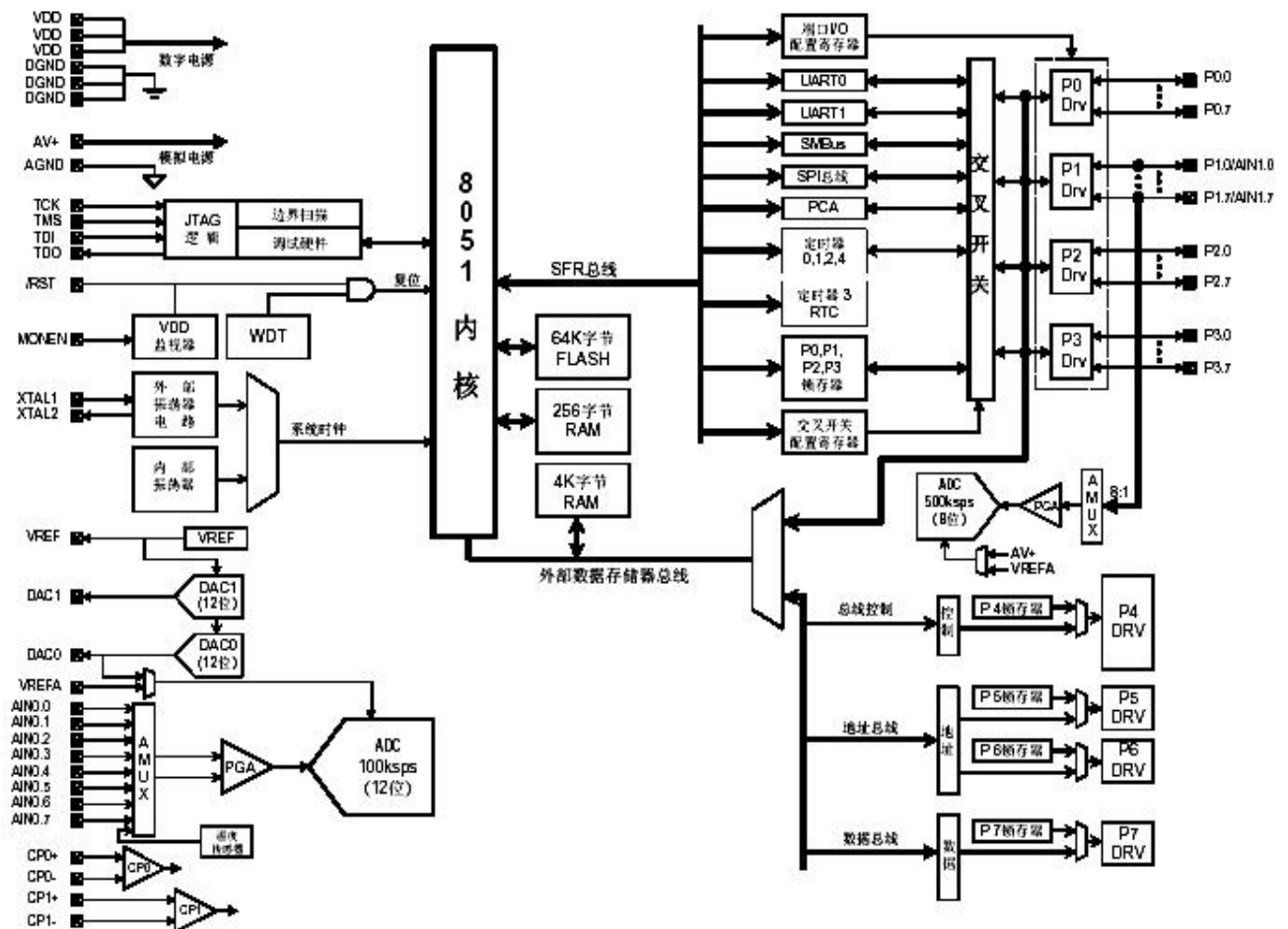


III. 单片机快速进阶

—C8051F02X 系列单片机基本应用指导 系统概述

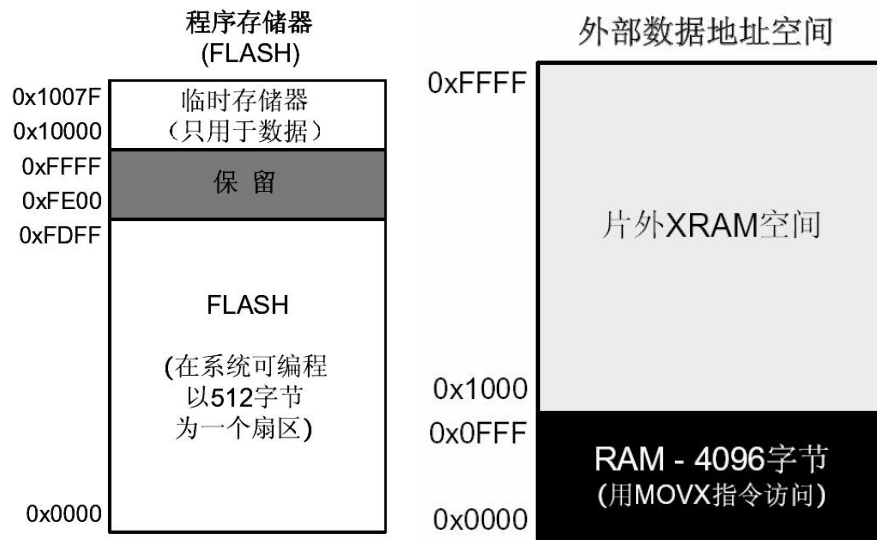
C8051F023 是完全集成的混合信号系统级 MCU 芯片，具有 32 个数字 I/O 引脚，主要特性：

1. 高速、流水线结构的 8051 兼容的 CIP-51 内核（可达 25MIPS）
2. 全速、非侵入式在系统调试接口（片内）
3. 真正 10 位、100ksps 的 8 通道 ADC，带 PGA 和模拟多路开关
4. 两个 12 位 DAC，可编程更新时序
5. 64K 字节可在系统编程的 FLASH 存储器
6. 4352（4096+256）字节的片内 RAM
7. 可寻址 64K 字节地址空间的外部数据存储器接口
8. 硬件实现的 SPI、SMBUS/I2C 和两个 UART 串行接口
9. 5 个通用的 16 位定时器
10. 具有 5 个捕捉/比较模块的可编程计数器/定时器阵列
11. 片内看门狗定时器、VDD 监视器和温度传感器

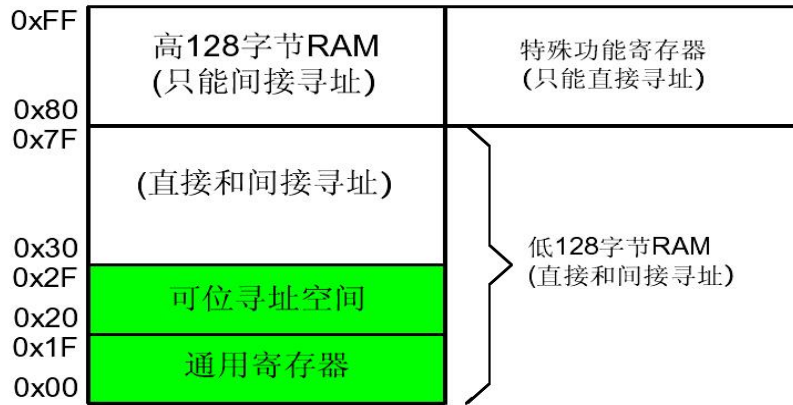


存储器组织

CIP-51 有标准的 8051 程序和数据地址配置。数据存储器包括 256 字节的数据 RAM，其中高 128 位字节为双映射：用直接寻址可以访问 128 字节的 SFR 地址空间；用间接寻址可以访问通用高 128 字节 RAM。低 128 字节 RAM 可以通过直接或间接寻址的方式访问，其中前 32 个字节为 4 个通用寄存器（0x00~0x1f），接下来 16 字节为位寻址区、也可以按字节寻址（0x20~0x2f）。c8051f023 中的 CIP-51 还有一个位于外部存储器寻址空间的 4k 字节的片内 RAM 和一个用于访问外部数据存储器的外部存储器接口 emif。外部存储器地址空间可以只映射到片内、只映射到片外或两者组合（4K 以下地址指向片内，4K 以上地址指向片外）。程序存储器包括 64K+128B 的 Flash，64K 空间以 512B 为一个扇区，可以在系统编程。从 0xFE00~0xFFFF 的 512B 为保留区，由工厂使用。0x10000~0x1007F 为 128B 的扇区，用于存储数据。



