〔19〕中华人民共和国专利局

〔51〕Int.Cl.⁴
H04M 3/42
G10L 5/00

CN 86 1 01238 A

〔12〕发明专利申请公开说明书

〔11〕CN 86 1 01238 A

〔43〕公开日 1986年11月12日

〔21〕申请号 86 1 01238
〔22〕申请日 86.1.29
〔30〕优先权
〔32〕85.1.29 〔33〕英国 〔31〕8502228
〔71〕申请人 普列斯海外有限公司
地址 英国英格兰埃赛克斯郡伊尔福德·维卡拉治径
〔72〕发明人 尼尔·凯利特

〔74〕专利代理机构 中国专利代理有限公司
代理人 李先春 杜有文

〔54〕发明名称 语音合成组件

〔57〕摘要

一种用于远程通讯交换机中的语音合成组件，具有一些联结装置，使得组件能联结到交换机上，以便交换机控制该组件驱动组件内部的电路装置，使其产生一个与特定的交换机工作状态相适应的语音响应音信息对状态指示音调加以补充，从而将交换机状态告诉用户。该语音合成组件由计算机程序控制，其中程序是利用结构程序方法设计的，该方法包括产生数据结构图，将这些数据结构图组合成为基本的程序结构，并扩展以包括能从中写出伪代码的所有程序操作。

北京市期刊登记证第1405号

权　利　要　求　书

1。用于远程通信交换机中的一种语音合成组件，其特征在于：该组件具有一些联结装置，使得组件能联结到交换机上，以便交换机控制该组件驱动组件内部的电路装置，使其产生一个与特定的交换机工作状态相适应的语音响应音信息，以补偿用来通知交换机用户该状态的指示音调状态。

2。根据权利要求1所说的语音合成组件，其特征在于：该组件包括一个中央处理单元，一个包含有语言库和多个语言合成器的存储器，中央处理单元是这样设置的，当在它扫描程序时，它从交换机接收信息，访问含有语言库的存储器，并将访问所得的信息分配给指定的语言合成器，语言合成器根据这些数据进行计算，并产生合成的语音响应音信息。

3。根据权利要求2所说的语音合成组件，其特征在于：各个语言合成器利用多路传输转接到含有语言库的存储器上，并被安排到中央处理单元的存储器中，这样，语言数据由含有语言库的存储器向指定的多路开关转接器的传送，就由中央处理单元控制，中央处理单元按照分配给合成器的存储单元地址，访问指定的合成器。

4。根据权利要求2或3所说的语音合成组件，其特征在于：该语音合成组件包括用于控制由交换机接收到的短语和通道号信号的逻辑装置，分别储存短语号信号和通道号信号的短语存储器和通道存储器，其中组件还包括合成器识别装置，它一方面响应由通道存储器产生的通道号信号，以识别哪个语音合成处理器应将被启动，另一方面响应由短语存储器产生的短语号信号，从而指出短语指针表中指向所需的语言数据的地址。

5。根据权利要求4所说的语音合成组件，其特征在于：该逻辑

装置包括一个输出选择器电路，该电路能允许指定的输出多路开关转接器将语言数据从有关的移位寄存器先传送至一个A－律转换电路，然后再将经A－律转换过的语言数据存放在一个缓冲存储器中，该缓冲存储器位于语言数据将被传送到的交换机系统总线之前。

6．根据前面任何一条权利要求所说的语音合成组件，其特征在于：该组件由一个计算机程序所控制，而该程序是利用一种结构程序设计方法设计的，该方法包括产生数据结构图，将数据结构图组合成基本的程序结构，并把它扩展到包括所有程序操作，并由此写出伪代码。

7．根据权利要求6所说的语言合成组件，其特征在于：该伪代码将被翻译成程序代码，从而使得语音合成组件能执行该程序。

8．一个本质上如上面所述的语音合成组件其特征参阅附图1至16。

9．一个包括前面任何一条权利要求所说的语音合成组件的远程通信交换机。

说　　明　　书

语 音 合 成 组 件

本发明涉及一种用于远程通信交换机中的语音合成组件。

语言重放技术和方法，业已问世多年。但是，采用模拟录音的磁带和磁盘机械驱动装置，却十分昂贵、笨重、耗电和容易损坏。而且，这些装置只能顺序地重放录音。编码语言的数字方法看来要比相应的模拟录音装置便宜，但是，对存储器的要求很高。例如，脉冲编码调制（PCM）技术通常在频率为8千赫（KHZ），相应的带宽为4千赫的条件下对语言数据取样。并将每一取样信号量化成8位数字信号，这就形成了每秒64K位的数据速率。这样高的数据速率，使得对大量词汇进行数字编码无法实现。数字编码方法有时被称作波形合成方法，但是，这并不是真正的合成方法，因为它们是由被存储的代码语言数据简单地重新组成原来的波形来。真正的合成技术，利用参量数据产生语言波形，该语言波形并不类同于原来的语言波形，只不过人的耳朵对这两种波形的感觉几乎相同。

语言合成基本上有两种不同类型，即时域合成和频域合成。时域合成是利用为时间函数的语言波形的压缩表达式作为语言存储数据。合成装置只是简单地取出这些存储波形以产生语言输出信号。这种方法的目的不单是为了减少存储原来语言波形所需的位数目，并进而企图逼近这种波形，而是试图产生一种由少量二进制位表示的合成语言波形，这种波形听起来像原来波形的声音，但是，看起来不一定要类似于原来的波形。实施这种方法的硬件非常简单，困难之处在于对语言波形进行分析，使其能以高度压缩的形式进行存储。

频域合成是基于对人的声道的模拟。语言波形以参量数据方式加以储存，这些参量数据表示出声音的倍频振动和声道的响应情况。复

杂的积成电路将这些频域的参量转换成接近于原来语言波形频谱的时域音频信号。

目前存在着各种基于时域合成和频域合成的语言合成方法，它们被用于语言合成积成电路中。这些方法大体上可以称作构成方法，或分析／合成方法。各构成合成法是依据一组规则，从一组规定的语言声段产生出语言输出。利用这样的一组规则，语言要素（例如音素）便能连结成任一单词。利用这些方法，不需要原始的人的语言波形。这种方法也能称之为“规则合成”法。

各分析／合成方法是由实际的人的语言波形导出语言波形的词汇。这些词汇是以单词，短语或者整个句子的形式出现的，因此，导出的合成语言，从音调，语调，易懂性和质量上讲，更加近似于原来的语言。分析／合成方法对实际的口语单词或短语进行分析，从而产生出表示该种语言波形的编码参量。这些参量可以存储起来以备后用，和／或马上合成，以控制语言合成器组件。这些方法比起构成合成方法，需要大得多的存储容量。

在语言合成积成电路中，采用四种基本的语言合成方法。这就是音素合成法，共振峰合成法，波形数字化法和线性预测编码法。

音素合成器是一种能够产生语音的基本单位—— 音素，并将它们连结起来的装置。音素合成器的整个位速率可以很低，大约是每秒100 位，因为在大多数语言中，音素的数目大致为40至100个。音素合成器强调“字母”的简单性，而不注重语言质量。基本音素的发音，在不同的词汇中稍有变化。由如此少量的基本发音所形成的词汇，显得有些生硬。借助于大的存储容量，利用音素变形〔allophon〕对音素进行修正，可以改善语言质量。44个基本的英语音素

可以扩展成１２８个音素变形。这需要大约３Ｋ字节的存储单元，而且，为了存放用于连结音素变形的规则，还进一步需要７Ｋ字节的存储单元。音素变形语言所起来仍然不自然，但是，规则设置一直在改进，以便改善语言质量。通过将音调曲线叠加在音素变形连结串上，可以使语言听起来更为舒服。但是，如果运用得不恰当，这会降低语言的易懂性，而且，从算法实施的角度考虑，这是一个困难的过程。音素变形合成器在词汇量，存储量要求，语言质量，通用性和灵活性诸项指标之间提供了一个折衷的选择。而且，音素变形合成器特别适合于文章—— 语言系统。由键盘输入的文章，可以自动地转换成适当的音素变形，继而再利用音素变形产生语言。

共振峰合成器的目的是从电子学角度模拟形成声道各特征的声音。大多数共振峰合成器对于浊音采用三个共振峰滤波器而对于摩擦音采用一个共振峰滤波器。鼻音共振峰滤波器可有可无，但是，通常不含有动态共振峰功能元件。这里，采用两种驱动功能元件，一种是浊音的脉冲源，一种是清音的噪声源。这两种驱动信号被时间变化的共振峰滤波器滤波成特定的频带或共振峰。共振峰频率的带宽由算法提供，但是严格模拟这些带宽，对语言的易懂性相对说来并不重要。控制器将数据送至含有九个控制参量的模块。这些参量确定了共振峰频率和振幅，它们是：

AH — 送气音振幅

AV — 元音振幅

AV — 摩擦音振幅

AN — 鼻音振幅

FI — 共振峰１频率

F2 — 共振峰2频率

F3 — 共振峰3频率

FF — 摩擦共振峰频率

FV — 振荡器频率

这些参量是通过让输入信号经过时间变化滤波器产生的。这些滤波器的系数随后储存在只读存储器（ROM）中，以供合成时使用。这一方法之所以被称之为规则合成，是因为提取词汇信号的规则，被存储在硬件中。共振峰合成法产生的语言易懂，且较之音素合成法的语言更接近于人的声音。此种方法的位速率大约为每秒400位。更精确地模拟共振峰带宽，会提高语言的质量，不过，也提高了位速率。

波形数字化法理论上是语言合成法的最简单形式。这是最早的一种语言合成途经，它依赖于有效的模—数转换。输入源被转换成数字数据，并存储在存储器中以便以后检索和输出。为了精确地表示语言波形，需要很多的数字编码，也就是说需要大容量的存储器。为了减少大量的存储数据，采用一种压缩算法，可以将需要的存储器容量减至原来的百分之一。利用这种方法，并不打算模拟人的声道。在国家半导体语言处理芯片（National Semicondncfors Speech processor Chip）中，采用三种压缩技术。一种技术是除去多余的音调周期和部分周期以及多余的音素和部分音素。第二种技术是通过在前面波形的幅值上加上一个微小附加量，来得到后来波形的幅值。第三种方法除去了语言波形的方向分量（相角）。由压缩的数据，在合成时，产生一个听起来类似于原来波形的波形。波形数字化法是一种语言合成的分析／合成法。

声音中的相角，对优质的音乐重放是重要的，但是，对于语言来

说，这一信息并不重要。利用语言波形的富里哀分析，可以得到一种波形，此种波形至少在半个周期内具有镜象对称和低振幅。利用镜象对称，可以将半个波形舍去，而在“重放”时，先向前，然后向后播放剩下的半个波形。通过将信号的低振幅部分减至零，储存数据的量可以减少一半而不降低语言的易懂性。这一方法被称之为半周期零化法。

利用重复周期，除去一些音素和音调周期。若干十分相似的周期可以用一个多次产生的周期来代替。对于浊音波形，这种重复可以发生3或4次，对于清音波形，则可达7或8次。清音也被规格化。在压缩算法中，将清音与现成的声音加以比较，由于清音几乎没有什么说话人明确的特征，所以清音可以利用置换而被合成。

自适应δ调制是一种用来由一个波形产生另一个波形的方法。由于语言波形是相当平滑和连续的波形，如果存储的量是相邻数字化信号之间振幅的差，而不是实际的振幅值，那末需要被存储的信息将大大减少。相邻数字信号之间的差叫做δ值，它可以简单地加至前面的数字信号上。利用波形数字化时，每一词汇和短语必须分别合成，因而所需的存储器容量比起音素合成或共振峰合成要大得多。但是，语言质量要好得多，而且能识别说话人。压缩算法允许具有合理低的数据速率，其速率可低达每秒1000位。

语言频谱特性变化的速度取决于人的声道肌肉的活动情况。这一特征联同人的感觉特征一起，意味着，在20至25毫秒期间内，语言的频谱包络可以被看成是不变的。线性预测编码就是利用这些特征，并通过重新产生一个随时间变化的频谱包络来模拟人的声道。

在有限的时间间隔内，一个语言取样信号，可以由多个过去的语

言取样信号的线性组合来表示，这是线性预测编码分析的基础。通过求解实际语言取样信号和适当加权的线性预测值之间半方差的和的最小值，可以确定一组唯一的预测系数（K参量）。用于分析的有限时间间隔被称作分析帧（analysis frame），对于每一分析帧，可以导出一组新的K参量。

利用与在分析时采用的人的语言产生的同样模型或表达式，LPC（线性预测编码）参量共同能量（或增益）和音调参量以及浊音／清音指示器一起，都被用于语言合成。合成语言是以对应于原来分析取样信号的帧的方式产生的。这些帧是由已被预测算法加权3的先前语言取样信号的线性组合重新产生的。

语言合成模型用两种能源表示人的声道，其中一种能源用于激励时间变化的滤波器。在这种滤波器中，包含着语言信号发射，声道转换功能和声门激励的组合频谱效应。

脉冲发生器通过在每一音调周期的开始时刻产生一固定振幅的脉冲，激励起浊音。音调周期由一缓慢变化的音调周期参量所确定。白噪声发生器激励起清音。浊音／清音控制电路选择这两种音源中的一种。增益参量影响所产生的信号的各种频谱分量，它表示语言的“响度”。十个预测系数（K参量），通过一个十极网格滤波器，通常被用来模拟声道。为了使各组参量之间能较为平滑的过渡，控制参量更改的速率通常比帧速率更快。

线性预测编码词汇被存储在帧格式中，每一帧含有必须的参量信息。一个典型的浊音帧格式可能需要多达５０位来表示浊音的帧。在最坏的情况下，最大的位速率是每秒４０帧（相当于２５毫秒的帧周期）乘以每帧４０位，即每秒2000位。不过帧长度可以短些。清音

帧，重复帧，无声帧和停顿码帧皆比浊音帧所需的位数为少。为了给出能识别说话人的语言，典型的线性预测编码系统需要1200位／秒。

利用存储的程序控制（SPC），现代专用自动交换分机（PABX）能够处理16至4000路分机电话。同许多现代交换机一样，这种交换机能为用户提供许多新的附属设备，比如转拨电话，无串音设备，分机简化拨号，以及其它附属设备等。

由于现代交换机中附属设备复杂性的增加，向用户提供有意义的反馈信息的要求也相应增加。现在，已足以可用声音来提供这一信息，目前正在对不够完善的地方加以改进。

一种试图解决问题的方法可以利用录音通知来实现，但是这有很多缺点。由于各种音信息（message）的性质，每一短语必须在其周期的开始时刻，由收听人相联结起来。这就要求，对每一音信息，必须将磁带回绕或快速前绕至所需要的短语的开始位置。利用这种系统的最大音信息存取时间等于整个磁带的回绕时间。这一方法的另一个缺点就是每次只能输出一个短语。

另外一种方法是，每一短语可以被包含在一个隔开的，连续运转的磁带环上。用这种方法可以将每一短语的最大存取时间减至该短语的长度。虽然这种方法一次可以输出多于一条短语，但是，每一短语一次只能被一个通道访问，从而使得短语连结在实际上不可能实现。还有，每一条短语需一个磁带驱动。这样的系统，对于不需要短语连结的大型通用交换机来说是可行的，但是，对于PABX来说，无论是体积还是成本都是不允许的。

