常见问题解答

μ'nSP 问答集内容如下：

- μ'nSP™单片机的组成原理及功能介绍
- 寻址方式和指令系统
- SPCE061A 片内外设部件
- 中断系统
- 汇编语言程序设计
- 实验中遇到的问题

◆ μ'nSP™单片机的组成原理及功能

Q: 系统工作电压对工作频率有无影响，对于较低的工作电压，系统能否工作于任意频率？

A: SPCE061A 的工作电压为 2.6~3.6V(cpu), V_{DDH} （IO 参考电压）为 V_{DD}~5.5V；系统工作的频率范围是：0.32MHz~49.152MHz，当工作电压超出规定范围，系统停止工作。

Q: 内核结构图中，SB 与 SHIFTER 是什么关系？

A: 内核结构图中，SHIFTER 是移位器，在进行移位操作时需有移位缓存器 SB (Shifter Buffer) 的配合操作。可参见 SPCE061A 教材 3.4.5 对几种移位操作指令（ASR-ALU 等）的叙述。

Q: CPU 的“休眠”状态靠什么条件结束？

A: 系统靠外设的触发来结束 CPU 的“休眠”，使其重新进入工作状态。这里“外设的触发”泛指为唤醒源，譬如键唤醒，实时时钟定时唤醒等等。

◆ 寻址方式和指令系统

Q: Rd、Rs 是否完全为目的、源寄存器或存储器指针？
A: Rd、Rs 是完全为目的、源寄存器或存储器指针。

Q: LOAD 指令的 Rd＝IM16、Rd[16]的指令格式的第一字组中的 Rs 是什么？因指令中并没有 Rs，编译成机器码 Rs 从何而来？其它涉及到 IM16、A16 的指令，也都有此问题。
A: Rs 为源寄存器 ，因为 IM16 指令支持 Rd=Rs alu_op IM16，A16 指令支持 Rd=Rs alu_op [A16]。在单纯的 LOAD 中，Rs 是不存在的；但是在一般指令中（譬如：ADD），Rs 是有意义的。alu_op 指算术逻辑指令符号如：ADD、SUB、AND、OR、TEST 等

Q: 指令 Rd＋＝Rs 是否有 Rd＝Rd＋Rs 的书写格式？（资料只给了 Rd＋＝Rs 一种格式）？类似有 Rd&＝Rs 等。
A: 有！亦即：Rd += Rs 与 Rd = Rd+Rs 是相同的。

Q: 指令 CMP Rs，IM16 和 CMP Rs，[A16]中为什么不用 Rd？Rs 在这里为源还是目的寄存器（与第一个问题对应），指令格式第一字组中 Rd 又是什么？与此类似的还有 TEST Rs，IM16 、TEST Rs，[A16]。
A: 在这种只用到单一寄存器的情形时，不论 Rd 或 Rs 都会是指向同一个寄存器，因此称呼为 Rd 或 Rs 均可。但因为 CMP 与 TEST 指令只影响状态标志位而不改变缓存器之内容，所以我们一般用 Rs 表示。

Q: LSL-ALU 只是逻辑左移，如何满足算术左移？例如 1000，0000，0000，1111B 左移 4 位将变为正数，如何满足算术左移？
A: LSL 和算数左移功用是相同，符号扩展在 16-bit 范围内都会存在，而上列该数左移 4 bits 时已经溢出，超过 16-bit 数字范围，所以已经无法正确呈现。

Q: ASR－ALU 和 LSL-ALU 指令带进位位操作时，是先移位再对 Carry 加或减，还是先对 Carry 加或减后再移位？如 Rd＋＝Rs ASR nn，Carry。
A: ASR－ALU 和 LSL-ALU 指令带进位位操作时，是先移位再对 Carry 加或减，因为从 μ'nSP™的内核结构图中可以看到移位器 SHIFTER 是串接在 ALU 前面。

Q: 循环指令 ROL 中移位寄存器 SB 参与运算，那么 SB 中的值究竟是什么？如何确定？在何处得知 SB 的内容？若 SB 内容不知，循环后结果将无意义。ROR 也有此问题。
A: 移位缓存器 SB 参与算逻运算前的移位操作或乘法运算中的移位操作，它只起中间缓存的作用，其值在运算后是不确定的，因而用户也就不能对其进行任何读操作。若用户要进行循环指令时, 要先对移位缓存器 SB 进行初始化的动作(先透过移位或循环指令), 然后再进行移位操作，或者可说它负责担负中间缓存的作用, 但在乘法或 FIR 指令后, 会对移位缓存器 SB 产生破坏动作, 因此其值在乘法或 FIR 指令运算后是不确定的，因而用户不适宜在此时对其进行任何读操作。