数据存储
内部数据地址空间

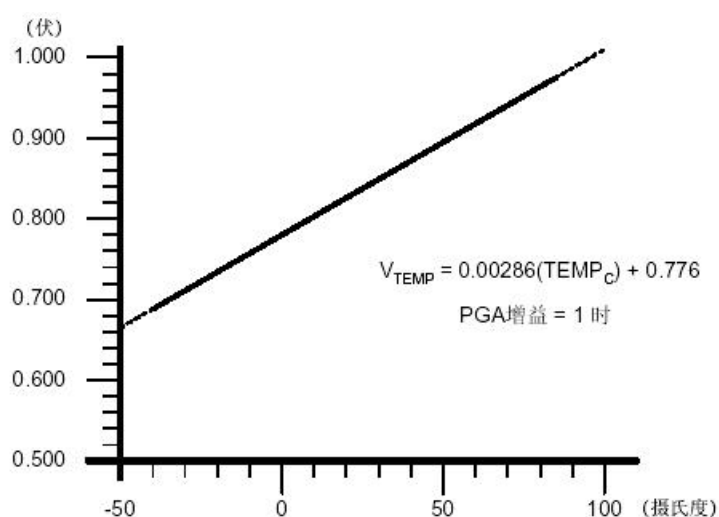


ADC—高速 10 位模/数转换

C8051F023 的 ADC0 子系统包括一个 9 通道的可编程模拟多路选择器 (AMUX0)，一个可编程增益放大器 (PGA0) 和一个 100ksps、10 位分辨率的逐次逼近寄存器型 ADC。

AMUX 中的 8 个通道用于外部测量，而第九通道在外部被接到片内温度传感器。注意，PGA0 的增益对温度传感器也起作用。可以将 AMUX 输入对编程为工作在差分或单端方式。这就允许用户对每个通道选择最佳的测量技术，甚至可以在测量过程中改变方式。在系统复位后 AMUX 的默认方式为单端输入。

温度传感器的传输函数示于下图。当温度传感器被选中 (AMX0SL 中的 AMX0AD3-0) 时，其输入电压 (V_{TEMP}) 是 PGA 的输入；PGA 对该电压的放大倍数由用户编程 PGA 设置值决定。



ADC 的工作方式

ADC0 的最高转换速度为 100ksps，其转换时钟来源于系统时钟分频，分频值保存在寄存器 ADC0CF 的 ADCSC 位。

启动转换

有 4 种转换启动方式，由 ADC0CN 中 ADC0 启动转换方式位 (AD0CM1, AD0CM0) 的状态决定。转换触发源有：

1. ADC0CN 的 AD0BUSY 位写 1；
2. 时器 3 溢出 (即定时的连续转换)；
3. 外部 ADC 转换启动信号的上升沿，CNVSTR；
4. 定时器 2 溢出(即定时的连续转换)。

AMUX0SL:AMUX0C 通道选择寄存器

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	复位值
-	-	-	-	AMX0AD3	AMX0AD2	AMX0AD1	AMX0AD0	00000000
位7	位6	位5	位4	位3	位2	位1	位0	SFR地址: 0xBB

位 7-4: 未使用。读=0000b;写=忽略

位 3-0: AMX0AD3-0: AMUX 地址位

AMX0AD3-0								
0000	0001	0010	0011	0100	0101	0110	0111	1xxx
AIN0	AIN1	AIN2	AIN3	AIN4	AIN5	AIN6	AIN7	温度 传感器

ADC0CF: ADC0 配置寄存器 (C8051F023)

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	复位值
AD0SC4	AD0SC3	AD0SC2	AD0SC1	AD0SC0	AMP0GN2	AMP0GN1	AMP0GN0	11111000
位7	位6	位5	位4	位3	位2	位1	位0	SFR地址: 0xBC

位 7-3: AD0SC4-0: ADC0SAR 转换时钟周期控制位

SAR 转换时钟来源于系统时钟,由下面的方程给出,其中 AD0SC 表示 AD0SC4-0 中保持的数值,CLKSAR0 表示所需要的 ADC0 SAR 时钟(注:ADC0 SAR 时钟应小于或等于 2.5MHz)。

$$AD0SC = \frac{SYSCLK}{CLK_{SAR0}} - 1$$

位 2-0: AMP0GN2-0: ADC0 内部放大器增益

000: 增益=1

001: 增益=2

010: 增益=4

011: 增益=8

10x: 增益=16

11x: 增益=0.5

ADC0CN: 控制寄存器

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	复位值
AD0EN	AD0TM	AD0INT	AD0BUSY	AD0CM1	AD0CM0	AD0WINT	AD0LJST	00000000
位7	位6	位5	位4	位3	位2	位1	位0 (可位寻址)	SFR地址: 0xE8

位 7: AD0EN: ADC0 允许位

0: ADC0 禁止。ADC0 处于低功耗停机状态。

1: ADC0 允许。ADC0 处于活动状态,并准备转换数据。

位 6: AD0TM: ADC0 跟踪方式位

0: 当 ADC0 被允许时,除了转换期间之外一直处于跟踪方式。

1: 由 ADSTM1-0 定义跟踪方式。

位 5: AD0INT: ADC0 转换结束中断标志

该标志必须用软件清 '0'。

0: 从最后一次将该位清 0 后,ADC0 还没有完成一次数据转换。

1: ADC0 完成了一次数据转换。

位 4: AD0BUSY: ADC0 忙标志位

读:

0: ADC0 转换结束或当前没有正在进行的数据转换。AD0INT 在 AD0BUSY 的下降沿被置 '1'。

1: ADC0 正在进行转换。

写:

0: 无作用

1: 若 ADSTM1-0=00b 则启动 ADC0 转换。

位 3-2: AD0CM1-0: ADC0 转换启动方式选择位。

如果 AD0TM=0:

00: 向 AD0BUSY 写 1 启动 ADC0 转换。

01: 定时器 3 溢出启动 ADC0 转换。

10: CNVSTR 上升沿启动 ADC0 转换。

11: 定时器 2 溢出启动 ADC0 转换。

如果 ADC0TM=1:

00: 向 AD0BUSY 写 1 时启动跟踪, 持续 3 个 SAR 时钟, 然后进行转换。

01: 定时器 3 溢出启动跟踪, 持续 3 个 SAR 时钟, 然后进行转换。

10: 只有当 CNVSTR 输入为逻辑低电平时 ADC0 跟踪, 在 CNVSTR 的上升沿开始转换。

11: 定时器 2 溢出启动跟踪, 持续 3 个 SAR 时钟, 然后进行转换。

位 1: AD0WINT: ADC0 窗口比较中断标志。

0: 自该标志被清除后未发生过 ADC0 窗口比较匹配。

1: 发生了 ADC0 窗口比较匹配。

位 0: AD0LJST: ADC0 数据左对齐选择位。

0: ADC0H: ADC0L 寄存器数据右对齐。

1: ADC0H: ADC0L 寄存器数据左对齐。

ADC0H: ADC 数据字 MSB 寄存器

位 7-0: ADC0 数据字高字节。

当 AD0LJST=0: 位 7-2 为位 1 的符号扩展位。位 1-0 是 10 位 ADC0 数据字的高 2 位。

当 AD0LJST=1: 位 7-0 为 10 位 ADC0 数据字的高 8 位。

ADC0L: ADC 数据字 LSB 寄存器

位 7-0: ADC0 数据字低字节。

AD0LJST=0: 位 7-0 是 10 位 ADC 数据字的低 8 位。

AD0LJST=1: 位 7-6 是 10 位 ADC 数据字的低 2 位。位 5-0 读出值总是为 0000b。

ADC0 可编程窗口检测器

ADC0 可编程窗口检测器不停地将 ADC0 输出与用户编程的极限值进行比较，并在检测到越限条件时通知系统控制器。窗口检测器中断标志（ADC0CN 中的 ADC0AD0WINT 位）也可被用于查询方式。参考字的高和低字节被装入到 ADC0 下限（大于）和 ADC0 上限（小于）寄存器（ADC0GTH、ADC0GTL、ADC0LTH 和 ADC0LTL）。窗口检测器标志既可以在测量数据位于用户编程的极限值内时有效，也可以在测量数据位于用户编程的极限值以外时有效。

