

一 S3C9488 的硬件资源

CPU 内核	Watch Timer(1 个)
SAM88RCRI 内核	输出 4 种频率 BUZ 产生 LCD 时钟驱动
存储空间	LCD 控制/驱动电路
208 个字节的通用寄存器	8COM×19SEG
4K/8K 的程序存储空间	4COM×19SEG
时钟频率	A/D 转换 (9 路)
陶瓷晶振、石英晶振、RC 振荡 时钟分频器 (1/1、1/2、1/8、1/16)	9 路 A/D 转换 12.5us 转换速度 (f=4MHZ)
指令集	同步 UART(1 个)
41 条指令 IDLE(空闲模式)、STOP(冻结模式)	波特率可变 支持串口接受/传送数据
指令执行周期	Watchdog Timer
500ns(fosc=8MHZ)	2 个时钟源选择 (Smart Option)
中断	低电压复位 (LVR)
10 个中断源、一个中断向量、一个中断 优先级	内部集成低电压复位电路 V _{LVR} =2.6V/3.3V/3.9V
I/O 口	低电压检测
38 个可编程引脚 (44QFP) 36 个可编程引脚 (42DIP) 26 个可编程引脚 (32DIP/32SOP)	可检测 2.4v/2.7v/3.3v/3.9v
Bsiac Timer(1 个)	操作温度
1 个可编程 8 位 Basic Timer(BT)	-25 ---85
定时器 (8 位 A/B , 2 个)	操作电压
1 个 8 位定时/计数器 A (3 种工作方式)	2.2v—5.5v fosc=4MHZ
1 个 8 位定时/计数器 B	2.7v—5.5v fosc=8MHZ

二 S3C9488 地址结构

S3C9488 有 2 类地址空间

程序存储空间

通用寄存器卷

一 程序存储空间

起始 2 个字节的地址单元为中断向量区 (0000H—0001H)