语言数字可以以数字格式存储在存储器中。能在电话线上给出适合质量的语言所需的最大频率为4千赫。这就意味着一个8千赫的取

样频率。若采用每一取样信号为8位，则给出的位速率是每秒64000位（b·P·S）。 PABX 语言库最初所期望的容量大约是110秒的语言时间，这就需要860K字节或430K字的存储空间。当前 PABX 的存储容量是1兆字，这就是说，在不考虑任何未来的扩展条件下，差不多一半有效存储单元将被用来存储语言数据。

利用语言合成方法，可以克服上述缺点。与存储语言相比，语言合成方法大大减低了对存储容量的要求，而且各短语都可从头开始，不需要把任何实际装置回绕。

语言合成方法在四个主要方面可以用来使语音响应能够用在 PABX 上。这四个方面是：

故障音调补充，

附属设备启动指示，

通路指示，以及

辅助信息的给定

目前，在呼叫建立或附属设备启动期间，用两种音调来指示一百种以上不同的故障。这些故障的原因各不相同，有资源无效，设备故障，直至用户操作错误等。

故障音调必须在故障被检出之后马上提供出来，不能有明显的延迟，同时，任一补充音信息也应同样迅速给出。

在补充音信息给出之后，接着给出通常所加的那种音调序列。如果由于某种原因不能提供语言响应，那么，上述音调序列立即开始。

附属设备的启动，用三种方式中的一种进行指示。当设备已被启动，通常利用一个开关音调加以证实。而当设备不用时，一般就允许该设备继续运行，在这些情况下，给出的音调为一正常的或特别的

（间断）拨号音调。在大多数其他情况下，重新接通呼叫是给出的唯一证实。

在第一种情况下，可以提供一个词语信息，证实附属设备在成功地运行，同样，为了一致起见，当设备不用时，也希望提供一类似的指示，只要由此而引入的时间延迟不是长得不可接受即可。在重新接通发生时，任何延迟都被认为是不允许的。

补充采用特殊拨号音调指出哪台附加设备已被启动，也是有用的。例如，当无串音设备〔DND〕和呼叫转换设备都被启动之处，则应当只指示出后者。

因此，本发明的目的就是提供一种用于远程通信系统中的语音合成组件，该合成组件避免了先有技术的合成器中所遇到的那些问题。

根据本发明，得到一种用于远程通信交换机中的语音合成组件，其中该组件具有一些联结装置以使该组件联结至交换机中，这样，使交换机控制组件，以便启动组件中的线路装置，产生一个适合于特定交换机状态的语音响应信息，从而将交换机状态情况告诉交换机用户。

根据本发明的一种方式，得到一种语音合成组件，其中，线路装置的启动，使得将要产生的语音响应信息，对状态指示音调加以补充，以将交换机状态告诉用户。

根据本发明的另一种方式，得到一种由计算机程序控制的语言合成组件，其中，程序是利用结构程序方法设计的，该方法包括产生数据结构图，将这些数据结构图组合成为基本的程序结构，并扩展以包括能从中写出伪代码的所有程序操作。

现在将在下面参照附图说明本发明的实施例。其中：

图1表示本发明的语音合成组件的方块图，

图2表示图1所示组件的CPU（中央处理单元），存储器和地址译码逻辑电路的方块图，

图3表示语音合成组件的存储器分配情况，

图4表示用于语音合成组件中的语音合成处理机的方块图，

图5表示图4的处理机语言输出流的时序图形，

图6表示图1所示的定时，控制和接口逻辑电路的方块图，

图7示出表示语音合成组件的指令结构表，

图8表示图6所示的方块图中接收，发送指令周期的时序图，

图9[illegible]（Jackson）结构程序设计的表示，

图10 a表示输入数据结构，

图10 b表示对外说话指令的输出数据结构，

图10 c表示语言数据输出数据结构，

图11表示语音合成组件基本程序结构的方块图，

图12 a和12 b表示复位程序的程序结构方块图，此时图12 b位于图12 a的右边，在接头A处彼此相联，

图13表示语音合成组件主程序结构的方块图，

图14表示P·VCOMM程序结构的方块图，

图15表示P·VSTAT程序结构的方块图，

图16表示P·SPDAT程序结构的方块图。

以下的讨论为利用语音合成方法的PABX中实现语音响应系统。确定了对硬件和硬件／软件结口的要求。线路板被称作语音合成组件（VSM）。

VSM由8个语言合成通道组成，它以类似于音调发生器插件的方式位于内架总线（Intershelf bus — ISB）上。这使得一

个VSM 能被所有扩展插件访向。

在VSM 中包含有足够多的存储单元，至少可以存放200 秒时间长度的语言数据。该存储器能够利用数据总线的高八位字节进行寻址。VSM 至少能够缓冲寄存两个短语地址。每一语言通道能够访问任一短语，而每一短语又能被一个以上通道访问。VSM 允许进行短语连结。

每一短语是从短语周期的开始被访问的，且任何音信息能够在任意时刻被中止。语言数据的定时和控制，是在VSM 中完成的，但是接口定时则由PABX系统控制。

指令由地址总线接收，这些指令详细说明了由从PABX数据总线接收和向PABX数据总线传送数据，由通道向PABX数据总线传送数据，上／下插件（up and down Card）和中止通道输出等。接收（从数据总线）指令指出，短语地址已出现在数字总线上，发送（向数据总线）指令，是请求将已装入的状态寄存器的信息输出送到总线上。状态信息提供了指示，指出哪条通道正在“对话”和哪个缓冲器有效。从数据总线接收和向数据总线发送数据，通道发送以及中止通道等指令，也都含有所有必需的通道信息。

待使用的短语需要慎重考虑，并需要一特制的语言数据库。

参看图1的方块图，图中示出了语音合成组件，下面对此加以讨论。VSM 含有8个语言合成通道SYN 和一个包含在高达三个128K EPROMS （可擦可编程序只读存储器）中的语言数据库SL。这三个EPROMS 中的一个还含有VSM 的系统固件SF 。

CPU 为一16位微处理机，采用它是因为需要将8个语言合成器经多路传输至一个语言数据库上。为了避免互相争用总线的问题，

由微处理机控制语言数据从数据库SL 向每一合成器SYN 的传送操作。合成器安排在存储器当中，因此能按存储器地址进行寻址。

地址译码逻辑电路ADL 对由CPU来的地址进行译码，并允许对语言合成器SYN进行操作、向状态寄存器SR 输入以及从通道和短语输入逻辑电路CPIL输出信息。定时、控制和接口逻辑电路 TCI 控制由地址总线AB和数据总线DB来的信息，向通道和短语输入逻辑电路CPIL输入和由语言输出逻辑电路SOL 及状态寄存器SR 来的信息向数据总线DB输出。

当VSM 上电或从PABX接收复位信号RST 时，定时、控制和接口逻辑电路TCI就使CPU 及通道和短语逻辑电路CPIC 复位。然后CPU 输入一个复位程序，使得语言合成器SYN 复位，并使状态寄存器SR 清零。定时，控制和接口逻辑电路TCI还从PABX 接收系统时钟SCK 信号。然后，由CPU依次开始对各个语言合成器SYN 进行扫描。对每一合成器SYN，检验其是否有数据请求。在每次合成器访问期间，对通道和短语逻辑电路CPIL，检验其新的短语信息。在每次通道访问之后，状态寄存器SR的值将被更新。

利用PABX软件控制，定时、控制和接口逻辑电路TCI 将通道号和短语号计入通道和短语输入逻辑电路CPIL。显示新通道号和短语号有效的指示信号将在CPU 上被标置出来。这些号数将被CPU在其扫描程序中读出，于是CPU 启动所需要的合成器SYN“说话”。

数字语言输出由串行转换成并行，并加以锁存，以备向数据总线DB 输出。定时、控制和接口逻辑电路TCI 将通道数和输出指令送至语言输出逻辑电路SOL。语言数据在输出至数据总线之前，已按A—律（A—law）加以压缩扩展。

状态寄存器SR 由PABX系统软件将输出指令经定时、控制和接口逻辑电路TCL 传送至状态寄存器SR 将其读出。来自状态寄存器SR 的数据，直接输出至数据总线DB上。

VSM 中使用的中央处理单元（CPU），是一个40个脚，由单极5伏供电的单片16位微处理机，它具有256字节的片载RAM（随机读写存储器）。指令系统是基于“存储器至存储器”的结构体系，其特征是多个寄存器文件驻留在存储器中。外部数据总线宽度为8位，但是指令系统允许按字和字节进行运算。变化位宽度的串行I/O 是利用通信寄存器单元（CRU）实现的。系统中需要一个4至12兆赫的时钟，为此，可采用内部晶控振荡器或外部频率源。产生的CLKOUT（时钟输出）信号是时钟基频的四分之一。对于外部存储周期，外部CRU 周期和外部指令周期，能够产生WAIT（等待）状态。当用在每一外部存储周期时，能产生自动WAIT状态。

被称之为工作空间的存储器区，被CPU 用来对指令操作数进行处理。工作空间能够占有存储器任何部分的十六个连续字区域，这部分存储区不再留作它用。数据和地址可以被包含在各自的工作空间寄存器中，即这些寄存器可以用作累加器或者操作数寄存器，地址寄存器或变址寄存器。在执行一定的指令时，某些工作空间寄存器，具有特定的意义。工作空间在存储区中的地址利用硬件工作空间指针（WP）指示。硬件工作空间指针是一个16位寄存器，它包含工作空间中第一个字的地址。工作空间概念在需要上下文转换的操作过程中特别显得重要。这种转换操作可以简单地通过改变工作空间指针来实现。VSM 利用位于CPU 内部RAM 区的两个工作空间。图2表示出VSM 存储器，CPU 和地址译码逻辑电路之间的关系而图3 是存

储单元的分配情况。参照图2和图3，系统固件SF 和语言库SL包含在三个EPROM（可擦可编程只读存储器）中。VSM 固件分配 在存储器底部2K字节。短语地址表分配在从地址BE00开始的512个字节中。这为语言数据留下45.5K字节的存储空间。8个语音合成处理器（VSP），通道和短语FIFO区（先进先出）以及对话状态寄存器，也由于CPU 存储器结构上的优点，而按照存储器地址加以 排列。

CPU 控制图1中存储器，语音合成处理器，通道和短语FIFO以及状态寄存器SR之间的所有数字传送。它也担负地址计算任务和储存所有动态信息。

当通电或从图1中定时、控制和接口逻辑电路TCI接收－CARD RESET（插件复位）信号时，CPU 就使所有语音合成处理器复位。CPU 通过以下步骤实现这一点，即首先，将数据的8个字节全部写“1”。然后，依次将每一语音合成处理器的字节复位。在将每一语音合成处理器复位以后，CPU 读入语音合成处理器的状态字节，以便检查复位是否已经发生。如果语音合成处理器还未复位，CPU 将设法再次使其复位。

一旦所有的语音合成处理器已经复位，CPU 将检查它们的CRUIN（通信寄存器单元输入）输入，以便确定新通道号和短语号是否可利用。如果没有可利用的数据，CPU 征对数据请求信号 将从0通道开始，依次扫描每一语音合成处理器。在每一语音合成处理器通道被访问以后，图1的VSM 状态寄存器SR 即被更新，CRUIN 输入再次由CPU进行检验。

如果发现新的通道号和短语号，CPU 将依次读通道FIFO和短

语FIFO。通道号数示意出，哪个语音合成处理器将说出指定的短语。短语号为短语指针表地址提供一变址，根据这一变址，依次又指出所需的语言数据的第一字节地址。短语号数与短语指针表之间的关系，可用以下方程表示：

$$(\text{短语号}_{16} \times 2) + \text{BE00}_{16} = \text{短语指针表地址}_{16}$$

短语指针表地址和短语指针表地址加1，向CPU提供了与所需短语有关的语言数据第一字节的地址。短语指针表地址加2和短语指针表地址加3，向CPU提供了下一个紧连的短语号的起始地址，这也就是所需短语的终地址。

当这些地址已被算出，并存储在CPU的工作空间后，CPU就向由通道号所指示的语音合成处理器通道送去一个对外说话指令。然后，十六字节的语言数据就从语言数据库传送至语音合成处理器，于是语音合成处理器开始说话。此后，CPU更新VSM状态寄存器，并从它原来停止的地方开始，再次扫描语音合成处理器通道。

在扫描程序中，如果语言数据由某一个语音合成处理器所请求，CPU将使另一个8个字节的语言数据传送至那个通道。在每一语言数据字节传送至语音合成处理器之前，CPU将该数据字节的地址与下一个依次紧连的短语号的起始地址加以比较。如果这两个地址相等，CPU便知道，需要的短语已经传送完毕，并确保不再有语言数据需要传送。如果8个字节还未传送完毕，CPU将发送全1字节，直至整个8字节全部传送完毕。

CPU继续以这种方式循环，依次扫描每一语音合成处理器通道，在每一次语音合成处理器通道访问之后，检查CRUIN输入，直至复

位发生或者VSM掉电。

由于语音合成处理器，通道FIFO，短语FIFO和VSM 状态寄存器，都安置在存储器中，因此，CPU和VSM之间的数据传送，可以看成分为三种类型。

(1) 读存储器—— 由EPROM，VSP 或 FIFO向CPU传送；

(2) 写存储器—— 由CPU向VSP或状态寄存器传送；

(3) 读CRU 。

在从存储器读出时，CPU 将适当的存储器地址放到CPU地址总线上，并对信号 $\overline{\text{MEMEN}}$ 进行断言。于是地址译码逻辑电路ADL将该地址译码，并置有关的存储器件为允许操作状态。然后CPU 对信号 $\overline{\text{DBIN}}$ 断言，并在信号READY变为真之后读出数据字节。在每次对存储器进行访问期间，将产生一个自动WAIT状态。这就使得地址，信号 $\overline{\text{DBIN}}$ 和信号 $\overline{\text{MEMEN}}$ 能够保持一个额外的CLKOUT周期，而与READY输入状态无关，从而保证在数据总线上能出现有效的数据。而在大多数存储器访问中，插入WAIT周期是不必要的，它只是用来减慢CPU 的速度。但是语音合成处理器至少在6微秒期间内，不可能在数据总线上提供有效的数据。在这种情况下，CPU的READY输入被用来保持CPU 继续等待，直至语音合成处理器能够提供有效 的数据。在CPU第一个CLKOUT周期，语音合成处理器不可能用CPU的READY输入来减慢，于是，在此时刻需要一个自动WAIT状态，由于这一自动WAIT状态，整个存储周期都因此而被延迟。

在写入存储器时，CPU将有关的存储器地址放到地址总线上，且对信号 $\overline{\text{MEMEN}}$ 进行断言。于是置有关的存储器件为允许操作状态，CPU 将待写入存储器的数据字节放到数据总线上，并对信号 $\overline{\text{WE}}$ 断

言。当READY信号被断言时，存储器写周期即告完成。这里，在每次存储器访问时，也产生一个自动WAIT周期，因为语音合成处理器至少要花2 3微秒时间从数据总线上读入数据。

当从外部CRU设备读入时，在此情况下，通道FIFO和短语FIFO将发生数据有效标置信号，CPU 将适当的地址输送到地址总线上。信号$\overline{\text{MEMEN}}$保持高电平。CPU选通信号$\overline{\text{DBIN}}$并按数位通过输入端CRUIN读数据。