使用通道 0 进行 A/D 转换的实例

注意：输入电压要大于 0V 小于 1.2V，否则会永久损坏芯片

```
#include <c8051f020.h> // SFR declarations
//-----
// C8051F02X 的 16 位 SFR 定义
//-----
sfr16 DP = 0x82; // 数据指针
sfr16 TMR3RL = 0x92; // 定时器 3 重装值
sfr16 TMR3 = 0x94; // 定时器 3 计数器
sfr16 ADC0 = 0xbe; // ADC0 数据
//-----
// 全局常量
//-----
#define SYSCLK 6000000 // 系统时钟频率 Hz
#define SAMPLERATE0 50000 // ADC0 采样频率 Hz
#define NUM_SAMPLES 30 // ADC0 采样次数
//-----
// 函数原型
//-----
void SYSCLK_Init (void);
void ADC0_Init (void);
void Timer3_Init (int counts);
void ADC0_ISR (void);
//-----
// 全局变量
//-----
xdata unsigned samples[NUM_SAMPLES]; // 存储 ADC0 结果数组
bit ADC0_DONE; // 当 NUM_SAMPLES 次被采集为真
//-----
// 主程序
//-----
void main (void) {
int i; // 循环计数器
WDTCN = 0xde; // 禁止看门狗定时器
WDTCN = 0xad;
```

```

SYSCLK_Init (); // 初始化振荡器
Timer3_Init (SYSCLK/SAMPLERATE0); // 初始化定时器 3 溢出作为 ADC0 采样率
ADC0_Init (); // 初始化 ADC
EA = 1; // 允许全部中断
while (1) {
ADC0_DONE = 0;
EIE2 |= 0x02; // 允许 ADC0 中断
while (ADC0_DONE == 0); // 等待采样结果
}
}
//-----
// 初始化化子程序
//-----
// 时钟初始化
//-----
// 此程序初始化系统时钟使用 6MHz 晶体为时钟源
//-----
void SYSCLK_Init (void)
{
int i; // 延时计数器
OSCXCN = 0x66; // 开启外部振荡器 6MHz 晶体
for (i=0; i < 256; i++); // 等待振荡器启振
while (!(OSCXCN & 0x80)); // 等待晶体振荡器稳定
OSCIEN = 0x88; // 选择外部振荡器为系统时钟源并允许丢失时钟检测器
}
//-----
// 配置 ADC0 使用定时器 3 溢出作为转换开始信号,转换结束产生一个中断,
// 使用左对齐输出模式
// 使能 ADC 转换结束中断, 使能 ADC0, 但禁止 ADC0 转换结束中断
//-----
void ADC0_Init (void)
{
ADC0CN = 0x05; // ADC0 禁止; 正常跟踪模式定时器 3 溢出 ADC0 转换开始
REF0CN = 0x07; // 使能温度传感器, 片内 VREF,和 VREF 输出缓冲器
AMX0SL = 0x00; // 选择 AIN0 作为 ADC 多路转换输出
ADC0CF = (SYSCLK/2500000) << 3; // ADC 转换时钟 2.5MHz
ADC0CF &= ~0x07; // PGA 增益= 1
EIE2 &= ~0x02; // 禁止 ADC0 中断
AD0EN = 1; // 使能 ADC0
}
//-----
// 配置定时器 3 自动重装间隔由<counts> 决定(不产生中断) 使用系统时钟为时基
//-----
void Timer3_Init (int counts)

```



```

{
TMR3CN = 0x02; // 停止定时器 3; 清除 TF3;
// 使用系统时钟作为时基
TMR3RL = -counts; // 初始化重装值
TMR3 = 0xffff; // 立即开始重装
EIE2 &= ~0x01; // 禁止定时器 3 中断
TMR3CN |= 0x04; // 启动定时器 3
}
//-----
//中断服务程序
//-----
// ADC0 中断服务程序
//-----
// ADC0 转换结束中断服务程序
// 得到 ADC0 采样值并存储到全局数组<samples[]>
// 并更新局部采样计数器<num_samples> 当<num_samples> == <NUM_SAMPLES>时,
// 禁止 ADC0 转换结束中断并置 ADC0_DONE = 1
//-----
void ADC0_ISR (void) interrupt 15 using 3
{
static unsigned num_samples = 0; // ADC0 采样计数器
AD0INT = 0; // 清除 ADC0 转换结束标志
samples[num_samples] = ADC0; // 读和存储 ADC0 值
num_samples++; // 更新采样计数器
if (num_samples == NUM_SAMPLES) {
num_samples = 0; // 复位采样计数器
EIE2 &= ~0x02; // 禁止 ADC0 中断
ADC0_DONE = 1; // 设置 DONE 标志
}
}
}

```

```

$INCLUDE (C8051F020.INC)
        ORG    0000H
        LJMP   MAIN
        ORG    0030H
MAIN:    MOV    WDTCN,#0DEH
        MOV    WDTCN,#0ADH
        MOV    XBR0,#04H
        MOV    XBR1,#00H
        MOV    XBR2,#40H
        MOV    ADC0CN,#05H
        MOV    REF0CN,#07H
        MOV    AMX0SL,#00H

```

```

MOV    ADC0CF,#18H
MOV    EIE2,#00H
SETB   AD0EN
MOV    TMR3CN,#02H
MOV    TMR3RLL,#136
MOV    TMR3RLH,#0FFH
MOV    TMR3H,#0FFH
MOV    TMR3L,#0FFH
MOV    TMR3CN,#04H
MOV    OSCXCN,#01100110B
MOV    R2,#00
LD:    MOV    R3,#00
        DJNZ  R3,$
        DJNZ  R2,LD
        MOV   A,OSCXCN
        JNB  ACC.7,LD
        MOV  OSCICN,#88H
LOOP:  MOV   R0,#10H
        MOV   R2,#10H
        ADC0: JNB AD0INT,$
        CLR  AD0INT
        MOV  @R0,ADC0H
        INC  R0
        MOV  @R0,ADC0L
        INC  R0
        DJNZ R2,ADC0
        LJMP LOOP
        END

```

12 位电压输出 DAC—数/模转换

每个 C8051F020/1/2/3 器件都有两个片内 12 位电压方式数/模转换器(DAC)。每个 DAC 的输出摆幅均为 0V 到 (VREF-1LSB)，对应的输入码范围是 0x000 到 0xFFF。

VREFA 引脚为 DAC0 和 DAC1 提供电压输入。通过配置 VREFA 模拟开关，DAC0 还可以使用 DAC0 的输出作为内部基准，DAC1 可以使用模拟电源电压作为基准。

DAC0H: DAC0 高字节寄存器

DAC0L: DAC0 低字节寄存器

DAC0CN: DAC0 控制寄存器

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	复位值
DAC0EN	-	-	DAC0MD1	DAC0MD0	DAC0DF2	DAC0DF1	DAC0DF0	00000000
位7	位6	位5	位4	位3	位2	位1	位0	SFR地址: 0xD4