3CH、3DH、3EH、3FH 用于 Smart Option 选择

程序起始地址为 0100H

整个程序地址空间分配如下：



程序存储空间中没有用到的存储区，如 0002H—003CH 可以用来存放程序。程序存储单元的缺省值为‘0FFH’。而 NOP 指令也是‘FFH’。

Smart Option

003CH 没有用到。3CH、3EH、3FH 中没有用到的必须初始化为 1

003DH 决定 P3CONH 的复位值

003EH 禁止/允许低电压复位（当允许 LVR 时，必须设定相应的低电压复位值）

003FH 内部晶振选择、Watchdog Timer 时钟选择、复位脚选择、必须决定 P0.0~P0.2 的功能

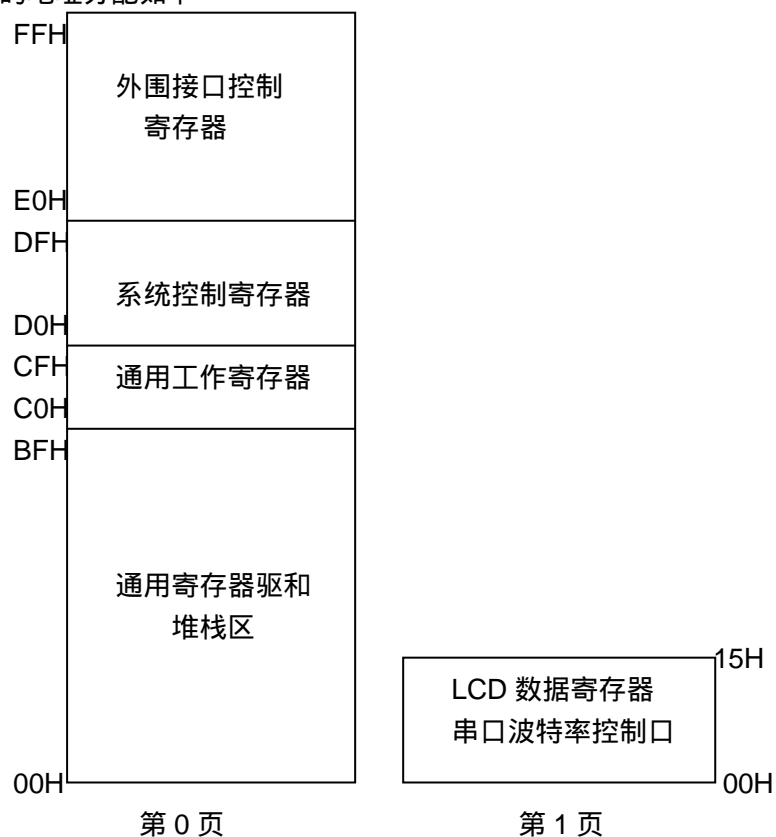
二 数据存储空间

数据存储空间分 2 页，其中第 1 页用于存储液晶显示内容，在不用于液晶显示时，可以用作普通的存储单元。

工作寄存器可以合并为 16 位的寄存器，但起始寄存器地址必须为偶数。如，RR0，RR2 组成的 16 位地址中 R_{n+1} 为低字节， R_n 为高字节

S3C9488 的堆栈区在通用寄存器卷中。系统复位后，堆栈指针寄存器的值是不确定。S3C9488 的堆栈是向下生成的。初始化程序必须把 SP 初始化为 00H—0C0H 之间的任意值，不可为缺省值。在压栈时，指针先减 1；在出栈时，弹出数据之后，指针再加 1。初始化 SP 时，其值不可为 00H，因为在压栈时，指针减 1 之后，将会进入到非法操作区。调子程序，2 个字节压栈；中断时，3 个字节压栈。

数据存贮空间的地址分配如下



三 地址访问模式

S3C9488 有 7 种地址访问模式

寄存器访问模式 (R)

间接寄存器访问模式

偏址访问模式

直接访问模式

间接访问模式

相对地址访问模式

立即数访问模式

寄存器访问模式

寄存器访问模式即操作数在寄存器中或寄存器对中。

DEC R1

DEC RR0

ADD R1,R2

间接寄存器访问模式

间接寄存器访问模式,寄存器中存放的值是操作数的地址。

RL @SHIFT

OR R1,@R6

LD R1,@R6

偏址访问模式

偏址访问模式,指令中提供的基地址加上指令中的偏移地址即为操作数的地址,它分为短指令格式合长指令格式。短指令格式中,偏移范围为-128~127;长指令格式中,偏移范围

为 16 地址形式。支持偏址访问的指令有 LD、LDC、LDE 3 条指令。

```
LD    R0,#100[R1]
LD    R4,#1000[RR2]
```

直接访问模式

直接访问模式，操作数在指令提供的地址存储空间中（该地址为直接地址，非寄存器）。

```
LDC   R1,1234H
JP    C,JOB1
CALL  DISPLAY
```

间接访问模式

间接访问模式,指令中直接提供转移的地址，地址只能为程序存储空间的低 256 字节。支持此种访问方法的指令只有 CALL 指令。

```
CALL  #40H
```

相对地址访问模式

相对地址访问模式，指令中会提供-128~127 之间的偏移地址，偏移量将会加到当前 PC 上，最后的 PC 值即为要调转到的地址。只有 JR 指令支持此地址访问方法。

```
JR    ULT,$+OFFSET
```

立即数访问模式

立即数访问模式，即指令中直接提供操作数。

```
LD    R1,#0AAH
```

四 编译器的伪指令

三星编译器有 2 种， SAMA； SASM

对 SAMA 汇编器需要一个驱动文件.DEF 文件（.def 文件描述微控制器）。汇编源文件的扩展名必须为.SRC。

(一) SAMA 汇编器

SAMA 可在 M-DOS 环境下运行，直接在 M-DOS 环境下输入 SAMA 即可。调用此汇编器，在此编译环境下，输入文件名即可。在 DOS 环境下，汇编文件有以下缺省配置：

```
/-A, /E, /K, /-M, /P, /HEX
```

