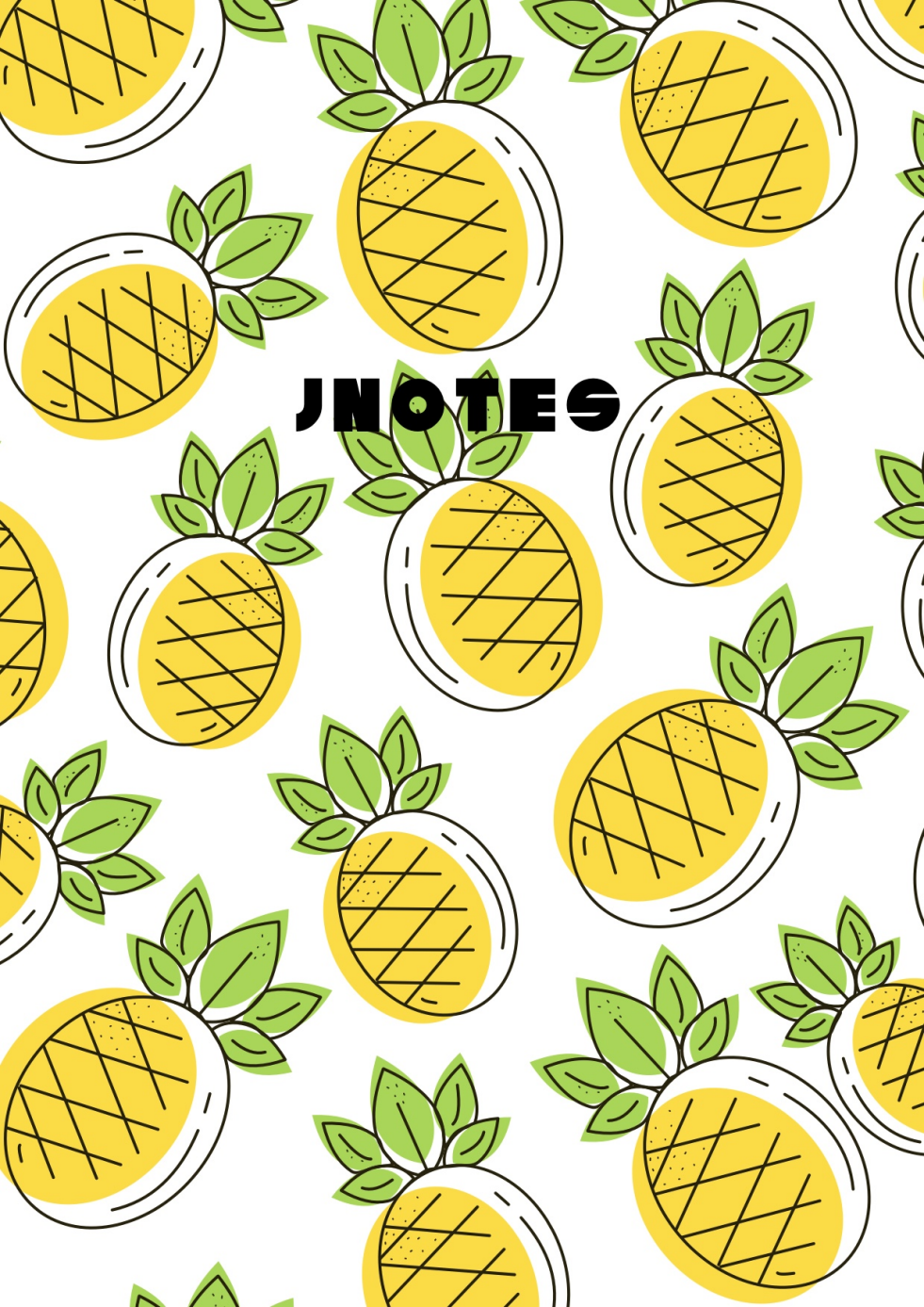

JNOTES

日期: /

微处理器：具有运算器和控制器的中央处理器

- 中央处理器（CPU）
- 微控制器（MCU）
- 数字信号处理器（DSP）

单片机 → 将微处理器、存储器、I/O接口电路集成在一块芯片上，称为单片微型计算器

微处理器的工作原理：取指令 → 指令译码 → 取操作数 → 回送结果

单片机的分类

1. HMOS 和 CHMOS（前缀有C） 功耗更低
2. 冯·诺依曼结构：程序与数据共用同一个存储器
   哈佛结构 - - - 各用一个独立的存储器（大多数）

8051 是低功耗微处理器（HMOS）

8051 CPU从功能上分为两个部分

总线接口部件 BIU：Bus Interface unit 负责与存储器、I/O传送数据

执行部件：EU：Execution unit 完成指令的译码与执行工作

日期: /

8086 CPU 内部结构

内/外总线

一个地址对应一个字节

bit: 位 byte 字节 => 1B

1B = 8bit

20根AB 地址总线 address bus $2^{20}B = 1MB$

16根DB 数据总线 data bus $2^{16}B = 2^6KB = 64KB$

64KB I/O空间

1个字节 0~5个字节

13条指令 (操作码. 操作数) 1~6个字节不等

16位 AX/BX/CX/DX 通用寄存器 存数据 or 存地址.

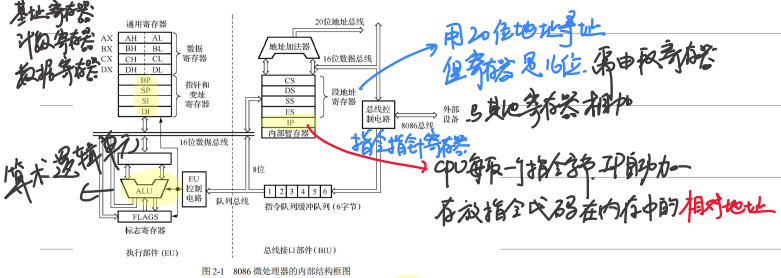
AH/AL、BH/BL、CH/CL、DH/DL 可分成8个8位

累加器

基址寄存器

计数寄存器

数据寄存器



算术逻辑单元

用20位地址寻址

但寄存器是16位. 需由段寄存器与其他寄存器相加

指令指针寄存器

CPU每取一个指令字节, IP自动加一

存放指令代码在内存中的相对地址

图 2-1 8086 微处理器的内部结构框图

SP: 堆栈指针寄存器, 指示堆栈栈顶在内存中的相对位置 (Stack Pointer)