$\overline{\text{READY}}$选择器，$\overline{\text{RESEL}}$，多路开关转换器$\overline{\text{WS}}$ mux 和多路开关转换器$\overline{\text{RS}}$ mux ，被用来在语音合成处理器访问周期中驱动适 当的语音合成处理器。地址位A13 — A15和ENABLE 0与ENABLE 1（在存储器写周期）及ENABLE 2（在存储器读周期）一起，被用来选择适当的语音合成处理器通道。信号$\overline{\text{DBIN}}$由多路开关转换器$\overline{\text{RS}}$ mux 进行切换，以便为选定的语音合成处理器通道提供一个读选 通信号，类似地，输出$\overline{\text{WE}}$ 提供一写选通信号。$\overline{\text{READY}}$选择器从VSP通道选择READY信号，并将其反相，再提供$\overline{\text{READY}}$给CPU 。

如果当$\overline{\text{CARD RESET}}$ 发生时，CPU 正处于语音合成处理器访问周期，那么有可能当CARD RESET 变成逻辑1时，REACY信号处于逻辑0。这将不会产生自动等待周期。另外，还必须将由$\overline{\text{READY}}$选择器给出的READY输出，计入锁存器RL，而在复位时，必须利用$\overline{\text{CARD RESET}}$ 信号将锁存器RL 的输出预置到逻辑1。这样安排的好处是，CPU 因此将与语音合成处理器同步。

参看图4，语音合成组件采用8个排列成如图所示的语音合成处理器VSPI — 8。每一处理器为一28根引脚，双供电的TTL兼容积成电路，它利用线性预测编码语言数据产生合成语言。它含有以下的

功能电路：用作与外部CPU 接口的CPU I/O 控制逻辑电路，保存由CPU 来的指令码的指令寄存器，向CPU提供有关语音合成处理器状态的最新信息的状态寄存器，保存当对外说话指令发生后，由CPU传送过来的语言数据的16字节FIFO，恢复LPC 语言数据的 语言合成器，产生模拟语言波形的数/模转换器和定时控制逻辑电路。

语音合成处理器能够使用由LPC 技术在8千赫或10千赫采 样率时编码的语言数据。处理器可以直接由加至OSC （振荡器）输入端的外部时钟驱动。这一外部时钟可以从PABX系统的一个时钟派生得到。这样安排的好处是VSP 输出与PABX 同步。

图1所示的CPU 数据总线DB 为8个双向缓冲器BFF1－8 所公有，这些缓冲器通常组合成一定的方式，使得当被信号EN置于 允许操作状态，能将数据从CPU 传送至所有的语音合成处理器。通过将$\overline{RS}$ 信号加至这些缓冲器中某一个的DIR 脚，数据能够由该处理器传送至CPU 。

根据CPU 的请求，每个处理器提供一个状态字节。这一字节只有3位含有某种意义的数据。这三位是对话状态（ TS ），缓冲器低（ BL ）和缓冲器空（ BE ）。当VSP 正在处理语言数据（亦即正在说话）时，TS 置于逻辑1。当16字节FIFO一半以上变空时，BL 置于逻辑1。当16字节FIFO在执行对外说话指令时，全部数据已经用完，则BE置于逻辑1。

在上电或PABX系统复位以后，如果将9个全1字节写入语音合成处理器，随后给出复位指令，那么语音合成处理器一定复位。如果语音合成处理器在对外说话方式（ Speak External mode ）上电，那么在VSP 开始处理FIFO中的数据之前，16字节的FIFO 必须

被填满一半以上。因此需要将9个全1字节写入语音合成处理器。当语音合成处理器识别出它开始处理停止码的第一字节（全1）时，它就停止处理数据，使FIFO清0，并使自己置于指令方式，从而立即接收一条复位指令。

在复位状态下，对话状态TS被清零，任何语言话动均告停止，16字节FIFO被清零，从而使得缓冲低BL和缓冲空BE都变成有效。语音合成处理器被置位于这样的状态：即允许数据写入指令寄存器，以及允许数据从状态寄存器读出。

为了读出状态字节，CPU将相对某一适当的语音合成处理器产生出信号$\overline{RS}$。这一信号也使得有关的双向数据总线缓冲器改变方向，以允许数据由语音合成处理器向CPU传送。当接收到信号RS时，语音合成处理器将$\overline{READY}$信号置于逻辑1，并将状态数据放到数据总线上，当数据稳定之后，再将$\overline{READY}$线变为低电平。然后，CPU读出状态字节，并将信号RS置于逻辑1。于是语音合成处理器将处于禁止输出状态。但是，这些输出禁止状态的建立需要10.5微秒的时间，这一延迟是由于双向数据缓冲器的原因，当RS信号变成无效时，双向数据缓冲器将把语音合成处理器输出从数据总线隔开。

利用CPU将有关指令送至语音合成处理器，从而使信号WS加至语音合成处理器上，并将指令放到数据总线上。语音合成处理器使得$\overline{READY}$信号变成无效（逻辑1），并将数据读入指令寄存器。在读过程完成之后，语音合成处理器将启动READY信号，解释并执行指令。VSM所使用的两条指令是复位指令和对外说话指令。

当接收到对外说话指令时，语音合成处理器被置于对外说话方式，并将16字节FIFO缓冲器清零。随后被语音合成处理器接收到的所

有数据都放在该缓冲器中，直至语音合成处理器接收并执行一条停止码操作。一旦有一个字节被写入缓冲器，缓冲器空信号BE 就变得无效（低电平）。当CPU 把九个字节的数据都装入缓冲器后，缓冲 器低信号BL 就变成无效，对话状态信号TS 变成有效，于是语音合成处理器开始处理来自缓冲器的语言数据。另外七个字节的数据亦被装入语音合成处理器，以充满缓冲器。

来自缓冲器的8字节语言数据，将产生至少25毫秒时间的合成语言。当FIFO含有少于九个字节数据时，缓冲器低信号BL 变成有效。然后，CPU 至少需要25毫秒的时间通过读语音合成处理器 状态字节来识别缓冲器低信号BL 已变为有效，并开始将剩下8个字节输送至语音合成处理器。这25毫秒的要求也限制了处理器的速度。在最坏的情况下，所有8个通道都需要将16字节的数据传送至通道之中。这一情况发生在当所有语言通道同时开始处理数据的时候。为了保证第一通道在它需要数据的时候，能够得到新的数据供应，CPU 必须能够发送8×16字节的语言数据，并且在25毫秒时间内完成对第一通道的扫描程序。利用现有的固件，这一扫描程序可以在大约3.5毫秒时间内实现。

CPU 将继续以每次8个字节的方式把语言数据装入缓冲器，直至达到正被输出的特定短语语言数据的末尾为止。每一短语在其末尾都有一个停止码，但是，为了保证语音合成处理器的确接收到停止码，CPU 发送出全1字节，直至语音合成处理器确实识别出停止码 并且停止处理语言数据为止。

与语言数据处理并行工作的是合成的语言输出。处理器能够以两种方式进行语言输出，第一种是10位串行数字输出格式，第二种是

经8位数／模转换之后，以模拟波形输出。由于在PABX中所有内部语言传送都是以数字方式进行的，因此VSM 采用数字输出。

PABX将所有语言波形转换成8位并行A—律压缩扩展数字数据。为了能与此种格式兼容，由语音合成处理器给出的数字输出，必须从十位串行数字数据转换成8位A—律压缩扩展数字数据。A—律转换正好发生在数据被写入PABX系统总线之前，下面在有关定时、控制和接口逻辑电路中，将对这种转换加以介绍。

语言输出流的时序脉冲示于图5中。ROMCLK 为一由语音合成处理器主振荡器派生出来的160 千赫时钟，该时钟被用来产生信号SHCLK，而SHCLK 信号用于计数进入16位移位寄存器的数字语言输出。信号TII提供一同步脉冲，并使新的数据以8千赫的数据速率到达移位寄存器输出端。

在信号TII 的下降沿，时钟发生器PAL（可编程序阵列逻辑）上的11位计数器被触发。该计数器的时钟由信号ROMCLK 提供。当计数器允许操作时，将允许ROMCLK 信号经时钟发生器PAL传播，并从SHCLK 输出端出来。当计数到达11时，SHCLK 输出被禁止，因此，向16位移位寄存器LSR 提供一个如图5所示的12位时钟信号。由语音合成处理器给出的串行数据输出，在SHCLK 信号的上升沿，被移入16位移位寄存器。信号SHCLK 的第一个和最后一个上升沿将无效数据移入移位寄存器，因此，只有移位寄存器的2至11位才被采用。

在信号TII的上升沿，16位移位寄存器的内容被装入与移位寄存器同处一个芯片上的16位输出缓冲器。然后，从语音合成处理器给出的数字输出，将以十位并行格式出现，以为定时、控制和接口

逻辑电路所用。

语音合成组件（VSM）需要能够从PABX的CCU（指令和控制单元）接收短语和通道信息，并将状态和合成的语言分别传送至PABX CCU和接收分机中。定时、控制和接口逻辑电路控制这类信息交换，它还允许诸如RESET这样的控制信号从PABX系统通过它送至VSM。

图6示出了定时、控制和接口逻辑电路的方块图。ULA（无约束逻辑阵列）为一特定设计的积成电路，该积成电路执行指令译码，接口定时和产生控制信号。通道号和短语号被计入通道FIFO和短语FIFO中，状态寄存器在ULA的控制下，将状态字节发送至PABX系统数据总线上。输出选择器使8个输出多路开关转接器OM1—8的某一个被允许操作。这些多路开关转接器允许来自各有关16位移位寄存器的输出，被送至A－律ROM地址输入端。A－律ROM能将10位并行语言数据转换成8位A－律压缩扩展语言数据。该压缩扩展语言数据又被传送至PABX系统总线上。双向缓冲器BDB决定数据是被传送至PABX系统总线上还是由PABX系统总线传出，BDB由ULA的$\overline{\text{BTE}}$输出信号置于允许操作状态。

PABX系统总线为分时多路传输。在这些总线上的地址和数据每244毫微秒改变一次。这一周期被称作一个时隙，总共有512个时隙，这就意味着，每隔125微秒对每一时隙进行一次访问。如此给出的数据速率是8千赫兹。PABX系统数据总线被分成两个8位总线，分别称作数据总线高字节和数据总线低字节。语言联系是以下列方式进行的。首先，一条电话通道被指定在除去0至7时隙的512个时隙中一个时隙期间内，用半个数据总线发送语言数据，而用另外半个数据总线接收语言数据。而另一条电话通道被指定在与第一条通道相同的时隙内，用第一条通道发送语言数据的那半个数据总线接收数据，而用另外半个数据总线发送数据。于是两个电话通道之间有了语言联系。VSM和电话通道之间的语言联系是以近乎同样的方式实现的，不同的是VSM不能接收语言数据，所以不能实现部分联系。

在PABX系统中，每一电话插件都标有特定号码，该号码由插件

插入的架和槽所确定。为了将信息传送至系统特定的插件上，或从特定的插件将信息送出，插件的架和槽地址应被送至系统的16位地址总线。地址总线的0－4位为架号，5－9位为槽号。其余10－15位地址，用于传送指令。

架地址由架接口插件译码，一旦架接口插件识别出是其自己的架地址时，它便将地址总线的5－15位全部缓冲寄存在架地址总线上。这些位表示成$\overline{\text{BAB5—15}}$。在该架上的每一插件都将地址总线位$\overline{\text{BAB5—9}}$与其本身的插件地址CAD1—5加以比较。这一比较是在ULA中进行的。当插件识别出其本身地址时，它便将地址总线位$\overline{\text{BAB10—15}}$锁存在ULA中，这些位表示成$\overline{\text{LAB10—14}}$。$\overline{\text{LAB15}}$供ULA内部使用。ULA还对包含在这些地址位中的指令进行解释，并产生控制和定时信号去执行该指令。

由VSM识别的指令汇列如下，指令结构示于图7的表中。

(1) 上／下（UP／DOWn）

(2)从地址和数据总线中分别读出通道号和短语号

(3)将状态字节发送至数据总线上

(4)将语言数据发送至数据总线上

ULA还能对两个控制信号作出反应，这两个控制信号是P3 RESET和$\overline{\text{SYN}}$（系统正常）。这两个信号都能产生一个$\overline{\text{CARD RESET}}$脉冲，该脉冲将使VSM，CPU和通道FIFO及短语FIFO立即复位。

当执行DOWN指令时，ULA输出被禁止。只有RESET或UP指令能使逻辑功能恢复。其它指令不起作用。在VSM上的LED（发光二极管）指示，将显示出VSM所处的状态。

当这一指令被接收并译码时，ULA产生一个$\overline{\text{RECEIVE}}$脉冲和一个$\overline{\text{BTE}}$（总线收发器允许操作）脉冲。$\overline{\text{RECEIVE}}$脉冲被VSM用来将锁存的地址总线$\overline{\text{LAB10—14}}$计入通道FIFO中。锁存地址总线的12—14位，被CPU用作通道号，第11位是空位。通道FIFO是一个16字深，5位宽先进先出存储器。这就允许多达16个通道的地址，被堆存在里面等待CPU读出FIFO。$\overline{\text{BTE}}$脉冲使双向缓冲器BDB被允许操作。这个缓冲器的传送方向由来自ULA的TRANSMIT信号决定。在正常情况下，TRANSMIT信号置于逻辑0，这时允许数据由架数据总线向VSM数据总线传送。由于VSM不能同时从架数据总线发送和接收数数，因此，只有一半数据总线被缓冲寄存到VSM上。在发送周期，TRANSMIT信号置于逻辑1，数据从VSM数据总线传送至架数据总线。

接收指令的时序图示于图8中。信号$\overline{\text{DCLK}}$的下降沿，开始将$\overline{\text{RECEIVE}}$脉冲计入锁存器；而在信号$\overline{\text{BTE}}$的上升沿，使锁存器清零。这就产生如图8所示的短语FIFO时钟脉冲，该脉冲具有122毫微秒的成熟时间（high time），它允许数据在开始装入到短语FIFO之前，有61毫微秒的建立时间。

短语FIFO是一8位宽，32字节深的先进先出存储器。因为通道FIFO只有16字深，且对于每一短语号，也必然存在一个通道号，所以，这一存储器只用了16字节。短语FIFO输出永远处于允许状态，并被缓冲寄存到CPU数据总线上。当缓冲器需要读短语号时，该缓冲器由CPU置于允许操作状态，当短语号被CPU读出后，下一个有效的短语号即允许出现在FIFO输出端，以等待CPU的下次访问。

发送状态指令导致ULA产生一个TRANSMIT脉冲。这就使得双向缓冲器BDB反向，允许在VSM数据总线上的数据传送至架数据总线上。信号BTE被用来置缓冲器于允许操作状态。TRANSMIT指令由信号LABII选通，所得到的信号，将允许状态寄存器向VSM数据总线输出信息。

发送语言数据指令也导致TRANSMIT脉冲产生，从而产生双向缓冲器反向。信号BTE再次用来置缓冲器于允许操作状态。在这种情况下，TRANSMIT脉冲由信号$\overline{\text{LABII}}$选通，所得到的信号，将允许输出选择器OS的8个输出端之一有输出。究竟那个输出被允许，则由信号$\overline{\text{LAB12-14}}$决定。然后被选定的输出使8个输出多路开关转接器OM1—OM8中的一个置于允许操作状态，这就可以将10位并行语言数据从它相应的16位移位寄存器送至A—律ROM的地址输入端。该ROM也可以由信号$\overline{\text{LABII}}$选通的TRANSMIT脉冲置于允许操作状态。当被置于允许操作时，ROM可以用作由10位并行语言数据寻址的查寻表。ROM的数据输出提供了8位并行A—律压缩扩展语言数据，这一数据经过双向缓冲器BDB再传送至架数据总线。

