	; 停止定时器
	LD	B, # STALO	; 指向起始值低字节
	LD	A, [X+]	; 装载结束值低字节
	X	A, [B]	; 送起始值低字节到 ACCU
			; 和送结束值低字节到 STALO
	SC		
	SUBC	A, [B]	; 从起始值低字节中
			; 减去结束值低字节
	X	A, [B+]	; 结果存入 STALO, 指

		;向 STAHI
LD	A,[X]	;送到 ACCU
X	A,[B]	;送起始值高字节到 ACCU
		;和送起始值高字节到 STAHI
SUBC	A,[B]	;从起始值高字节减去
		;结束值高字节
X	A,[B]	;结果存入 STAHI
.END		

§ 6.4 COP8 单片机脉冲调宽 A/D 转换方法

采用单片机中的脉冲调宽 PWM 功能,也可以实现低价高效的 A/D 转换,约在 10ms 时间完成 8 位 A/D 转换。通过电容把 PWM 输出波形充电到平均电压作为比较器的输入模拟量的比较值(若 RC 时间常数远大于脉冲宽度),其计算如下:

$$V_{in} = V_{ref} [T_{on} / (T_{on} + T_{off})]$$

为了测量和计算方便,则上式可为:

$$V_{in} = V_{ref} [T_{on} \text{ 和 } / (T_{on} + T_{off}) \text{ 和}]$$

图 6-8 给出基本的原理图,采用了 LM2901 比较器,两个 100kΩ 电阻及 0.047μF 薄膜电容。采样时间为 20μs,时间常数为 4.7ms。

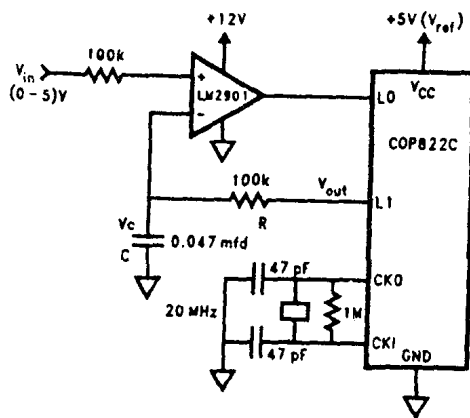


图 6-8 PWM 脉冲调宽 A/D 转换基本原理

图 6-9 是脉冲调宽 A/D 转换的流程图。首先,将软件计数器 T_{on} 和 TOTAL 预置为 FF,接着将累加器和 0F1 寄存器初值为 2。然后配置 L 端口来完成单片机初始化。

用 IFBIT 0,0D2 指令检查比较器的输出。从而调整 PWM 输出。每一次采样,DRSZ 指令将 TOTAL 计数器减 1。当 TOTAL 计数器减到 0,初始循环完成了,第一次转换之后,IFEQ A,0F1 指令将为真,并将 FF 值重新加载到 T_{on} 及 TOTAL 寄存器。电容器上电压等于 V_{in} ,且开始实际转换,当 TOTAL 寄存器减到 0 时(采样 255 次后),转换就完成了。 T_{on} 将不再重新加载,而用 T_{on} 加载累加器,用 X A,00 指令存储在 00 单元之中。

最后的两条指令(RBIT 1,LONF 和 RBIT 1[B])是可选的,用来防止电容上电荷在转换期内明显地减少,并允许加快电容初始化时间。

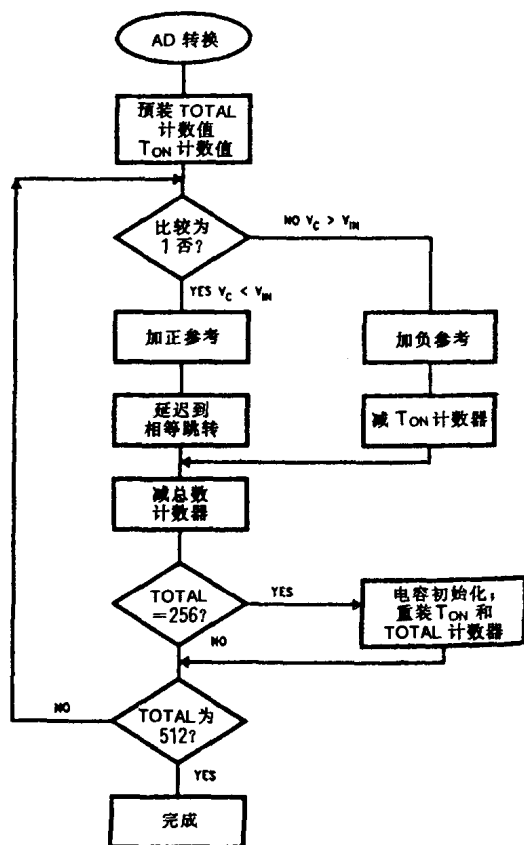


图 6-9 脉冲调宽 A/D 转换流程图

相应的程序如下:

; 以下的程序清单可工作在所有 COP800 微控制器中

; 首先为第一个转换初始化, 其次将结果存入 RAM 单元 00。

.CHIP820

LCONF=0D1

LDATA=0D0

TON=0F2

TOTAL=0F0

;

LD A, #02	; 用于确定何时再装入
LD TOTAL, #0FF	; 预加载总计数器
LD 0F1, #2	; 乘数(255 送入 INIT, 结果加 255)
LD TON, #0FF	; 预加载 Ton
LD 0FE, #0D0	; 加载 B 寄存器指向 LDATA 寄存器
LD LDATA, #01	; L 口数据寄存器, L0=弱上拉, L1=高

	LD LCONF, #02	;L 口配置寄存器, L0=输入, L1=输出
LOOP:	IFBIT 0, 0D2	;测试比较器输入
	JP HIGH	;假如 L0=1 跳转
	NOP	
	NOP	;置位和复位的补偿时间
	RBIT 1, [B]	;使 L1 为低
	DRSZ Ton	;当为低时 Ton 减 1
	JMP COUNT	
HIGH:	SBIT 1, [B]	;使 L1 为高
	NOP	
	NOP	
	NOP	
	NOP	
	NOP	
	NOP	;补偿高和低循环
COUNT:	DRSZ TATAL	;总计数减 1
	JP LOOP	;
	RBIT 1, LCONF	;三态 L1 使额外周期错误最少
	RBIT 1, [B]	;
	IFEQ A, 0F1	检查初始化循环是否完成
	JP RELOAD	;逻辑真时跳转
	JP DEC	;如不是第二个循环的结束则跳转
RELOAD:	LD 0F2, #0FF	;用 FF 重加载 T on
	LD 0F2, #0FF	;同步 TOTAL 和 T on计数器
DEC:	SBIT 1, [B]	;设置 L1 为高
	SBIT 1, LCONF	;恢复 L1 为输出
	DRSZ 0F1	;乘数减 1 直至为 0
	JMP LOOP	;继续 A/D 直至第二次转换后
	LD A, TON	;将 T on装入 A
	X A, 00	;将结果存入 RAM 单元 00 中
	.END	

NS 公司 AN-607 的“Pulse Width Modulation A/D Conversion Techniques with COP800 Family Microcontrollers”应用文选中,还介绍了 8 通道的方法,以及采用该方法的提高精度途径。

§ 6.5 COP8 单片机定时器应用方法

COP8 单片机具有功能较强的 16 位定时器,有输入捕获,外事件计数及 PWM 等三种工作方式,本节介绍定时器的几种应用方法。

6.5.1 COP8 单片机定时器 PWM 应用方法

用 COP8 单片机定时器的 PWM 工作方式,可以实现 D/A 转换,马达控制和 AC 马达可控硅控制等应用。

一、D/A 转换

通过控制脉冲调宽 PWM 中的 T_{ON} 及 T_{OFF} 的占空比,来调整 PWM 输出波形。在 PWM 输出端接上 RC,在电容上可得到相应模拟值,从而达到简单的 D/A 转换的目的。

上一节的用 PWM 的 A/D 转换实际上就是采用 PWM 的 D/A 转换来实现的。图 6-10 是 PWM 实现 D/A 转换示意图。

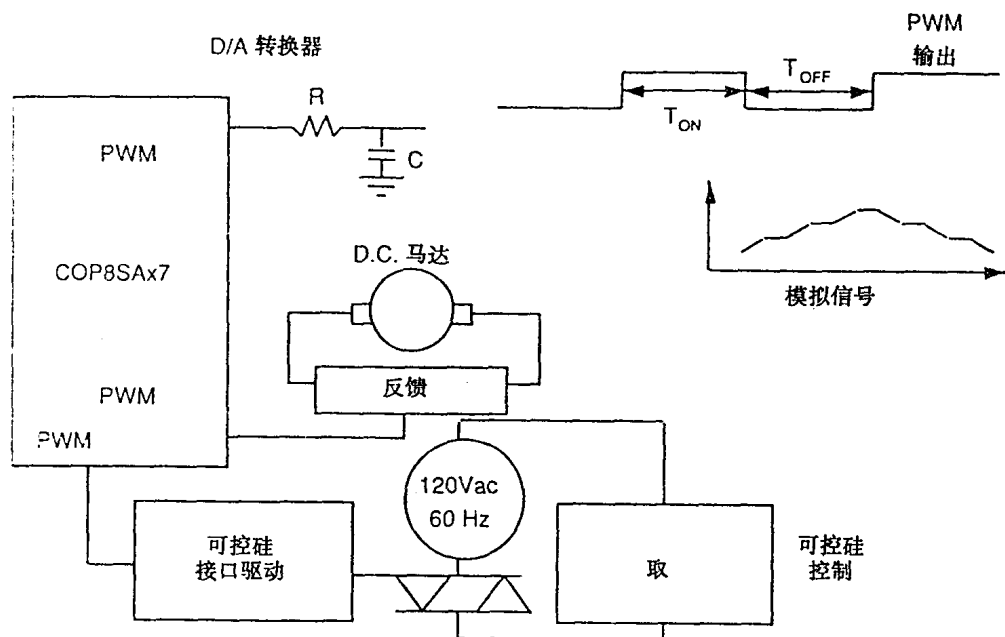


图 6-10 PWM 的 D/A 转换及马达控制示意图

二、马达控制

用 PWM 来实现 DC 马达的伺服控制,同样,通过变化 PWM 的 ON 及 OFF 时间(变化占空比),以及变化周期时间。高输出(100%占空比)可使马达转得最快,低输出(0%占空比)使马达停止,动态地变化 PWM 占空比,可以很平滑地控制马达转速。

相应的控制程序如下:

UPTML0 =020

UPTMHI	=021
UPTMLO	=022
UPTMHI	=023

T1L	=0C0
T1H	=0C1
T1RALO	=0C2
T1RAHI	=0C3
T1RBLO	=0C4
T1RBHI	=0C5
T1CNTRL	=0C6

PORTLD	=0D0
PORTLC	=0D1

INITIALIZATION:

字节/周期数

T1INIT:	LD	PORTLC, #0FF	3/3
	LD	PORTLD, #0	3/3
	LD	A, DNTMLO	2/3
	X	A, T1L	2/3
	LD	A, DNTMHI	2/3
	X	A, T1H	2/3
	JSR	UPDATE	2/5
			16/23
			11/46
		全部初始化时间 =	27/69

CHANGE:

PWMCHG:	_____	设置新的运行周期时间
	_____	UPTMLO, UPTMHI,
	_____	DNTMLO, DNTMHI

	LD	T@CNTRL, #0B5 3/3

INTERRUPT:

. =OFF(中断地址)