BP: 基址指针寄存器: 存放堆栈数据在内存中的偏移地址 (Base Pointer)

SI: 源操作数变址寄存器 (Source Index)

DI: 目的操作数变址寄存器 (Destination Index)

日期: /

一. 8086 CPU 内部寄存器

二. 标志寄存器 (Flag Resister)

CF	Carry Flag	仅无符号数有用	进位标志
PF	Parity Flag	→ 偶数为1 (低8位中1的个数为偶数)	奇偶校志
AF	Auxiliary Flag	仅BCD有用 (十进制)	半进位标志
ZF	Zero Flag	运算结果为0, ZF=1	零标志
SF	Sign Flag	仅有符号数有用 最高位为1(负数)	符号标志位
OF	overflow Flag	仅有符号数有用	溢出标志: 最高位进位 ⊕ 次高位进位 (异或)
TF	Trap Flag	这3个是控制位, 不反映CPU的状态, 而是由程序设置的	单步标志
IF	Interrupt Flag	为1响应外部可屏蔽中断 (=1)	中断标志
DF	Direction Flag		

三. 存储器分段

BX, SI, DI ...

分成了逻辑段, 每个逻辑段有逻辑地址

日期: /

内存空间的一地址 (20位)

四 物理地址的形式 → 从00000H到FFFFFH 一个逻辑段64KB不超过64KB

逻辑地址: 段基地址:偏移地址 不唯一: 1200:0345H 1234:0045H

物理地址: 段基地址×10H+偏移地址 唯一: 12345H

传入内存中找到data

段寄存器: CS DS SS ES [Code 代码段 Data 数据段 Stack 堆栈段 Extra 附加段

偏移地址: IP. BX. BP. SI. DI

8086 CPU 引脚功能

MN/MX = 1 最小模式

MN/MX = 0 最大模式

AD15 ~ AD0 数据引线和地址引线的低16位是分时复用的

地址/数据总线

低地址信号 → 锁存器 → 低数据

ALE 地址锁存允许信号 (Address Enable) 地址使能

8086带载能力: 加缓冲器: 漏门极限

DT/R Data Transmit / Receive = 1 CPU对外发数据 写操作

= 0 收数据

日期: /

其中. $S_6=0 \Rightarrow$ 8086与总线相连
S5表明IF状态
$S_4 \cdot S_3$ 组合起来指示当前选择的段寄存器

输出CPU当前的状态信息

$\overline{DEN}$ Data Enable 控制 cpu 收发数据

$A_{16}/S_3 \sim A_{19}/S_6$ T_1: $A_{19} \sim A_{16}$ → 锁存 $T_2 \sim T_4$: $S_6 \sim S_3$
address/state 发送地址信号

数据

$\overline{BHE}/S_7$ Bus High Enable =0 在读写期间 $D_{15} \sim D_8$ 有效 控制高8位
(还有与A0组合控制字/字节所选)

$M/\overline{IO}$ =1 cpu在访问存储 =0, cpu在访问IO
memory / IO

$\overline{RD}$ =0 允许 CPU 从存储 or I/O 读出数据
read

$\overline{WR}$=0 允许 ··· 对 ——or—— 写入数据
write

READY =0 存储 or I/O 没准备好. =1, 进入T_4. 完成数据传输

INTR Interrupt Request =1 外设向cpu提出中断请求. 若IF=1 响应

$\overline{INTA}$ Interrupt Acknowledge cpu响应可屏蔽中断请求的回应

NMI Non-Maskable Interrupt 不可屏蔽

HOLD Hold Request

HLDA Hold Acknowledge

总结: 带 Acknowledge 的信号. 由CPU向外发出

读: cpu 收数据

写: cpu 发数据.

日期: /

2.3 8086最小系统的配置

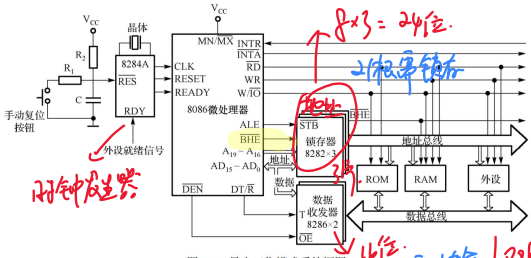


图 2-2 最小工作模式系统框图

掌握cpu发data/收data流程

掌握 8284A, 8282, 8286作用

8086的工作模式: MN/MX (33引脚)

最小工作模式: 微机系统中只有一个CPU, 总线所有信号直接由8086产生 系统中总线控制逻辑电路被省到最少 (单处理器模式)

最大 - : 系统有2个及2个以上的微处理器存在, 主处理器为8086, 其他称为协处理器

(多处理器模式)

输入/输出: 8089, 数值: 8087

总线信号由一个总线控制器8288根据8086CPU输出的总线周期状态产生

(S2. S1. S0)

2.4 总线操作时序

CPU在总线上每完成一次数据的传送称为一个总线周期, 由四个时钟周期

T1~T4组成 (T1出现的是地址信号, T2. T3. T4出现的是数据信号)

日期: /

2.5 存储器组织及数据存取过程

- 字节数据（8位）
- 字数据（16位）高八位+低八位　字地址=低字节的地址

字地址 = 偶数：规则字
　　奇数：不规则字

低	32H
高	20H

2032H

- 双字数据　16×2=32位

512KB×2 ⟶ 1MB

2^{20} Byte = 1MB

高存储体　　低存储体

A0 用作片选　$\overline{SEL}=0$ ⇒ 选低存储体

$\overline{BHE}=0$ ⇒ 选高存储体

存取过程：

规则字只需存取一次，不规则字需存取二次。

日期: /

微处理器的工作原理：取指令 → 指令译码 → 取操作数 → 执行运算 → 回送结果

3.1 指令格式

机器码 [操作码 1个字节
操作数 数值 or 数值地址

[标号:] [前缀指令] 指令助记符 操作数 [;注释]
标号地址 指令助记符 操作数 目的 源

3.2 8086寻址方式 操作数可存放在：①指令代码 ②寄存器 ③内存 ④I/O口

7个

源操作数：7个 目的操作数：6个（除立即寻址外的6个）

立即寻址 操作数在指令队列 CPU 快
寄存器寻址 在寄存器中
直接寻址
寄存器间接寻址
寄存器相对寻址
基址变址寻址
相对基址变址寻址
操作数在存储器中 慢

日期:

