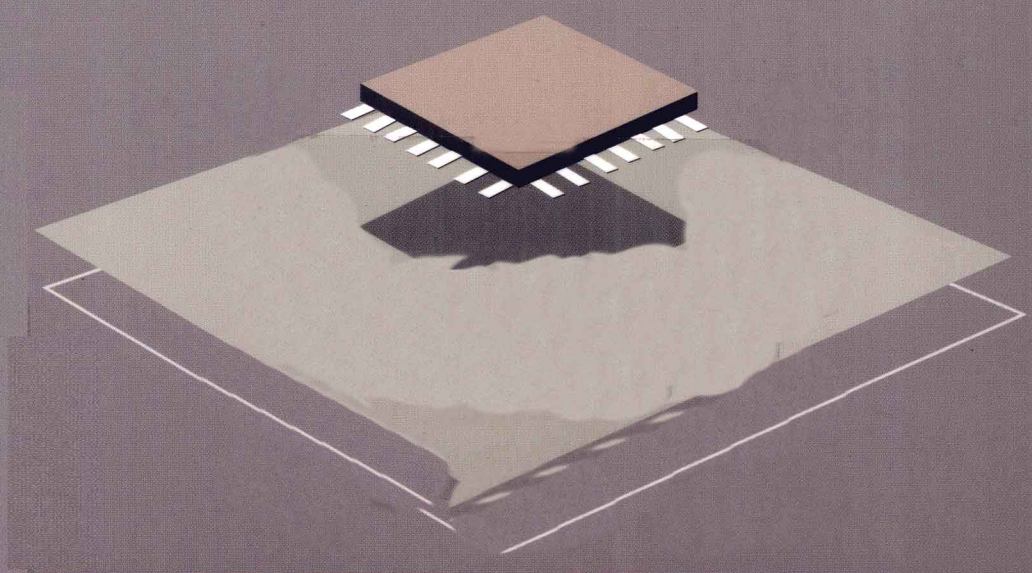
嵌入式系统基础

——ARM与Realview MDK (Keil for ARM)

任 哲 张永忠 编著



北京航空航天大学出版社
BEIHANG UNIVERSITY PRESS

嵌入式系统基础

——ARM与Realview MDK(Keil for ARM)

任　哲　张永忠　编著

北京航空航天大学出版社

内 容 简 介

本书在介绍微型计算机一般原理的基础上，重点介绍目前在应用中流行的 ARM RISC 体系结构及其嵌入式处理器，同时介绍当前流行的 ARM 工程开发工具 Realview MDK，并以该开发工具为基础介绍 ARM 汇编语言程序设计、工程开发及 RTX 操作系统的简单应用。

为了拓展学生的知识和培养学生的自学能力，本书还在附录中简要地介绍另一个常用的基于 ARM 体系结构的嵌入式处理器 S3C44B0X 的构成及其主要接口以及 ARM 体系结构的高级存储管理部分，以便为读者学习高档 ARM 处理器核建立必要的基础。

本书适合高等院校电气自动化、仪器仪表、电子技术等专业微机原理课程使用，也可作为其他以计算机嵌入式应用为教学目标的专业教学用书，当然也可供对 ARM 体系结构及其嵌入式处理器感兴趣的专业人士阅读、参考。

图书在版编目(CIP)数据

嵌入式系统基础 ：ARM 与 Realview MDK(Keil for ARM) / 任哲，张永忠编著. -- 北京 ：北京航空航天大学出版社，2012.2

ISBN 978-7-5124-0692-6

Ⅰ.①嵌… Ⅱ. ①任… ②张… Ⅲ. ①实时操作系统 Ⅳ.①TP316.2

中国版本图书馆 CIP 数据核字(2011)第 278300 号

嵌入式系统基础

——ARM 与 Realview MDK(Keil for ARM)

任 哲 张永忠 编著

责任编辑 王慕冰 龚荣桂 王平豪

*

北京航空航天大学出版社出版发行

北京市海淀区学院路 37 号(邮编 100191) http://www.buaapress.com.cn

发行部电话：(010)82317024 传真：(010)82328026

读者信箱：bhpress@263.net 邮购电话：(010)82316936

北京时代华都印刷有限公司印装 各地书店经销

*

开本：787×960 1/16 印张：27.75 字数：622 千字

2012 年 2 月第 1 版 2012 年 2 月第 1 次印刷 印数：4 000 册

ISBN 978-7-5124-0692-6 定价：56.00 元

前　言

近年来，由于计算机技术的发展，嵌入式系统已在汽车、航空、航天、造船、仪器仪表、家用电器等领域得到了极其广泛的应用，进而也激发了社会对嵌入式系统人才的巨大需求。所以，近几年来我国各高等院校相继开设了嵌入式系统的相关课程。本书就是一本专为高等院校教学使用的、专门介绍 ARM 体系结构及其嵌入式处理器的教科书。

本书可以说是作者原作《ARM 体系结构及其嵌入式处理器》(北京航空航天大学出版社 2008 年出版)一书的改写版。本书除了保留原作起点低、通俗易懂、信息量大的特点之外，还借助书中指令系统及编程例题介绍了 ARM 工程开发方法和开发工具 Realview MDK(Keil for ARM)的使用；更具特色的是，本书借助开发工具的介绍以通俗的语言详细地介绍了 ARM 程序结构、映像文件、分散加载等一些其他书籍涉及较少，但又需要嵌入式系统开发者了解并掌握的计算机基本理论和知识。