不同缺省值配置，在输入命令时，中间用空格隔开。

```
如 SAMA TEST /E /HEX
```

输出文件

```
TEST.HEX    16 进制代码文件
TEST.LST    列表文件
```

} 汇编之后，自动生成这 2 个文件

.MSG, .SRM, .FIL, .INX 等生成文件用于程序调试。

.LST 文件中包含以下四个部分：

行号：程序的行数

ROM 地址：ROM 的物理地址空间

生成的代码：可执行文件代码

源文件代码：程序的汇编源文件代码

其列表文件的格式如下：

```

1 0000          CHIP    C:\def\88C4116.DEF
2 0000
3 0000
4 0020          org     0020h
5 0020  8D 01 00  jp     t,initial
6 0023
7 0023
8 0088          ORG     0088h
9 0088  02 45     vector  pwmov_int
10 00BA  02 4E     vector  pwmcap_int
.....
50 0104  5F          sb1
51 0105  08 F9     ld     r0, adin
52 0107  4F          sb0
53 0108  E6 D0 55   ld     0d0h, #055h
54 010B  0C D0     ld     r0, #0d0h
55 010D  D6 C0 AA   ld     @r0, #0aah
56 0110  E5 C0 12     ld     12h,@r0
57 0113  FF          nop
58 0114  E5 C0 55     ld     55h,@r0
59 0117  D6 C0 55     ld     @r0, #055h
60 011A  FF          nop
.....

No warnings issued  No errors were found
Equates: 1   Labels: 38   Program size: 379 bytes

```

16 进制文件主要包括以下几个部分

标号：起始位置标号

长度：2 个 ASCII 字符指定数据字节段的长度

装载地址：4 个 ASCII 字符指定开始装载的起始地址

类型：指定段类型；20 或 00 对应数据段，22 对应代码段

总数检查：长度段、装载地址、类型、数据段等的二进制总和，而后把二进制总和转换为 ASCII 码。

源文件格式

典型的源文件指令包括以下几个部分： 标号， 指令， 操作数， 注释

```

TABLE1:      LD      R2,      #12H      ;指令
  ↓             ↓       ↓         ↓           ↓
  标号         指令  目的操作数  源操作数   注释

```

标号：可以是数字、字母或下划线。但起始位置必须为字母，最长为 31。

标号不可重复，不可用保留字符，指令前没有标号时，前面要有空格。

指令：指令可以从第 2 列开始写，但我们建议从第 8 列写，前面用空格或 TAB

操作数：操作数之间要用逗号或空格隔开。如果 16 进制开始为 A~F，则前面要加 0。

```
LD 0A0H,#0AH ;正确
```

```
LD A0H,#0AH ;错误
```

特殊符号：\$：代表当前地址；#：立即数；@：间接寻址符号

```
JR $-3H
```

```
JP $+100H
```

系统缺省设置为十进制，如果用 16 进制或 2 进制，需在数字后面加相应后缀。

在写程序时，SAMA 汇编器不分大小写，除非用 DB 定义字符串时，区分大小写。

伪指令

SAMA 用到的伪指令有：

CHIP, END	程序开始加载驱动文件和程序终止
EQU	符号定义
ORG, REG_ORG, VENT	定义地址
DB, DW, DH, DL, DCS, DRS,	数据定义及地址空间分配
TCALL, TJP	表参考
EJECT, OBJECT, TITLE, FROM, LISTING, INCLUDE	列表控制伪指令
IF, ELSE, THEN	条件汇编
MACRO, ENDM	宏指令

.CHIP 伪指令定义.DEF 驱动文件

```
.CHIP C:\OPENICE\INCLUDE\DEF\57C21132.DEF
```

.END 说明程序代码完毕，只能在程序结尾用，前面不能有标号。

.EQU 定义符号，在定义位时，后面要有'.'，特殊功能寄存器的定义在.DEF 文件中，在.DEF 中用到的定义符号，在程序中不能再用。

```
例： MOT      EQU    23H
      TAB      EQU    MOT+5H
      BITM     EQU    P4.2
      RAMA     EQU    0FF0H.3
      RAMB     EQU    0FAH.3
```

.ORG 定义程序代码起始地址

.VENTn 定义中断入口地址向量（4 位微控制器）

```
ORG 0H
VENT0 1,1,RESET
VENT1 1,0,INTB
```

DB 字节定义

DW 字定义

DH 定义高字节，把程序存储空间标号的高字节定义到当前位置

DL 定义低字节

DCS 定义程序代码空间，后面的数字告诉汇编器要用到多少空间，也可以初始化存储单元的值，如果没有指定值，则为'0FFH'。

```
DCS 10H,23H ;定义 10H 个存储单元且值为 23H
```