高字节存放在高地址,
低字节存放在低地址

注:① 操作数只能用于SRC
② SRC与DST的字长要一致

立即寻址 MOV AX, 1000H

立即数 ← A~F开头-加0 MOV AX, 0F000H

寄存器寻址 MOV AX, BX 注:SRC与DST的字长要一样

↳ A~DX, SI, DI, SP, DP

下面几种情况需先求出有效地址

直接寻址: MOV AX, [1000H] 加[]代表EA(有效地址)

默认DS 16×DS+EA

变更 CS, SS, ES 段超越前缀 MOV AX, ES:[1000H]

寄存器间接寻址 MOV AX, [SI]

EA = (BX, BP, SI, DI)

寄存器中的值是操作数的地址

注 { BX, SI, DI: DS的段地址 (不同)
BP: SS的段地址

寄存器相对寻址 MOV AX, [BP+34H]

EA = (BX, BP, SI, DI) + disp 偏移量

基址变址寻址 MOV AX, [BX+SI]

EA = (BX ↔DS, BP ↔SS) + (SI, DI)

相对基址变址寻址 MOV AX, [BX+SI+2]

EA = (BX, BP) + (SI, DI) + disp 偏移量

基址 + 变址

日期: /

变址寻址可以有多种格式:

MOV AX, 0200H + [BX]

MOV AX, [BX + 0200H]

MOV AX, 0200H[BX]

总结: 在计算物理地址时, 为 10H × 段基址 + 偏移地址 (PPEA)

在自动寻址中, 段基址默认在DS中.

在等待类间接寻址中, 对BX、SI、DI 默认在DS中.

基址变址 对即 SS中.

⇒只要有[BP]段基址中用SS

日期: /

单片机的引脚功能:

1. 主电源引脚 V_{CC} 和 V_{SS}

V_{CC}: 40脚 电源输入端. (+5V)

V_{SS}: 20脚 为用接地端 (GND)

2 时钟振荡电路引脚 XTAL1 和 XTAL2

XTAL1 和 XTAL2 分别用于晶体振荡电路的 反相器输入端 和 输出端

Extern Crystal

(a) 内部时钟方式 (b) 外部时钟方式

图 3-2 单片机时钟工作方式

3. 控制引脚.

① 复位信号输入端 RST (Reset)

保持两个机器周期以上的高电平. 完成复位操作

② 地址锁存允许信号 ALE (Address Latch Enable)

访问外部存储器时, 用来锁存P0口送出的低8位地址信号. } 现在几乎不用

③ 读选通信号 $\overline{PSEN}$ (Program Store Enable)

外部程序存储器的读选通信号

④ 外部程序存储器控制信号 $\overline{EA}$ (Enable Address)

$\overline{EA}$=0, 访问外部程序存储器 $\overline{EA}$=1, 访问片内与片外存储器 (先内后外) 该引脚常接高电平.

日期: /

④ P0. P1. P2. P3 端口

P0口(P0.0~P0.7): ①8位漏极开路型的双向I/O口, 作成用户数据总线

②提供低8位地址和8位双向数据总线(先地址后数据)

P1口(P1.0~P1.7): 内部带上拉电阻的8位准双向I/O口.

P2口(P2.0~P2.7): ①内部带上拉电阻的8位准双向I/O口. ②访问外存时, 输出高8位地址

P3口(P3.0~P3.7): ①内部带上拉电阻的8位准双向I/O口. ②第二功能: 外部中断, 计数脉冲输入端, 读写外部数据存储器, I/O控制端.

并行I/O口的特点:

(1) 4个都是双向口, P0是漏极开路的, P1. P2. P3均具有内部上拉电阻(准双向口)

内部无上拉电阻. 作为I/O口时, 需要外接上拉电阻 (10k)

(2) 8根端口线都可以用作输入、输出, 还可以进行位操作

(3) 当并行I/O口作为输入时, 须向锁存器必须先写入"1".

注:

(1) P2口作为地址线使用时, 剩下不可以用作I/O口使用.

(2) P3口 - - 第二功能时, 剩下的可以单独作为I/O口使用

日期: /

中央处理器（CPU）

由运算器、控制器、布尔(位)处理器组成

1. 运算器：算术逻辑单元（ALU）Arithmetic Logical Unit

① 累加器（Acc）Accumulator 一个8位寄存器（注：在指令系统中助记符为ACC，其余为A）

② 程序状态字寄存器（PSW）（Program status Word）

是一个8位寄存器，用来存放运算结果的一些特征

D7	D6	D5	D4	D3	D2	D1	D0
CY	AC	F0	RS1	RS0	OV	保留	P

- CY：进位标志位 Carry
- AC：半进位标志位 Auxiliary Carry
- F0：用户自定义标志位 Flag
- RS1、RS0：工作寄存器组选择 Register Select
- OV：溢出标志位 Overflow
- P：奇偶校验标志位 Parity

日期: /

CY: 最高位有进位(借位)时, 由硬件置1, 否则清零 { ①判断有无进位 ②作为位累加器

AC: 辅助进位, 低四位向高四位产生进位(或借位)时, 由硬件置1 { ①判断有无 ②BCD码调整运算.

F0、F1: 用户指定的状态标志

RS1. RS0: 工作寄存器组指针: 指定CPU当前工作的寄存器组

OV: 溢出标志: 有符号数加减, 无符号数乘除有异常结果为1 一判断结果是否正确.

P: 有奇数个"1", 则置为1, 否则置为0。 用于串行通讯中的数据校验, 判断传输是否有错误

③B寄存器: 乘除运算中, 作为ALU的输入之一, 与Acc配合完成运算

2. 控制器:

①程序计数器(PC) Program Counter

用来存放下一条要执行的指令的地址

②堆栈指针(SP) Stack Pointer 栈顶地址

③数据指针(DPTR) Data Pointer

可以外接64KB的数据存储器和I/O接口, 故在其内部设置16位的DPTR

④特殊(位)处理器: { CY: 进位标志位
位寻址寄存器
位寻址单行I/O口
位操作指令系统

日期: /

3. 存储器:

一. 8051在物理上有4个存储空间:

1片内、片外程序存储器 (ROM)

2.片内、片外数据存储器 (RAM)

二. 从用户使用角度上来看. MSC-51有3个存储空间

1. 片内外统一编址的64KB程序存储器 (16位地址)

2. 256B片内数据存储器 (8位地址)

3. 64KB片外数据存储器地址 (用16位地址)