INTR:	VIS	1/7
T2INTR:	JSR UPDATE	2/5
	RETI	1/5
UPDATE:		4/17
	LD B, #T1RALO	2/2
	LD X, #UPTMLO	2/3
UPDLUP:		
	LD A, [X+]	1/3
	X A, [B+]	1/2
	IFBNE #6	1/1
	JP UPDLUP	1/3-1
	LD [B], #0B0	2/2
	RET	1/5

11/46

4/17

11/46

整个中断时间

=15/63

全部周期数

=16+3+ +15=34+

三、AC 马达可控硅控制

图 6-11 是 AC 马达可控硅控制的示意图。

为了能控制可控硅的周期基(cyclic basis),需要有一个精确时间基准。这可以通过交流电源(50Hz/60Hz)来提供,经过过零检测得到一个简单实时时钟。

采用 COP8SA×7 单片机能对噪声或半精确的过零检测电路进行补偿,通过软件的延迟和去抖动算法来完成。用所给的 AC 交流波形的参数点,把波形划分成有效可分配处理时间变得容易了。

相应的程序如下:

;这是通用的光亮度调节程序

;时钟 10MHz(1μs 指令周期)

.INCLD COP888. INC

.TITLE TIMER, 'TIMER APPLICATION EXAMPLE'

.SECT TRIAC, REG ;初始化

TEMP; .DSB 1

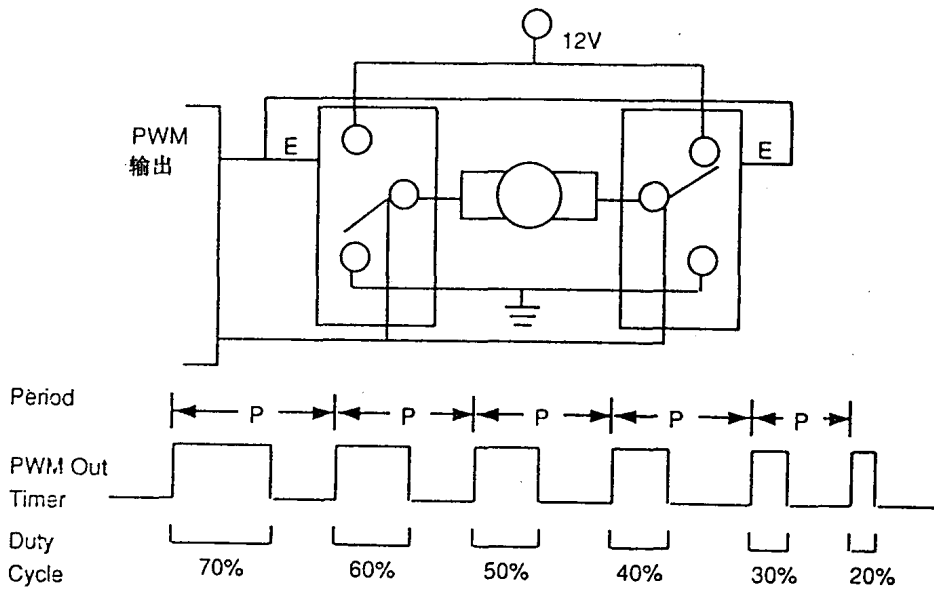


图 6-11 AC 马达可控硅控制示意图

```

LEVEL:      .DSB 1
FIN:        .DSB 1
REG1:       .DSB1
.SECT       MAIN,ROM
LD          FIN, #000
LD          LEVEL, #040
LD          PORTGC, #000
LD          PORTGD, #004
LD          CNTRL, #080
LD          PSW, #000
LD          TMR1LO, #07D
LD          TMR1HI, #000
LD          T1RALO, #0EB
LD          T1RAHI, #003

BEG:        IFBIT    2,PORTGP          ;上电同步或复位同步
            JP        HI
            JP        BEG
HI:          IFBIT    2,PORTGP
            JP        HI

            ;测试是否真的过零(有效

```

;转送)为零出错检测去抖动

```
        JSR      DELAY
        IFBIT    2,PORTGP
        JP       BEG
DOIT:    JMP      INIT
```

```
LO:      IFBIT    2,PORTGP
        JP       D1
        JP       LO
D1:      JSR      DELAY
        IFBIT    2,PORTGP
        JP       DOIT
        JP       LO
```

;明/暗度主程序

;真正过零被检测

;延迟 1ms 以取得最大值

```
INIT:    JSR      TIMER
        LD       A,FIN
        IFEQ     A,#015
        JP       THER
BEGG:    INC      A
        X        A,FIN
        JP       FIRE
THER:    LD       FIN,#000
        LD       A,LEVEL
        DEC      A
        X        A,LEVEL
        LD       A,LEVEL
        IFEQ     A,#000
        JP       LP2
        JP       LP3
LP2:     LD       LEVEL,#040
        JP       FIRE
LEP3:    IFBIT    5,LEVEL
        JSR      ADD
        JSR      SUB
        NOP
        NOP
```

;点火子程序

```
FIRE:  LD    PORTD, #0FF
        X     A, FEMP
        CLR   A
LP6:    INC   A
        IFEQ  A, #03
        JP    LP5
        JP    LP6
LP5:    CLR   A
        LD    PORTD, #00
TWO:    X     A, TEMP
HI1:    IFBIT 2, PORTGP
        JMP   HI
        JMP   LO
```

;延迟子程序

```
DELAY: LD    REG1, #00F
LOOP:  DRSZ  REG1
        JP    LOOP
        RET
```

;由要求的延迟对定时器减 1

```
SUB:   LD    A, T1RALO
        SUBC  A, #07D
        X     A, T1RALO
        LD    A, T1RAHI
        SUBC  A, #000
        RC
        X     A, T1RAHI
        RET
```

;由要求的延迟对定时器加 1

```
ADD:   LD    A, T1RALO
        ADC   A, #07D
        X     A, T1RALO
        LD    A, T1RAHI
        ADC   A, #000
```

```

RC
X      A,T1RAHI
RETSK

```

;定时器子程序

```

TIMER:  SBIT      T1C0,CNTRL
LP1:     IFBIT     T1PNDA,PSW
        JP        LP4
        JP        LP1
LP4:     RBIT      T1C0,CNTRL
        RBIT      T1PNDA,PSW
        RET
        .END

```

6.5.2 COP8 单片机定时器输入捕获应用方法

COP8 单片机定时器捕获工作方式,可用来测量捕获事件间的时间,图 6-12 给出了 COP8SA×7 单片机用来测量马达的转速。每转一圈,通过磁传感器来产生一个脉冲。

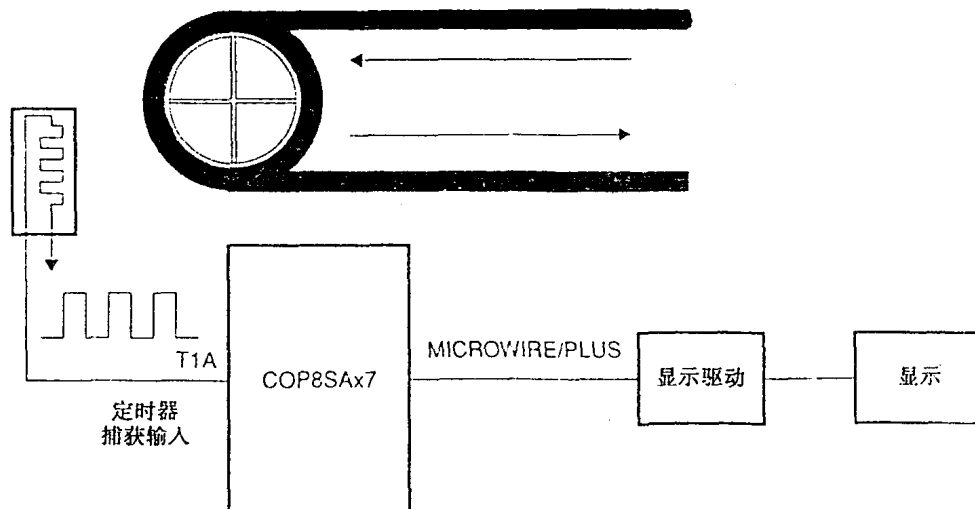


图 6-12 定时器捕获应用示意图

单片机中的定时器 T1 被设置为 T1AE 跳沿产生中断,其初值为 0FFFFH,定时器 T1A 输入端的跳沿,使现时的定时器值送至 R1A 寄存器,并置起定时器中断挂起标志,从而使程序转至 0FF 存储单元,通过 VIS 指令实现向量中断。中断服务子程序将把定时器中断挂起标志 T1PNDA 复位,然后读 R1A 寄存器的值存入 RAM,RETI 指令返正常程序执行,并使能下一中断(把 GIE 位置位)。

在下一上升沿出现在 T1A 时,程序再次进入中断服务子程序。R1A 寄存器的值再次读

入,并与先前的值进行比较。两次捕获的不同值乘以指令周期时间,给出了一次转动时间,这也很容易转换成频率,通过 MICROWIRE/PLUS 微总线接口来显示驱动。

相应的程序如下:

```
.INCLD  COP888XX.INC
.SECT   TMR,ROM

                                ;定时器 T1 构置
        LD      PORTGC,#00
        LD      PORTGD,#08
        LD      TMR1LO,#0FF
        LD      TMR1HI,#0FF
        LD      ICNTRL,#00
        LD      CNTRL,#0C0
        LD      PSW,#011
SELF:    JP      SELF

.SECT   INTERRUPT,ROM,ABS=0FF    ;定时器中断处理子程序
        VIS

                                ;定时器服务子程序
T1SERV:  IFBIT    T1C0,CNTRL      ;若 T1 溢出
        JP      UNDFLW          ;则处理定时器溢出
        RBIT     T1PNDA,PSW      ;否则复位 T1PNDA 挂起标志
        :                               ;(定时器捕获过程)
        RETI                    ;从中断返回
UNDFLW:  RBIT     T1C0,CNTRL      ;复位定时器溢出挂起标志
        .                               ;(定时器溢出过程)
        RETI                    ;从中断返回

                                ;出错子程序(无中断挂起)
ERROR:   RETI                    ;从中断返回
        .SECT   INTTABLE,ROM ABS=01E0 ;向量表

.ADDRW  ERROR
        . . .
.ADDRW  T1SERV
        . . .
.ENDSECT
```


6.5.3 COP8 单片机定时器外部事件计数器应用方法

COP8 单片机定时器外部事件应用方法十分类似于 PWM 工作方式的操作。唯一的不同是定时器时钟源来自于外部。该方式有能力执行控制一个系统的计数,预测外部事件的数目。例如,磁盘的第几扇区搜索或装配生产线上第几道工序的测试。

相应的程序如下:

```

        .INCLD  COP888XX. INC
        .SECT   EEC,ROM

                                           ;定时器 T1 构置
RBIT    3,PORTGC
RBIT    3,PORTGD
LD      CNTRL,#00
LD      ICNTRL,#00
LD      TMR1LO,#COUNT0
LD      TMR1HI,#COUNT1
LD      T1RALO,#Count0
LD      T1RAHI,#Count1
LD      T1RBLO,#Count0
LD      T1RBHI,#Count1
SBIT    T1C0,CNTRL
LD      PSW,#011
SELF:   JP      SELF

        . SECT   INTERRUPT, ROM,
        ABS=0FF                                           ;定时器中断处理子程序

        VIS

T1SERV:  RBIT    T1PNDA,PSW                                ;定时器服务程序
        RBIT    T1C0,PSW                                  ;复位 T1A 挂起标志
                                           ;停止定时器
                                           ;(定时器中断过程)
        SBIT    T1C0,PSW                                  ;启动定时器
        RETI                                              ;中断返回

ERROR:  RETI                                              ;出错子程序
                                           ;(无中断挂起)
        . SECT   INTTABLE,ROM,
        ABS=01E0                                           ;从中断返回
                                           ;向量表