固件是VSM的控制程序，或叫操作系统。固件是一个静态存储在积成电路即ROM，PROM或EPROM（只读存储器，可编程序只读存储器）上的软件程序。VSM固件驻留在EPROM中，起始地址为0000H（16进制），这就是CPU再起动地址。在存储器安排中，为固件保留了2K字节的存储单元。

固件程序是利用杰克逊（Jackson）结构程序技术设计的。这是一种将系统的数据结构作为程序基础来考虑的程序设计方法。这一方法包括产生数据结构图，并将这些结构图组合成一个基本的程序

结构。然后，将其扩展包括所有的程序操作，最后，将能写出逻辑原理或伪码。由于它与程序语言和硬件无关，因址，还需要一个产生实际程序编码的步骤，不过，这只是一个简单的伪码翻译过程。

数据结构图和程序结构图代替了流程图。在数据结构图和程序结构图中，程序的流向是从上至下，从左至右。在杰克逊结构程序设计中，只采用三种组成类型，即序列，迭代，和选择。序列指的是按次序第一步，然后，……直至最后的执行顺序，迭代指的是反覆多次重覆某一过程，选择是指对这一或那一过程的选择。这些组成类型的图形表示，示于图9中。

A表示整个程序，它的执行次序是B，然后C，然后D。R本身为一含有用星号表示的迭代部分E的迭代过程。迭代部分根据迭代条件的不同，可以发生零次或多次。C为含有用圆圈表示的选择部分F各G的选择过程。这两部分中，只有一个部分被执行，哪部分被执行，则要取决于条件。只有一个选择部分的选择过程也是存在的，它可以用一个空的，无条件选择部分来描述。

每一方块并未指出它是属于那种组成类型，组成类型要通过检查下面一级来确定。

图10a，10b，和10c表示出VSM的数据结构。它们又划分成输入和输出数据结构，输出数据结构进一步分成对外说话指令输出和语言数据输出。还有一个来自CPU的输出，对语音合成处理器的RESET指令，但是，由于这一输出指令只有当CPU被复位时才发生一次，因此它将被放在一个独立的复位程序中处理。

图10a表示VSM的输入数据结构。第一个标以VSM INPUT的方块，是一个具有迭代部分INPUT的迭代过程。输入本身又是一个

COMMAND STATUS（指令状态），CHANNEL & PHRASE（通道和短语）以及SPEECH STATUS（语言状态）的序列过程。COMMAND STATUS 表示向CPU的一个输入，该输入指示出新的通道号和短语号是否有效。CHANNEL & PHRASE为一选择过程。用圆圈标注的选择部分，其选择条件亦写在方框中。如果新的通道号和新的短语号存在，就执行最左边的选择部分。如果这一条件被满足，通道号和短语号就顺序地被读入CPU中。最右边的选择部分是空的。最后，SPEECH STATUS 表示向CPU的一个输入，该输入指示出VSP是否需要更多的语言数据。

图10b表示对外说话指令输出结构。如果新的通道数和短语数存在，CPU就将16字节的语言数据写入有关的语言合成处理器。这由结构的左边指明。结构的右边是为了完整起见而包含在这里的一个空选择部分。

图10c 表示语言数据输出结构。如果语音合成处理器需要更多的语言数据，CPU就将8字节语言数据写入语音合成处理器。同样，这里也存在一个空选择部分，它在这里是为了指示出，如果未接收到语言数据请求，那么什么也没有发生。

图10a，10b，和10c所示的数据结构，现在可以被用来设计程序结构。图11 示出了基本的程序结构。

参看图11，P. VSM代表整个程序，它由两个顺序的单元P. RESET和P. VBODY组成。P. RESET是VSM的复位程序，留在后面介绍。P. VBODY是VSM固件的主体，并重复执行，直至硬件或软件复位。P. LOOP表示VSM扫描程序的开始，对每通过P. VBODY一次，这一迭代就发生8次。P. LOOP的迭代部分是

P。VCHLP，这是一个由5个主要程序组成的序列，这5个主要程序是：P。COMMAND，P。VCOMM，P。STATUS，P。VSTAT和P。CHAN。

P。COMMAND是检查通道号和短语号有效性的程序。如果这两者有效，则执行P。VCOMM，且通道号和短语号被读入CPU。P。VCOMM还产生对外说话指令，并将16字节语言数据写入相应的VSP中。

P。STATUS用于检查来自被扫描的VSP需要更多的语言数据的请求信号。如果需要更多的语言数据，就执行P。VSTAT，将8字节语言数据写入VSP。

P。CHAN标志VSM扫描程序的结束。通道号增加，那么重复执行P。LOOP程序，这次访问的是下一个通道。当P。LOOP执行8次以后，通道号复位为0。

现在，可以将基本的程序结构扩展到包括由CPU所执行的操作。复位功能P。RESET可以扩展到图12所示的程序结构中。

VSP通道号首先被置0。这是用来在复位程序中访问相应的语音合成处理器。然后执行P。ROBODY，直至所有8个语音合成处理器通道都被复位为止。P。RBODY是一个具有迭代成分P。RBODYITER的迭代过程，如果通道号数小于8，就执行这一迭代过程。

BYTE COUNT被置0，该数被用来确定P。RLOOP的迭代次数，因而就是写入由通道号选定的语音合成处理器的字节数。如果BYTE COUNT不等于9，就执行P。RLOOP。对于每通过P。RLOOP一次，就将一个全1字节写入语音合成处理器。这9个字节被用来填满语音合成处理器缓冲器，且如果在对外说话方式时上电的

话，该9字节将置语音合成处理器于说话状态。在每一字节被写入语音合成处理器之后，将发生一个299微秒的延迟，这一延迟的时间长短，是由通过P。RCNTI的次数决定的。这一延迟的原因是如果语音合成处理器是在清0条件下上电的，那么每一个FF（全1）字节都被语音合成处理器当作是复位指令，而且在该复位指令被语音合成处理器执行之前，不能送入任何新的指令。这就需要300微秒的延迟。在CPU打算再次写入语音合成处理器之前固件编码中还要执行其它一些指令，299微秒的固件延迟，超过了为此目的所需的时间。在299微秒延迟发生之后，BYTE COUNT加1，P。RLOOP重新迭代。

当所有9个FF字节被送入后，就将复位指令写入语音合成处理器。语音合成处理器能够识别这是一个复位指令，或者是因为它把每一个FF字节看作复位指令，因而将复位指令本身简单地看作又是一个FF字节，或者是因为它把FF字节看作是数据，当在第9个FF字节被写入语音合成或处理器之后，语音合成处理器开始说话时，它把第一个FF字节看作是停止码。在第9个FF字节之后的299微秒的延迟时间内，语音合成处理器有时间停止说话，并为指令输入作好自身的准备。在复位指令之后，还有另一个299微秒的延迟，以允许语音合成处理器执行它的复位程序。

在这一延迟之后，读状态指令被写入语音合成处理器，这使语音合成处理器将其状态字节放到它的输出端。CPU读出状态字节，并检验以确定语音合成处理器是否已被正确地复位。在状态字节被读出以后，产生一个13微秒的延迟，以允许语音合成处理器停止驱动它的输出。

如果语音合成处理器已被正确地复位，那么，通道号增加，P。RBODY的下一次迭代开始。如果语音合成处理器还没有被正确地复位，那么，对该通道就再次重复执行复位程序。

P。VBODY标志VSM固件主体的开始。它的程序结构示于图13中。在P。RESET程序执行之后，立即开始执行P。VBODY程序，执行过程一直持续到硬件或软件复位出现为止。

通道号首先被置0，P。VCHLP程序开始执行。P。VCHLP是一个具有迭代成分P。VCHLPITER的迭代程序，它将重复执行迭代，直至通道号达到8为止。然后，P。VBODY重新迭代，将通道号置0，又重新执行P。VCHLP程序的另一个8个循环。

另外，还要检验，以确定开始一个新的短语的通道号和短语号是否有效。如果有效，则执行P。VCOMM。为了清楚起见，这一段的程序结构示于图14中。

在作出是否执行P。VCOMM，以及P。VCOMM 已按照要求被执行以后，需对由通道号选定的语音合成处理器进行检验，以确定它是否需要短语数据。这一检验是通过读相应的语音合成处理器状态字节来完成的。在读操作之后，出现一个13微秒的延迟，以允许语音合成处理器停止驱动其输出。如果语音合成处理器已请求了短语数据，则执行P。VSTAT。P。VSTAT的程序结构示于图15中。

在P。VSTAT按需要被执行之后，通道号增加，P。VCHLP重新迭代。

P。VCOMM 是一个启动特定的语音合成处理器说话的程序段，它只有在新通道号和短语号有效时才执行。在此情况下，CPU 通过检查其CRU输入来确定这一点，CRU输入简单说来就是来自通道

FIFO和短语FIFO的数据有效标志的共同选通。

为了使得扫描程序能够在其已离开的那点继续下去，当前的通道号首先被保存下来。CPU通过改变工作空间指针以及在短语数据写操作时利用第二个工作空间作为寄存器存储区域，简单地达到这一目的。然后，从通道FIFO读出新的通道号，从短语FIFO读出短语号。

正如先前所述，通过对短语号的运算，产生出短语指针地址。从这个短语指针地址，可以读出待说短语的起始地址码。CPU读入这个地址码，并将它存储在寄存器中。这个地址码以后将递增并被用来从EPROM中提取短语数据。待说短语之后的下一个短语的起始地址，也被读入CPU，这个地址将被视作CPU的停止检验地址。

BYTE COUNT被置于16，SPEAK EXTERNAL指令被写入VSP。然后，将产生一个26微秒的延迟以允许语音合成处理器接收和解释指令，并使其本身作好接收短语数据的准备。

标以P. SPDAT的程序段，是一个用来将若干短语数据字节写入语音合成处理器的程序，被写入的字节数目由BYTE COONT的值确定。P. SPDAT是用在固件中的唯一的子程序。其程序结构示于图16中。

在P. SPDAT被执行之后，通过恢复工作空间指针，将通道号置于它的扫描值，扫描程序继续执行。

P. SPDAT执行将短语数据传送至选定的语音合成处理器的任务。传送的字节数由BYTE COUNT的初始值决定。P. SPDAT1将一直执行，直至BYTE COUNT减至0。

P. SPEAK是一个具有两个选择部分的选择过程。或者是所有

的短语数据都被写入语言合成处理器，或者是还有更多的短语数据有待发送，除此之外不存在其它情形。

假如全部短语数据已被发送，（当短语数据地址变得等于下一条短语的起始地址时，即发生上述情况），FF 字节将逐一被写入语音合成处理器，直至被发送的总的字节数等于BYTE COUNT的初始值为止。假如仍然存在有效的短语数据，那么下一个短语数据字节就被发送至语音合成处理器。然后，短语数据地址加1，这就完成了一次P. SPEAK过程。

每执行一次P. SPDATI，BYTE COUNT 被减1，如果BYTE COUNT不等于0，那末，就开始P. SPDAI的进一步迭代。如果BYTE COUNT等于0，那末正确的字节数就被传送至语音合成处理器。为了在语音合成处理器下次请求短语数据时，能够从正确的地址将数据送去，需要将当前的短语数据地址和紧接已经说出的短语之后的那条短语的起始地址加以保存。这两个地址都存放在随机访问存储器中。

P. VSTAT是一个程序段，当语音合成处理器请求数据时，它能将8字节短语数据送往语音合成处理器。如果语音合成处理器的状态字节指示出，FIFO的填满数少于一半，那末，CPU就将 8 字节数据送往语音合成处理器。

CPU把短语数据地址和下一条短语的起始地址从它们所在的RAM存储器中检索出来。再把它们存放到第二工作空间的寄存器中，然后将工作空间指针指向第二工作空间的首地址。这就使得固件16工作空间寄存器工作时不以任何方式对用于扫描操作中的那些寄存器发生影响。

然后，BYTE COUNT 置于8，执行P. SPDAT。一旦P. SPDAT 执行完之后，工作空间复位至第一工作空间。这可以通过简单地将工作空间指针改变，使其指向第一工作空间的首地址来完成。

从上述程序可以导出伪码，该伪码表示如下。

VSM 伪代码

```
P. VSM 序列
  P. RESET 序列
    置通道号为0
    P. RBODY 迭代 直至通道号＝8
        P. RBODYITER 序列
          置字节计数为0
          P. RLOOP 迭代直至字节计数＝9
            将“FF”字节写入VSP
            置延迟计数为22
            P. RCNTI 迭代直至延迟计数＝0
              延迟计数＝延迟计数－1
            P. RCNTI 结束
            字节计数＝字节计数＋1
          P. RLOOP 结束
          将复位指令写入VSP
          置延迟计数为22
          P. RCNT2 迭代直至延迟计数＝0
                    延迟计数＝延迟计数－1
          P. RCNT2 结束
```

从VSP读状态字节

置延迟计数为1

P. RCNT3迭代直至延迟计数=0

延迟计数=延迟计数－1

P. RCNT3结束

P. STATUS选择VSP正确地复位

通道号=通道号+1

P. STATUS结束

P. RBODYITER结束

P. RBODY结束

P. RESET结束

P. VBODY迭代，当不复位时

P. VBODYITER序列

置通道号为0

P. VCHLP迭代直至通道号=8

P. VCHLPITER序列

读通道和短语FIFO标志

P. VCOMM选择新的通道号和短语号有效

存当前通道号

从通道FIFO读通道号

从短语FIFO读短语号

短语指针地址=短语号×2 + BE00H

读短语起始地址

（短语数据地址）

读下一条短语起始地址

置字节计数为16

将外部说话指令写入VSP

置延迟计数为4

P．VCNT1迭代直至延迟计数＝0

延迟计数＝延迟计数－1

P．VCNT1结束

P．SPDAT1序列

调用子程序P．SPDAT

P．SPDAT1结束

通道号复位

P．VCOMM结束

从VSP读状态字节

置延迟计数为1

P．VCNT2迭代直至延迟计数＝0

延迟计数＝延迟计数－1

P．VCNT2结束

P．VSTAT选择VSP需要更多数据

检索存储的短语数据地址

检索存储的下一条短语起始地址

置字节计数为8

P．SPDAT2序列

调用子程序P．SPDAT

```
            P. SPDAT2 结束
          P. VSTAT 结束
          通道号=通道号+1
        P. VCHLPITER 结束
      P. VCHLP 结束
    P. VBODYITER 结束
  P. VBODY 结束
P. VSM 结束
P. SPDAT 序列
  P. SPDAT1 迭代直至字节计数=0
    P. SPDAT1ITER 序列
      P. SPEAK 选择没有更多的短语数据有效
        将“FF”字节写入 VSP
        置延迟计数为1
        P. SPCT1 迭代直至延迟计数=0
          延迟计数=延迟计数-1
        P. SPCT1 结束
      P. SPEAK 或更多的短语数据有效
        将短语数据字节写入 VSP
        置延迟计数为1
        P. SPCT2 迭代直至延迟计数=0
          延迟计数=延迟计数-1
        P. SPCT2 结束
          短语数据地址=短语数据地址+1
```