1. 程序存储器: 用于存放编好的程序或表格常数

在程序存储器的开始部分, 一般具有特殊功能的地址段. 用作程序运行和各种中断

编写程序时必须跳过这个区域。

(1) 0000H~0002H: 单片机系统复位后, PC=0000H, 即程序从0000H单元开始执行程序, 若程序不从0000H单元开始执行, 则应在这3个单元中存放一条无条件转移指令, 让系统跳过这个区域, 直接去执行用户指定的程序, 这主要针对汇编语言编程, 用C51编程就不需要考虑了。

(2) 0003H: 外部中断0入口地址。

(3) 000BH: 定时器T0溢出中断入口地址。

(4) 0013H: 外部中断1入口地址。

(5) 001BH: 定时器T1溢出中断入口地址。

(6) 0023H: 串行口中断入口地址。

(7) 002BH: 定时器T2溢出中断入口地址 (仅52系列单片机有)。

8n+3

日期: /

2. 数据存储器

用于在运算中间运算结果、数据暂存和缓冲、标志位等。

分为片内数据存储器、片外数据存储器、特殊功能存储器(SFR)

① 片内

3个区域 { 工作寄存器区 / 位寻址区 / 数据缓冲区

表3-4 片内通用数据存储器的结构

RAM地址	D7	D6	D5	D4	D3	D2	D1	D0	区域
7FH~30H									数据缓冲区
2FH	7F	7E	7D	7C	7B	7A	79	78	可位寻址区
2EH	77	76	75	74	73	72	71	70	
2DH	6F	6E	6D	6C	6B	6A	69	68	
2CH	67	66	65	64	63	62	61	60	
2BH	5F	5E	5D	5C	5B	5A	59	58	
2AH	57	56	55	54	53	52	51	50	
29H	4F	4E	4D	4C	4B	4A	49	48	
28H	47	46	45	44	43	42	41	40	
27H	3F	3E	3D	3C	3B	3A	39	38	
26H	37	36	35	34	33	32	31	30	
25H	2F	2E	2D	2C	2B	2A	29	28	
24H	27	26	25	24	23	22	21	20	
23H	1F	1E	1D	1C	1B	1A	19	18	
22H	17	16	15	14	13	12	11	10	
21H	0F	0E	0D	0C	0B	0A	09	08	
20H	07	06	05	04	03	02	01	00	
1FH~18H	寄存器3组								工作寄存器区
17H~10H	寄存器2组								
0FH~08H	寄存器1组								
07H~00H	寄存器0组								

通用存储区

(1) 工作寄存器:

四个通用寄存器,供用户编程时使用,用于临时存储8位数据信息

表 3-6 工作寄存器及 RAM 地址对照表

RS1 RS0	区 号	寄存器名	字节地址	RS1 RS0	区 号	寄存器名	字节地址
0 0	0区	R0~R7	00H~07H	0 1	1区	R0~R7	08H~0FH
1 0	2区	R0~R7	10H~17H	1 1	3区	R0~R7	18H~1FH

通过改变PSW中的RS1、RS0来实现选择CPU当前的工作寄存器组

(2) 位寻址区

内部RAM中地址为20H~2FH的16单元。

CPU不仅有字节寻址功能,还有位寻址功能

表 3-7 RAM 位寻址区地址映像

字节地址	位地址							
	D7	D6	D5	D4	D3	D2	D1	D0
2FH	7F	7E	7D	7C	7B	7A	79	78
2EH	77	76	75	74	73	72	71	70
2DH	6F	6E	6D	6C	6B	6A	69	68
2CH	67	66	65	64	63	62	61	60
2BH	5F	5E	5D	5C	5B	5A	59	58
2AH	57	56	55	54	53	52	51	50
29H	4F	4E	4D	4C	4B	4A	49	48
28H	47	46	45	44	43	42	41	40
27H	3F	3E	3D	3C	3B	3A	39	38
26H	37	36	35	34	33	32	31	30
25H	2F	2E	2D	2C	2B	2A	29	28
24H	27	26	25	24	23	22	21	20
23H	1F	1E	1D	1C	1B	1A	19	18
22H	17	16	15	14	13	12	11	10
21H	0F	0E	0D	0C	0B	0A	09	08
20H	07	06	05	04	03	02	01	00

日期: /

(3) 数据缓冲区

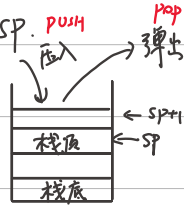
30H ~ 7FH 是数据缓冲区，即用户RAM区，共80个单元

52系列片内RAM有256个单元 工作寄存器区和位寻址区的单元数与地址和引脚一致.

而数据缓冲区有256个单元. 地址范围为 30H ~ FFH

(4) 堆栈与堆栈指针 SP. PUSH 压入 POP 弹出

"后进先出"



SP总是指向最后一个压入栈的数据所在的数据单元 — 栈顶

2. 特殊功能寄存器 (SFR) Special Function Register

带*号的SFR. 地址能够被8整除. 可位寻址

寄存器符号	寄存器名称	地 址	寄存器符号	寄存器名称	地址
* B	B 寄存器	F0H	TH1	定时/计数器 1（高字节）	8DH
* ACC	累加器	E0H	TH0	定时/计数器 0（高字节）	8CH
* PSW	程序状态字	D0H	TL1	定时/计数器 1（低字节）	8BH
* IP	中断优先级控制寄存器	B8H	TL0	定时/计数器 0（低字节）	8AH
* P3	P3 口	B0H	TMOD	定时/计数器工作方式寄存器	89H
* IE	中断允许控制寄存器	A8H	* TCON	定时/计数器控制寄存器	88H
* P2	P2 口	A0H	PCON	电源控制寄存器	87H
SBUF	串行口数据缓冲寄存器	99H	DPH	数据地址指针（高字节）	83H
* SCON	串行口控制寄存器	98H	DPL	数据地址指针（低字节）	82H
* P1	P1 口	90H	SP	堆栈指针	81H
			* P0	P0 口	80H