```

```
.ADDRW      ERROR
           . . .
.ADDRW      T1SERV
           . . .
.ENDSECT
```

§ 6.6 COP8 单片机微总线 MICROWIRE/PLUS 应用方法

COP8 单片机的微总线 MICROWIRE/PLUS 是很有特色的串行接口,可以通过它,外扩存储器或 I/O。其应用实例可见 3.2.2 节的新型 COP8 单片机 MICROWIRE/PLUS 微总线接口。

第七章 COP8 单片机的应用实例

本章通过若干例子说明 COP8 单片机在无绳电话、直流步进马达控制、洗衣机、空调机中的应用。目的是让读者通过这些例子熟识 COP8 的指令系统和编程方法。本章还给出了 COP8 单片机的算术运算程序包。读者可以直接引用这些子程序。

§ 7.1 COP87A×7 单片机在无绳电话设计中的应用

COP8 系列单片机能满足无绳电话的要求,提供有竞争价格的多种应用方案。COP8A×7 支持无绳电话的双机功能,一个手机和一个座机。手机完成标准的电话功能,如电话网信号传送,拨号(DTMF),安全,上次号码重拨,振铃,全双工语音通信。座机连接电话网,使用联邦通信委员会 FCC 确定的十个通道之一的 46/49MHz 无绳电话波段,通过射频与手机连接。

图 7-1 和图 7-2 给出了 COP8SAB7 应用于无绳电话的逻辑框图。

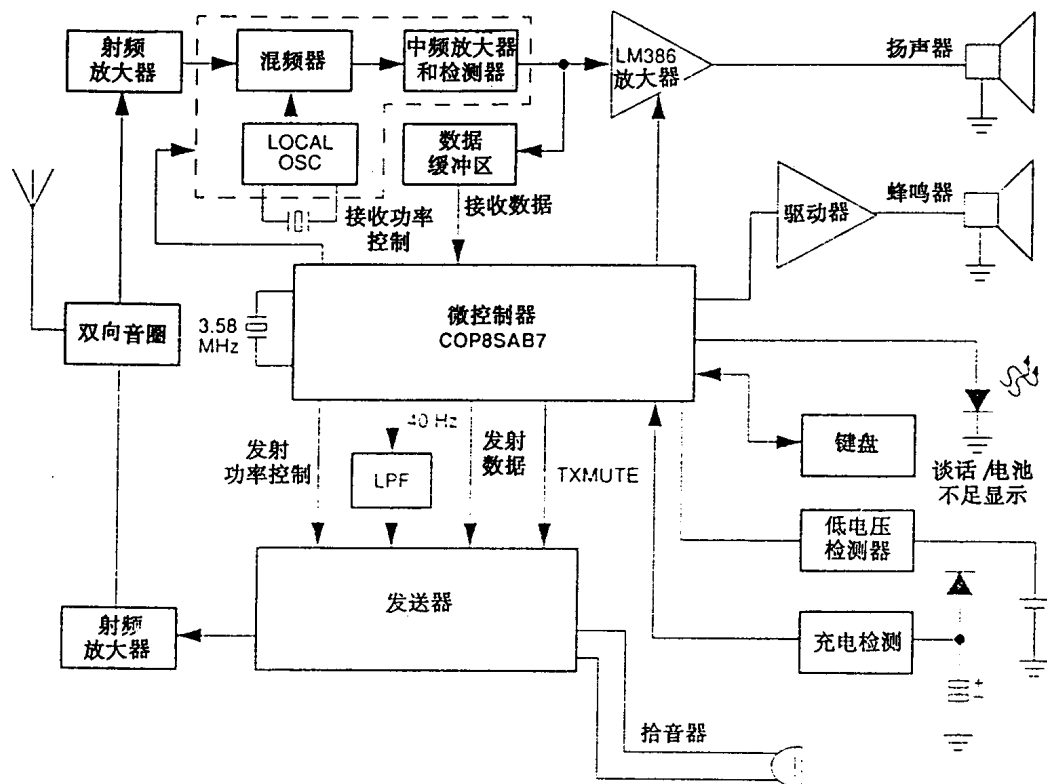


图 7-1 无绳电话逻辑框图(座机)

美国国家半导体公司提供完整的低功耗、低电压的解决方案,包括 COMBO,EEP-ROM,线性音频和 8 位单片机。

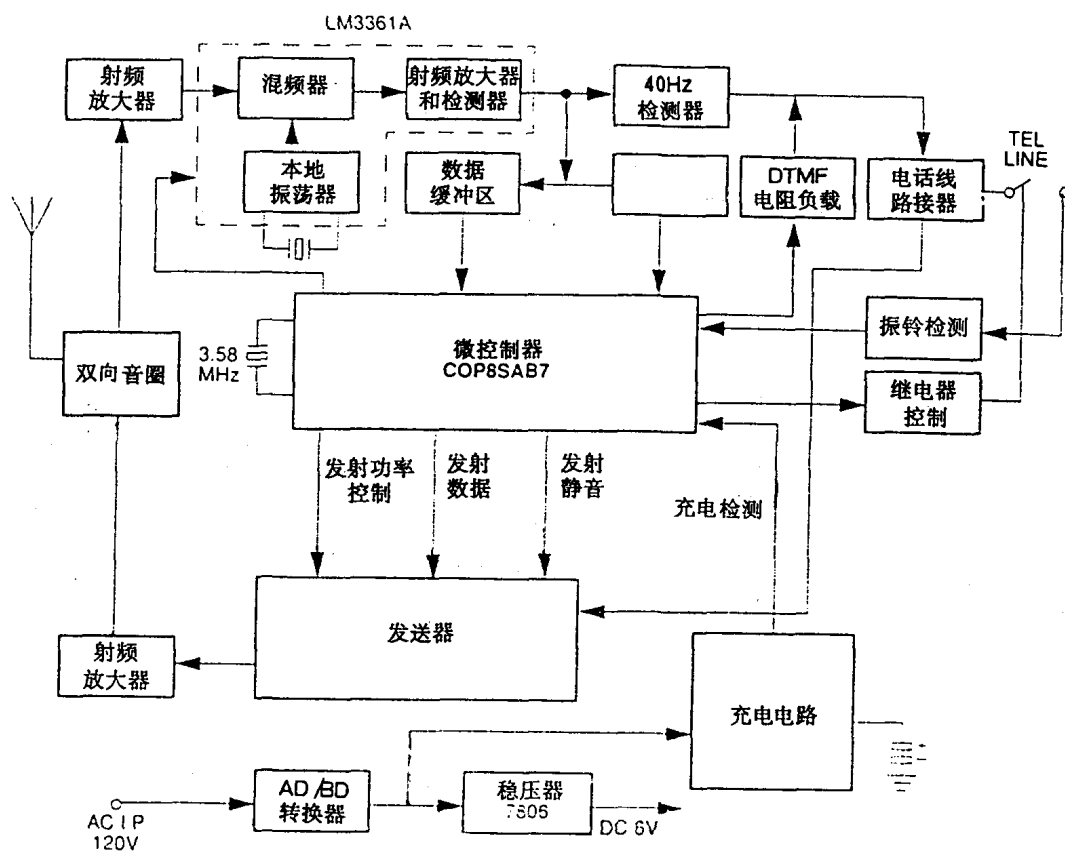


图 7-2 无绳电话逻辑框图(手机)

§ 7.2 COP8 单片机在直流步进马达控制设计中的应用

直流步进电机接收输入的电流脉冲信号转换成转子的运转。一般电机有四级绕组线圈，分别标为红，黄/白，红/白，黄色。为了使电机转动，按一定的顺序给线圈绕组通电（施加脉冲电流）。施加脉冲的顺序可以改变电机转动的方向。改变脉冲的频率可以改变电机转动的速度。

以下给出一个控制步进电机转动的例子。该例子控制电机逆时针或顺时针单步运转,运转速度分为四档。

图 7-3 和图 7-4 分别给出了直流步进马达控制原理图和程序流程图。

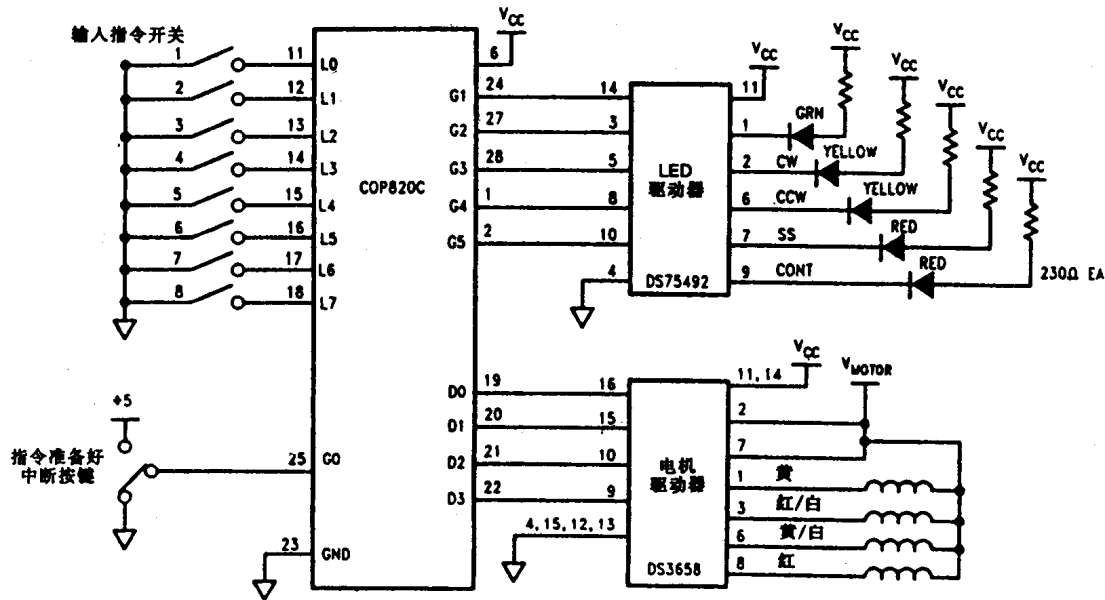


图 7-3 直流马达控制原理图

步进电机运转的速度由 COP8 的 16 位定时器控制。用不同的值设定定时器的计数值,在计数器溢出时产生中断,控制电机转动一步。本例中电机运行的速度是 25 步/秒,100 步/秒,200 步/秒,400 步/秒。表 7-1 给出了电机四组绕组导通和断开的四种组合。这四种组合保存在四个连续的内存单元。在定时器产生中断时,依次取出一种组合控制步进电机,就可以驱动步进电机连续转动。如果按递增的方向取数,则电机按顺时针方向转动。如果按递减的方向取数,则电机按逆时针方向转动。

表 7-1 步进电机控制序列

步数	黄	红/白	黄/白	红	输出值
0	On	Off	Off	On	9
1	On	On	Off	Off	C
2	Off	On	On	Off	6
3	Off	Off	On	On	3
4	On	Off	Off	On	9

在单运行时,每次步进转子转动 1.8 度。转子转动 360 度需要执行 200 次操作。

以下为控制程序清单。

;RAM 定义

```
;00      (MS0)步进电机驱动值 09H
;01      (MS0)步进电机驱动值 0CH
;02      (MS0)步进电机驱动值 06H
```