.DRS 分配 RAM 和寄存器存储空间，前面需要标号。

```

REG_ORG      40H
A      DRS      1      ;40H
B      DRS      1      ;41H
E      DRS      2      ;42H
C      DRS      1      ;44H
D      DRS      1      ;45H
ORG      20H
LD      A,B      ;=LD      40H,41H
LD      C,D      ;=LD      44H,45H

```

.TCALL 用于程序存储空间 20H~7FH 存储区的参考表。

```

ORG      20H
RTCI0: TCALL ABC
RTCI1: TCALL XY2
.....
ORG      80H
.....
ABC: LD      A,#04H
.....
XYZ: LD      A,#0AH
.....
REF      RTCI0      ;=CALL      ABC
REF      RTCI1      ;=CALL      XYZ

```

.TJP TJP 的用法同 TCALL

.INCLUDE 伪指令用于调入其它源程序到当前程序中。在 INCLUDE 的文件中不可用 END 位指令。

.TITLE 该指令可以在输出文件中输出自己编写的程序标题，最长 40 个字符。

.LISTING 用于源程序部分的列表。如果源程序中有一部分不要出现在列表文件中则用 LISTING OFF 伪指令；如果源程序中有一部分要出现在列表文件中，则用 LISTING ON 位指令。默认情况下，源程序都出现在列表中。

.FROM 该指令可以改变列表页的行数。默认情况下，列表文件通常为 56 行。
FROM ON, 10H

.EJECT 该伪指令告诉编译器从当前位置起产生新一页列表文件。

.OBJECT 该伪指令将会在列表文件中生成每条指令所占的字节数。默认情况下，列表中都会生成指令所占字节数。打开 ON，关闭 OFF。

.MACRO 宏定义

```

例 1 : ADD16:      MACRO      A1,A2,A3,A4
                        ADD      A2,A4
                        ADC      A1,A3
                        ENDM
.....
                        NOP
                        ADD16      R0,R1,R2,R3

```

(二) SASM 汇编器

SASM 汇编器命令选择

-exxx 设置最大的错误输出个数 (缺省为 15 个)
 -Exxxx 错误信息输出到文件名为xxx的文件
 -m 设置列表文件中的宏扩展行 (缺省值为 64 行)
 -s 在目标文件中输出 SLO 格式的调试信息
 -v 显示版本信息

输出文件 .hex 可执行程序代码, 二进制文件
 .SIT,DBG 适用于调试环境文件
 .LST 源文件和程序代码文件
 .MAP 段和全局符号表信息

源文件编写

一个完整的指令语句包括 4 个部分:

(标号) 指令 操作数 (注释)

标号区分大小写, 指令、伪指令、条件转移代码、工作寄存器不区分大小写。标号可以用字母、数字或下划线, 但起始位置必须为字母或下划线。一个程序行最长为 255 个字符, 超过的字符将被忽略; 标号最长为 31 个字符, 标号在顶行写时, 可不要冒号; 不顶行时, 要冒号。伪指令开始可以用 '.'; 注释行用 ';' 与指令分开。

SASM 伪指令

ORG/ORIGIN 定义程序起始地址
 EQU/EQUAL/EQUATE 定义符号值

HOURS:	.EQUATE	24	
MINS:	.EQUAL	60	
NUMBER:	.EQU	65536*2/128	
EQU1:	.EQU	EQU2	;错误
EQU2:	.EQU	0	
EQU3:	.EQU	EQU2	;正确

SEC/SECTION 定义数据段或程序段, PROGRAM 定义程序体, DATA 定义数据段

格式: SECTION OPTION
 SEC

例: ORG 100H
 ADD R0,#0

PROGRAM: .SECTION ; 定义程序段名
 DATA: .SEC ; 同上
 ADD R0,#0
 .DATA ; 定义数据段
 DW 1234H
 .PROGRAM ; 定义程序段
 ADD R0,#0