位 7: DAC0EN: ADC0 允许位

0: DAC0 禁止。DAC0 输出引脚为高阻态，DAC0 处于节电停机方式

1: DAC0 允许。DAC0 正常输出；DAC0 处于工作状态。

位 6-5: 未用。读=00b；写=忽略。

位 4-3: DAC0MD1-0: DAC0 方式位。

00: DAC 输出更新发生在写 DAC0H 时。

01: DAC 输出更新发生在定时器 3 溢出时。

10: DAC 输出更新发生在定时器 4 溢出时。

11: DAC 输出更新发生在定时器 2 溢出时。

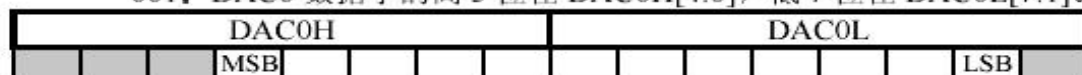
位 2-0: DAC0DF2-0: DAC0 数据格式位:

000: DAC0 数据字的高 4 位在 DAC0H[3:0]，低字节在 DAC0L 中。

000: DAC0 数据字的高 4 位在 DAC0H[3:0]，低字节在 DAC0L 中。



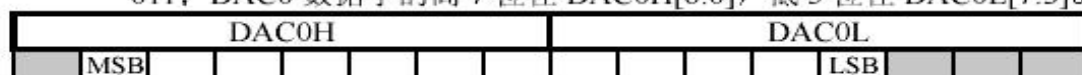
001: DAC0 数据字的高 5 位在 DAC0H[4:0]，低 7 位在 DAC0L[7:1]。



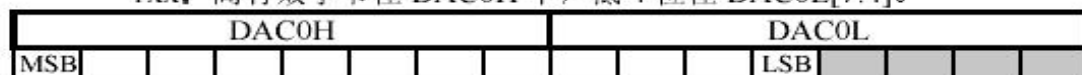
010: DAC0 数据字的高 6 位在 DAC0H[5:0]，低 6 位在 DAC0L[7:2]。



011: DAC0 数据字的高 7 位在 DAC0H[6:0]，低 5 位在 DAC0L[7:3]。



1xx: 高有效字节在 DAC0H 中，低 4 位在 DAC0L[7:4]。



REF0CN: 电压基准控制寄存器

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	复位值
-	-	-	AD0VRS	AD1VRA	TEMPE	BIASE	REFBE	00000000
位7	位6	位5	位4	位3	位2	位1	位0	SFR地址: 0xD1

位 7-5: 未用。读=000b, 写=忽略。

位 4: AD0VRS: ADC0 电压基准选择位

0: ADC0 电压基准取自 VREFA 引脚。

1: ADC0 电压基准取自 DAC0 输出。

位 3: AD1VRS: ADC1 电压基准选择位

0: ADC1 电压基准取自 VREFA 引脚。

1: ADC1 电压基准取自 AV+。

位 2: TEMPE: 温度传感器允许位

0: 内部温度传感器关闭。

1: 内部温度传感器工作。

位 1: BIASE: ADC/ DAC 偏压发生器允许位 (使用 ADC 和 DAC 时该位必须为 1)

0: 内部偏压发生器关闭。

1: 内部偏压发生器工作。

位 0: REFBE: 内部电压基准缓冲器允许位

0: 内部电压基准缓冲器关闭。

1: 内部电压基准缓冲器工作。内部电压基准提供从 VREF 引脚输出。

DAC0 产生锯齿波的实例

```
#include <c8051f020.h> // SFR declarations
//-----
// C8051F02X 的 16 位 SFR 定义
//-----
sfr16 RCAP4 = 0xe4; // 定时器 4 重装值
sfr16 T4 = 0xf4; // 定时器计数器
sfr16 DAC0 = 0xd2; // DAC0 数据
//-----
// 全局常量
//-----
#define SYSCLK 2000000 // 系统时钟频率 Hz
#define SAMPLERATED 100000L // DAC0 转换频率 Hz
//-----
// 函数原型
//-----
void Timer4_Init ( int counts );
void Timer4_ISR ( void );
//-----
// 全局变量
//-----
```

```

unsigned int   DPTR0 = 0x0000;//存放 DAC 数据
//-----
// 主程序
//-----
void main ( void )
{
    WDTCN = 0xde; // 禁止看门狗定时器
    WDTCN = 0xad;
    REF0CN = 0x03;//使能 DAC0
    DAC0CN = 0x90;//设置 DAC0
    Timer4_Init ( SYSCLK/SAMPLERATED );//初始化定时器 4
    EA = 1;
    while ( 1 );
}
//-----
// 初始化化子程序
//-----
// 配置定时器 4 自动重装间隔由<counts> 决定
//-----

void Timer4_Init ( int counts )
{
    CKCON |= 0x40;
    RCAP4 = -counts;
    T4 = RCAP4;
    EIE2 |= 0x04;
    T4CON |= 0X04;
}
//-----
//中断服务程序
//-----
// T4 中断服务程序
//-----

void Timer4_ISR ( void ) interrupt 16 using 3
{
    unsigned int   dptrl,dptrh;
    dptrl = DPTR0;
    dptrh = DPTR0;
    dptrh = dptrh >> 8;
    DAC0L = dptrl;
    DAC0H = dptrh;
    if ( DPTR0 <= 0x0FFF )DPTR0++;
    else DPTR0 = 0x0000;
}

```

```

$INCLUDE      (C8051F020.INC)
      ORG      0000H
      LJMP     MAIN
      ORG      0030H
MAIN:  MOV      WDTCN,#0DEH;禁止看门狗
      MOV      WDTCN,#0ADH
      MOV      REF0CN,#03H;使能内部电压发生器和内部电压基准缓冲器
      MOV      DAC0CN,#90H;使能 DAC0,使用定时器 4 更新 DAC0
      MOV      CKCON,#40H
      MOV      RCAP4L,#236
      MOV      RCAP4H,#0FFH
      MOV      0F4H,#0FFH
      MOV      0F5H,#0FFH
      MOV      EIE2,#04H
      MOV      T4CON,#04H
LOOP:  MOV      A,T4CON
      JNB ACC.7,LOOP
      MOV      T4CON,#04H
      INC DPTR
      MOV      DAC0L,DPL
      MOV      DAC0H,DPH
      LJMP     LOOP
      END