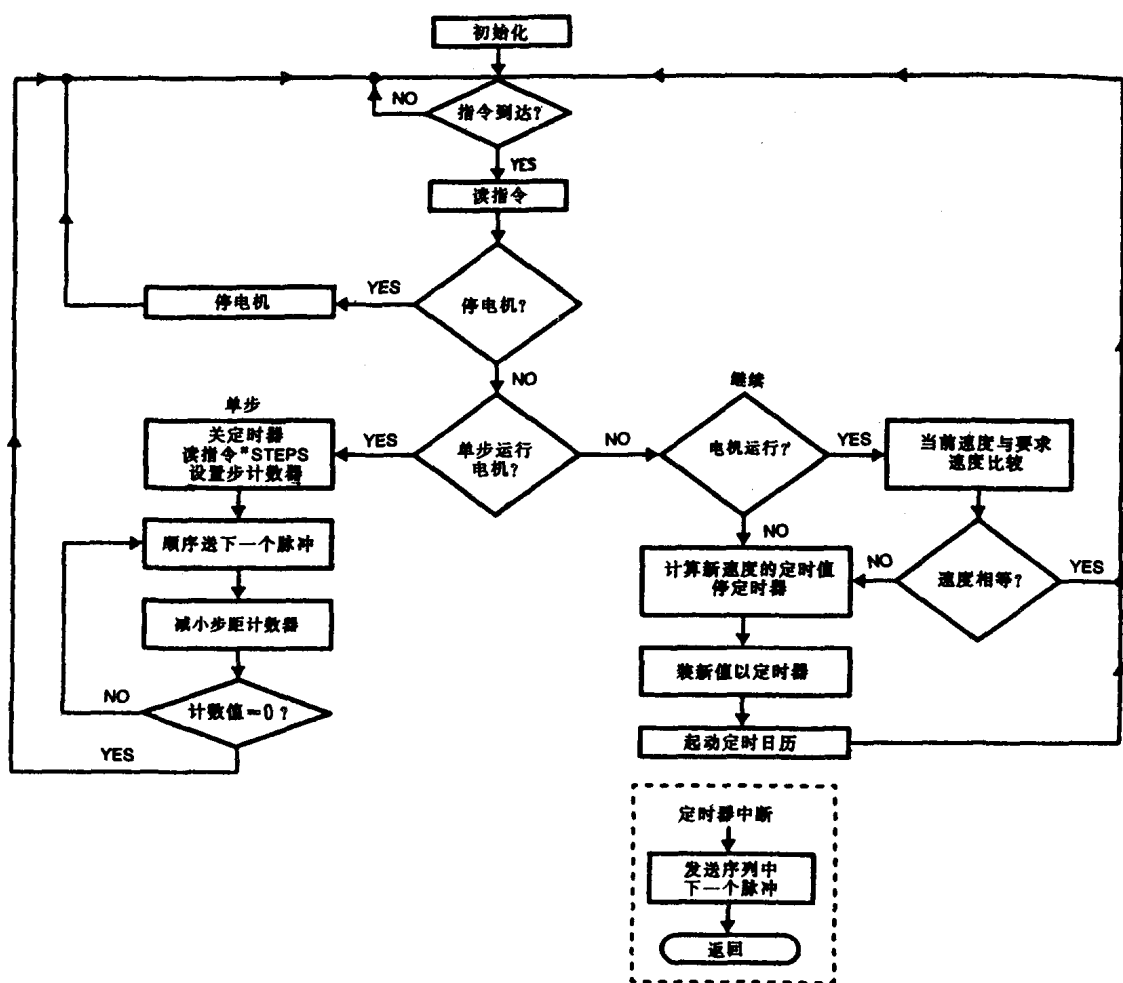


图 7-4 直流马达控制程序流程图

```

;03      (MS0)步进电机驱动值 03H
;04      (CMD)控制命令:
;         7-4 电机速度,某单步的高半字节
;         3 未使用
;         2 =1 单步, =0 连续
;         1 =1 顺时针, =0 逆时针
;         0 =1 电机停止, =0 电机运转
;05      单步数目的低字节
;07      标志寄存器
;         7-1 未使用
;         0
;14      定时器低字节数值
;15      定时器高字节数值

```

;D2 命令口状态
 ;D4
 ;DC 脉冲数据输出口
 ;F0 步进计数寄存器 0
 ;F1 步进计数寄存器 1
 ;F2 步进电机控制序列数据指针(RAM00~03)

 ;命令位

GO=0

DIR=1

MODE=2

 ;G 口

INT=1

READY=2

CW=3

SS=4

NS=5

 ;寄存器

CMD=04

STEPS=05

CREG0=0F0

GREG1=0F1

FLGREG=07

STPPTR=0F2

TVAL0=014

TVAL1=015

MS0=00

MS1=01

MS2=02

MS3=03

 ;COP820 的口分配

PORTGC=0D5

PORTGD=0D4

PORTGP=0D6

PORTLC=0D1

PORTLD=0D2

PORTD=0DC

PORTI=0D7

SIOR=0E9

TMRLO=0EA

TMRHI=0EB

TAULO=0EC

TAUHI=0ED

CNTRL=0EE

PSW=0EF

GIE=0

ENI=1

BUSY=2

IPND=3

TPND=5

IEDG=2

MSEL=3

TRUN=4

TC3=5

TC2=6

TC1=7

.CHIP 820

;寄存器初始化

; SO, #02F

LD CNTRL, #080

LD SPW, #03

;允许外部中断

LD PORTGD, #01

LD PORTGC, #03E

LD PORTD, #09

;输出脉冲值

LD PORTLC, #00

LD PORTLD, #0FF

LD B, #MS0

;生成步进电机控制序列数据

LD [B+], #09

LD [B+], #06

LD [B], #03

LD STPPTR, #00

;初始化步进电机控制序列数据指针

LD FLGREG, #00

;初始化标志寄存器

;读命令,译码并执行

TOP: SBIT READY,PORTGD
JSR WAUT
RBIT READY,PORTGD

	XA,CMD	;将命令存入CMD寄存器
	IFBIT GO,CMD	;停止位是1?
	JP STOP	;是,停止,
	IFBIT MODE,CMD	;检查方式位
	JSR SSTEP	;单步
	JSR CONT	;继续
	JP TOP	;等待下一条命令
STOP:	JSR TMRSET	;停止定时器工作
	LD PORTGD,#01	;关全部LED
	JS TOP	
	;电机单步运行	
SSTEP:	JSR TMRSET	;停止定时器
	LD PORTGD,#010	
	JSR WAIT	
	INC A	
	X A,CREG0	
	LD A,CMD	
	SWAP A	
	AND A,#0F	
	INC A	
	X A,CREG1	
TP2:	DRSZ CREG0	
	JP DO2	
	RETSK	
DO2:	LD GREG0,#0FF	
	DO:JSR NXTVAL	
	JSR DELAY	
	JP TP2	
	;电机连续运行	
CONT:	IFBIT TRUN,CNTRL	;是否已是运行状态
	JP CHKSPD	;是,检查当前速度
	JP SETGO	
CHKSPD:	JSR SPEED	
	RET	
SETGO:	JSR TMRSET	;停止定时器
	LD PORTGD,#020	
	JSR TIMVAL	;计算定时器值
	LD A,[B]	;送定时器初值

```

        X A,TMRHI
        LD A,[B-]
        X A,TAUHI
        LD A,[B]
        X A,TMRLO
        LD A,[B]
        X A,TAULO
        SBIT ENT1,PSW                      ;允许定时器中断
        SBIT TRUN,CNTRL                    ;启动定时器
        RET
        ;中断处理
        . =0FF
        IFBIT TPND,PSW
        JP TMRINT
        IFBIT IPND,PSW
        JP EXTINT
        SBIT GIE,PSW
        RETSK
TMRINT:  X A,0F9
        LD A,B
        X A,0FA
        RBIT TPND,PSW
        JSR NXTVAL
        LD A,0FA
        X A,B
        LD A,0F9
        RETI
EXTINT:  SBIT INT,FLGREG
        JSR DELAY
        RBIT IPND,PSW
        RETI
        ;以每秒步进 100 次为例,设定定时器常数
        ;1μs 定时器计数一次,1 秒/100 次=10000 微秒/次
        ;10000=02718H,故 02718H 写入定时器可获得 100 步/秒
TIMVAL: LD B,#TVAL0
        IFBIT 4,CMD
        JP SLOWER
        IFBIT 5,CMD

```

```

        JP SLOW
        IFBIT 6,CMD
        JP FAST
FASTER: LD[B+],#02                ;400 步/秒(2 转)
        LD[B],#08
        RET
FAST:   LD[B+],#088
        LD[B],#013
        RET
SLOW:   LD[B+],#018                ;100 步/秒(1 转)
        LD[B],#027
        RET
SLOWER:
        LD[B+],#054                ;25 步/秒(1/4 转)
        LD[B],#09C
        RET
        ;比较当前指令要求的电机速度和上次电机速度是否相等
        ;相等则不改变定时器的值,不相等则重新设定定时器的值
SPEED:  JSR TIMVAL                ;计算原定电机速度
        LD A,TVALO
        IFEQ A,TAULO                ;是否相等
        JP TSTHI
        RETSK                        ;不相等,重新设定定时器的值
TSTHI:  LD A,TVAL1
        IFEQ A,TAUH1
        RET                        ;相等,不改变定时器的值
        RETSK                        ;不相等,重新设定定时器的值
DELAY:  LD 0F3,#01
DLY1:   LD 0F4,#0FF
DLY2:   DRSZ 0F4
        JP DLY2
        DRSZ 0F3
        JP DLY1
        RET
        .END