GLOBAL/PUBLIC 全局标号定义
 EXTERN/EXTERNAL 外部符号定义

DEFVAR	定义并初始化标号 ;如果没有初始化用 DEFVAR 定义的变量,将会被初始化为 0。用 DEFVAR 定义的变量值可以用 SETVAR 再次设定变量的值;而用 EQU 定义的变量则不可以,其值是恒定的。
格式:	.DEFVAR [=NUM,] ; .SETVAR 用于给.DEFVAR 定义的变量重新赋值
例:	.DEFVAR =10H,A,B,C,D ;定义 A , B , C , D 并初始化其值为 10H LD R0,#A ;R0=10H A .SETVAR 20H LD R0,#A ;R0=20H
例:	YEAR .EQU 1998 .DEFVAR =YEAR,DATE DATE .SETVAR DATE-1 ;DATE=1997 DATE .SETVAR DATE+2 ;DATE=1999
VECTOR	定义中断向量地址
格式:	VECTOR VECTOR_ADDR,SERVICE_ADDR ;(SAM8) VENT EMB,ERB,SERVICE ;(SAM4)
.BYTE/DB	定义 8 位数据
	.DB n,..... .BYTE n,.....
例:	DATA1 .DB 1,2,3 DATA2 .DB 11,12,13
	在 SAM8 中
	LDC R0,DATA1 LDC R1,DATA2+1
WORD/DW	定义 16 位数据
	.DW n,..... .WORD n,.....
例:	DATA1 .DW 1,2,3 DATA2 .DW 1234H
DL/LONG	定义 32 位数据
	.DL n,..... .LONG n,.....
例:	DATA3 .DL 1,2,3 DATA4 .LONG 9ABCH
.ASCII/ASCIZ	定义字符串。ASCII 不在最后产生空格,ASCIZ 在最后产生空格
例:	TITLE .ASCII "SAM8 test program" ;不产生空格 TITLE .ASCIZ "I am", "aby" ;在每句后面产生空格
.FLOAT	把值转化为 32 位数据
.DOUBLE	把值转化为 63 位数据
.BLKB/BLOCK	保留几个字节的存贮空间。如果指定有值,把相应的值填入此空间;如果没有指定的值,则把 0 填入此空间。

格式： | .BLKB n[,value]
 | .BLOCK n[,value]
 .BLKB 10 ;10 字节的 0
 .BLOCK 4,33H ;4 字节的 33H
 BLKW 保留几个字的存储空间。同上。

例： | .BLKW n[,value]
 .BLKW 10
 .BLKW 5,1234H

RAM_DS 把当前的 RAM 地址或寄存器地址分配给相应的变量。
 .RAM_ORG 40H
 A: .RAM_DS 1 ;40H
 B .RAM_DS 1 ;41H
 E .RAM_DS 2 ;42H
 C .RAM_DS 1 ;44H
 D .RAM_DS 1 ;45H
 .ORG 20H
 LD A,B ;LD 40H,41H
 LD C,D ;LD 44H,45H

TACLL,TJP 只适应于 SAM4
 .MACRO 宏指令。在定义宏体参数时，各参数用‘,’隔开。利用符号传递参数，而不是具体的值。在调用宏的时候，必须先定义，而宏的编译是在调用的时候编译的，不是在定义的时候编译的。

格式： || .MACRO f,.....

 .ENDM

定义宏的时候，必须有标号。关于局部标号/全局标号，在三星编译器中，不区分局部标号和全局标号。

```
RAMCLR: .MACRO S1,S2
        LD    R0,S1
        LD    R1,S2
CLR:    LD    R0,#0
        INC   R0
        CP    R1,R0
        JR    NE,CLR
        .ENDM
```

```
.....
RAMCLR #00H,#80H
```

条件编译伪指令

IF/ELSE/ENDIF

条件编译伪指令将使编译器根据前面的定义条件来禁止/允许编程者编写的部分源代码。

格式： | .IF n
 ;如果 n 为真

```

或      |      .ENDIF
          |      .IF   n
          |      .....      ;如果 n 为真
          |      .ELSE
          |      .....      ;如果 n 为假
          |      .ENDIF
例 :     |      .IF   VALUE>40
          |      ADD  R0,R1      ;如果 n 为真
          |      .ELSE
          |      ADD  R3,R4      ;如果 n 为假
          |      .ENDIF

```

```
.IFDEF/ELSE/ENDIF      IFNDEF/ELSE/ENDIF
```

根据标号是否在前面定义来决定是否编译编程者编写的部分源代码。

```

格式 :   |   .IFDEF/.IFNDEF   |
          |   .....      ; 标号如果定义，则编译/标号没有定义，则编译
          |   .ENDIF
或      |   .IFDEF/.IFNDEF   |
          |   .....
          |   .ELSE
          |   .....
          |   .ENDIF