Q: 在循环指令 ROL、ROR 中若进位位参与运算，如 Rd＋＝Rs ROL nn，Carry，Carry 位是如何参与循环的？
A: ROL、ROR 确切地讲是循环移位指令，它们只对源寄存器 Rs 中的内容与 SB 一起进

行循环移位；因此，循环移位操作与进位 C 无关。

Q: GOTO 是在任何一页的页内转移？页是当前默认页？还是只在零页范围内转移？

A: 对 ISA 1.0 而言，GOTO 指令是在当前的页内转移指令。对 ISA 1.1 而言，GOTO，将可在各页转移。

◆ SPCE061A 片内外设部件

Q: 并行 I/O 结构图中，读引脚和读 Buffer 的三角形是否为三态门？图中的大三角形是什么？

A: 读引脚和读 Buffer 的三角形不是三态门，为 input buffer，用来 regulate 外部的 analog 信号为 0 或 1 数字信号后进 I/O 寄存器。图 1.1 中的大三角形则为 output buffer，用以放大输出信号，为三态门。

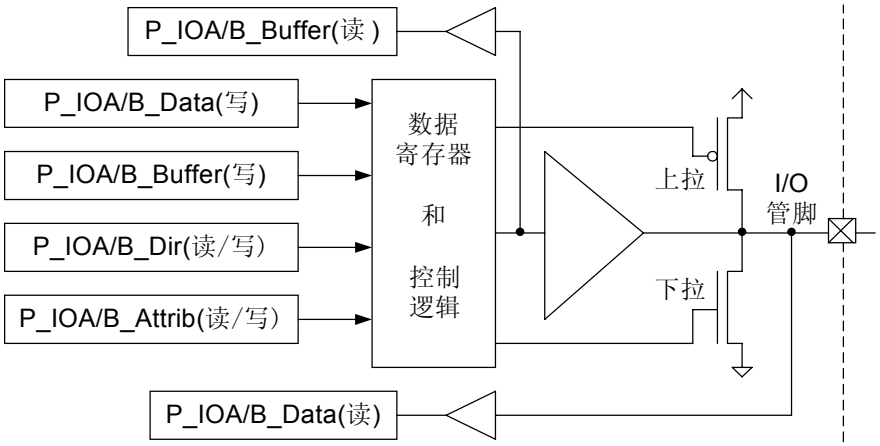

图 1.1 并行 I/O 结构图

Q: 并行 I/O 设置为输出时，写 Data 实际上就是输出的数据，为什么在设置时还写 Data？不写是否可以？因为写 Data 实际上就从管脚上输出了，但例子都写了。

A: 并行 I/O 设置为输出时，写 Data 实际上就是输出的数据，当设置完_Dir、_Attr 口位时，实际便已完成输出口位的设置，而此时再向_Data 口位写数据，实际上便已向端口的数据寄存器写输出数据了。

Q: IOB2 和 IOB4 设置为反馈，是否从 IOB2 输出脉冲，同时该脉冲从内部直接到中断系统，作为外部中断源 EXT1？IOB3 与 IOB5 也类似？

A:

如图 1.2 在 IOB2 和 IOB4 之间或者 IOB3、IOB5 之间增加一个 RC 振荡电路，便可在 EXT1 或 EXT2 端得到振荡频率信号。为使反馈电路正常工作，必须将 IOB2 或 IOB3 设置成反相输出口，且 IOB4 或 IOB5 须设成输入口。

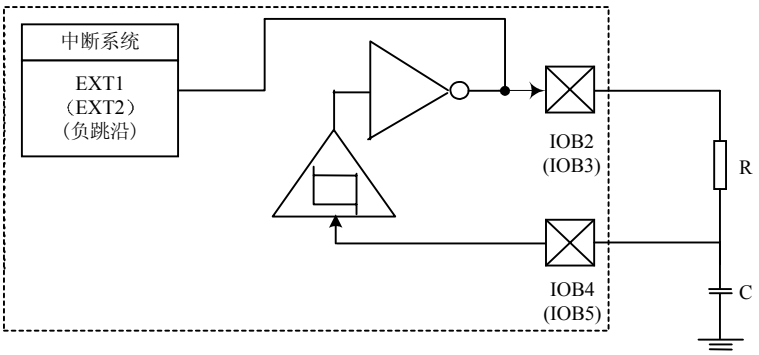

图 1.2 IOB2、IOB4 或 IOB3、IOB5 之间的反馈结构

Q: 定时/计数器 B 时钟源应为 ClkC，否则定时/计数器 A 和定时/计数器 B 对 ClkA 有不同的要求，ClkA 应如何设置？

A: 定时/计数器 B 时钟源虽然为与定时/计数器 A 的 ClkA 相同，但二者的选择设置却是在两个不同的单元中进行的。可参见定时器/计数器的配置介绍和定时器/计数器的控制后面的注释。