```

§ 7.3 COP8SAC7 单片机在洗衣机中的应用

COP8SAC7 可以作为主要控制部件用于自动洗衣机。使用 COP8SAC7, 可以提高整机的可靠性, 增加洗衣机的功能和特色。

图 7-5 给出了自动洗衣机控制单元的方框图。

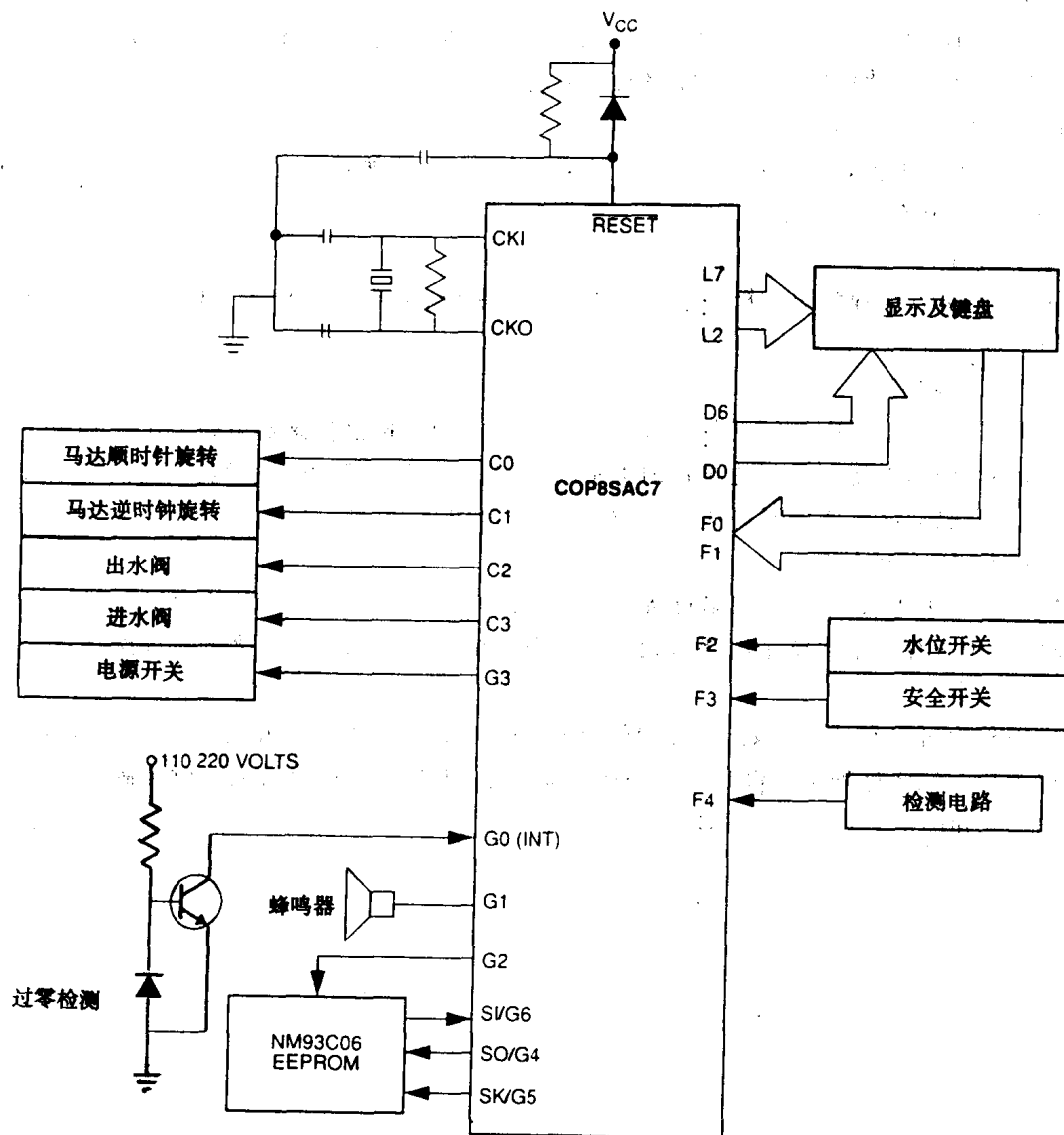


图 7-5 COP8SAC7 单片机的自动洗衣机控制单元

COP8SAC7 具有以下特点, 可以满足家电市场的需求:

- 16 位多功能定时器, 有两个自动装入寄存器
- 为 LED 和 LCD 配置的增强输出

- 大电流可编程输出
- 肖特基触发输入
- WATCHDOG/时钟监视电路
- 软件陷阱中断
- 附加的硬件中断
- MICROWIRE/PLUS 微总线界面

以下是一些主要功能的描述。

可靠性和安全性

WATCHDOG 电路用于防止应用程序进入死循环。通过初始化一个非屏蔽中断,软件陷阱可以查出程序计数器的差错(如超出正常退栈)。

LED 和 LCD 显示单元

利用 8 个大电流输出,drive-to-rail,可编程 I/O 口,功能强大的查表指令,加上两个指针寄存器可以组成一个多显示面板。

过零检测

G0 引脚可以编程为边沿极性触发的中断输入引脚,从而可以通过主电源的过零检测构成时间计数器。

其他 I/O 功能

引脚 F2 和 F3 可用于检测进水,排水,不平衡,并构成机盖传感器电路。引脚 F4 可用来监视马达的负载,以便判断所洗衣物的重量,选择适当的漂洗和甩干过程。引脚 C0~C3 和 G3 作为输出驱动继电器控制马达,出水阀,进水阀和电源开关。

外部 EEPROM 接口

MICROWIRE/PLUS 总线串行接口用来连接一个外部 256 位 EEPROM(C93C06)。该 EEPROM 用来存储上次整个洗涤过程。下次重开电源还可以按照原来的过程洗涤。

软件考虑

应用程序可以在洗涤开始选择或预置洗涤过程,也可以在洗涤中途选择或预置洗涤过程。这意味着洗涤过程不仅仅是一个顺序控制过程,也是一个可交替的实时控制过程。

图 7-6 和图 7-7 为两个程序流程图。

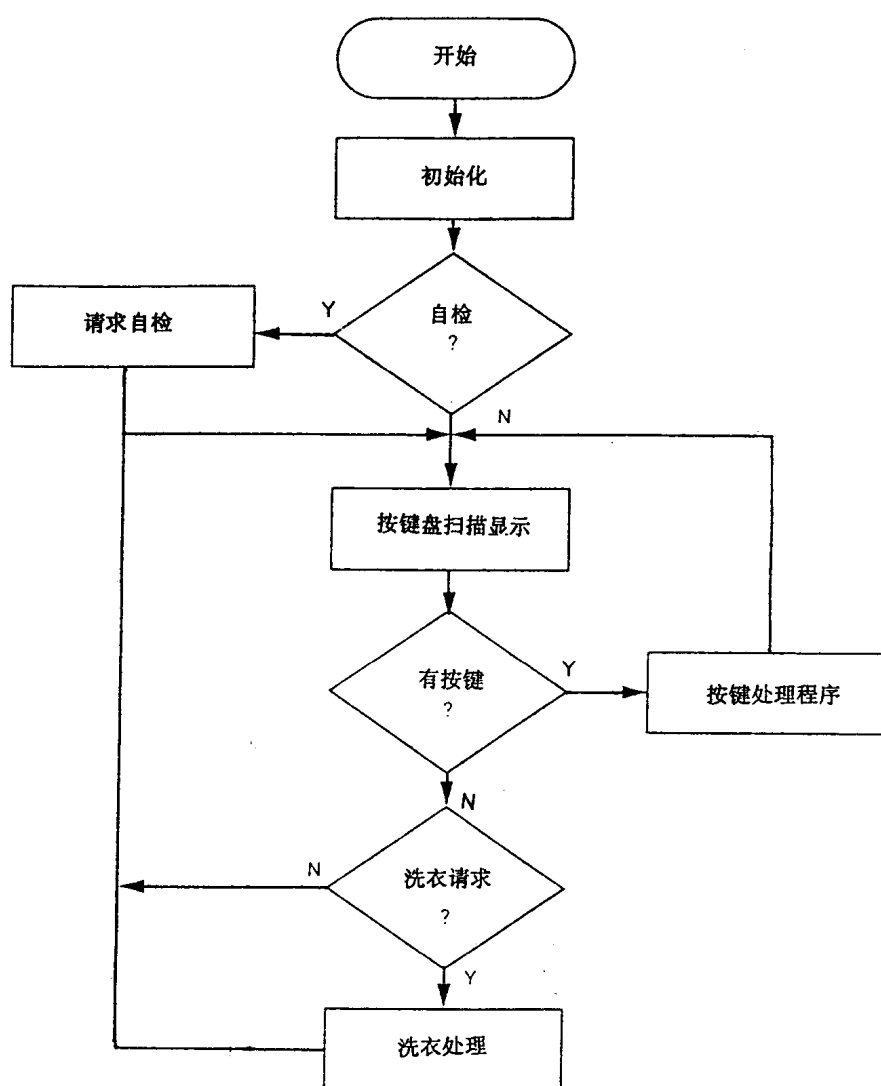


图 7-6 全自动洗衣机控制主程序流程图

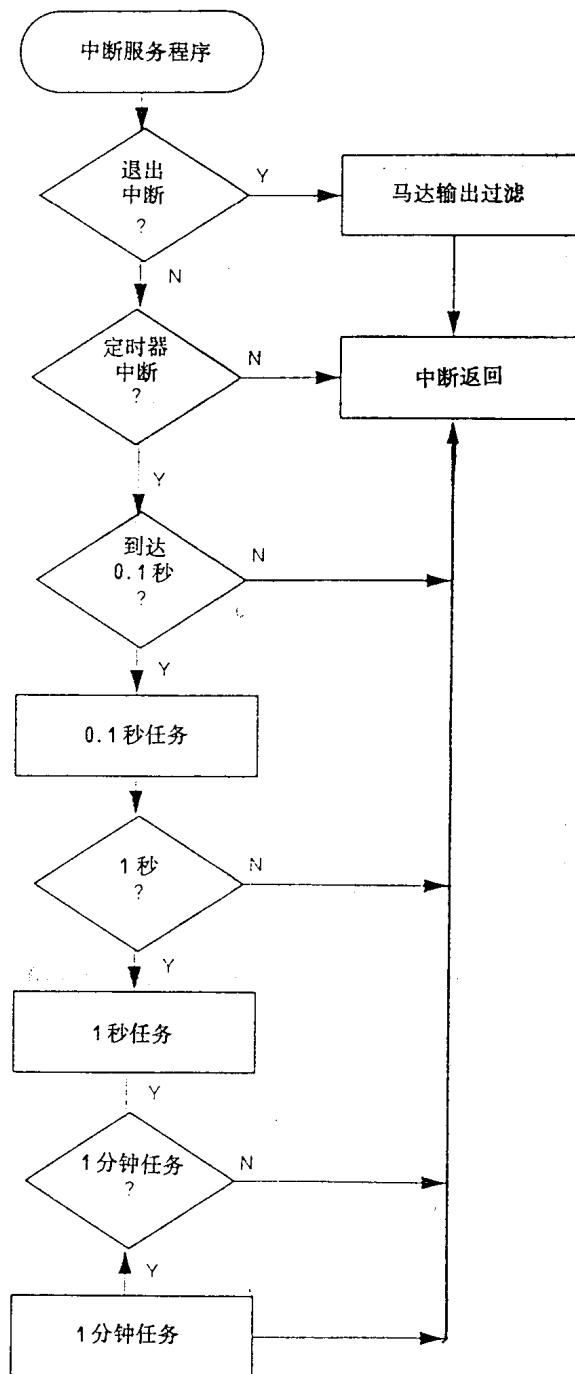


图 7-7 全自动洗衣机控制的中断程序流程图

§ 7.4 COP8SAB7 单片机在空调机中的应用

COP8SAB7 适用于空调控制器。COP8SAB7 提供控制传感器、制冷和除湿的功能。COP8SAB7 还提供电压过低和过高控制,保护压缩泵和电扇。图 7-8 是使用 COP8SAB7 组成的空调机系统图。