```

中断系统

CIP-51 包括一个扩展的中断系统，支持 22 个中断源，每个中断源有两个优先级。

某些中断标志在 CPU 进入 ISR 时被自动清除。但大多数中断标志不是由硬件清除的，必须在 ISR 返回前用软件清除。

中断源	中断向量	优先级	中断标志	使能	优先级控制
复位	0x0000	最高	无	始终使能	总是最高
外部中断 0 (INT0)	0x0003	0	IE0 (TCON.1)	EX0 (IE.0)	PX0 (IP.0)
定时器 0 溢出	0x000B	1	IE0 (TCON.5)	ET0 (IE.1)	PT0 (IP.1)
外部中断 1 (INT1)	0x0013	2	IE1 (TCON.3)	EX1 (IE.2)	PX1 (IP.2)
定时器 1 溢出	0x001B	3	TF1 (TCON.7)	ET1 (IE.3)	PT1 (IP.3)
UART0	0x0023	4	RI (SCON0.0) TI (SCON0.1)	ES0 (IE.4)	PS0 (IP.4)
定时器 2 溢出 (或 EXF2)	0x002B	5	TF2 (T2CON.7)	ET2 (IE.5)	PT2 (IP.5)
串行外设接口	0x0033	6	SPIF (SPI0CN.7)	ESPI0 (EIE1.0)	PSPI0 (EIP1.0)
SMBus 接口	0x003B	7	SI (SMB0CN.3)	ESMB0 (EIE1.1)	PSMB0 (EIP1.1)
ADC0 窗口比较	0x0043	8	AD0WINT (ADC0CN.2)	EWADC0 (EIE1.2)	PWADC0 (EIP1.2)
可编程计数器阵列	0x004B	9	CF (PCA0CN.7) CCFn (PCA0CN.n)	EPCA0 (EIE1.3)	PPCA0 (EIP1.3)
比较器 0 下降沿	0x0053	10	CP0FIF (CPT0CN.4)	ECP0F (EIE1.4)	PCP0F (EIP1.4)
比较器 0 上升沿	0x005B	11	CP0RIF (CPT0CN.3)	ECP0R (EIE1.5)	PCP0R (EIP1.5)
比较器 1 下降沿	0x0063	12	CP1FIF (CPT1CN.4)	ECPIF (EIE1.6)	PCPIF (EIP1.6)
比较器 1 上升沿	0x006B	13	CP1RIF (CPT1CN.3)	ECPIR (EIE1.7)	PCPIR (EIP1.7)
定时器 3 溢出	0x0073	14	TF3 (TMR3CN.7)	ET3 (EIE2.0)	PT3 (EIP2.0)
ADC0 转换结束	0x007B	15	AD0INT (ADC0CN.5)	EADC0 (EIE2.1)	PADC0 (EIP2.1)
定时器 4 溢出	0x0083	16	TF4 (T4CON.7)	ET4 (EIE2.2)	PT4 (EIP2.2)
ADC1 转换结束	0x008B	17	AD1INT (ADC1CN.5)	EADC1 (EIE2.3)	EADC1 (EIE2.3)
外部中断 6	0x0093	18	IE6 (PRT3IF.5)	EX6 (EIE2.4)	PX6 (EIP2.4)
外部中断 7	0x009B	19	IE7 (PRT3IF.6)	EX7 (EIE2.5)	PX7 (EIP2.5)
UART1	0x00A3	20	RI (SCON1.0) TI (SCON1.1)	ES1(EIE2.6)	PS1(EIP2.6)
外部晶体振荡器准备好	0x00AB	21	XTLVLD(OSC XCN.7)	EXVLD (EIE2.7)	PXVLD (EIP2.7)

中断一览表

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	复位值
EA	IEGF0	ET2	ES0	ET1	EX1	ET0	EX0	00000000
位7	位6	位5	位4	位3	位2	位1	位0	SFR地址: 0xA8
							(可位寻址)	

IE: 中断允许寄存器

位 7: EA: 允许所有中断。

该位允许/禁止所有中断。它超越所有的单个中断屏蔽设置。

0: 禁止所有中断源。

1: 开放中断。每个中断由它对应的中断屏蔽设置决定。

位 6: IEGF0: 通用标志位 0。

该位用作软件控制的通用标志位。

位 5: ET2: 定时器 2 中断允许位。

该位用于设置定时器 2 的中断屏蔽。

0: 禁止定时器 2 中断。

1: 允许 TF2 标志位 (T2CON.7) 的中断请求。

位 4: ES0: UART0 中断允许位。

该位设置 UART0 的中断屏蔽。

0: 禁止 UATR0 中断。

1: 允许 RI0 标志位 (SCON0.0) 或 TI0 标志位 (SCON0.1) 产生的中断。

位 3: ET1: 定时器 1 中断允许位。

该位用于设置定时器 1 的中断屏蔽。

0: 禁止定时器 1 中断。

1: 允许 TF1 标志位 (TCON.7) 的中断请求。

位 2: EX1: 外部中断 1 允许位。

该位用于设置外部中断 1 的中断屏蔽。

0: 禁止外部中断 1 中断。

1: 允许/INT1 引脚的中断屏蔽。

位 1: ET0: 定时器 0 中断允许位。

该位用于设置定时器 0 的中断屏蔽。

0: 禁止定时器 0 中断。

1: 允许 TF0 标志位 (TCON.5) 的中断请求。

位 0: EX0: 外部中断 0 允许位。

该位用于设置外部中断 0 的中断屏蔽。

0: 禁止外部中断 0。

1: 允许/INT0 引脚的中断请求。

IP: 中断优先级寄存器

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	复位值
-	-	PT2	PS0	PT1	PX1	PT0	PX0	11000000
位7	位6	位5	位4	位3	位2	位1	位0 (可位寻址)	SFR地址: 0xB8

位 7-6: 未用。读=11b, 写=忽略。

位 5: PT2: 定时器 2 中断优先级控制

该位设置定时器 2 中断的优先级。

0: 定时器 2 为默认优先级。

1: 定时器 2 为高优先级。

位 4: PS0: UART0 中断优先级控制。

该位设置 UART0 中断的优先级。

0: UART0 为默认优先级。

1: UART1 为高优先级。

位 3: PT1: 定时器 1 中断优先级控制

该位设置定时器 1 中断的优先级。

0: 定时器 1 为默认优先级。

1: 定时器 1 为高优先级。

位 2: PX1: 外部中断 1 优先级控制

该位设置外部中断 1 的优先级。

0: 外部中断 1 为默认优先级。

1: 外部中断 1 为高优先级。

位 1: PT0: 定时器 0 中断优先级控制

该位设置定时器 0 中断的优先级。

0: 定时器 0 为默认优先级。

1: 定时器 0 为高优先级。

位 0: PX0: 外部中断 0 优先级控制

该位设置外部中断 0 的优先级。

0: 外部中断 0 为默认优先级。

1: 外部中断 0 为高优先级。

EIE2: 扩展中断允许 2

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	复位值
EXVLD	ES1	EX7	EX6	EADC1	ET4	EADC0	ET3	00000000
位7	位6	位5	位4	位3	位2	位1	位0	SFR地址: 0xE7

位 7: EXVLD: 允许外部时钟源有效 (XTLVLD) 中断

该位设置 XTLVLD 的中断屏蔽。

0: 禁止 XTLVLD 中断。

1: 允许 XTLVLD 中断。

位 6: ES1: 允许 UART1 中断

该位时钟 UART1 的中断屏蔽。

0: 禁止 UART1 中断。

1: 允许 UART1 中断。

位 5: EX7: 允许外部中断 7

该位设置外部中断 7 的中断屏蔽。

0: 禁止外部中断 7 的中断屏蔽。

1: 允许外部中断 7 的中断屏蔽。

位 4: EX6: 允许外部中断 6 的中断屏蔽

0: 禁止外部中断 6 的中断。

1: 允许外部中断 6 的中断。

位 3: EADC1: 允许 ADC1 转换结束中断

该位设置 ADC1 转换结束的中断屏蔽。

0: 禁止 ADC1 转换结束中断。

1: 允许 ADC1 转换结束中断。

位 2: ET4: 允许定时器 4 中断

该位设置定时器 4 的中断屏蔽。

0: 禁止定时器 4 的中断屏蔽。

1: 允许 TF4 标志位 (T4CON.7) 产生的中断请求

位 1: EADC0: 允许 ADC0 转换结束中断

该位设置 ADC0 转换结束的中断屏蔽。

0: 禁止 ADC0 转换结束中断。

1: 允许 ADC0 转换结束产生的中断请求。

位 0: EX3: 允许定时器 3 中断

该位设置定时器 3 中断屏蔽。

0: 禁止定时器 3 中断。

1: 允许 TF3 标志位 (TMR3CN.7) 的中断请求。

EIP2: 扩展中断优先级 2

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	复位值
PXVLD	PS1	PX7	PX6	PADC1	PT4	PADC0	PT3	00000000
位7	位6	位5	位4	位3	位2	位1	位0	SFR地址: 0xF7