日期: /

时钟电路及其时序

1. 时钟周期：即振荡周期，时钟脉冲发生器所产生的脉冲信号的频率是单片机中最基本、最小的时间单位。

振荡周期用 P（period）表示

$$P = \frac{1}{f_{osc}}$$

2. 状态周期：将时钟周期二分频，用 S（Status）表示，前一个时钟周期定义为 P1

$$S = 2P = \frac{2}{f_{osc}}$$

后一个 …… 定义为 P2

3. 机器周期：完成一个基本操作所需时间，一个机器周期有 6 个状态

$$T_{cy} = 6S = \frac{12}{f_{osc}}$$

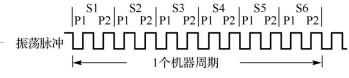


图 3-3 时钟周期、状态周期和机器周期之间的关系

4. 指令周期：执行一条指令所需时间

- 单周期指令
- 双周期指令
- 4 周期指令

1Tcy 或 2Tcy 或 4Tcy

日期: /

汇编语言程序代码.

```
AGA: SETB  P1.1   ;先对P1.1口写入“1”,
                  ;以便能正确读入P1.1口数据
     MOV  C, P1.1 ;读P1.1口状态（0或1），
     CPL  C       ;将读进来的内容取反
     MOV  P1.0, C ;写P1.0口
     SJMP AGA     ;循环执行，方便反复调整
                  ;观察执行结果开关状态
```

Again 标号　　Carry (进位/借)

Set Bit 位置1　　Complement 取反

Move 传送数据　　Short Jump 短跳转

1. 数据传送 Mov. MovX. Movc
2. 堆栈指令: pop. push
3. 数据交换指令
4. 算术运算
5. 逻辑运算
6. 循环移位
7. 控制转移指令

日期: /

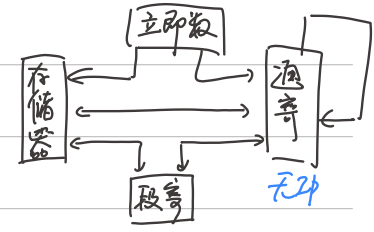
8086指令系统:

DST: 目的操作数 destination

SRC: 源操作数 source

MOV DST, SRC

方向:



PUSH SRC 不可立即数

POP DST

日期: /

8. 设单片机的（70H）=60H，（60H）=20H，P1 口为输入口，当输入状态为 B7H，执行下面程序。

(R0) = 70H
(A) = ((R0)) = (70H) = 60H
(R1) = (A) = 60H
(B) = ((R1)) = (60H) = 20H


```
MOV  R0, #70H
MOV  A, @R0
MOV  R1, A
MOV  B, @R1
MOV  P1,#0FFH
MOV  @R0,P1
```

(P1) = 0FFH
((R0)) = (R1) = 0FFH
(70H) = 0FFH B7=1

试分析单片机的（70H）、（B）、（R1）、（R0）的内容是什么。

9. 有 4 个变量 U、V、W、X 分别从单片机的 P1.0～P1.3 输入，阅读如下程序，写出逻辑表达式并画出逻辑电路图。

(P1) = 0FFH

```
MOV  P1,  #0FFH
MOV  C,  P1.0
ANL  C,  P1.1
CPL  C
MOV  ACC.0,C
MOV  C,  P1.2
ORL  C,  /P1.3
ORL  C,  ACC.0
MOV  F,  C
```

(C) = (P1.0)
(C) = C) AND (P1.1)

10. 若（R1）=30H，（A）=40H，（30H）=60H，（40H）=08H。试分析单片机执行下列程序段后上述各单元内容的变化。

```
MOV  A, @R1
MOV  @R1, 40H
MOV  40H, A
MOV  R1, #7FH
```

(A) = ((R1)) = (30H) = 60H
((R1)) = (40H)
(30H) = (40H) = 08H
(40H) = (A) = 60H
(R1) = 7FH

日期:　　/

单片机的最小系统：

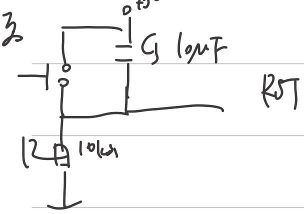
单片机、晶振电路、复位电路、电源。

其他输入/输出信号

单片机部分、复位电路、晶振电路、电源电路 —— 构成最小系统

将单片机恢复到初始状态

把系统的时钟信号决定单片机的运行速度



日期: /

中断系统

1. 5个中断源 a. $\overline{INT0}$ 外部中断0

b. $\overline{INT1}$ 外部中断1

c. T0 定时器/计数器0溢出中断

d. T1 — 〃 — 1 —

e. 串行口中断请求

2. 中断请求标志

TCON: Timer Control Register (88H)

7	6	5	4	3	2	1	0
TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0

IT0: 外部中断0的中断触发方式控制位

IT0=0 电平触发方式

IT0=1 边沿触发方式

IE0: 外部中断0的请求标志位

P3.2引脚信号有效时，IE0=1，执行完后自动清零。

IT1: 外部中断1的中断触发方式控制位

IT0=0 电平触发方式

IT0=1 边沿触发方式

日期: /

IE1：外部中断1的请求标志位

P3.3引脚信号有效时，IE1 =1，执行完后自动清零。

TR0：定时器0的运行控制位。

TR0=1 启动 TR0=0 停止

TR1：定时器1的运行控制位。

TR1=1 启动 TR1=0 停止

TF0：定时器0溢出中断标志位；计满溢出后硬件置位。

TF0=1，同时向CPU发出中断请求，响应后自动清0（也可软件清0）

TF1同理

SCON

1 0

TI RI

Serial control register 串行口

RI：串行口接收中断标志位

TI：串行口发送中断标志

IE

7 6 5 4 3 2 1 0

EA ES ET1 EX1 ET0 EX0

interrupt enable

1 0 0 0 0 0 0 1

EX0：外部中断0中断允许位

EX0=1 允许

EX0=0 禁止

EX1：外部中断1 —

日期: /

ET0: 定时器/计数器T0中断允许位 ET1:

ET0=1 允许.

ET0=0 禁止

ES: 串口中断允许位.

ES=1 允许. ES=0 禁止

EA: 总中断允许控制位.

EA=1. 开放所有中断. EA=0. 禁止所有中断

3. 中断优先级 (发生后低5位清0)

IP(B8H) Interrupt priority 优先寄存

7	6	5	4	3	2	1	0
		PS	PT1	PX1	PT0	PX0	

为1: 设置为高优先级中断

为0: 设置为低优先级中断

同级优先级: 外部中断0 > 定时器T0中断 > 外部中断1 > 定时器T1中断 > 串口中断

INT0 > T0 > INT1 > T1 > UART

日期: /

定时器/计数器　　T0: P3.4　T1: P3.5

1. CSJ中有2个16位可编程定时器/计数器.

TH0 TL0　TH1 TL1

定时器：对机器周期进行计数. 一个机器周期计数器加1

计数器：对外部脉冲进行计数. T0, T1引脚上从高电平到低电平跳变时. 计数器内容加1.　从P3.5引脚输入

(1) C/T=0 定时　C/T=1 计数.　TMOD：定时器/计数器的寄存器

counter/timer　TMOD (SFR)　不可位寻址

D7	D6	D5	D4	D3	D2	D1	D0
GATE	C/T	M1	M0	GATE	C/T	M1	M0
→ 定时器1				→ 定时器0			

(2)