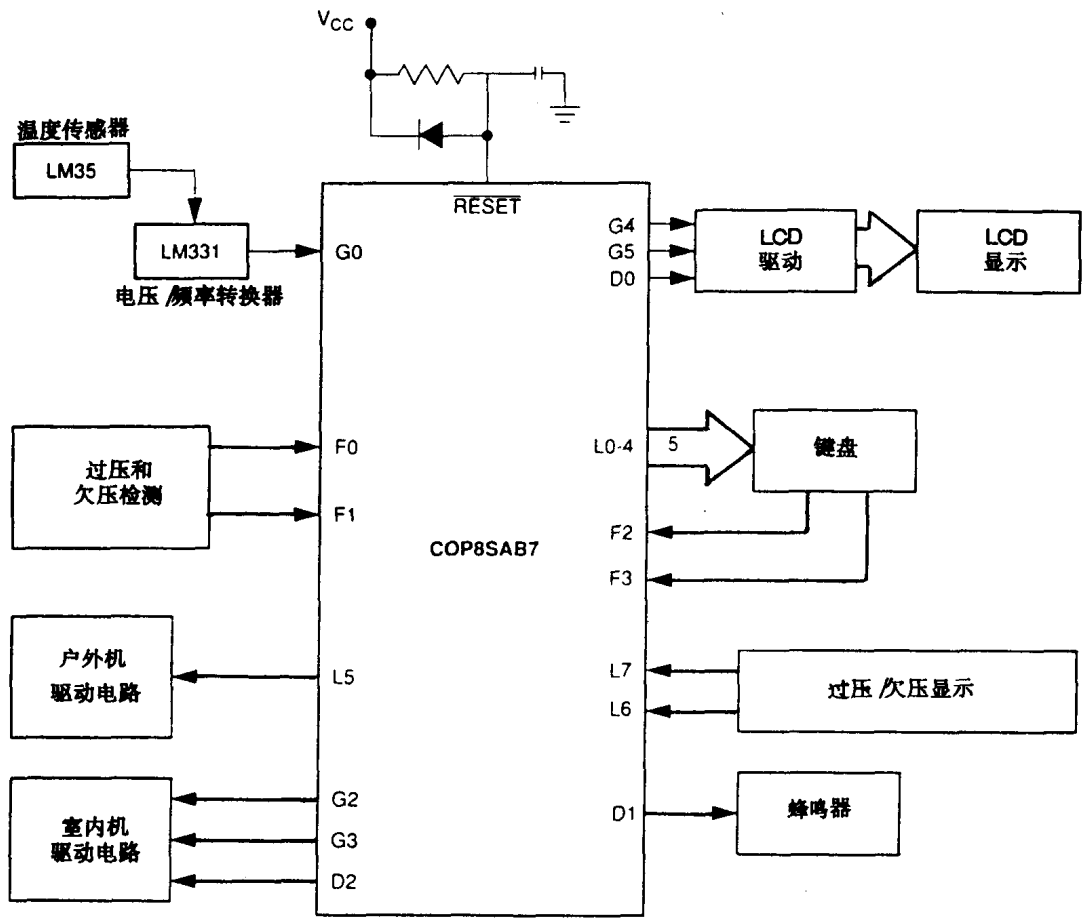


图 7-8 空调机控制模块框图

温度检测

系统使用温度传感器 LM35 产生电压作为 LM331 V/F 转换器的输入,供 LM331 进行转换。根据输入的电压值,LM331 产生成比例的输出频率,输入到 COP8SAB7 的 G0/INT 引脚。程序根据 INT 引脚产生中断的次数计算出该频率,并转换成对应的温度值。

图 7-9 为温度检测电路。

键盘扫描

图 7-10 给出了键盘的结构。引脚 L0~L4 正常情况下输出高电平。输入引脚 F2 和 F3 无按键时为低电平,有按键时为高电平。

电压过高过低检测

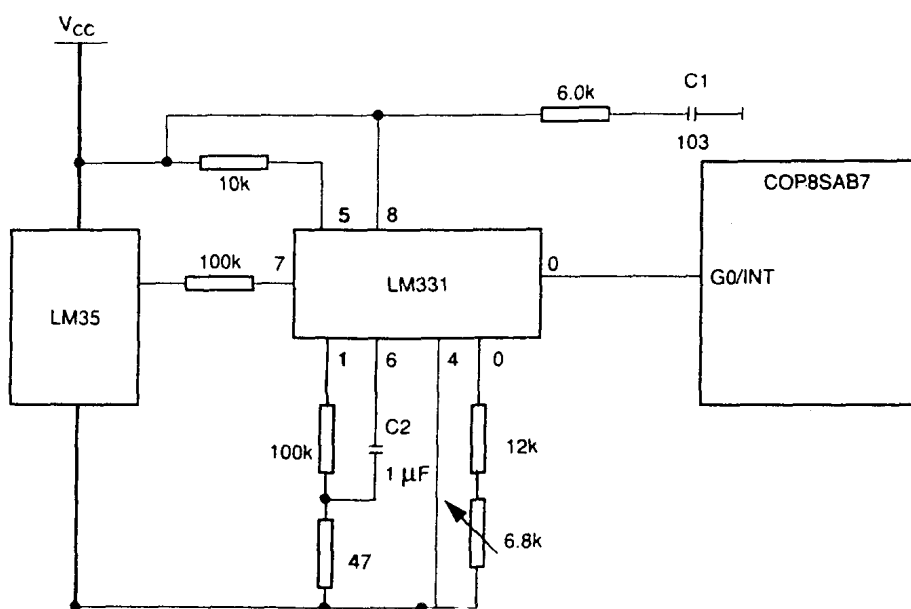


图 7-9 温度检测电路

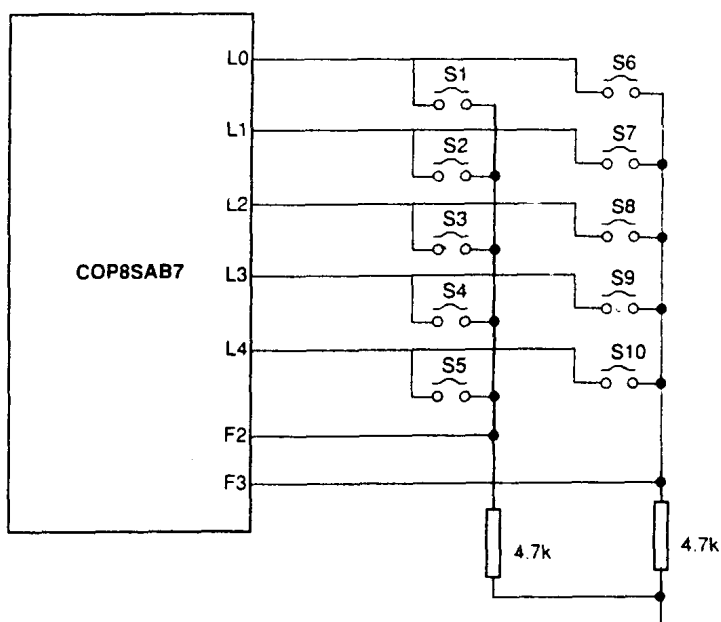


图 7-10 键盘扫描电路

主电源的电压变化由引脚 F0 和 F1 输入。在这两个引脚上读到低电平表示线电压超出正常范围。读到高电平表示线电压处在正常范围之内。原理图见图 7-11。

风扇、压缩机和蜂鸣器的驱动电路

图 7-12 是一个适当的驱动电路。继电器 J 控制压缩机的开关。三个可控硅用于控制风扇的高、中和低状态。图中只画出一个可控硅。MOC3041 是带有过零触发的光电隔离驱动

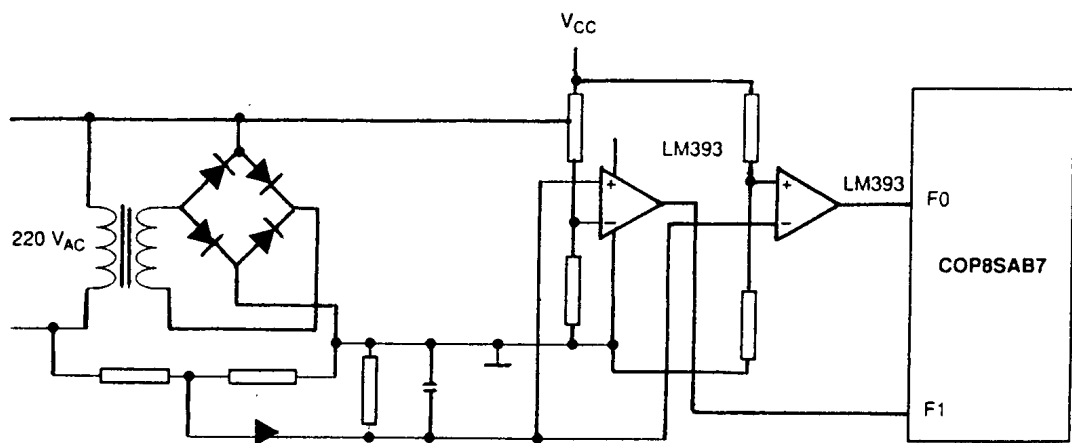


图 7-11 电压过高过低检测电路

器。当有按键时,D 引脚输出一个正脉冲使蜂鸣器鸣响。

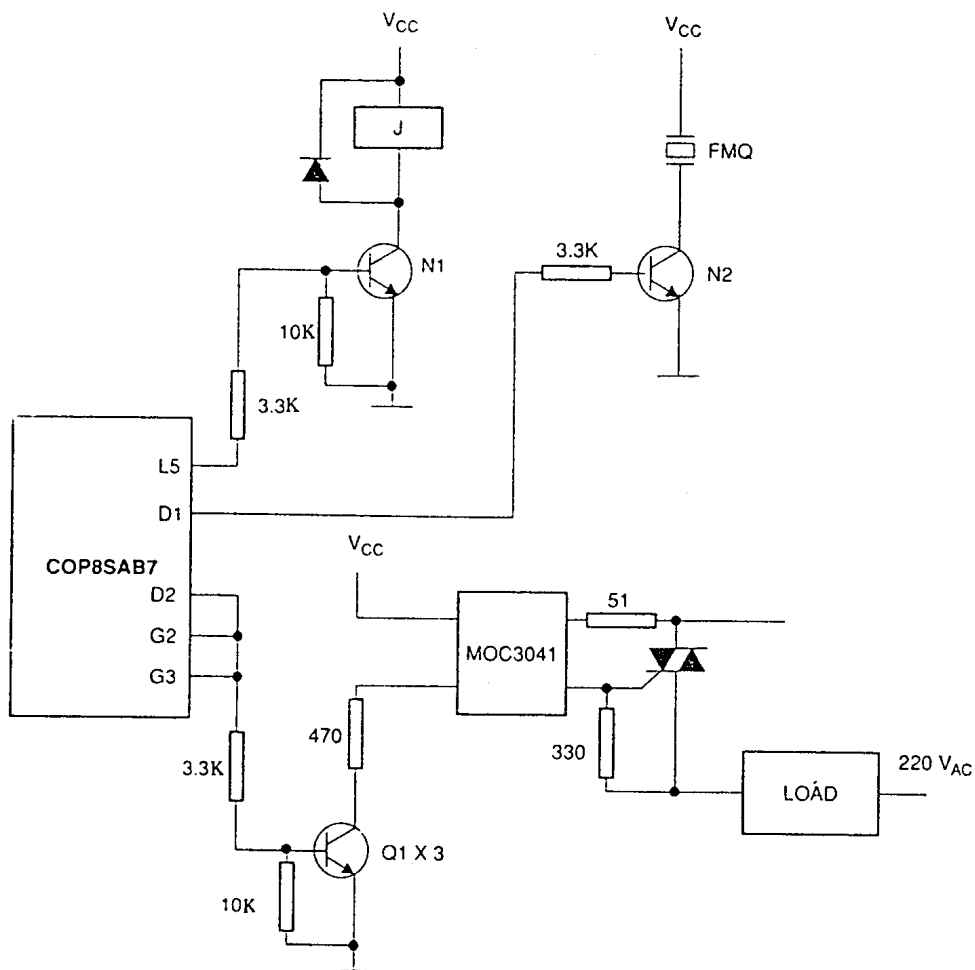


图 7-12 风扇、压缩机和蜂鸣器的驱动电路

§ 7.5 COP8 单片机 MathPak

本节讨论美国国家半导体公司的 COP8 系列微控制器的各种数学运算。其中包括二进制和 BCD 码(二进制编码的十进制数)运算。文中详细讨论了四则基本运算(加法、减法、乘法和除法),给出了二进制和 BCD 码的加法和减法的几个例子。同时提供乘法、除法和 BCD 码转换的算法。BCD 码转换为二进制码,二进制码转换为 BCD 码的程序也包括在内,还有各种乘法和除法的子程序。

四组最佳子程序为:

1. 乘法
2. 除法
3. 十进制(压缩 BCD 码)转换为二进制
4. 二进制转换为十进制(压缩 BCD 码)