```

符号的定义用.EXTERN 或.DEFVAR 伪指令，不可用.GLOBAL 伪指令

INCLUDE INCLUDE 可以加载.EXTERN 文件，宏文件，EQU 列表文件，REG 文件。

```

RADIX            设置缺省进制
格式 :           .RADIX  X
例 :             .RADIX  H
                  ADD   R0,#10      ;16 进制
                  .RADIX DEC
                  ADD   R0,#10      ;10 进制

```

.END 结束程序体。在 INCLUDE 文件中，不要用 END 伪指令。

.算术操作运算

操作符号	单目/双目	优先级	描 述	操作符号	单目/双目	优先级	描 述
(--	1 最高		<<	双目	4	左 移
)	--	1 最高		>>	双目	4	右 移
+	单目	1	正	&	双目	5	与 ⁽³⁾
-	单目	1	负		双目	5	或 ⁽⁴⁾
>	单目	1	低字节 ⁽¹⁾	^	双目	5	异 或
<	单目	1	高字节 ⁽²⁾	&&	双目	5	逻辑与
~	单目	1	取 反		双目	6	逻辑或
*	双目	2	乘 法	==	双目	6	相 等
/	双目	2	除 法	!=	双目	6	不 等
%	双目	2	取 模	>	双目	6	大 于
+	双目	3	加 法	<	双目	6	小 于
-	双目	3	减 法	>=	双目	6	大于等于
				<=	双目	6	小于等于

(1) 此单目运算所进行的操作为 (num&0FFH)

(2) 此单目运算所进行的操作为 ((num&0FF00H) >>8)

(3) 此双目运算所进行的操作为操作数之间按相对应的位进行与运算

(4) 此双目运算所进行的操作为操作数之间按相对应的位进行或运算

```
例： LD    R0,#1+2*3/4
      LD    R1,#0F0H&INPUT
      LD    R1,#03H|INPUT
      LD    R1,#(>1234H) ;34H
      LD    R1,#(<1234H) ;12H
```

```
mins .EQU 60*24
LD    R1,#1234>>4 ;0123H
```

```
.IF(VAL==VAL2)&&(VAL2==VAL3)
LD    R3,#45H
.ENDIF
```

```
.IF(VAL1==VAL2)
    .IF(VAL2==VAL3)
        LD    R3,#45H
    .ENDIF
.ENDIF
```

五 中断

复位之后,系统禁止中断,故在初始化程序中要用 EI 指令允许中断,虽可以操作 SYM.3 但并不推荐这样操作。建议愚弄 EI 或 DI,允许/禁止中断。

中断标志需软件清 0,硬件不自动清除中断标志位。中断优先级由中断服务程序的编排顺序决定,在中断中还可以响应其他中断。

SAM88RCRI 响应中断后,将会禁止其他中断,在中断处理完毕之后,允许其他中断。

中断源有:

Timer A	溢出/相等/捕捉中断
Timer B	定时中断
P3.3/P3.4/P3.5/P3.6	外部中断
Watch Timer	中断
UART	串口接受/发送中断

六 时钟电路

当选用主晶振的时候,不能通过对 OSCCON 寄存器的操作来终止主晶振,只能通过 'STOP' 指令来停止主晶振。当选用辅晶振的时候,可以通过操作 OSCCON 来停止辅晶振。

当把 P0.0 和 P0.1 选作辅晶振的时候,而又没有外接晶振时,要把该脚接地。当 CPU 工作与辅晶振的时候,可以用 STOP 指令,但要设置好 Basic Timer,选择它的时钟频率为 $f_{xx}/128$ 。晶振的稳定时间为 $62.5ms ((1/32768) \times 128 \times 16)$ 。

七 复位电路

RESET 脚为低电平的时候,系统复位。复位后,系统状态如下:

禁止中断

允许看门狗功能 (Watchdog Timer)

P0~P4 位输入模式 (除 P0.0—P0.2,P3.3-p3.6)