M1	M0	方式	功能
0	0	0	13位
0	1	1	16位
1	0	2	自动重装初值8位
1	1	3	定0：分成2个8位
			定1：停止计数

(3) GATE：门控位

GATE=0　TR0/TR1置1可启动

GATE=1　TR0/TR1须置1

且(P3.2)/(P3.3)为高电平

是否受 INT0 和 INT1 引脚输入电平的控制

日期:　　　/

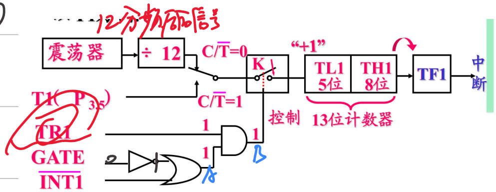
2.定时器工作方式

a.工作方式0：M1M0=00时.

13位定时器（TH0高8位，TL0低5位）

计满溢出后TF0置1.　　满进位

①



①GATE=0，A恒为1

B只取决于TRx：
- TRx=1 B为高电平.
- TRx=0. B为低电平.

②GATE=1，B取决于TRx和$\overline{INT0}$.

TRx 且 $\overline{INTx}$ =1，B为高电平.

b.工作方式1：M1M0=01

时间最长　16位定时器/计数器（结构与方式0相同 区别在于计数位数不同）

c.工作方式2：M1M0=10.

自动重装载初值　16位加法计数器 TH0和TL0

重装初值　8位计数器

缺点：计数范围小，适用于
需重复定时，而定时范围不大的场合.

日期: /

d. 工作方式3 ： $M_1M_0 = 11$ (只适用于 T_0, T_1 无工作方式3)

T_0 被分解成2个独立的8位计数器 TL_0 和 TH_0

3. 应用设计 (编程)

0:13
1:16
2:8
3:8

① 计算初值 ： a. 计数器计算初值

计数器位数 (与工作方式有关)

$$X = 2^n - N$$

X → 初值

N → 计满加所需计数值

b. 定时器计算初值.

$$T = (2^n - X)T_p$$

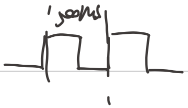
T → 定时时间

n → 定时器位数

T_p → 机器周期

$$X = 2^n - \frac{T}{T_p}$$

X → 初值



1s

1000ms

500ms. 50ms × 10次

12MHz → $T_p = 1\mu s$.

例. 已知晶振 6MHz, 要求定时 0.5ms 试分别求出 T_0 工作于方式0, 方式1, 方式2, 方式3 的初值

1) 方式0. T_0 初值. $2^{13} - \frac{500\mu s}{2\mu s} = 8192 - 250 = 7942 = 1F06H$ (16进制)

$1F06H = 0001111100000110B$

高位、低位

$TL_0 = 00000110B = 06H$ 低5位

$TH_0 = 11111000B = F8H$ 高8位

日期:　　　/

(2)方式1:

例:要求在P1.0引脚输出周期为400μs脉冲方波,已知fosc=12MHz

1Ms.

试分别用T1工作方式0、方式1、方式2编程.

(1)方式0: 算初值: $2^{13}-\frac{400\mu s}{1\mu s}=8192-400=$

TMOD=0

确定工作方式

初始化:

(1) 确定工作方式: 对TMOD进行赋值

(2) 预置定时或计数初值, 将其写入T0、T1中

中断

(3) 根据需要对中断允许寄存器有关位进行赋值, 以开放/禁止定时/计数器中断.

(4) 启动定时/计数器, 将TRi赋值为1

00

日期: /

串行口控制寄存器 SCON (可位寻址) (98H)

SCON	D7	D6	D5	D4	D3	D2	D1	D0
	SM0	SM1	SM2	REN	TB8	RB8	TI	RI

SM0, SM1：串行口4种工作方式选择位

SM0	SM1	方式	功能	波特率
0	0	0	移位寄存器方式(用于并行扩展)	fosc/12
0	1	1	10位UART	可变
1	0	2	11位UART	fosc/32或fosc/64
1	1	3	11位UART	可变

SM2：多机通信控制位（允许多机通信）

SM2=1：当第9位数据(RB8)=1时，RI置1，产生中断请求，并将接收到的前8位数据送入SBUF中；若RB8=0，则将其丢弃。

SM2=0：（无多机通信），直接将前8位数据送入SBUF中，并将RI置1，产生中断请求

在方式0时，SM2必须为0

(3) REN：允许串行接收位（1：允许，0：禁止）

(4) TB8：发送的第9位数据

(5) RB8：接收的第9位数据

(6) TI/RI：表示一帧数据发送/接收完毕，由必须由软件清0

日期: /

电源控制寄存器

2. PCON 最高位 SMOD

波特率：每秒钟传送二进制位数。

SMOD=0：波特率不变。SMOD=1：波特率加倍。

波特率计算

方式0的波特率 = $\frac{f_{osc}}{12}$

方式2的波特率 = $f_{osc} \times \frac{2^{SMOD}}{64}$ （$\frac{1}{12}f_{osc}$ 或 f_{osc}）

方式1或3的波特率 = $\frac{2^{SMOD}}{32} \times$ 定时器T1的溢出率

定时器T1的溢出率 = $\frac{f_{osc}}{(2\times(256-TH1))}$

发送数据 → 通信时钟由高→低 将数据一位一位发送到总线

接收 上升沿 采样

串行方式：单工 / 半双工 / 全双工

异步 / 同步

一帧数据：起始位、数据位、奇偶位、停止位。

"0" "1"

串口通讯标准：

① RS-232C：-3V ~ -15V 逻辑"1"；+3V ~ +15V 逻辑"0"

RXD TXD 地线(GND)

转换电平

② RS-485

日期: /

串行口程序设计:

(1) 选工作方式 → 确定 SCON 中 SM0 和 SM1

(2) 其他位. 如 SM2. REN. TI. RI

(3) 波特率 → 1. 2. 3 (确定 SM0)

1. 3 确定初值 TH1. → 确定了 T1 的工作方式和初值.

(4) 中断方式 → 初始化中断 IE. IP.