P。SPEAK结束

字节计数＝字节计数－1

P。SPDATlITER结束

P。SPDATl结束

保存短语数据地址

保存下一条短语起始地址

P。SPDAT结束

用于VSM的语言数据是利用得克萨斯仪器公司轻便语言实验室（Texas Instraments Partable Speech Lab)(PSL)的方法进行编码的，该PSL包括语言分析系统，语言合成器，EPROM编程器以及CPU电路板。该PSL可以外接受话器，喇叭和VDU（直观显示装置）。任何长达10秒的面对受话器说出的短语，都可以立即被用户分析、利用音调激励的LPC编码成“线性预测”系数、存储和进行合成。然后，用户可以或者通过经受话器输入端重新输入的方法，或者通过修改线性预测系数的方法，改变说出的短语，以提高语言的质量。最后编码的短语，可以利用EPROM编程器装入EPROM中。

总共有12个合成参量以编码形式，先存储在PSL中，然后存储在EPROM中。这些参量是音调，能量以及反射系数K1－K10。每一参量占有3至6位，在合成时，从语音合成处理器查寻ROM中选择一个10位参量。如果所有编码参量对每一个25毫秒帧都送出的话，那末总共需要50个编码位。但是，有三种特殊情况不必对全帧发送。它的优点在于可以大大降低位速率。

第一种特殊情况是由于下面的实际情况产生的，即声道形状的变

化比较缓慢，因此，利用一个重复位来重复先前的反射系数是可能的。当该位如此设置时，那么在合成时，只需要从EPROM读取能量和音调数据，先前的K1－K10数值将被保留下来。

第二，清音语言比起浊音语言所需的反射系数少。当音调参量为0时，只从EPROM中读出系数K1－K4。

最后，当能量参量为0时，也就是说在词间或音节间无声时，便不需要任何其它的数据。

这三种情况的组合，对于男性语音平均可以减低40%的位速率，对于女性语音，减低量稍少一点。

对于在VSM库中短语所需的语言质量，由于电话网络带宽的限制而不如短语可能被产生时那样高。为了保证得到尽可能清楚的录音，利用PSI将库中语言记录在高质量的录音带上。录音是在隔声室，而不是消声室中进行的。所产生的语言质量，适合于所希望的应用场合，特别是在经过编辑之后，更是如此。

这一方法的主要优点之一就是，简单地通过建立另一段录音时间，短语库能够被改变到即接到很容易地利用同一个喇叭的状态。

当短语库已经内用PSL进行录音和修改之后，它必须要存储在EPROM中。利用PSI EPROM编程器，很容易作到这一点。由于PSL编程器能够编程的最大EPROM只有32K位，因此，需要把每一短语放进一个32K的EPROM中，再把它装入一个能处理较大EPROMS的编程器中，并且由这个编程器对用在VSM上的128 K EPROM编程。短语地址表被加在128 K EPROM上，该EPROM将驻留在VSM存储器中的地址8000处。短语地址表起始地址为BE00，它是用手工编程的。

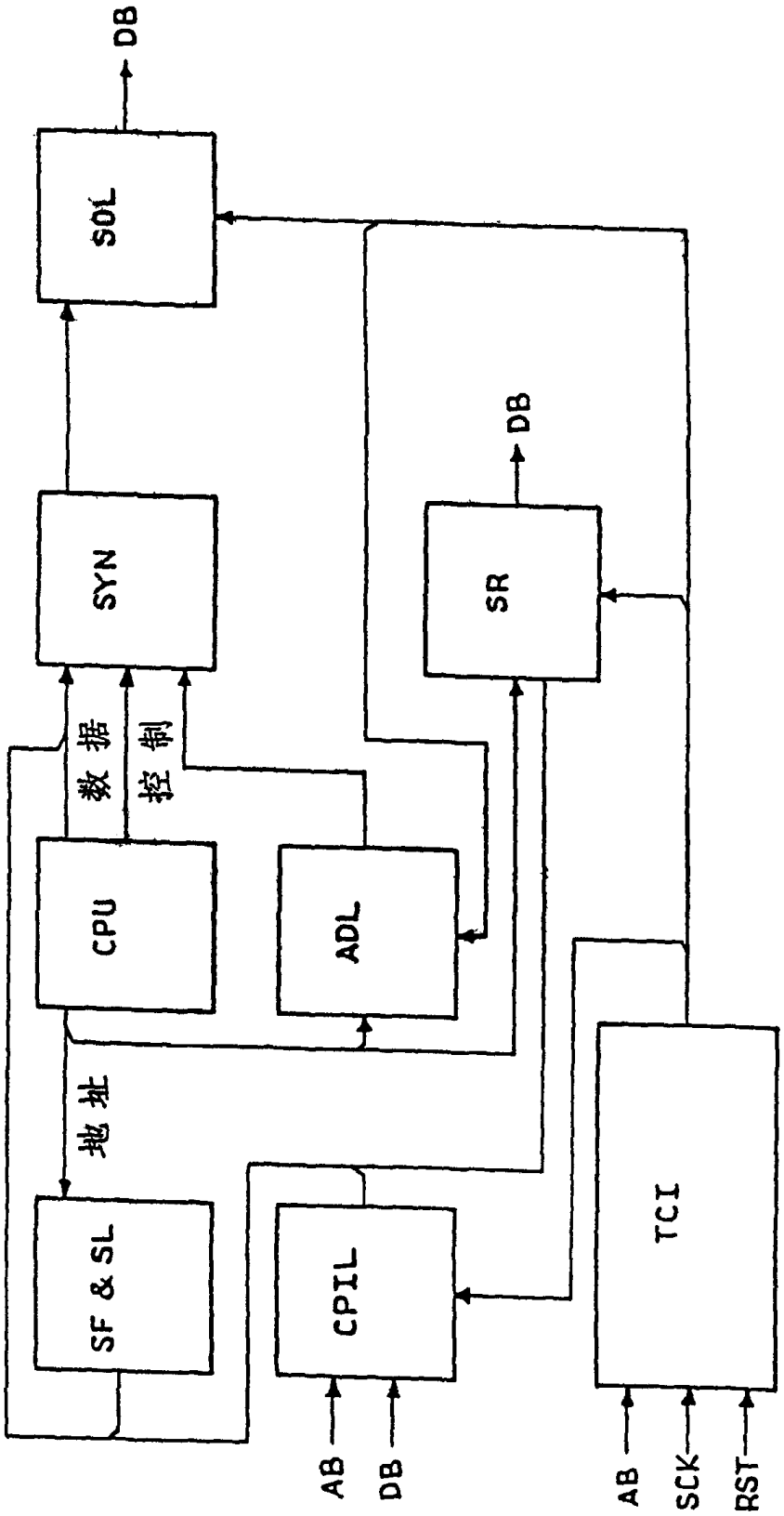


图 1

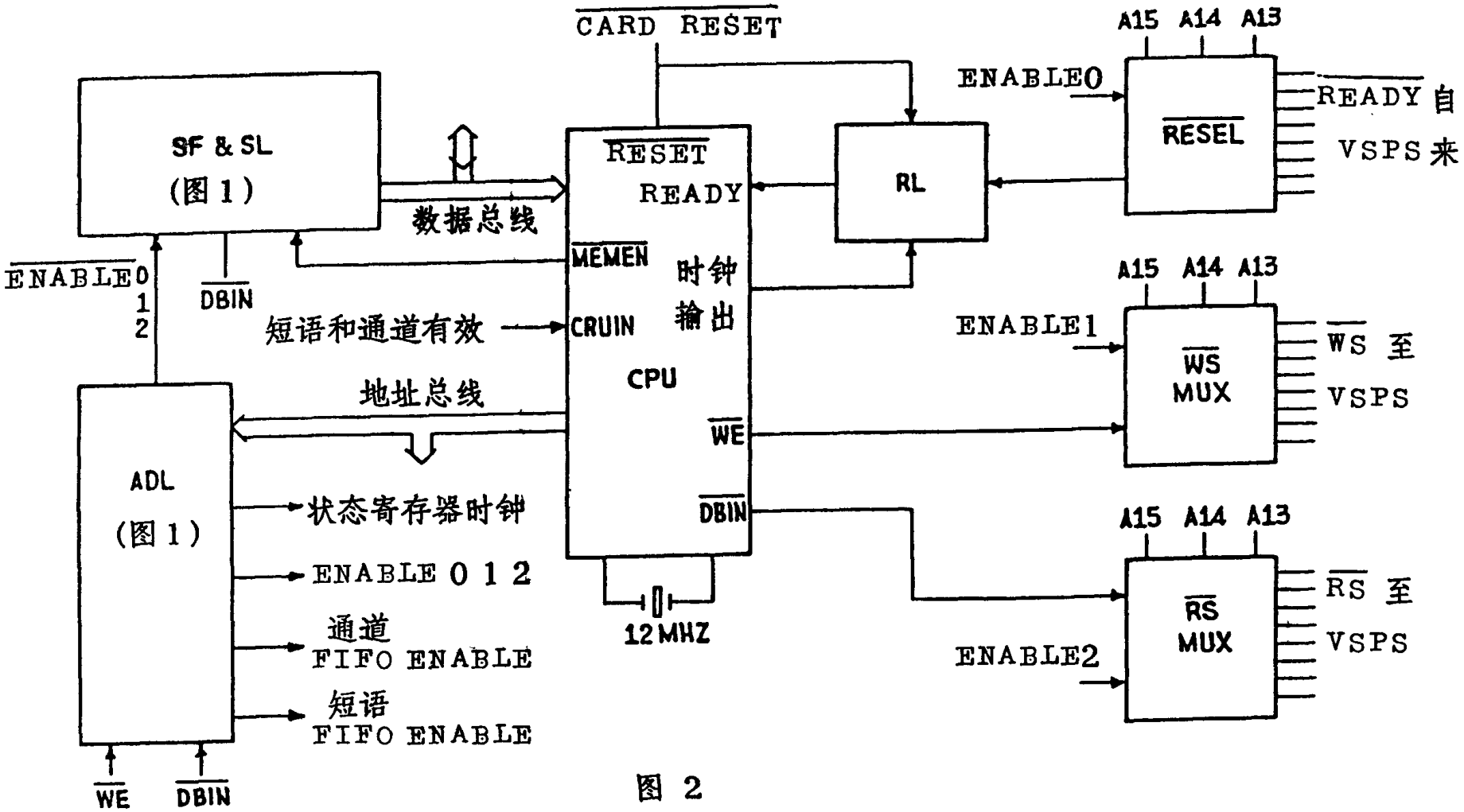

CARD RESET
A15 A14 A13
ENABLE0
READY 自
VSPS 来
SF & SL
(图 1)
数据总线
RESET
READY
RESEL
RL
MEMEN
时钟
输出
ENABLE 0
1
2
DBIN
短语和通道有效
CRUIN
CPU
A15 A14 A13
ENABLE1
WS 至
VSPS
WS
MUX
地址总线
WE
ADL
(图 1)
状态寄存器时钟
DBIN
A15 A14 A13
ENABLE 0 1 2
RS 至
VSPS
RS
MUX
通道
FIFO ENABLE
12 MHZ
ENABLE2
短语
FIFO ENABLE
WE
DBIN

图 2

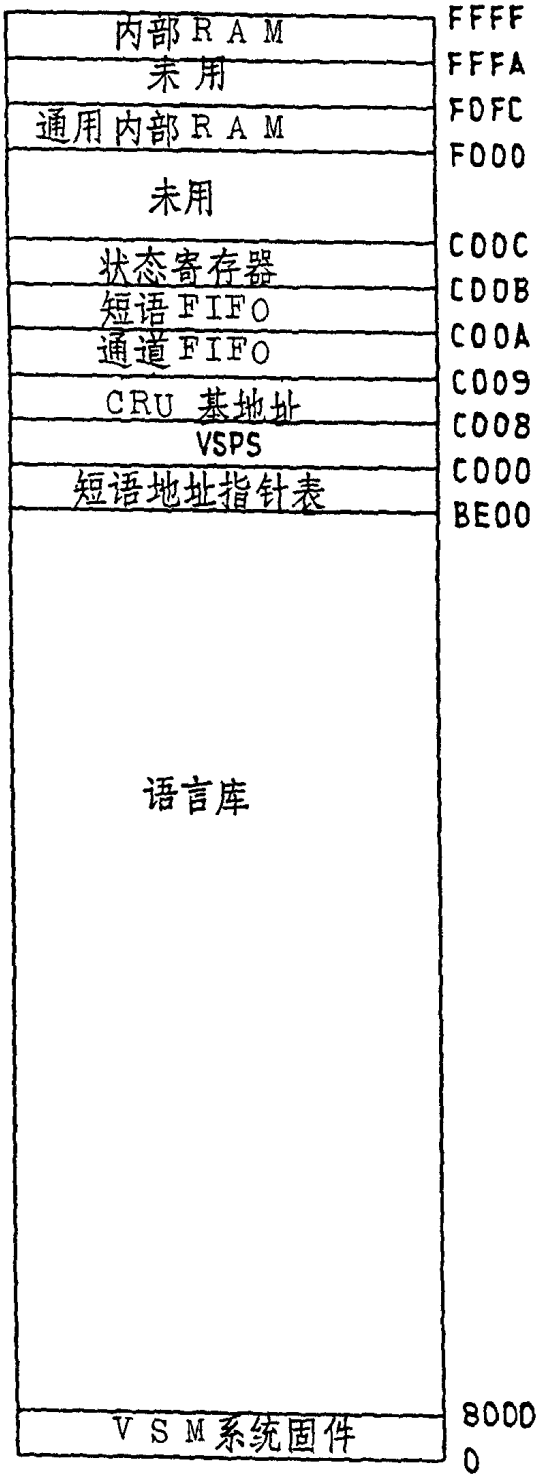

内部RAM
FFFF
未用
FFFA
通用内部RAM
FDFC
未用
FD00
状态寄存器
C00C
短语FIFO
C00B
通道FIFO
C00A
CRU 基地址
C009
VSPS
C008
短语地址指针表
C000
BE00
语言库
VSM系统固件
8000
0