禁止对控制寄存器和相应的数据寄存器进行操作

PC=0100H

晶振稳定,执行程序

当用外部时钟源输入时,不要用 STOP 指令,因为用 STOP 指令将会使 X_{IN} 脚内部接地。

用外部复位退出 STOP 状态时,内部寄存器卷的数据不变。在 STOP 状态,为减少电流消耗,对 42 封装,设置 P0.3—P0.4 为输出模式,设置其值为低;对 32 封装, P4.0—P4.6/P0.3—P0.7 设为输出模式,且其值为低。

外部中断使系统退出 STOP 状态后,系统执行中断服务程序,在中断返回后,执行 STOP 后的指令。

S3C9488 的 TEST 脚必须接地。

八 I/O 口

P3,P4 为 7 位 I/O 口; P0,P1,P2 为 8 位 I/O 口。

对 P0 口的 P0.0—P0.2 必须在 Smart Option 中定义其用途。

P0, P1 口有相应的控制寄存器来决定该口是否带上拉电阻,因此在编程时,要特别注

意是否选择了上拉电阻。

在使用 I/O 口的其他功能时，要设置相应的控制寄存器。

九 Basic Timer

用于系统复位或从 STOP 状态退出时，稳定晶振。进入 STOP 状态时，需要设置该寄存器的时钟频率输入。在退出 STOP 状态后，启动 Basic Timer，其计数器开始计数，当 BTCON.4 被置起的时候，CPU 开始工作。

通过对 BTCON.1 写‘1’可以清除 Basic Timer 的计数值。

十 8 位定时器 A/B

定时器 A 有 3 种工作方式：定时模式，PWM 输出模式和捕捉模式

定时器 A 的中断标志需要软件清除，硬件不自动清 0。

A 工作于定时器模式时，需要向 TADATA 寄存器写入定时数据，TA 计数器会与 TADATA 寄存器的值比较，当计数值与定时值相等时，产生中断，同时计数器清 0。

TA 工作于 PWM 输出模式时，TA 计数器会与 TADATA 寄存器的值比较，当计数值与预制值相等时，产生中断，但不清除计数值，仍然计数，直到计数溢出，产生溢出中断，再从 00H 开始计数。当计数值小于 TADATA 中的值时，TAOUT 输出高电平，当计数值大于等于 TADATA 中的数据时，TAOUT 输出低电平。

TA 工作于捕捉模式时，仍会产生溢出中断；当 TADATA 寄存器有值时，仍然会产生相等中断。在这种模式下，通过读捕捉到的数据，根据 TA 的时钟频率可以知道脉宽宽度。

对 TACON.0 写‘1’，启动 TA。

定时器 B，是一个减计数器，可用作普通定时器，也可用作远程控制信号。

作普通定时器时，根据控制寄存器的设置，可把 TBDATAH, TBDATAL 寄存器的值装入计数器，当产生下溢时，产生中断，同时控制 T—FF 输出。

当 TBCON.5--.4 为‘10’时，TBDATAL 寄存器的值装入计数器，执行减操作，产生下溢时，装入 TBDATAH 的值，执行减操作。

TB 可选择为单次模式，也可设置为循环模式。

十一 串口

S3C9488 的串口为全双工串口，有 3 种工作方式：

移位寄存器输出模式	波特率为 $f_{xx}/(16 \times (16\text{Bit BRDATA} + 1))$
8 位串口输出模式	波特率为 $f_{xx}/(16 \times (16\text{Bit BRDATA} + 1))$
9 位串口输出模式	波特率为 $f_{xx}/(16 \times (16\text{Bit BRDATA} + 1))$

S3C9488 的串口接收、发送的数据寄存器的地址相同，但物理存储空间是分开的。

当把 UDATA 作为目的操作数时，将会启动串口发送数据。在方式 0 下，串口数据接受的条件为 UARTPND.1=0(中断标志位为 0),UARTCON.4=1；在方式 1、2 下，串口数据接受条件为 UARTCON.4=1 (UARTPND 寄存器标志发送/接收中断，UARTCON.0=1 产生发送中断，UARTCON.1=1 产生接收中断)。