工作方式 0: 移位寄存器方式

同步通信. 将串行口变成一个移位寄存器 I/O 口. 半双工

RXD → 数据传送

TXD → 同步时钟脉

工作方式 1: 10 位异步收发通信模式

1 位起始位 — 0

8 位数据位 — 有用信息

1 位停止位 — 1

全双工

TXD 发送 (P3.1)

RXD 接收 (P3.0)

发送: TI=0

接收 RI=0 REN=1 接收有效条件: ① RI=0 ② SM2=0 时, 停止位=1

日期:　　　/

方式2:

1位起始位 —— 值0　　　8位数据位 —— 有用的

1位校验位 —— 对有用位进行奇偶校验　　　1位停止位 —— 值1

接收条件: 1) RI=0　　2) SM2=0时 接收到的停止位为1

方式3: 波特率设置方式同方式1. 其他与方式2类似

CD4094:

当STB=0　　允许串行数据从D输入. 但关闭输出端.

STB=1.　　　关闭输入端, 但允许8位数据并行输出

SCON: 发送A机: 0x40

工作方式1: ① 设 SM0, SM1=01　　　接收B机: 0X50

② 设定波特率 —→ 设定T1的溢出率.

设定: SMOD

$X=256-f_{osc}/(384\times 波特率)$

$X=244.$

设定定时器的工作方式为方式2 → 2为8位计数器

TH1, (TL1) → 计数.

重装载

$$\frac{2^{SMOD}}{32}\cdot\frac{f_{osc}}{12(256-TH1)}=波特率$$

常见波特率及对应TH1的初值 (默认SMOD=0)

1200bps ←→ 232　　2400bps ←→ 244　　4800bps ←→ 250　　9600bps ←→ 253

日期: /

处理串行口中断: void isr_uart( ) interrupt 4

工作方式3.

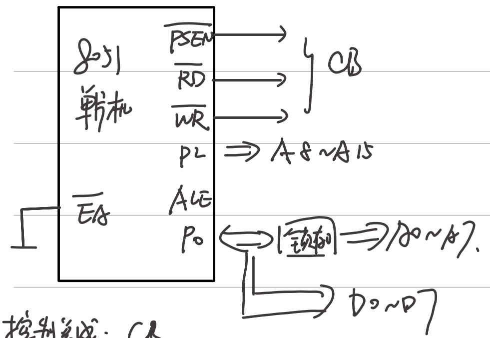
sbit p=PSW^0; PSW中的最低位是奇偶校验标志位(P)

反映ACC中前8位数据的1的个数的奇偶性.

TB8=p; 奇偶校验位.

外部并行总线:

并行总线结构: 地址总线(AB). 数据总线(DB). 控制总线(CB)



地址总线: P0口提供低8位..(分时复用DB)

P2口提供高8位

通过外接一个锁存器锁存. (低8位D)

控制总线: CB

(1) ALE: 做地址锁存 P0口输出的低8位地址用于分离地址和数据

(2) $\overline{PSEN}$: 扩展外部程序存储器的读控制 —ROM

(3) $\overline{EA}$: $\overline{EA}$=0: 只访问外部程序存储器 $\overline{EA}$=1. 从内部程序存储器开始执行

片内: 0000H~FFFH 共4KB 片外: 1000H~FFFFH: 60KB

日期: /

RAM

(4) (5) $\overline{RD}$ 和 $\overline{WR}$：作为对片外数据存储器和 I/O 端口读/写选通信号

对外部数据存储器和 I/O 设备进行寻址

①读选信 ②译码作 →为对片选片和片选端提供信号.

A/D、D/A 转换器

校准 一般: 逐次比较型 / 双积分

①分辨率. 基准电压(V_{REF}) n：转换位数

分辨率：$\frac{V_{REF}}{2^n}$

②量化误差：真实值与转换出来误差

③转换精度

④转换时间 完成一次 A/D 转换所需的时间.

D/A 转换器.

电压输出/电流输出

①分辨率. $\frac{满量程}{2^n}$

②建立时间 ③转换精度

日期: /

SPI通信: serial peripheral interface 串行外设接口

一主多从.

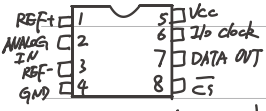
CS: 片选信号: 低电平有效.

SCK: 串行时钟线, 由主设备产生

MOSI: 发送信号线 Master output Slave input
主机输出 从机输入

MISO: 接收信号线 Master input Slave output

TLC549: A/D转换器: 逐次逼近型



$V_{in} = \frac{V_{REF(+)} - V_{REF(-)}}{2^n} N + V_{REF(-)}$

N: 输出的数字信号, 通常 $V_{REF(+)} = 5V$ $V_{REF(-)} = 0V$

8位 输入电压为: $V_{in} = N \cdot \frac{5}{256}$

接口函数.

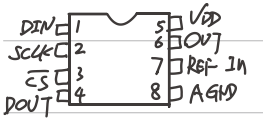
日期: /

```
uchar TLC549_ADC(void)
{
    uchar i, temp;          ← 存i读取和定义数据
    TLC549_CLK=0;
    TLC549_CS=0;            ← 片选
    for(i=0;i<8;i++)
    {
      temp<<=1;             ← 腾出最低位
      temp|=TLC549_DO;      ← 当有data out 加入最低位.
      TLC549_CLK=1;         ]→ 通知 TLC549 输出下一个数据
      TLC549_CLK=0;
    }
    TLC549_CS=1;            → 结束通信
    delay_us(20);
    return temp;
}
```

日期:

10位的数模转换器

TLC5615 (DAC) 一款数模转换器



DIN: 串行数据输入端

DOUT: 用于级联的串行数据输出端

OUT: DAC模拟电压输出端

10位

日期: /

两线: SCL 时钟线
SDA 数据线

I²C通信协议

Inter-Integrated Circuit 芯片间总线

SCL: 时钟线

SDA: 数据线

上拉电阻 → 空闲时确保SCL和SDA为高电平.