图 3

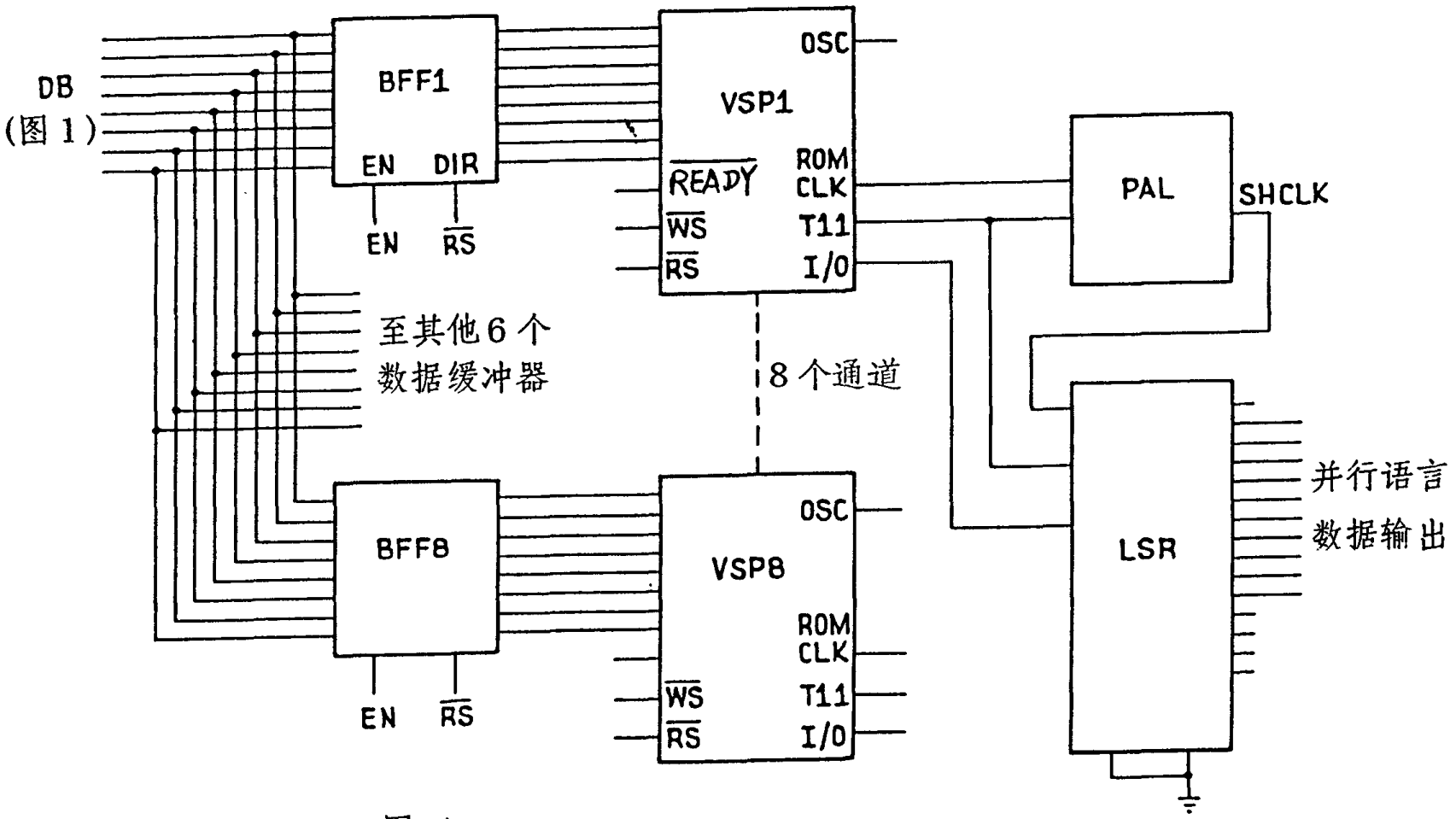

DB
(图 1)
BFF1
EN
DIR
EN
RS
至其他6个
数据缓冲器
BFF8
EN
RS
OSC
VSP1
READY
WS
RS
ROM
CLK
T11
I/O
8个通道
OSC
VSP8
WS
RS
ROM
CLK
T11
I/O
PAL
SHCLK
LSR
并行语言
数据输出

图 4

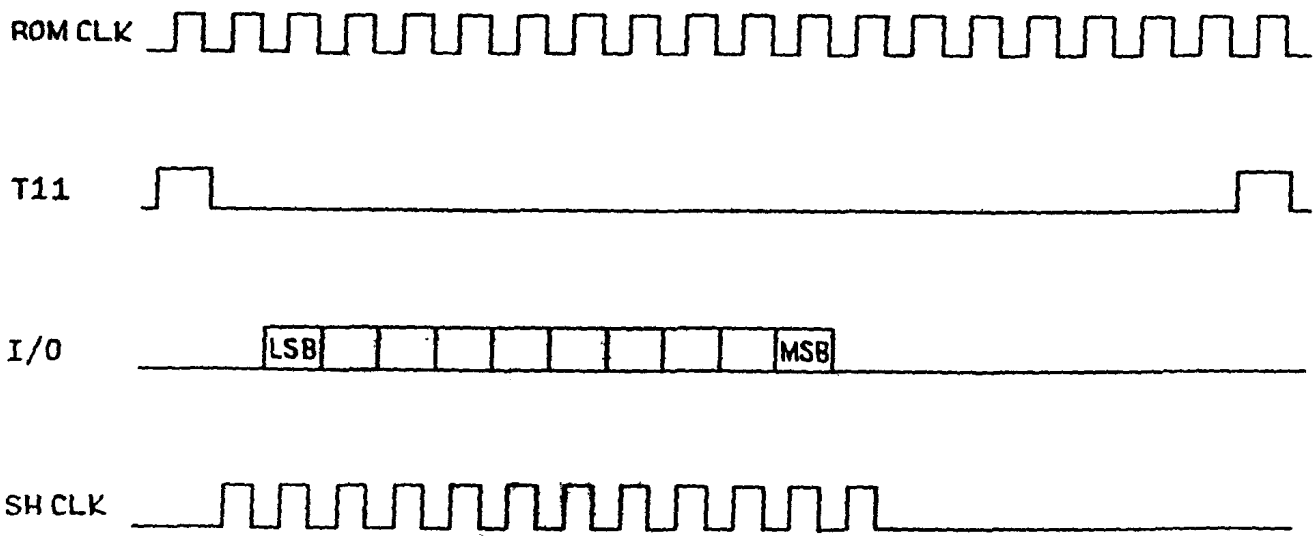


图 5

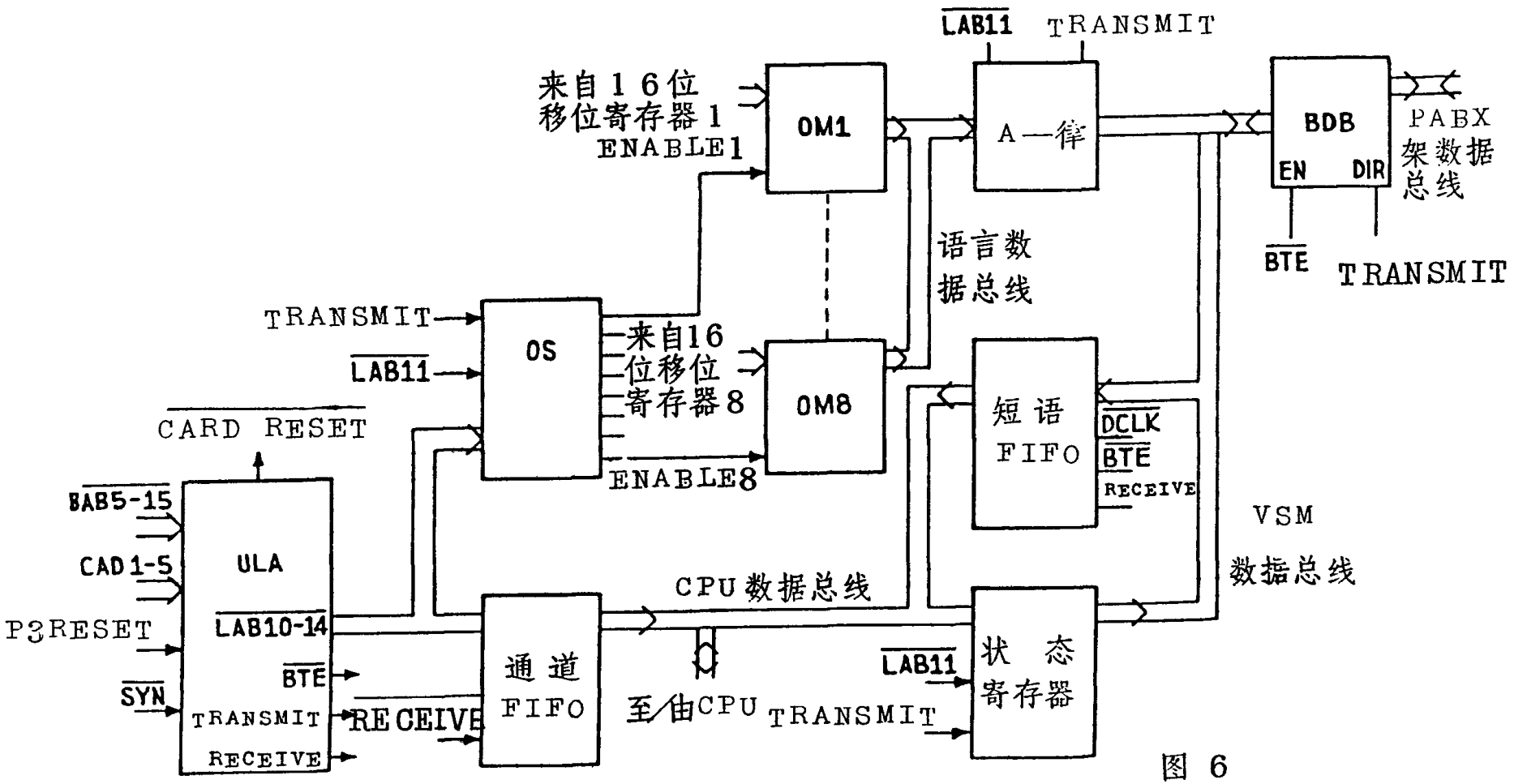

LAB11
TRANSMIT
来自16位
移位寄存器1
ENABLE1
OM1
A—律
BDB
EN
DIR
PABX
架数据
总线
BTE
TRANSMIT
语言数
据总线
TRANSMIT
LAB11
OS
来自16
位移位
寄存器8
OM8
ENABLE8
短语
FIFO
DCLK
BTE
RECEIVE
CARD RESET
VSM
数据总线
BAB5-15
CAD1-5
P3RESET
SYN
ULA
LAB10-14
BTE
TRANSMIT
RECEIVE
CPU数据总线
通道
FIFO
RECEIVE
至/由CPU
TRANSMIT
LAB11
状态
寄存器

图 6

BAB	10	11	12	13	14	15	
	1	1	1	1	1	0	上
	1	1	1	1	1	1	下
	0	X	Y	Y	Y	0	接收通道和短语号
	0	0	Y	Y	Y	1	发送状态字节
	0	1	Y	Y	Y	1	发送语言数据

X—不恰当组合

Y Y Y—通道号

图 7

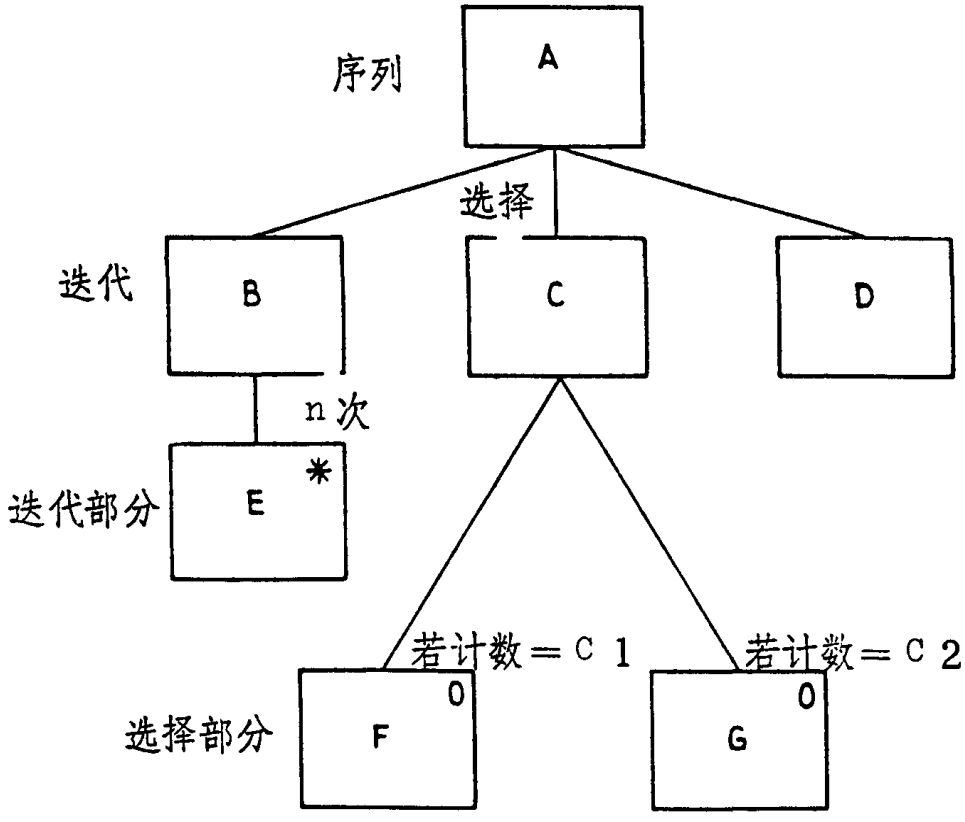


图 9

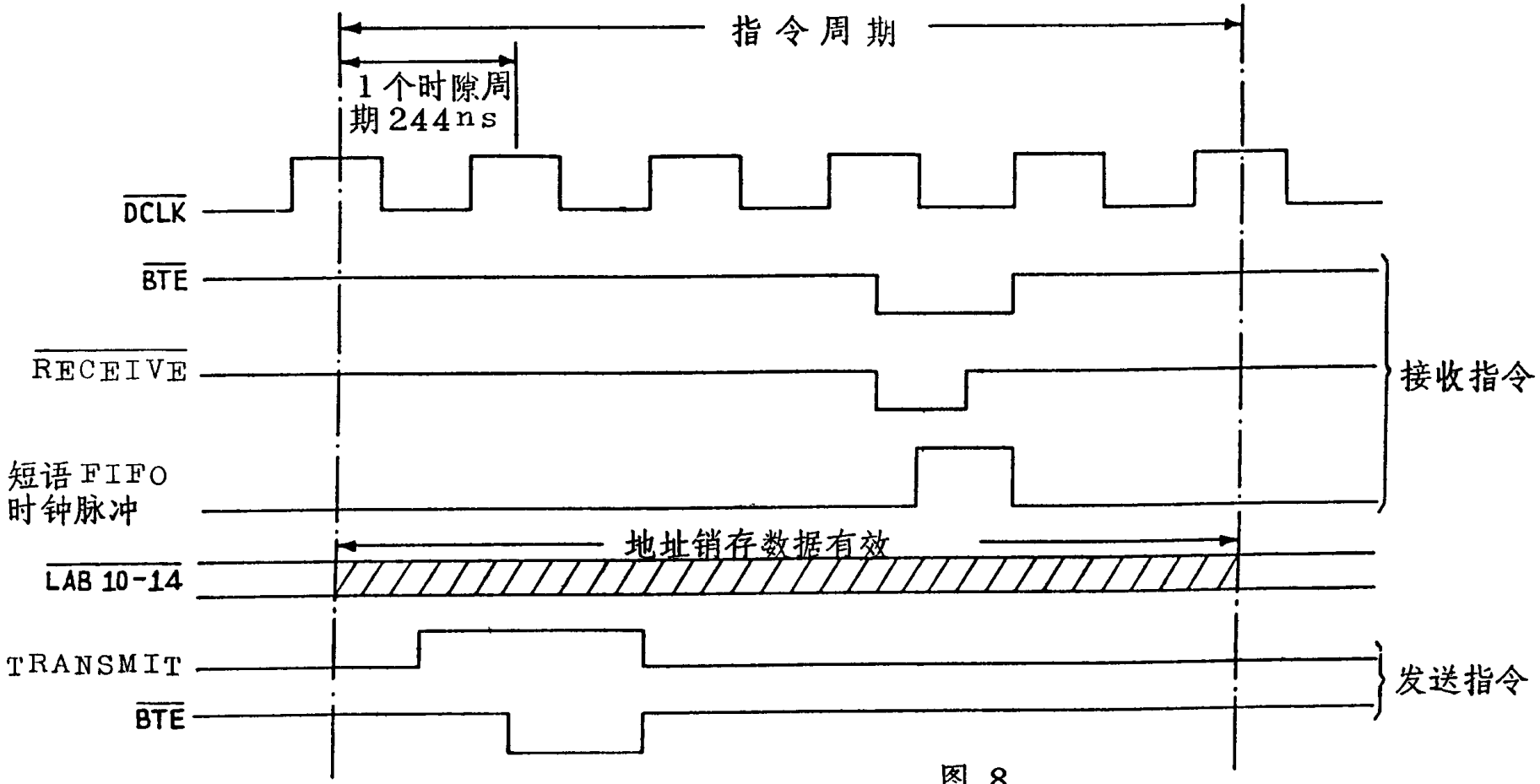

指令周期
1个时隙周期244ns
DCLK
BTE
RECEIVE
接收指令
短语FIFO时钟脉冲
地址销存数据有效
LAB 10-14
TRANSMIT
发送指令
BTE