位 7: PXVLD: 外部时钟源有效 (XTLVLD) 中断优先级控制

该位设置 XTLVLD 的中断的优先级。

0: XTLVLD 中断为低优先级。

1: XTLVLD 中断为高优先级。

位 6: PS1: UART1 中断优先级控制

该位设置 UART1 的中断的优先级。

0: UART1 中断为低优先级。

1: UART1 中断为高优先级。

位 5: PX7: 外部中断 7 优先级控制

该位设置外部中断 7 的中断的优先级。

0: 外部中断 7 的为低优先级。

1: 外部中断 7 的为高优先级。

位 4: PX6: 外部中断 6 的优先级控制

0: 外部中断 6 的为低优先级。

1: 外部中断 6 的为高优先级。

位 3: EADC1: ADC1 转换结束中断优先级控制

该位设置 ADC1 转换结束的中断的优先级。

0: ADC1 转换结束中断为低优先级。

1: ADC1 转换结束中断为高优先级。

位 2: PT4: 允许定时器 4 中断优先级控制

该位设置定时器 4 的中断的优先级。

0: 定时器 4 的中断为低优先级。

1: 定时器 4 的中断为高优先级请求

位 1: PADC0: ADC0 转换结束中断优先级控制

该位设置 ADC0 转换结束的中断的优先级。

0: ADC0 转换结束中断为低优先级。

1: ADC0 转换结束中断为高优先级。

位 0: PX3: 定时器 3 中断中断优先级控制

该位设置定时器 3 中断的优先级为低优先级。

0: 定时器 3 中断为低优先级。

1: 定时器 3 中断为高优先级。

复位源

软件强制复位

向 PORSF 位写 1 将强制产生一个上电复位。

时钟丢失检测器复位

时钟丢失检测器实际上是由 MCU 系统时钟触发的单稳态电路。如果未收到系统时钟时间大于 100 微秒，单稳态电路将超时并产生一个复位。在发生时钟丢失检测器复位后，MCDRSF 标志（RSTSRC.2）将被置位。

看门狗定时器复位

MCU 内部有一个使用系统时钟的可编程看门狗定时器（WDT）。当看门狗定时器溢出时 WDT 将强制 CPU 进入复位状态。为了防止复位，必须在溢出发生前由软件重新触发 WDT。

禁止 WDT

向 WDTCN 寄存器写入 0xAD 将禁止 WDT。

```
CLR    EA                ; 所有中断
MOV    WDTCN,#0DEH      ; 禁止软件看门狗定时器
MOV    WDTCN,#0ADH
SETB   EA                ; 重新允许中断
```

写 0xDE 和写 0xAD 必须发生在 4 个时钟周期之内，否则禁止操作将被忽略。在这个过程中应禁止中断，以避免两次写操作之间有延时。

振荡器

每个 MCU 都有一个内部振荡器和一个外部振荡器驱动电路，每个驱动电路都能产生系统时钟。MCU 在复位后从内部振荡器启动。MCU 可以从内部振荡器或外部振荡器运行，可使用 OSCISN 寄存器中的 CLKSL 位在两个振荡器之间随意切换。外部振荡器需要一个外部振荡器、并行方式的晶体、电容或 RC 网络连接至 XTAL1/XTAL2 引脚。XTAL1 和 XTAL2 不耐 5V 电压。

OSCICN：内部振荡器控制器寄存器

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	复位值
MSCLKE	-	-	IFRDY	CLKSL	IOSCEN	IFCN1	IFCN0	00010100
位7	位6	位5	位4	位3	位2	位1	位0	SFR地址： 0xB2

位 7: MSCLKE: 时钟丢失检测器允许位

0: 禁止时钟丢失检测器。

1: 允许时钟丢失检测器；检测到时钟丢失时间大于 100 微秒时将触发复位。

位 6-5: 未用。读=00b，写=忽略。

位 4: IFRDY: 内部振荡器频率准备好标志

0: 内部振荡器频率不是按 IFCN 位指定的速度运行。

1: 内部振荡器频率按照 IFCN 位指定的速度运行。

位 3: CLKSL: 系统时钟源选择位

0: 选择内部振荡器作为系统时钟。

1: 选择内部振荡器作为系统时钟。

位 2: IOSCEN: 内部振荡器允许位

0: 内部振荡器禁止。

1: 内部振荡器允许。

位 1-0: IFCN1-0: 内部振荡器频率控制位

00: 内部振荡器典型频率为 2MHz。

01: 内部振荡器典型频率为 4MHz。

10: 内部振荡器典型频率为 8MHz。

11: 内部振荡器典型频率为 16MHz。

R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	复位值
XTLVLD	XOSCND2	XOSCND1	XOSCND0	-	XFCN2	XFCN1	XFCN0	00000000
位7	位6	位5	位4	位3	位2	位1	位0	SFR地址： 0xB1

OSCXCN：外部振荡器控制寄存器

位 7: XTLVLD: 晶体振荡器有效标志

(只在 XOSCND=1xx 时有效)

0: 晶体振荡器未用或未稳定。

1: 晶体振荡器正在运行并且工作稳定。

位 6-4: XOSCND2-0: 外部振荡器方式位

- 00x: 关闭。XTAL1 引脚内部接地。
- 010: 系统时钟为来自 XTAL1 引脚的外部 CMOS 时钟。
- 011: 系统时钟为来自 XTAL1 引脚的外部 CMOS 时钟的二分频。
- 10x: RC/C 振荡器方式二分频。
- 110: 晶体振荡器方式。
- 111: 晶体振荡器方式二分频。

位 3: 保留。读=无定义, 写=忽略。

位 2-0: XFCN2-0: 外部振荡器频率控制位。

XFCN	晶体 (XOSCMD=11x)	RC(XOSCMD=10x)	C(XOSCMD=10x)
000	$f \leq 12\text{kHz}$	$f \leq 25\text{kHz}$	K 因子= 0.44
001	$12\text{kHz} < f \leq 30\text{kHz}$	$25\text{kHz} < f \leq 50\text{kHz}$	K 因子= 1.4
010	$30\text{kHz} < f \leq 95\text{kHz}$	$50\text{kHz} < f \leq 100\text{kHz}$	K 因子= 4.4
011	$95\text{kHz} < f \leq 270\text{kHz}$	$100\text{kHz} < f \leq 200\text{kHz}$	K 因子= 13
100	$270\text{kHz} < f \leq 720\text{kHz}$	$200\text{kHz} < f \leq 400\text{kHz}$	K 因子= 38
101	$720\text{kHz} < f \leq 2.2\text{MHz}$	$400\text{kHz} < f \leq 800\text{kHz}$	K 因子= 100
110	$2.2\text{MHz} < f \leq 6.7\text{MHz}$	$800\text{kHz} < f \leq 1.6\text{MHz}$	K 因子= 420
111	$f > 6.7\text{MHz}$	$1.6\text{MHz} < f \leq 3.2\text{MHz}$	K 因子= 1400

选用外部 6MHz 晶振实例:

```
void SYSCLK_Init(void)
```

```
{
```

```
int i; // 延时计数器
```

```
OSCXCN = 0x66; // 开启外部振荡器 6MHz 晶体
```

```
for (i=0; i < 256; i++); // 等待振荡器启振
```

```
while (!(OSCXCN & 0x80)); // 等待晶体振荡器稳定
```

```
OSCICN = 0x88; // 选择外部振荡器为系统时钟源并允许丢失时钟检测器
```

```
}
```

```
MOV OSCXCN,#01100110B;开启外部振荡器 6MHz 晶体
```

```
MOV R2,#00;等待振荡器启振
```

```
LD: MOV R3,#00
```

```
DJNZ R3,$
```

```
DJNZ R2,LD
```

```
MOV A,OSCXCN
```

```
JNB ACC.7,LD;判断晶体振荡器稳定
```

```
MOV OSCICN,#88H;选择外部振荡器为系统时钟源并允许丢失时钟检测器
```

配置外部存储器接口

配置外部存储器接口的过程包括下面 4 个步骤：

1. 将 EMIF 选到低端口（P3、P2、P1 和 P0）
2. 选择复用方式或非复用方式。
3. 选择存储器模式（只用片内存储器、不带块选择的分片方式、带块选择的分片方式或只用片外存储器）。
4. 设置与片外存储器或外设接口的时序。
5. 选择所需要的相关端口的输出方式（寄存器 PnMDOUT 和 P74OUT）。