一类子程序已优化成最少的 COP800 程序编码。另一类子程序优化为最少的执行时间。本节组织成四个不同的部分。第一部分给出了各种加法和减法程序,包括二进制码和 BCD 码(二进制编码的十进制数)。第二部分给出了乘法的算法,提供若干最佳的 1,2,3 和 4 个字节的乘法子程序。第三部分给出除法算法并给出若干最佳的 1,2,3 和 4 个字节操作数的除法子程序。第四部分给出了十进制数(压缩的 BCD 码)到二进制数以及二进制数到十进制数(压缩的 BCD 码)转换算法,这一部分提供了若干最佳的 BCD 码转换子程序。

COP800 算术指令包括加法(ADD)、带进位加(ADC)、带进位减(SUBC)、增 1(INCR)、减 1(DECR)、十进制调整(DCOR)、清累加器(ACC)、置进位(SC)和清进位(RC)。移位和循环移位指令也可以看作是算术运算指令的派生指令。这些指令是带进位循环右移(RRC)、累加器半字节交换(SWAP)指令,有助于编写快速乘法子程序。

二进制和 BCD 码加法和减法

在减法运算中,借位由无进位标志表示。反之,有进位标志表示无借位。因此,在减法之前,有必要置进位(表示无进位)。在加法之前同样要清进位。ADD 指令没有把进位当作操作数,也不改变进位。同时还要注意到,进位和半进位标志(分别为 PSW 的第 6,7 位)在复位时被清除,并在执行指令 ADD, INC, DEC, DCOR, CLR 和 SWAP 后保持不变。DCOR 指令使用进位和半进位两个标志。SC 指令置进位和半进位,RC 指令清除进位和半进位。

下面的程序例子示范二进制码和 BCD 码(二进制编码的十进制数)两种 4 字节数的加法指令,数据存储器中地址为 24 到 27 的四个字节内容是加数或减数,而 16 到 19 的四个字节为被加数或被减数。结果保存在地址 24 到 27 单元中。这些操作由二进制方式或 BCD 码方式完成。应当注意,加 66 的 BCD 码预制作只用于 BCD 码加法,减法不需要。二进制编码的十进制调整指令 DCOR 使用进位和半进位,但不改变它们的状态。还要注意,IFBNE #12 这条指令决定了循环只执行 4 次。因为 IFBNE 操作数是模 16 的结果(除 16 后的余数)。当 B 从 #24 加到 #28 时,其模 16 的结果为 #12,这时 B=#12,循环结束。也可以说,IFBNE 只判断 B 寄存器的低半字节。

7.5.1 二进制和 BCD 加法及减法

一、二进制加法

4.3.2 节的加法和减法运算的结果取代第二个操作数而不是第一个操作数。下面的程序中,加法和减法运算的结果取代第一个操作数。对减法来说,确保减法的结果取代被减数而不是减数。程序中 X 和 B 指针已交换使用。

```
LD B, #16          ;前面无前导 0,表示十进制数
LD X, #24
RC
LOOP: LD A,[X+]
LDC A,[B]
X A,[B+]
IFBNE #4
JP LOOP
IFC
JP OVFLOW
```

二、二进制减法

```
LD B, #010          ;有前导 0,表示十六进制数
LD X, #018
SC
LOOP: LD A,[X-]
X A,[B]
SUBC A,[B]
X A,[B+]
IFBNE #4
JP LOOP
IFNC
JP NEGRSLT
```

三、BCD 加法

```
LD B, #010
LD X, #018
RC
LOOP: LD A,[X+]
ADD A, #066
ADC A,[B]
DCOR A
```

```

X A,[B+]
IFBNE #4
JP LOOP
IFC
JP OVELOW

```

四、BCD 减法

```

LD B, #16
LD X, #24
SC
LOOP: LD A,[X+]
X A,[B]
SUBC A,[B]
DCOR A
X A,[B+]
IFBNE #4
JP LOOP
IFNC
JP NEGRSLT

```

7.5.2 乘法运算

COP800 的乘法运算都是从乘数的低位开始,积是乘数的两倍。乘数放在积的低部分。积的高部分首先清为零,然后双倍长度的积右移一位。移出的一位是乘数的低位。如果该位为 1,则被乘数加到双倍乘积的高部分。这种移位和有条件的加法重复进行。重复的次数是乘数的位数加 1。最后一次增加的移位得到正确的乘法结果。M 字节的被乘数乘以 N 字节的乘数,积的字节数是 M+N。这些乘法子程序占用 2M+N+1 个字节,原因是乘数最初是使用积的低字节。另外需要一个字节存放移位计数器(CNTR)。

一、MPY88 8 乘 8 乘法子程序(19 字节编码,180 个指令周期)

```

0[0]:被乘数(ICAND)
[1]:乘数(IER)
[2,1]:积(PROD)
MPY88: LD CNTR, #9      ;装入计数值,其值为乘数位数加 1
RC
LD A, #2
CLR A      ;清乘积高字节

```

```

M88LP:      RRC A           ;右移积的高位
            X,[B-]
            LD A,[B]
            RRC A           ;右移积的低位和乘数
            X A,[B-]
            CLR A           ;清累加器,测试乘数的低位
            IFC
            LD A,[B]
            RC
            LD B,#2
            ADC A,[B]        ;积的高位加上被乘数
            DRSZ CNTR        ;循环控制
            JP M88LUP
            RET

```

二、MUT88 快速 8 乘 8 乘法子程序(42 字节编码,145 个指令同期)

```

[0]:      被乘数(ICAND)
[1]:      乘数(IER)
[2,1]:    积(PROD)

```

```

MUT88:      LD CNTR,#3      ;装入计数值,其值为乘数位数加上 1 的 1/3
            RC
            LD B,#2
            CLR A           ;清积的高字节
ML88LP:     RRC A           ;右移积的高字节
            X A,[B-]
            LD A,[B]
            RRC A           ;右移积的低位和乘数
            X A,[B-]
            CLR A
            IFC             ;测试乘数的低位
            LD A,[B]        ;低位为 1,取出被乘数
            RC
            LD B,#2
            ADC A,[B]        ;积的高位加上被乘数
            ;
            RRC             ;重复上述指令,共 13 个指令周期

```

```

X A,[B-]          ;重复两遍
LD A,[B]
RRC A
X A,[B-]
CLR A
IFC
LD A,[B]
RC
LD B,#2
ADC A,[B]          ;第二遍结束
;
RRC                ;第三遍开始
X A,[B-]
LD A,[B]
RRC A
X A,[B-]
CLR A
IFC
LD A,[B]
RC LD B,#2
ADC A,[B]          ;第三遍结束
;
DRSZ CNTR          ;循环控制
JMP ML88LP
RET

```

三、VFM88 最快 8 乘 8 乘法子程序(96 字节编码,116 个指令周期)

```

[0]:   被乘数 (ICAND)
[1]:   乘数 (IER)
[2,1]: 积 (PROD)

```

```

VFM88: RC
LD B,#2
LD [B-],#0        ;清高位
LD A,[B]
RRC A              ;右移积低位和乘数

```

X A,[B-]	
CLR A	;清累加器并测试乘数低位
IFC	
LD A,[B]	;低位为 1,则取出被乘数
RC	
LD B,#2	
ADC A,[B]	;积的高字节加上被乘数
;	
RRC A	;右移积的高位
X A,[B-]	
LD A,[B]	
RRC A	;右移积的低位和乘数
X A,[B-]	
CLR A	;清累加器,测试乘数低位
IFC	
LD A,[B]	
RC	
LD B,#2	
ADC A,[B]	;积的高位加上被乘数
;	
RRC A	;(第一次重复上述处理一位的指令)右移积的高位
X A,[B-]	
LD A,[B]	
RRC A	;右移积的低位和乘数
X A,[B-]	
CLR A	;清累加器,测试乘数低位
IFC	
LD A,[B]	
RC	
LD B,#2	
ADC A,[B]	;积的高位加上被乘数
;	
RRC A	;(第二次重复上述处理一位的指令)右移积的高位
X A,[B-]	
LD A,[B]	
RRC A	;右移积的低位和乘数
X A,[B-]	
CLR A	;清累加器,测试乘数低位

IFC	
LD A,[B]	
RC	
LD B,#2	
ADC A,[B]	;积的高位加上被乘数
;	
RRC A	;(第三次重复上述处理一位的指令)右移积的高位
X A,[B-]	
LD A,[B]	
RRC A	;右移积的低位和乘数
X A,[B-]	
CLR A	;清累加器,测试乘数低位
IFC	
LD A,[B]	
RC	
LD B,#2	
ADC A,[B]	;积的高位加上被乘数
;	
RRC A	;(第四次重复上述处理一位的指令)右移积的高位
X A,[B-]	
LD A,[B]	
RRC A	;右移积的低位和乘数
X A,[B-]	
CLR A	;清累加器,测试乘数低位
IFC	
LD A,[B]	
RC	
LD B,#2	
ADC A,[B]	;积的高位加上被乘数
;	
RRC A	;(第五次重复上述处理一位的指令)右移积的高位
X A,[B-]	
LD A,[B]	
RRC A	;右移积的低位和乘数
X A,[B-]	
CLR A	;清累加器,测试乘数低位
IFC	
LD A,[B]	

```

RC
LD B, #2
ADC A, [B]           ; 积的高位加上被乘数
;
RRC A                ; (第六次重复上述处理一位的指令) 右移积的高位
X A, [B-]
LD A, [B]
RRC A                ; 右移积的低位和乘数
X A, [B-]
CLR A                ; 清累加器, 测试乘数低位
IFC
LD A, [B]
RC
LD B, #2
ADC A, [B]           ; 积的高位加上被乘数
;
RRC A                ; 右移积的低字节
X A, [B-]
LD A, [B]
RRC A                ; 右移积的低字节和乘数
X A, [B]
RET

```

四、MPY168 快速 16 乘 8 乘法子程序

```

[1,0]:      被乘数  (ICAND)
[2]:        乘数    (IER)
[4,3,2]:    积      (PROD)

```

```

MPY168: LD CNTR, #9      ; 装入计数值, 其值为乘数位数加 1
RC
LD B, #4
LD [B-], #0             ; 清积的高字节
LD [B-], #0
JP MP168S
M168P:  RRC A            ; 右移积的高字节
X A, [B-]
LD A, [B]

```

```

RRC A ;右移积的中间字节
X A,[B-]
MP168S: LD A,[B]
RRC A ;右移积的低字节和乘数
X A,[B]
IFNC ;测试乘数的低位
JP MP168T
RC
LD B,#0 ;将被乘数的低字节送累加器
LD A,[B]
LD B,#3
ADC A,[B] ;将被乘数的低字节加到积的中间字节
X A,[B]
LD B,#1
LD A,[B] ;取被乘数的高字节
LD B,#4
ADC A,[B] ;将被乘数的高字节加到积的高字节
DRSZ CNTR ;循环控制
JP M168LP
;
MP168T: LD B,#4
LD A,[B] ;取积的高字节
DRSZ CNTR ;循环控制
RET

```

五、MP1616 16 乘 16 乘法子程序

```

[1,0]:被乘数 (ICAND)
[3,2]:乘数 (IER)
[5,4,3,2]:积 (PROD)

```

```

MP1616: LD CNTR,#17 ;装入计数值,其值为乘数位数加 1
RC
LD B,#5
LD [B-],#0 ;清积的两个高字节
LD [B-],#0
M1616X: LD A,[B] ;5 条循环指令,用于右移积和乘数
M1616L: RRC A