图 8

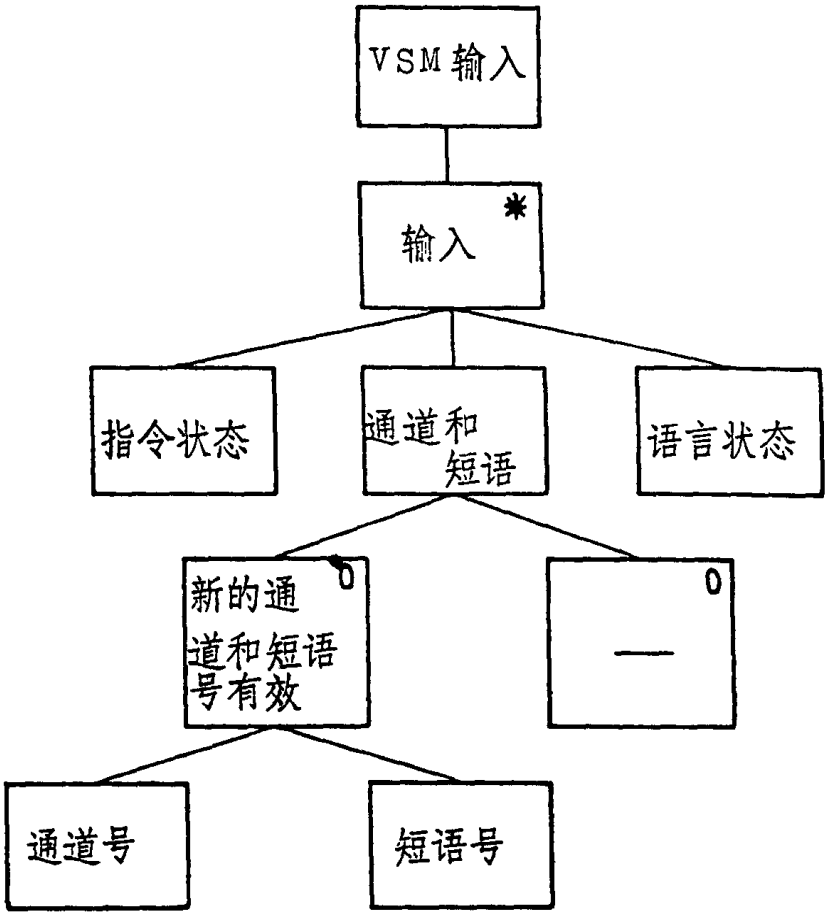


图 10a

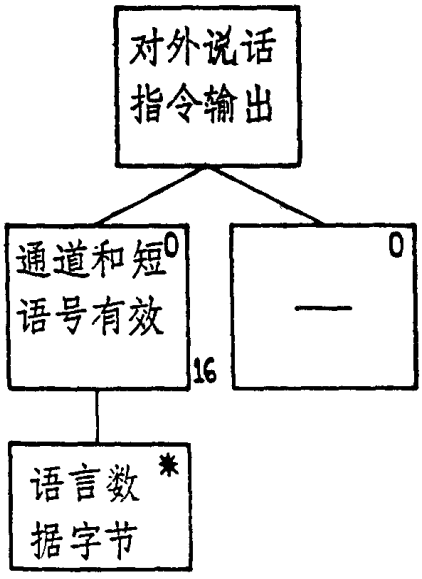


图 10b

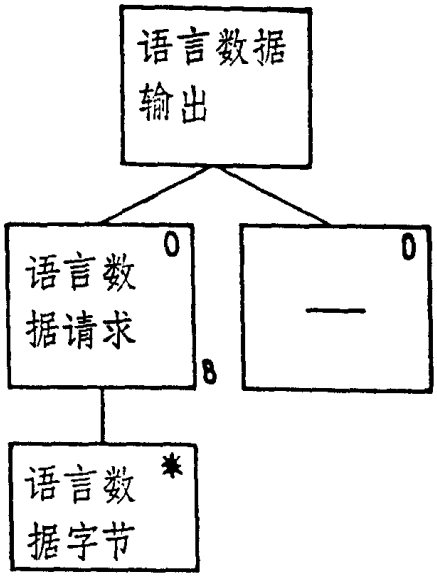


图 10c

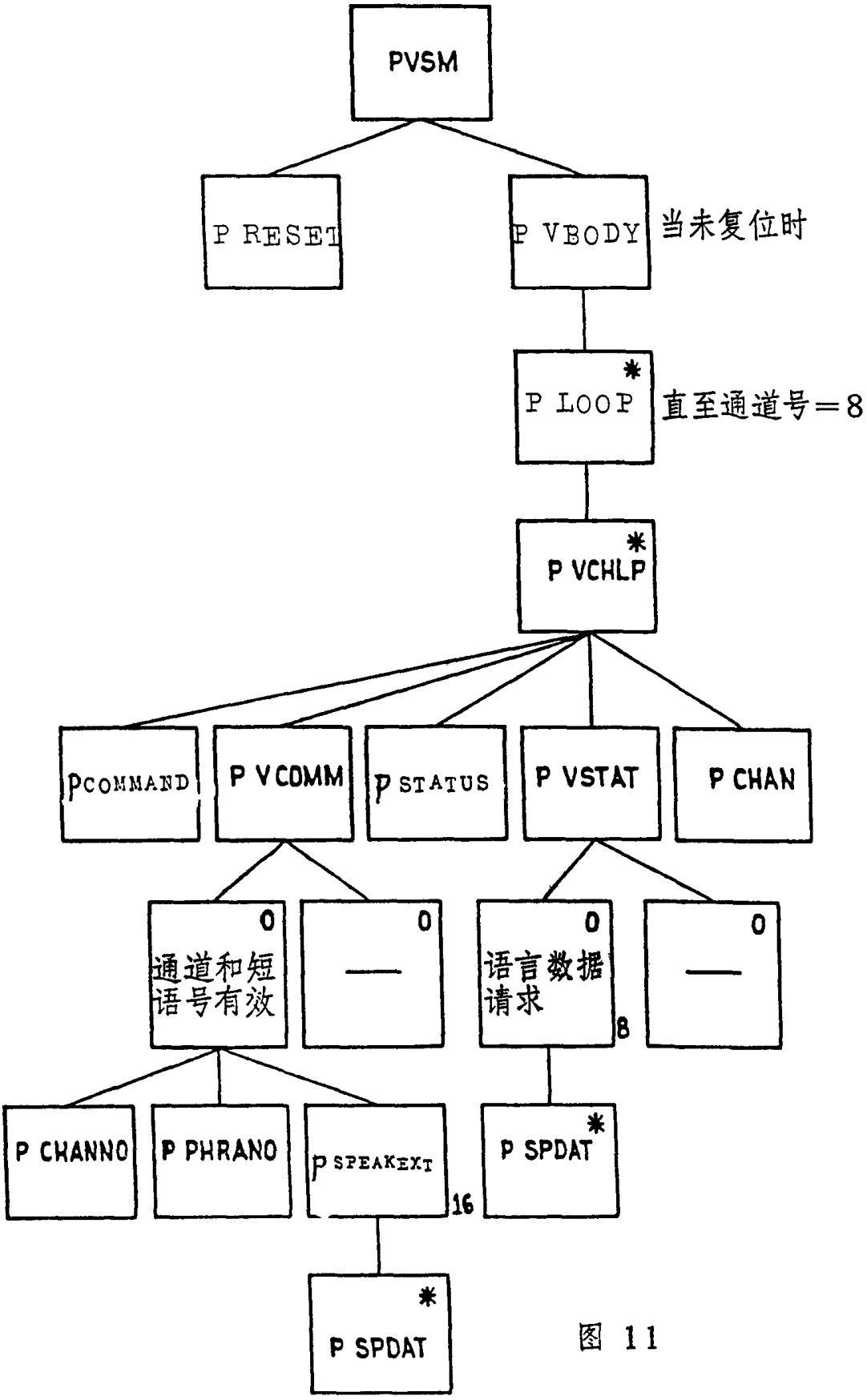

PVSM
P RESET
P VBODY
当未复位时
P LOOP
*
直至通道号＝8
P VCHLP
*
PCOMMAND
P VCOMM
P STATUS
P VSTAT
P CHAN
0
通道和短语号有效
0
—
0
语言数据请求
8
0
—
P CHANNO
P PHRANO
P SPEAKEXT
16
P SPDAT
*
P SPDAT
*

图 11

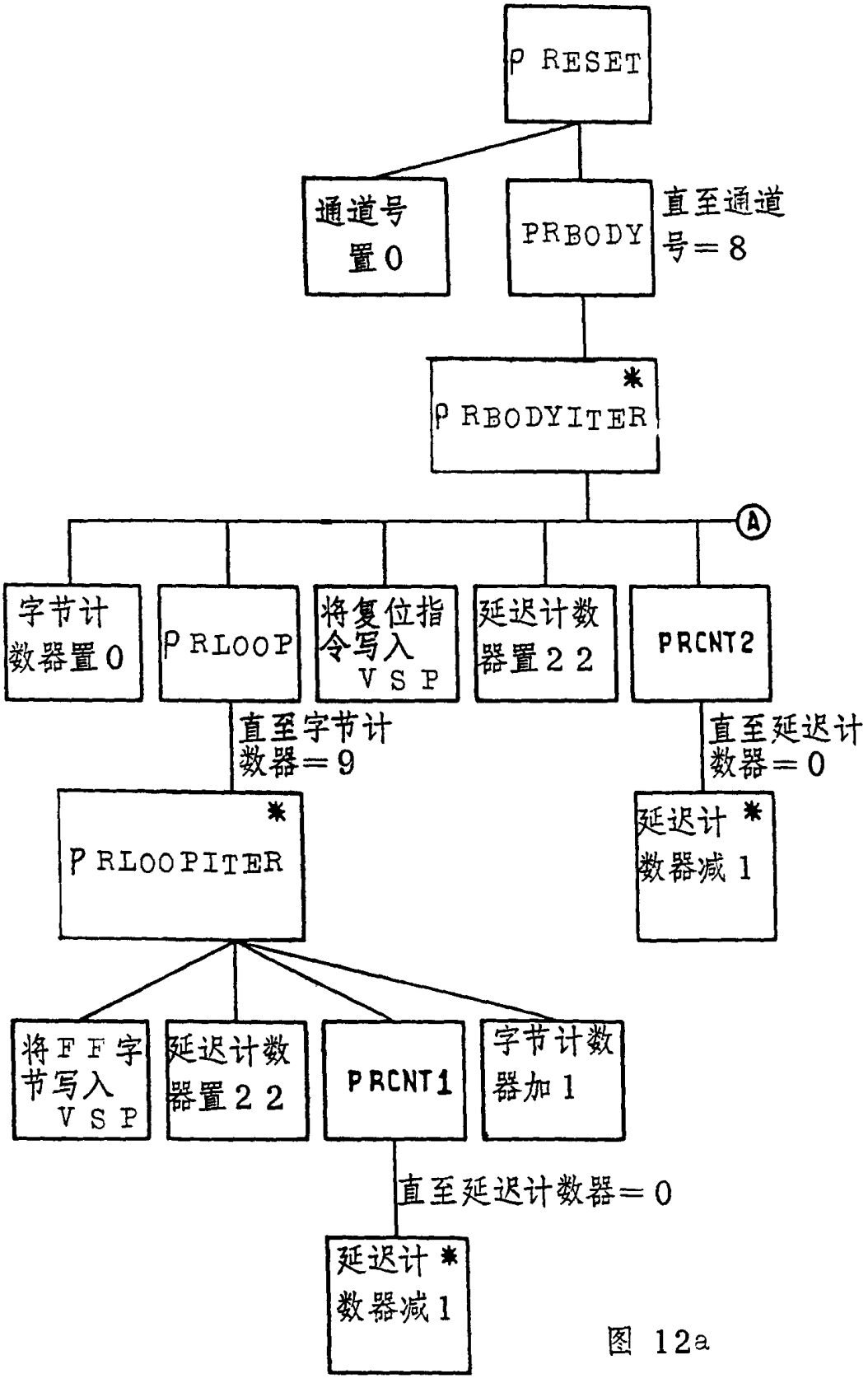

P RESET
通道号
置0
PRBODY
直至通道
号＝8
*
P RBODYITER
A
字节计
数器置0
P RLOOP
将复位指
令写入
V S P
延迟计数
器置22
PRCNT2
直至字节计
数器＝9
直至延迟计
数器＝0
*
P RLOOPITER
延迟计 *
数器减1
将FF字
节写入
V S P
延迟计数
器置22
PRCNT1
字节计数
器加1
直至延迟计数器＝0
延迟计 *
数器减1