EMI0CF：外部存储器接口配置

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	复位值
-	-	PRTSEL	EMD2	EMD1	EMD0	EALE1	EALE0	00000011
位7	位6	位5	位4	位3	位2	位1	位0	SFR地址： 0xA3

位 7-6：未用。读=00b，写=忽略。

位 4：EMD2：EMIF 复用方式选择位

0：EMIF 工作在地址/数据复用方式。

1：EMIF 工作在非复用方式（分离的地址和数据引脚）。

位 3-2：EMD1-0：EMIF 工作模式选择位

这两位控制外部存储器接口的工作模式。

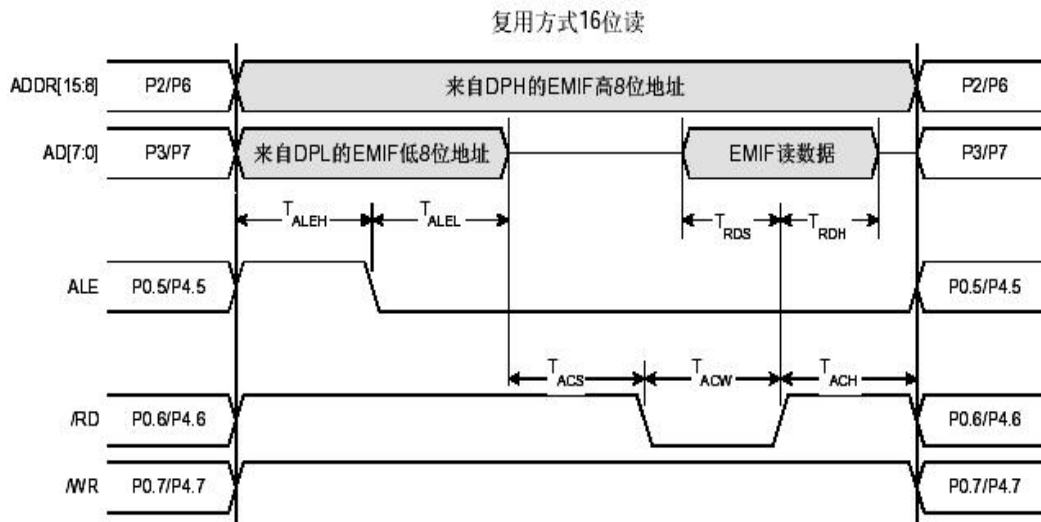
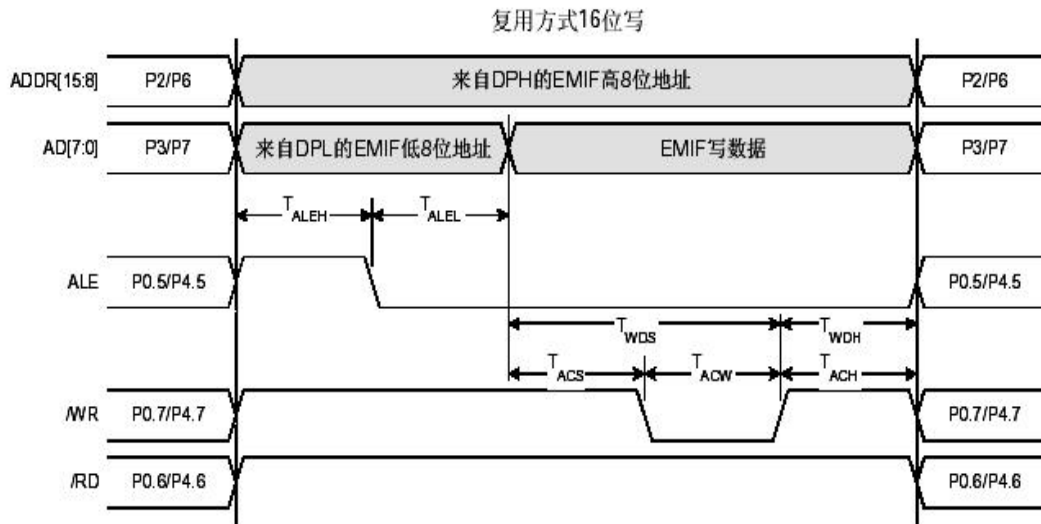
00：只用内部存储器：MOVX 只寻址片内 XRAM。所有有效地址都指向片内存储器空间。

11：只用外部存储器：MOVX 只寻址片外 XRAM。片内 XRAM 对 CPU 不可见。

只用外部存储器的设置

当 EMI0CF.[3:2]被设置为‘11’时，所有 MOVX 指令都将访问器件外部 XRAM 空间。片内 XRAM 对 CPU 为不可见。

16 位 MOVX 操作使用 DPTR 的内容确定有效地址 A[15:0]。在访问片外存储器时地址总线 A[15:0]的全部 16 都被驱动。



外部存储器接口的 AC 参数
复用方式时序图

端口 0-3 和优先权交叉开关译码器

C8051F023 器件有大量的数字资源需要通过 4 个低端 I/O 端口 P0、P1、P2 和 P3 才能使用。P0、P1、P2 和 P3 中的每个引脚即可定义为通用的端口 I/O(GPIO) 引脚，又可以分配给一个数字外设或功能（例如：UART0 或/INT1）。这种资源

参数	说明	最小值	最大值	单位
T_{SYSCLK}	系统时钟周期	40		ns
T_{ACS}	地址/控制建立时间	0	$3 * T_{SYSCLK}$	ns
T_{ACW}	地址/控制脉冲宽度	$1 * T_{SYSCLK}$	$16 * T_{SYSCLK}$	ns
T_{ACH}	地址/控制保持时间	0	$3 * T_{SYSCLK}$	ns
T_{ALEH}	地址锁存使能高电平时间	$1 * T_{SYSCLK}$	$4 * T_{SYSCLK}$	ns
T_{ALEL}	地址锁存使能低电平时间	$1 * T_{SYSCLK}$	$4 * T_{SYSCLK}$	ns
T_{WDS}	写数据建立时间	$1 * T_{SYSCLK}$	$19 * T_{SYSCLK}$	ns
T_{WDH}	写数据保持时间	0	$3 * T_{SYSCLK}$	ns
T_{RDS}	读数据建立时间	20		ns
T_{RDH}	读数据保持时间	0		ns

分配的灵活性是通过优先权交叉开关译码器实现的。

优先权交叉开关译码器，或称为“交叉开关”，按优先权顺序将端口 0-3 的引脚分配给器件上的数字外设（UART、SMBus、PCA、定时器等）。端口引脚的分配顺序是从 P0.0 开始，可以一直分配到 P3.7。UART0 具有最高优先权，而 CNVSTR 具有最低优先权。当交叉开关配置寄存器 XBR0、XBR1 和 XBR2 中外设的对应允许位被设置为逻辑‘1’时，交叉开关将端口引脚分配给外设。

端口 0-3 中所有未被交叉开关分配的引脚都可以作为通用 I/O (GPIO) 引脚，通过读或写相应的端口数据寄存器访问，这是一组既可以按位寻址也可以按字节寻址的 SFR。被交叉开关分配的那些端口引脚的输出状态受使用这些引脚的数字外设的控制。向端口数据寄存器（或相关的端口位）写入时对这些引脚的状态没有影响。

不管交叉开关是否将引脚分配给外设，读一个端口数据寄存器（或端口位）将总是返回引脚本身的逻辑状态。

交叉开关寄存器被正确配置后，通过将 XBARE(XBR2.4)设置为逻辑‘1’来使能交叉开关。在 XBARE 被设置为逻辑‘1’之前，端口 0-3 的输出驱动器被禁止，以防止对交叉开关寄存器和其它寄存器写入时在端口引脚上产生争用。