```

```

X A,[B-]
IFBNE #1
JP M1616X
CLR A           ;清累加器
IFNC           ;测试乘数低位
JP M1616T
RC
LD B,#0
LD A,[B]        ;取出被乘数的低字节
LD B,#4
ADC A,[B]        ;加到积低字节前一个字节
X A,[B]
LD B,#1
LD A,[B]        ;取出被乘数高字节
M1616TY:LD B,#5
ADC A,[B]        ;加上积的高字节
DRSZ CNTR       ;循环控制
JP M1616L
RET

```

7.5.3 除法运算

COP800 除法实现的方法:被除数逐位左移到一个测试字段,该测试字段的长度与除数相同。除数紧跟在测试字段之后。在每次移位之后,做减法测试,即测试字段减去除数。如果够减,即测试字段大于或等于除数,则从测试字段中减去除数,并置位被除数的低位为1来记商为1。这种移位、测试、记商的过程重复进行。重复次数是被除数的位数。除法的结果(商)生成在被除数字段,余数生成在测试字段。

M字节的被除数和N字节的除数产生M字节的商和N字节的余数。由于测试字段的字节数与除数相同,所以这种除法算法使用 $M+2N+1$ 个存储器(RAM)字节。其中一个字节用于移位计数。

一、DIV88 8除8除法子程序

```

[0]:被除数[DD]
[2]:除数  [DR]
[0]:商    [QUOT]
[1]:余数  [测试字段]

```

```

DIV88: LD CNTR,#8      ;装入计数长度,长度为被除数位数
LD B,#1

```

```

        LD [B], #0          ;清除测试字段
DIV88S: RC
        LD B, #0
        LD A, [B]
        ADC A, [B]          ;左移被除数
        X A, [B+]
        LD A, [B]
        ADC A, [B]          ;左移被除数
        X A, [B]
        LD A, [B+]          ;将测试位送 ACC
        SC
        SUBC A, [B]          ;测试是否够减
        IFNC
        JP DIV88B            ;不够减则不保留减法结果
        LD B, #1
        X A, [B-]            ;减法结果送测试字段
        SBIT 0, [B]          ;记商为 1
DIV88B: DRSZ CNTR            ;循环次数减 1
        JP DIV88S            ;继续循环
        RET

```

二、DIV168 16(或 24,32)除 8 除法子程序

改变参数可扩展 DIVXX8 子程序,并保持字节数(26)不变

```

[1,0]:被除数[DD]    16 位
[2,1,0]:被除数[DD]  24 位
[3,2,1,0]:被除数[DD] 32 位
[3]:除数  [DR]  16 位
[4]:除数  [DR]  24 位
[5]:除数  [DR]  32 位
[1,0]:商   [QUOT]16 位
[2,1,0]:商   [QUOT]24 位
[3,2,1,0]:商   [QUOT]32 位
[2]:余数  [测试字段]16 位
[3]:余数  [测试字段]24 位
[4]:余数  [测试字段]32 位

```

```

DIV168:    LD CNTR, #16      ;装入计数长度,长度为被除数位数
                                ;DIV168:16
                                ;DIV248:24
                                ;DIV328:32
                LD B, #2      ;DIV168:2
                                ;DIV248:3
                                ;DIV328:4
                LD [B], #0    ;清测试字段
DVXX8L:    RC
                LD B, #0
DXX8LP:    LD A, [B]          ;左移被除数和测试字段
                ADC A, [B]
                X A, [B+]
                IFBNE #3      ;DIV168:3
                                ;DIV328:4
                                ;DIV328:5
                JP DXX8LP
                LD A, [B-]    ;除数送 ACC
                IFC           ;测试是否有 1 移出测试字段(见注解 1)
                JP DVXX8S
                IFGT A, [B]   ;测试除数是否大于余数
                JP DVXX8T
                SC
DVXX8S:    X A, [B]
                SUBC A, [B]   ;从余数中减去除数
                X A, [B]
                LD B, #0
                SBIT 0, [B]   ;记商为 1
DVXX8T:    DRSZ CNTR         ;循环次数减 1
                JP DVXX8L     ;继续循环
                RET

```

注解 1:在被除数的字节数大于除数的字节数,而且除数的高位为 1 的特殊场合,移位后的被除数可能移出一个 1 在测试字段,但仍小于除数,因此没有执行减法。在这种情况下,一个 1 位将移出测试字段,而且必须先执行一次减法。

三、DV1616 16(或 24,32)除 16 除法子程序

[1,0]:被除数[DD]

[5,4]:除数 [DR]
 [1,0]:商 [QUOT]
 [3,2]:余数 [测试字段]

```
DV1616: LD CNTR, #16      ;装入计数值,其值为被除数位数
        LD B, #3
        LD [B-], #0      ;清测试字段
        LD [B], #0
DV616S: RC
        LD X, #2          ;初始化 X,B 指针
        LD B, #0
DV616L: LD A, [B]
        ADC A, [B]        ;左移被除数和测试字段
        X A, [B+]
        IFBNE #4
        JP DV616L
        SC
        LD A, [X+]        ;测试字段低字节送 ACC
        SUBC A, [B]        ;余数低字节减除数低字节
        LD A, [X]          ;测试字段高字节送 ACC
        LD B, #5
        SUBC A, [B]        ;余数高字节减除数高字节
        IFNC
        JP DV616T
        X A, [X-]          ;够减,做减法
        LD A, [X]
        LD B, #4
        SUBC A, [B]
        X A, [X]
        LD B, #0
        SBIT 0, [B]        ;记商为 1
DV616T: DRSZ CNTR        ;循环计数
        JP DV616S
        RET
```

四、最小通用除法子程序(40 字节)

用于任何字节的被除数和除数

以 DV3224 为例子

32 除 24 除法子程序

[3,2,1,0]:被除数[DD]

[9,8,7]:除数 [DR]

[3,2,1,0]:商 [QUOT]

[6,5,4]:余数 [测试字段]

```
DV3224: LD CNTR, #32      ;装入计数值,其值为被除数的位数
        LD B, #6
CLRLUP: LD [B-], #0       ;清测试字段
        IFBNE #3         ;被除数的最高字节
        JP CLRLUP
DVSHFT: RC
        LD B, #0
SHFTLP: LD A, [B]
        ADC A, [B]       ;左移被除数和测试字段
        X A, [B+]
        IFBNE #7         ;除数的最低字节
        JP SHFTLP
        IFC              ;测试是否 1 移出测试字段
        JP DVSUBT
        SC
        LD X, #4
TSTLUP: LD A, [X+]       ;测试字段减去除数
        SUBC A, [B]
        LD A, [B+]       ;B 指针加 1
        IFBNE #10        ;除数最高字节地址+1
        JP TSTLUP
        IFNC             ;判是否够减
        JP DVTEST
        LD B, #7
DVSUBT: LD X, #4         ;余数减去除数
        SUBC A, [B]
        X A, [X+]
        LD A, [B+]
        IFBNE #10
```



```

        JP SUBTLP
        LD B, #0
        SBIT 0, [B]          ;记商为 1
DVTEST: DRSZ CNTR           ;循环计数
        JP DVSHFT
        RET

```

7.5.4 十进制数(压缩 BCD 码)/二进制数转换

一、DECBIN 十进制(压缩 BCD 码)到二进制

这个 24 字节的子程序给出了非常少的编码,可将任意长度的以压缩 BCD 码表示的十进制数转换成二进制数。

算法:通过循环算法实现转换。转换结束后,二进制结果的位置正好在 BCD 码十进制数的后面。在该算法的每个循环中,十进制操作数和二进制结果右移一位。移位的结果是十进制操作数的低位移进二进制数的高位。接下来测试移位后的十进制数中各个半字节的高位(第 7 位和第 3 位)。如果该位为 1,则相应的半字节减去 3。这样就相当于右移 BCD 码一位,然后将结果修正为 BCD 码。整个过程反复的次数等于十进制数包含的二进制位数。

```

16 位   [1,0]:      二进制数
        [3,2]:      十进制(压缩 BCD 码)
24 位   [2,1,0]:    二进制数
        [5,4,3]:    十进制(压缩 BCD 码)
32 位   [3,2,1,0]:  二进制数
        [7,6,5,4]:  十进制(压缩 BCD 码)

```

24 字节

1030 指令周期(16 位)

```

DECBIN: LD CNTR, #16          ;装入计数值,其值为 BCD 字段中二进制位数
                                   ;16:2 个字节
                                   ;124/32:34/32 位
DB1:    LD B, #3              ;5/7:24/32 位
        RC
DB2:    LD A, [B]              ;右移 BCD 码和二进制字段的循环
        RRC A,
        X A, [B-]
        IFBNE #0F
        JP DB2
        LD B, #3              ;5/7:24/32 位
        SC
DB3:    LD A, [B]              ;测试 BCD 码半字节的高位,如果该半字节的高位

```

```

IFBIT 7,[B]          ;为 1,则半字节减去 3
SUBC A,#030
IFBIT 3,[B]
SUBC A,#3
X A,[B-]
IFBNE #1             ;24/32 位时送 2/3
JP DB3               ;循环测试各 BCD 字节
DRSZ CNTR             ;循环计数
JP DB1
RET

```

二、BINDEC 二进制到十进制(压缩 BCD 码)

这个程序以最少代码 25 个字节将任意长度的一个二进制数转换成压缩 BCD 码表示的十进制数。

算法:通过循环算法实现转换。压缩 BCD 码的十进制结果在二进制数的上方。必须保证有足够的字节保存十进制结果。在该算法的每个循环周期,二进制数左移一位,BCD 码的十进制结果也左移一位,移位的结果是二进制数的高位移进十进制数的最低位。移位产生的 BCD 码用 DCOR 指令来实现十进制调整。附加的 ADD A,#066 指令必须要和 DCOR 指令一起使用。这个过程重复执行,总的执行次数是二进制数包含的二进制位数。

```

16 位   [1,0]       十进制数
        [4,3,2]     十进制(压缩 BCD 码)
24 位   [2,1,0]     二十制数
        [6,5,4,3]   十进制(压缩 BCD 码)
32 位   [3,2,1,0]   二进制数
        [8,7,6,5,4] 十进制(压缩 BCD 码)
25 字节
856 指令周期(16 位)

```

```

BINDEC: LD CNTR,#16      ;装入循环计数值,其值为二进制数的位数
                        ;16:16 位
                        ;24/32:24/32 位
        RC
        LD B,#2          ;24/32 位时为 3/4
BD1:    LD [B+],#0        ;清 BCD 字段
        IFBNE #5         ;24/32 位时为 7/9
        JP BD1
BD2     LD B,#0

```

BD3:	LD A,[B]	;循环程序,用于左移二进制字段
	ADC A,[B]	
	X A,[B+]	
	IFBNE #2	;24/32 位时为 3/4
	JP BD3	
BD4:	LD A,[B]	;循环程序,用于左移
	ADD A,#066	;及 BCD 码字段左移结果的十进制调整
	ADC A,[B]	
	DCOR A	
	X A,[B+]	
	IFBNE #5	;24/32 位时为 7/9
	JP BD4	
	DRSZ CNTR	;循环控制
	JP BD2	
	RET	

参 考 文 献

1. National Semiconductor: "COP8SA $\times$ designer's guide", August 1997
2. National Semiconductor: "COP8 feature family user's manual", March, 1998
3. National Semiconductor: "Microcontrollers Databook", 1997/1998
4. National Semiconductor: "COP8 Development tools", 1995
5. 刘仁普主编: COP8 系列 8 位单片计算机技术应用手册, 电子工业出版社, 1995. 10



单片机系列丛书

National
Semiconductor

责任编辑：陆盛强

封面设计：赵丽丽

ISBN 7-309-02366-8



9 787309 023664 >

ISBN 7-309-02366-8/T·233

定价：20.00元