数据帧格式:

单向写: 常见 引脚 读写方向 0:写 1:读 (R/W) 终止信号

S	1	0	1	0	0	0	0	0	A	字节1	A	······	P

起始信号　设备地址(7位)　应答信号 0:有应答 1:无应答 由从器件发出　主器件发出

单向读:

S	1	0	1	0	0	0	0	1	A	字节1	A	······	P

起始信号　设备地址(7位)

双向读写

重新产生起始信号

S	1	0	1	0	0	0	0	0	A	字节1	A/Ā	Sr	从机地址	1	A	数据	Ā	P

起始信号　设备地址(7位)　重新产生的从器件地址

先发一个字节 再收一个字节数据. 起始信号和从器件地址都被重新产生一次, 而收读写方向相反

日期:　　/

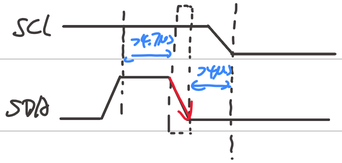
寻址字节的帧格式

寻址字节　　器件地址 → 4位　　引脚地址 → 3位　　方向

DA3 DA2 DA1 DA0　A2 A1 A0　R/$\overline{W}$

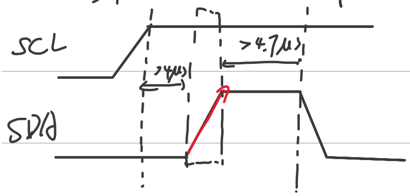
模拟时序:　　SCL 高电平
SDA 下降沿↓

(1)起始信号 S起始: SCL为高电平期间, SDA由下降沿表示起始信号



nop();

(2)终止信号 P终止: SCL为高电平期间, SDA由上升沿表示终止信号



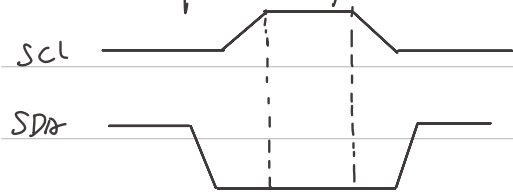
SCL 高电平
SDA 上升沿↑

日期: /

(3) 应答"0"函数，(发送数据"0")

SDA 低电平
SCL 正脉冲

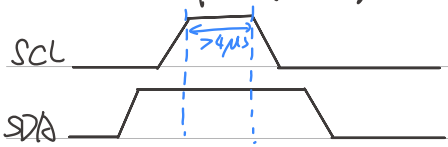
在SDA低电平期间，SCL发出一个正脉冲



(4) 非应答"1"函数 (发送数据"1")

SDA 高电平
SCL 正脉冲.

在SDA高电平期间，SCL发送一个正脉冲.



(5) 发送一字节数据函数

```
void SendByte (uchar dat)
{   uchar i;
    for(i=0; i<8; i++)
    {
        SDA = (bit)(dat & 0x80);   // 将最高位提取出来
        SCL = 1;
        _nop_(); _nop_(); _nop_(); _nop_(); _nop_();   // 发送
        SCL = 0;
        dat <<= 1;                 // 左移1位 => 次高位变成下一个要发送的位.
    }
}
```

日期: /

(6) 接收一字节数据函数:

```
void RcvByte(void)
{
    uchar i, dat=0;
    SDA=1;
    for(i=0;i<8;i++)
    {
        SCL=1;
        dat<<=1            → 腾出最低位
        if(SDA==1) dat|=0x01;
        SCL=0;
    }
    return dat;
}
```

高电平期间读取数据

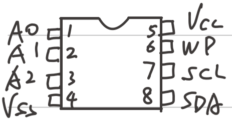
传输过程

若SDA为高电平, → 1

日期: /

I²C 通信:

AT24C02 芯片



A0、A1、A2：设置芯片的器件地址

①级联：通过A0、A1、A2进行寻址

②默认："0"　③最多级联 $2^3=8$个

SCL：时钟线。　SDA：数据输入/输出。

WP：写保护端 { WP=1　只能读,不能写入
　　　　　　　　WP=0　可写入/读

Vcc：+1.8V ~ +6.0V

256B = 32页 × 8B　　芯片寻址．片内子地址寻址 —— 选择具体的存储器
　　　　　　　　　　↑
　　　　　　　选择I²C上的特定AT24C02

7位地址码：1010 A2A1A0 (R/W)

片内子地址：00H~FFH (256个)

对AT24C02的读写操作

①字节写入

S	器件寻址字节	A	片内地址字节	A	Data	A	P

②页写入．写入一整页(8个字节)

同上　Data1 A Data2 A ... Data n A

日期: /

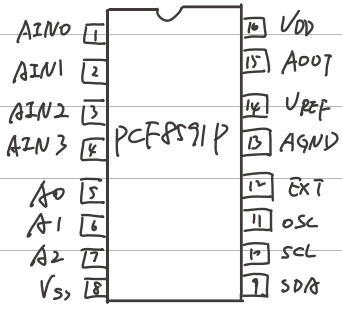
③ 指定地址读取:

S 器件寻址写 A 片内地址 A S 器件地址读 A data A P

④ 指定地址连续读取:

S 器件地址写 A 片内地址 A S 器件地址读 A data1 A data2 A ... dataN A P

PCF8591:



A0~A2: 地址端

OSC: 外部时钟输入端
内 出

EXT: 内部时钟选择线
接地: 使用内部时钟.

AGND: 模拟信号地

AOUT: D/A转换输出端.

器件地址:

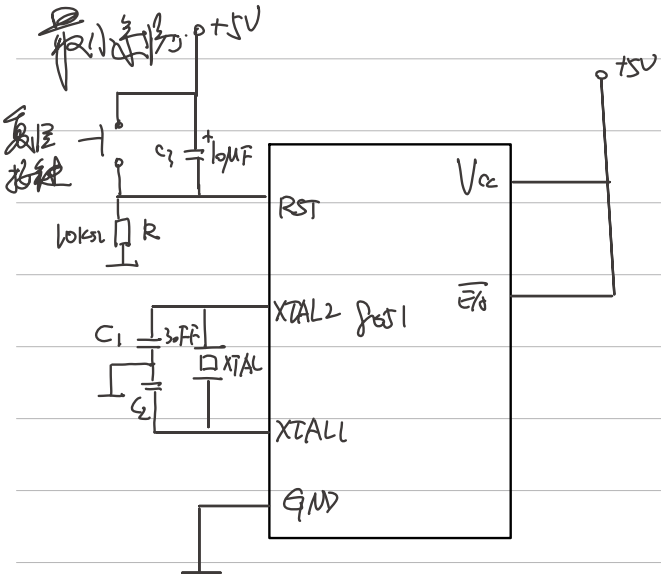
1	0	0	1	A2	A1	A0	R/W

控制寄存器:

0	X	X	X	0	X	X	X
D7	D6	D5	D4	D3	D2	D1	D0

D6: 模拟量输出允许
D5 D4: 模拟转换方式
D2: 自动增益
D1 D0: 模拟输入通道

日期: /


最小系统
+5V
复位按键
C3
10μF
10kΩ
R
RST
Vcc
+5V
XTAL2
XTAL1
GND
C1
30pF
C2
晶振XTAL