被交叉开关分配给输入信号（例如 RX0）的引脚所对应的输出驱动器被禁止；因此端口数据寄存器和 PNMDOUT 寄存器的值对这些引脚的状态没有影响。

每个端口引脚的输出方式都可被配置为漏极开路或推挽方式，缺省状态为漏极开路。在推挽方式，向端口数据寄存器中的相应位写逻辑‘0’将使端口引脚被驱动到 GND,写逻辑‘1’将使引脚被驱动到 VDD。在漏极开路方式，向端口数据寄存器中的相应位写逻辑‘0’将使端口引脚被驱动 GND,写逻辑‘1’将使端口引脚处于高阻状态。当系统中不同器件的端口引脚有共享连接，即多个输出

连接到同一个物理线时（例如 SMBus 连接中的 SDA 信号），使用漏极开路方式可以防止不同器件之间的争用。

通过设置输出方式为“漏极开路”并向端口数据寄存器中的相应位写‘1’将端口引脚配置为数字输入。

弱上拉

每个端口引脚都有一个内部弱上拉部件，在引脚与 VDD 之间提供阻性连接（约 100kΩ），在缺省情况下该上拉器件被使能。弱上拉部件可以被总体禁止，通过向弱上拉禁止位（WEAKPUD,XBR2.7）写‘1’实现。当任何引脚被驱动为逻辑‘0’时，弱上拉自动取消；即输出引脚不能与其自身的上拉部件冲突。对于端口 1 的引脚，将引脚配置为模拟输入时上拉部件也被禁止。

外部存储器接口引脚分配

如果外部存储器接口(EMIF)被设置在低端口(端口 0-3), EMIFLE(XBR2.5)位应被设置为逻辑‘1’,以使交叉开关不将 P0.7(/WR)、P0.6(/RD)和 P0.5(/ALE)(如果外设存储器接口使用复用方式)分配给外设。

XBR0: 端口 I/O 交叉开关寄存器 0

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	复位值
CP0E	EC10E	PCA0ME			UART0EN	SPI0EN	SMB0EN	00000000
位7	位6	位5	位4	位3	位2	位1	位0	SFR地址: 0xE1

- 位 2: UART0EN: UART0 I/O 允许位
- 0: UART0 I/O 不连到端口引脚。
- 1: RX0、TX0 连到 2 个端口引脚。

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	复位值
SYSCKE	T2EXE	T2E	INT1E	T1E	INT0E	T0E	CPIE	00000000
位7	位6	位5	位4	位3	位2	位1	位0	SFR地址: 0xE2

XBR1: 端口 I/O 交叉开关寄存器 1

- 位 7: SYSCKE: SYSCLK 输出允许位
- 0: SYSCLK 不连接到端口引脚。
- 1: SYSCLK 连接到端口引脚。
- 位 6: T2EXE: T2EX 允许位
- 0: T2EX 不连接到端口引脚。
- 1: T2EX 连接到端口引脚。
- 位 5: T2E: T2 允许位
- 0: T2 不连到端口引脚。
- 1: T2 连到端口引脚。
- 位 4: INT1E: /INT1 允许位。
- 0: /INT1 不连到端口引脚。

- 1: /INT1 连到端口引脚。
- 位 3: T1E: T1 允许位
 - 0: T1 不连到端口引脚。
 - 1: T1 连到端口引脚。
- 位 2: INT0E: /INT0 允许位
 - 0: /INT0 不连到端口引脚。
 - 1: /INT0 连到端口引脚。
- 位 1: T0E: T0 允许位
 - 0: T0 不连到端口引脚。
 - 1: T0 连到端口引脚。

XBR2: 端口 I/O 交叉开关寄存器 2

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	复位值
WEAKPUD	XBARE	-	T4EXE	T4E	UART1E	EMIFLE	CNVSTE	00000000
位7	位6	位5	位4	位3	位2	位1	位0	SFR地址: 0xE3

- 位 7: WEAKPUD: 弱上拉禁止位
 - 0: 弱上拉全局允许。
 - 1: 弱上拉全局禁止。
- 位 6: XBARE: 交叉开关允许位
 - 0: 交叉开关禁止。端口 0、1、2 和 3 所有引脚被强制为输入方式。
 - 1: 交叉开关允许。
- 位 4: T4EXE: T4EX 输入允许
 - 0: T4EX 不连接到端口引脚。
 - 1: T4EX 连接到端口引脚。
- 位 3: T4E: T4 输入允许位
 - 0: T4 不连接到端口引脚。
 - 1: T4 连接到端口引脚。
- 位 2: UART1E: UART1 I/O 允许位
 - 0: UATR1 I/O 不连接到端口引脚。
 - 1: UATR1 TX 和 RX 连接到两个端口引脚。
- 位 1: EMIFLE: 外部存储器接口低端口允许位
 - 0: P0.7、P0.6 和 P0.5 功能由交叉开关或端口锁存器决定。
 - 1: 如果 EMI0CF.4=0 (外部存储器接口为复用方式), 则 P0.7 (/WR)、P0.6 (/RD)、P0.5 (/ALE) 被交叉开关跳过, 它们的输出状态由端口锁存器和外部存储器决定。
 - 1: 如果 EMI0cf.4=1 (外部存储器接口为非复用方式), 则 P0.7 (/WR)、P0.6 (/RD) 被交叉开关跳过, 它们的输出状态由端口锁存器和外部存储器决定。
- 位 0: CNVSTE: 外部转换启动输入允许位
 - 0: CNVSTR 不连接到端口引脚。
 - 1: CNVSTR 连接到端口引脚。

PnMDOUT 端口 n 输出方式寄存器

PnMDOUT[7:0]: 端口 n 输出方式位

- 0: 端口引脚的输出方式为漏极开路。
- 1: 端口引脚的输出方式为推挽

P1MDIN: 端口 1 输入方式寄存器

P1MDIN[7:0]: 端口 1 输入方式位

0: 端口引脚被配置为模拟输入方式。数字输入通路被禁止（读端口位将总是返回 0）。
引脚的弱上拉 被禁止。

1: 端口引脚被配置为数字输入方式。读端口位将返回引脚的逻辑电平。弱上拉状态由
WEAKPUD (XBR2.7) 位决定。

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	复位值
WEAKPUD	XBARE	-	T4EXE	T4E	UART1E	EMIFLE	CNVSTE	00000000
位7	位6	位5	位4	位3	位2	位1	位0	SFR地址: 0xE3

P3IF: 端口 3 中断标志寄存器

位 7: IE7: 外部中断 7 标志位

0: P3.7 引脚没有检测到下降沿。

1: 当检测到 P3.7 引脚的下降沿时刻标志由硬件置位。

位 6: IE6: 外部中断 6 标志位

0: P3.6 引脚没有检测到下降沿。

1: 当检测到 P3.6 引脚的下降沿时刻标志由硬件置位。

位 3: IE7CF: 外部中断 7 边沿配置位

0: 外部中断 7 由 IE7 输入的下降沿触发。

1: 外部中断 7 由 IE7 输入的上升沿沿触发。

位 2: IE6CF: 外部中断 6 边沿配置位

0: 外部中断 6 由 IE7 输入的下降沿触发。

1: 外部中断 6 由 IE7 输入的上升沿沿触发。


```

void PORT_Init (void)
{
EMI0CF = 0x0F; //EMIF 工作在数据/地址复用方式
XBR0 = 0x04; //将 UART0 的 TX0 和 RX0 引脚连接到端口
XBR1 = 0x1E; //将 T0、/INT0、T1 和/INT1 连接到端口引脚
XBR2 = 0x42; //使能交叉开关并使交叉开关跳过 P0.5~P0.7 引脚
P1MDOUT = 0xFF; //设置 P1 的引脚输出为推挽方式
}

```

```

MOV    EMI0CF,#0FH
MOV    XBR0,#04H
MOV    XBR1,#1EH
MOV    XBR2,#42H
MOV    P2MDOUT,#0FFH

```

这样配置后的连接方式为:

TX0—P0.0	RX0—P0.1	T0—P0.2	/INT0—P0.3	T1—P0.4
/ALE—P0.5				
/RD—P0.6	/RD—P0.7	/INT1—P1.0		