Q: 请详细解释定时/计数器 TA_TimeOut/2 的确切含义？是否相当于占空比为 1/2 的信号？

A: 定时/计数器 TA_TimeOut/2 的确切含义是 TimerA 计数溢出频率被 2 分频输出。

Q: P_UART_Command1、P_UART_Command2 以及 P_UART_Data 是否为 8 位？若向 P_UART_Data 写入 16 位数，如何发送？高 8 位低 8 位？

A: P_UART_Command1、P_UART_Command2 以及 P_UART_Data 的有效位均为 8 位（低字节）。若向 P_UART_Data 写入 16 位数，则只发送低字节部分，高字节部分需经移位处理到低位字节部分方可发送。

Q: UART 数据格式只有一种？

A: UART 数据格式目前只规定了一种，见图 1.3 。其中 TX 端在发送数据时应与接收其数据的 RX 端严格地按照协议好的数据格式进行数据传输，见图 4.6，否则会引起数据帧错误。

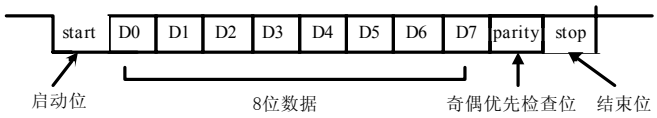

图 1.3　　UART 数据帧的格式

Q: SIO 接口中数据接受/发送端口 SDA 是双向？对应 IOB1 应设置为输入还是输出？

A: SIO 接口中数据接受/发送端口 SDA 可根据需要将其设置为输入或输出端口。

Q: 请给出 SIO 接口较详细的解释说明。请给出 SIO 设备地址的详细解释。SPRS512 的详细资料。其容量多大？地址如何确定？

A: SIO 设备地址的详细解释可参见 061 书的 2..1.3 串行设备输入输出端口 SIO 部分，其中有 SIO 接口较详细的解释说明。SPRS512 容量为 512KB，地址选通可通过对其 $\overline{CS}$ 管脚的控制来实现。连接图见图 1.4

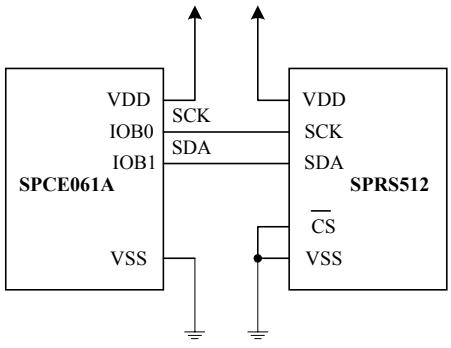

图 1.4　　SPCE061A 与 SPRS512C 的连接图

Q: ADC 的自动方式可以很方便的实现转换功能为什么还要手动方式？

A: 当 ADC 自动转换方式的速度（为多少？）与输入的模拟信号源形成的速度匹配时，自然可以使用自动方式转换；而当二者的速度不匹配（尤其是后者的速度低于前者的速度）时便可考虑使用手动方式，即以软件编程方式进行控制。

Q: ADC 的模拟量输入范围是什么？

A: A/D 模拟量输入范围： 1/2VDD+/-1/4VDD。

Q: 如何理解 ADC 的硬件转换速率限制是 Fosc/32/12 HZ？

A: ADC 的硬件转换速率限制 Fosc/32/12 Hz 是指控制 ADC 进行模/数转换的最高速率，它除了取决于系统时钟 Fosc 外，也取决于控制 ADC 转换的中断服务子程序的周期。如果超出了此最高转换速率，便会出现转换数据不准确之情况。