图 12a

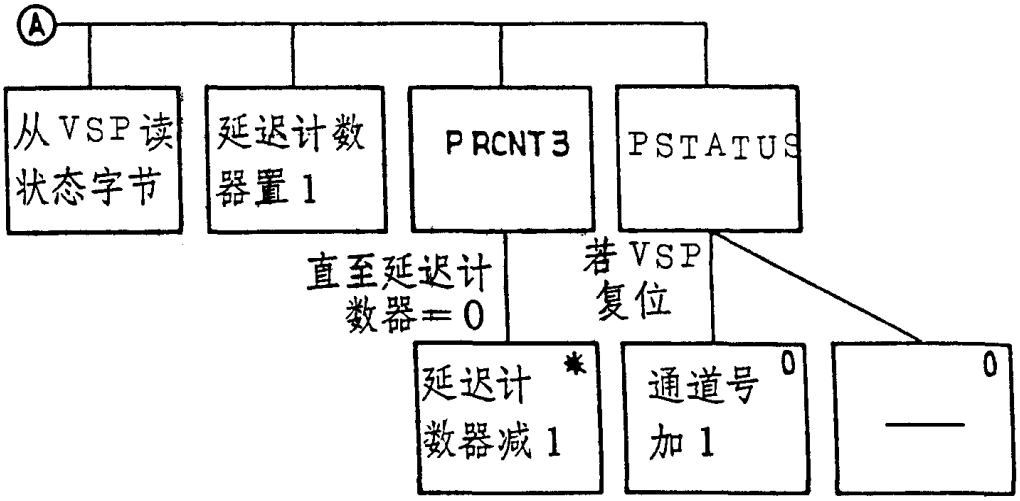

A
从VSP读状态字节
延迟计数器置1
PRCNT3
PSTATUS
直至延迟计数器=0
若VSP复位
延迟计数器减1
*
通道号加1
0
0

图 12b

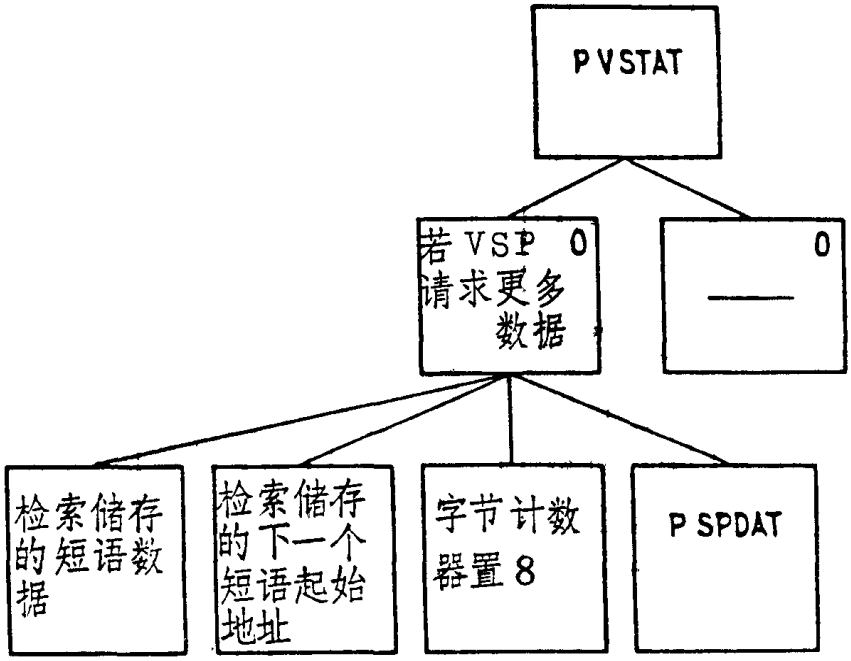

PVSTAT
若VSP请求更多数据
0
0
检索储存的短语数据
检索储存的下一个短语起始地址
字节计数器置8
PSPDAT

图 15

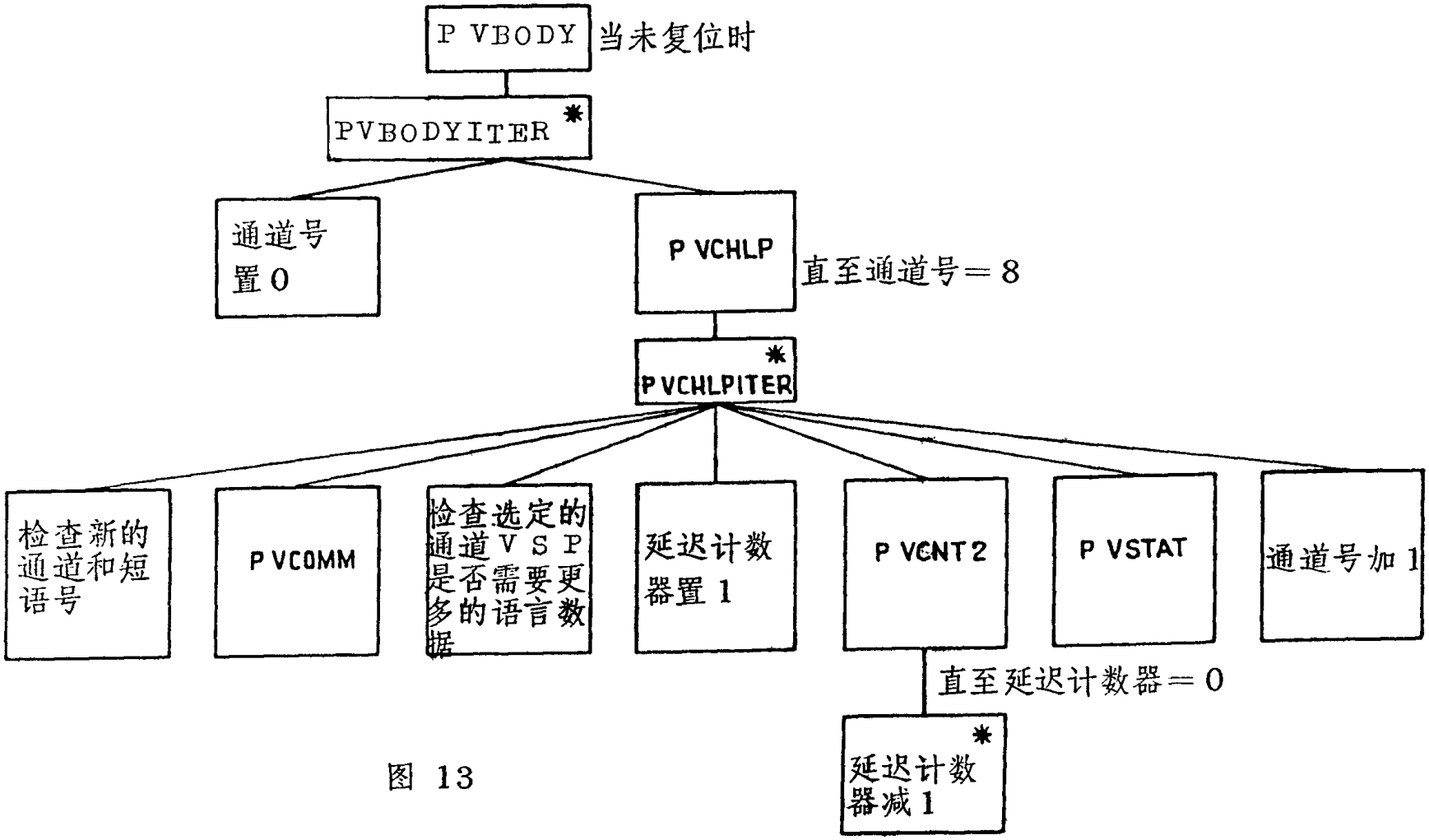

P VBODY
当未复位时
PVBODYITER *
通道号
置0
P VCHLP
直至通道号＝8
P VCHLPITER *
检查新的通道和短语号
P VCOMM
检查选定的通道VSP是否需要更多的语言数据
延迟计数器置1
P VCNT2
P VSTAT
通道号加1
直至延迟计数器＝0
延迟计数器减1 *

图 13

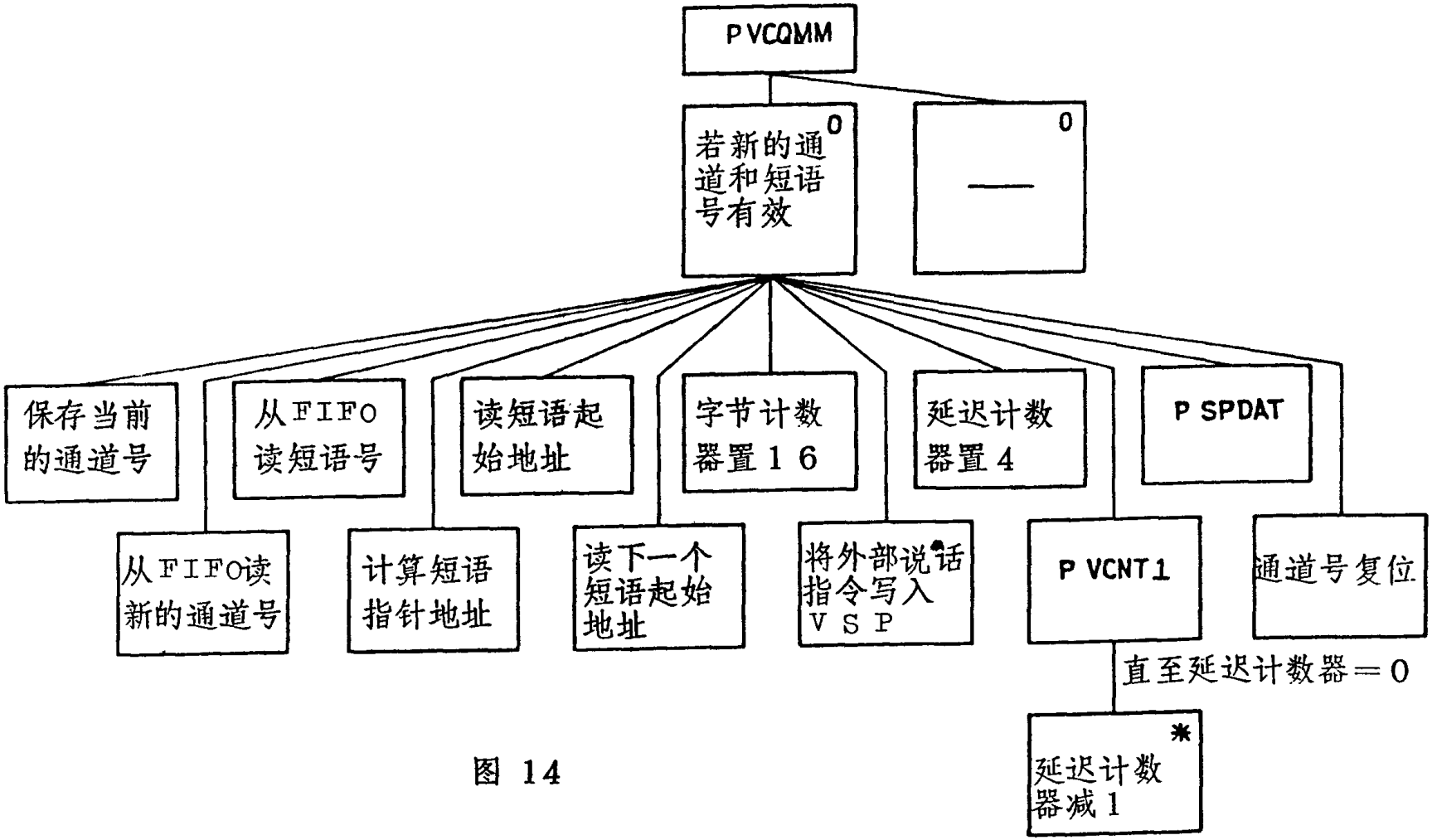

PVCQMM
若新的通道和短语号有效
0
—
0
保存当前的通道号
从FIFO读短语号
读短语起始地址
字节计数器置16
延迟计数器置4
P SPDAT
从FIFO读新的通道号
计算短语指针地址
读下一个短语起始地址
将外部说话指令写入VSP
P VCNT1
通道号复位
直至延迟计数器=0
延迟计数器减1
*

图 14

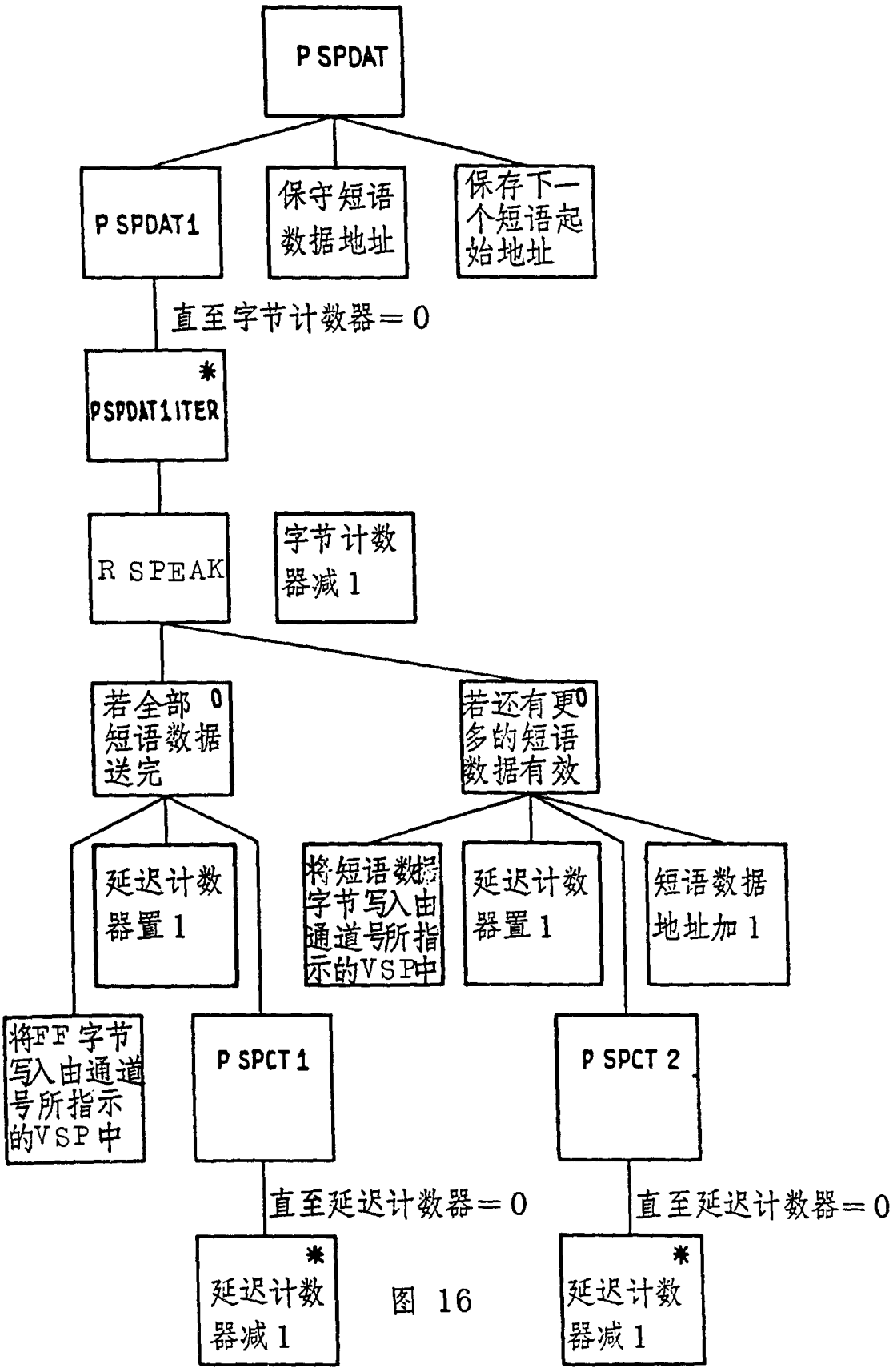

P SPDAT
P SPDAT1
保守短语数据地址
保存下一个短语起始地址
直至字节计数器＝0
*
PSPDAT1ITER
R SPEAK
字节计数器减1
若全部短语数据送完
若还有更多的短语数据有效
延迟计数器置1
将短语数据字节写入由通道号所指示的VSP中
延迟计数器置1
短语数据地址加1
将FF字节写入由通道号所指示的VSP中
P SPCT1
P SPCT2
直至延迟计数器＝0
直至延迟计数器＝0
*
延迟计数器减1
*
延迟计数器减1

图 16