在方式 0 下，当 8 位数据溢出完毕之后，产生中断。

在方式 1、2 下，对于发送，当发送停止位时，产生中断；对于接收，当接收到停止位

的中点时，产生中断。中断标志必须软件清除，硬件不自动清 0。

在方式 2 下，UARTCON.5=1 时，允许校验位，校验位有系统根据设置自动生成，作为发送数据的第 9 位。在方式 0、1 下，禁止用校验位。

奇偶校验的选择在 UARTCON 中设置。校验错误位（RPE）用于标志接收数据的正确性。如果禁止校验，则 PEN=0，第 9 位可以作为数据位发送/接收。

波特率为 $fx/(16 \times (16\text{Bit BRDATA} + 1))$ ，其中 BRDATA 为波特率数据寄存器。

串口工作方式

方式 0：在方式 0 下，RXD 为数据接收/发送口；TXD 为时钟传输口，传输数据只能为 8 位，最低位先发送。写数据到 UDATA，则开始发送数据。工作方式由 UATCON.6-7 决定。

方式 1：10 位发送模式，1 位起始位（“0”），1 位停止位（“1”）。在方式 1 下，TXD 发送数据，RXD 接受数据，波特率可变，硬件自动产生起始位和停止位。最低位先发送。写数据到 UDATA，则开始发送数据。当允许接受（RE=1）时，开始接受数据。

方式 2：11 位发送模式，第 9 位为数据位或校验位。当允许校验位时，根据设置系统自动产生校验位；对于接受方，第 9 位做校验位。

多机通信

多机通信只能工作在方式 2 下，这时要禁止校验位，允许多机通信位 MCE=1。当允许多机通讯位时，只有接收到第 9 位为‘1’时，才会产生中断。第 9 位用来区分地址和数据。

MCE 的设置方式 0 下，不起作用；在方式 1 下，可以用来检测是否接收到合法的停止位。当 MCE=1 时，只有接收到正确的停止位才产生中断。

十二 Watch Timer

WT 可以定时产生中断，也可以产生固定频率的蜂鸣输出，同时产生 LCD 的扫描时钟。如果用到 LCD 显示，则不能禁止 Watch Timer。

WT 的时钟输入可以选择主时钟或辅时钟，选择辅时钟，当系统进入 STOP 状态之后，仍可以驱动 LCD 显示。

对液晶显示要正确设置 LCDCON 和 LCDVOL，并根据设计的液晶玻璃，选择正确的偏压比，占空比驱动方式。

十三 A/D 转换

A/D 转换中，模拟信号输入电压应在 $AV_{REF} \sim V_{SS}$ 之间。

S3C9488 有 9 路 AD 输入，10 位转换结果。S3C9488 内部没有模拟信号保持电路，所以在 AD 转换时，电压波动要小。为保证正确转换，在 AD 转换时，要禁止中断。在 AD 转换时，不要用 STOP，IDLE 指令，以免产生漏电流。

模拟信号输入端口，建议加上滤波电容，以减小外界干扰。

建立 AD 转换须 10 个时钟周期，转换 1 位须 4 个时钟周期，所以 AD 转换共须 50 个时钟周期。

AD 转换最大时钟频率为 4M。

十四 Watchdog Timer

WDT 定时器可以用作看门狗，当系统以外进入 STOP/IDLE 或程序走“飞”的情况下，复位系统。其时钟输入可以选择内部 RC 震荡或 Basic Timer 溢出。WDT 计数器为 16 位，

当选择 Basic Timer 的溢出做 WDT 的时钟输入时，计数器第 3 位置 1 时，复位系统；当用内部 RC 时，计数器第 15 位置 1 时，系统复位。

当系统在 STOP/IDLE 模式下时，WDT 时钟为 Basic Timer 的溢出，则 WDT 溢出并不使系统复位，电流消耗很小；当用 RC 震荡市，WDT 溢出可是系统复位，在正常进入 STOP 状态时，要禁止 WDT，否则，WDT 溢出会使系统复位。

如果使用看门狗功能，在程序中，要定时清 0。

十五 低电压检测/低电压复位

如果系统要低电流消耗，则要禁止低电压检测；如果用低电压检测，则设定电压会与 VDD 比较，当 VDD 小于设定电压时，VLDCON.6 被置 1。在初始化低电压检测电路时，要先写入 3FH，再进行其它操作。

S3C9488 内部有低电压复位电路，通过设置 3EH 可以禁止/允许低电压复位。当允许低电压复位时，要设置相应的值，而不应该为缺省值。允许低电压复位时，如果电压没有达到额定值，MCU 不会工作，以免进行误操作。