Q: 音频输出有两种方式:Tone Mode 和 Speech Mode 二者的区别是什么？音频输出的两种方式下 Timer 的作用分别是什么？,

A: 音频输出的两种方式 Tone Mode 和 Speech Mode 的区别在于:其输出的控制机理不同。前者是通过控制 Timer 溢出所产生的不同频率来决定声波振动次数的多少，从而决定发出的声音音调的高低，譬如好听的乐音；后者则是用与声音（不论是音乐还是语音）数据采样率相同的速率将声音数据通过数/模转换（D/A）通道还原成音频电压或电流输出，其中声音数据采样率可决定声音音质的好坏，并决定了声音数据所需占据的存储空间。Timer 对于前者是必须要有的组件之一。其作用是控制 Tone Mode 中发声频率。DAC 或 PWM 方式无非是 D/A 转换的 2 种不同形式，它们自己本身都可用来实现 Speech Mode 的音频输出(只是形式不同);并且它们分别与 Timer 结合使用可实现 Tone Mode 的音频输出，其中 Timer 起音频调频的作用，而 DAC 或 PWM 则起调幅值的作用。两种方式的音频输出波形参见图 1.5

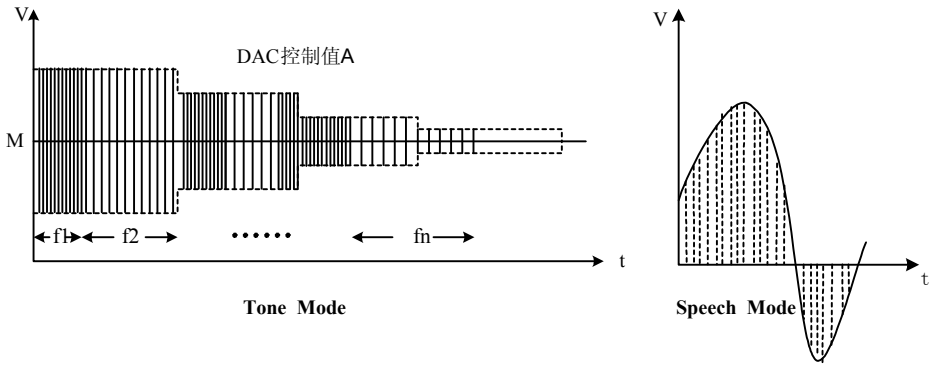

图 1.5 μ'nSP™ 两种音频输出方式的波形

Q: PWM 锁存方式所涉及的硬件是不是只有 PWM 预锁存器和 PWM 计数器？给出图还涉及其它的硬件像加法器、10 位数据锁存器、怎么理解？

A: user 将资料填入预锁存器，等待 PWM timer INT 发生将资料送入锁存器中，加法器依据 PWM 计数器的数值不断比较锁存器中的值以产生对应的周期方波，这里的加法器名称让人觉得奇怪，实际上做的是比较计数器与锁存器中的值，只是在 IC 硬件上是用半加法器来实现的。

◆ 中断系统

Q: 凌阳单片机是否在执行每条指令的最后检测中断请求信号？

A: 是的，所有中断请求信号都是在指令执行完后才检测。

Q: 在凌阳单片机的中断响应周期中，是否自动保护断点地址与标志位、关中断？

A: 是的，当中断发生时会将断点位置 PC 及 SR 存入堆栈后才会开始执行中断服务程序。详细的动作说明如下：凌阳单片机 CPU 在响应中断请求过程中，首先会做

```
PC→[SP]
SR→[SP+1]
[中断向量]→PC
0→CS
```

保护断点的举动。待从中断服务子程序返回时（RETI 指令的执行）便会做出[SP]→SR、[SP－1]→PC 的举动。

Q: 允许中断的中断源在中断请求被 CPU 响应，并按照中断优先级执行完中断服务程序后，是否会自动进入同级中断中下一级中断向量所指的中断服务程序，直到所有的中断服务程序都被执行完，才回到中断发生前程序指令处继续执行源程序？

A: 是的，在凌阳单片机中有两种不同性质的中断：FIQ、IRQ。FIQ 的优先级最高可以中断 IRQ 中断服务程序，而 IRQ 中断则只能中断使用者程序，并按照中断优先级依序执行。允许中断的中断源在中断请求被 CPU 响应，并按照中断优先级执行中断服务程序过程中，通常是由用户写的指令来引导同级中断服务程序的执行顺序，待从中断服务程序返回时，如果有下一级中断请求便会继续响应，直到响应完毕所有的中断请求之后，才会回到响应中断前的断点处继续执行下面的指令。

◆ 汇编语言程序设计

Q: 定义段是否可嵌套？

A: 为了使程序在链接时具有更大的灵活性，用户可以用伪指令.SECTION 来定义某些段。定义的段名最多不可超过 32 个字符，且最多可定义 4096 个段，但不可嵌套使用。

Q: 伪指令 END 在什么情况下是必须使用的？

A: 在欲停止编译的地方写上 END，即停止编译。若未撰写的话，则在遇到档尾时会自动停止编译。

◆ 实验中遇到的问题

Q：在试验中.define TIMER_DATA_FOR_8KHZ 0xFA23
R1=TIMER_DATA_FOR_8KHZ ;
[P_TimerA_Data]=R1;为什么“TIMER_DATA_FOR_8KHZ”能定义以 8KHZ 采样？
A：TIMER_DATA_FOR_8KHZ是指定时的初始值，你也可以用任何符号代替
当定时的时钟源选择为Fosc/2时，如果我们需要8K采样率则需记数1500次即（24M/2）/1500=8K
所以[P_TimerA_Data]寄存器初始值为（0xffff－1500）=0xFA23，而TIMER_DATA_FOR_8KHZ就是指0xFA23

Q: 在A/D转换的试验中：R1=[P_ADC_MUX_CTRL]
TEST R1 ,0x8000 此处比较R1 与0x8000 ,有何意义，起何作用?
A: 判断[P_ADC_MUX_CTRL]寄存器是否为0x8000,是因为该寄存器的B15位是用来判断10位转换是否结束当B15=0时，表示没有转换完毕继续返回进行转换，否则转换完毕。

Q:作键唤醒实验时有下列三语句不解，R1=[P-IOA-LATCH]；R1=0X0007；
[P-SYSTEMCLOCK]=R1，不知第一个语句的作用。
A: 键唤醒实验中的 r1=[P_IOA_RL] ,该指令语句只是通过读寄存器[P_IOA_RL]来激活A口的唤醒功能，并锁存IOA0-IO7的键状态 ，所以R1的值是没有任何意义的。

Q: 做AD转换实验时，我设置了LINE6-IN通道，利用TimerA触发A/D转换，同时利用TimerA开中断，读[P-ADC-MUX-Ctrl]时，发现其值为0X4000.
A:[P-ADC-MUX-Ctrl]的值为0X4000是指MIC的A/D转换，之所以会出现这种情况是因为：061的A/D转换有直接方式也有定时器A或B方式，但是电压模拟量输入只可以用直接方式，即通过读取P_ADC_MUX_Data单元触发A/D转换，而MIC输入则可以用直接和定时两种方式。

Q：对P_IOA_DATA和P_IOA_BUFFER读写的区别？
A： 向 P_IOA_DATA单元写入数据和向P_IOA_BUFFER单元写入数据的作用是相同的，都是写入数据寄存器,区别在于读,也就是说读出 P_IOA_DATA的数据来自I/O管脚,而P_IOA_BUFFER则来自于数据寄存器。例如：当A口设置为输入时，可以将任意端口接VCC或GND，运行程序从寄存器窗口会看到P_IOA_DATA的值改变，而P_IOA_BUFFER的值为初始化值。

Q: 在SACM-DVR实验中，你们给的例子中的SRAM是采用16位地址线，在此实验中调用的InitWriteSRAM(),WriteSRAM(),InitReadSRAM()等函数都是采用16位地址线编写的程序吗？为什么我在单步运行的时候不能看到上面的函数的原程序，如果我想改为17位地址线，我该怎么更改程序；
A: 在 SACM-DVR 实验中所给的对 SRAM 操作的几个函数是采用 16 位地址线编写的,对于InitWriteSRAM(),WriteSRAM(),InitReadSRAM()等几个函数的内容及如何采用 17 位地址线编程，则可参考第八章的 8.2.5 中语音录放部分内容。

Q: 在 FIR 滤波器算法例程中，要实现 N 阶 FIR 低通滤波器，采样频为 10KHz。我们理解这个采样率是通过定时/计数器溢出中断启动 ADC 来确保的。那幺在中断服务程序中是否还需要[P_ADC_CTRL]=ADC_START 来启动 ADC 呢？

A: 不用了！当定时/计数器溢出中断触发ADC时，你只需检查RDY是否为 1 代表转换完成，

然后直接读取P_ADC之转换结果即可。也就是说，在ADC自动方式下通过定时/计数器溢出中断服务子程序F_IRQ1_Service_10kHz中读取P_ADC单元就足以启动ADC了。因此，[P_ADC_CTRL]=ADC_START是多余的举动，已去掉。