全书共分 11 章。第 1、2 章为基础部分，重点介绍微型计算机系统的基本构成及基本工作原理；第 3～5 章介绍 ARM 体系结构、指令系统及汇编语言程序设计的基本知识；第 6 章介绍 ARM 工程及其开发工具；第 7 章在介绍微型机中断技术的基本概念基础上介绍了 ARM 的中断系统及其特点；第 8、9 章介绍基于 ARM 体系结构的嵌入式处理器 LPC2000 的构成、特点及其外围接口；第 10 章在简要介绍计算机固件的基础上介绍了 LPC2000 的固件；第 11 章介绍嵌入式操作系统基础及 MDK 提供的嵌入式操作系统 RTX 的简单应用。

为了拓展学生的知识和培养学生的自学能力，本书还在附录 C 中简要地介绍了另一个常用的基于 ARM 体系结构的嵌入式处理器 S3C44B0X 的构成及其主要接口，以使读者对 ARM 体系结构嵌入式处理器有一个更全面的认识；为保证 ARM 知识体系的完整性，在附录 D 中介绍 ARM 体系结构的高级存储管理，从而为读者进一步学习高档 ARM 处理器核建立了必要的基础。

本书编者为任哲、张永忠、章小卫和房红征，其中任哲和张永忠为主编。

本书在编写过程中参考和借鉴了大量的相关资料(见参考文献)及网络资源，并引用了其中一些文字和代码，在此谨对这些作者表示衷心的感谢。

由于作者水平有限，书中难免出现错误，希望广大读者能给予批评指正。作者的电子邮箱为 renzhe71@sina.com。

任　哲

2011 年 7 月

目　录

第1章　微型计算机基础知识

自 1946 年世界上出现第一台数字电子计算机 ENIAC 之后，它就以强大的运算能力使世界感到震惊。其后，经过半个多世纪的飞速发展，计算机技术应用已经深入到人们生产和生活的各个方面，以至于不管你是否情愿，都要与之打交道。

本章的主要内容有：

- 数字电子电路是计算机硬件的基础；
- 二进制数制及机器数；
- 有符号数的表示及溢出的概念；
- 存储器的基本电路及其作用，存储器存储单元的地址；
- 运算器、控制器的基本概念；
- 指令及指令系统，计算机程序的执行过程；
- 计算机的硬件组成；
- SISC 和 RISC 架构；
- 微型计算机的桌面系统应用与嵌入式应用。

1.1　微型计算机的发展历程

根据计算机所使用的电子器件，计算机的发展经历了电子管、晶体管、集成电路、大规模集成电路和超大规模集成电路等几个阶段。同时，计算机在体系结构、组成方式及其体积、性能、价格、应用领域等方面也发生了重大变化。目前，计算机可分为巨型机、大型机、中型机、小型机和微型机这 5 类。其中，人们接触最多的是叫做“微机”的微型计算机。

微型计算机出现在 20 世纪 70 年代。1971 年，以 Intel 公司设计的世界上第一个微处理器芯片 Intel 4004 为标志，电子数字计算机的发展便进入了微型计算机时代。微型计算机一出现，就表现出了强大的生命力，在短短的几十年间就经过了从第一代到第五代的发展历程：第一代是以 4004、4040 和 8008 微处理器为代表的字长为 4 位和 8 位的微型计算机；第二代是以微处理器 Z80、I8085、M6800 和 Apple－II 等为代表的中高档 8 位机；第三代是以 8086、8088、80286 微处理器为代表的 16 位机；第四代是以 80386、80486、Pentium、Pentium II、

Pentium III、Pentium IV 等微处理器为代表的 32 位机；第五代则是以 Itanium、MIPS 为代表的具有精简指令架构(RISC)的 64 位机。

1.2 数字电路与计算机

电子数字计算机，是一种用电子装置构成，能够按照机器操作者输入的命令进行数字信息处理的机器。从功能上看，它具有两个基本功能：一是表示和存储数字数据的功能；二是对数字数据进行运算的功能。从物理构成的角度来看，它是一种由数字电子器件和电路构成的运算装置。

1.2.1 数据在计算机中的表示——二进制

作为计算机的硬件基础——数字电路，只能用电路信号端子的电信号(电平)来表示数字信息，即规定，低电平代表数码 0(低电平也叫做电路的“0”状态)，高电平代表数码 1(高电平也叫做电路的“1”状态)，所以它的一个信号端也就只能表示两个数码——0 和 1。但是，如果电路有两个信号端子，那么这两个端子就可以有 4 个状态——00、01、10 和 11，如果用 00 表示十进制数 0，用 01 表示十进制数 1，用 10 表示十进制数 2，用 11 表示十进制数 3，那么这个电路就能表达 4 个十进制数了。这就是说，一个数字电路的信号端子越多，其状态就越多，所能表达的数也就越多。稍微分析一下就会知道，一个具有 n 个信号端子的数字电路所具有的状态数为 2^n，那么它所能表达的数的个数就为 2^n 个。

例如，有 8 个信号端子的数字电路就可以有如下的 256 个状态，并可像下面那样表示 256 个十进制数 0～255：

0000 0000 表示十进制数 0；
0000 0001 表示十进制数 1；
0000 0010 表示十进制数 2；
0000 0011 表示十进制数 3；
0000 0100 表示十进制数 4；
⋮
1111 1111 表示十进制数 255。

从上面左边的表示中可以看到，电路端子的表示是以“逢 2 进 1”为加法运算规则，以“借 1 当 2”为减法运算规则的二进制数制。

1.2.2 运算器及二进制数运算

计算机的核心任务是进行数的运算，该功能是由一个叫做算术逻辑运算单元(ALU)的电路来实现的。

1. 算术逻辑运算单元 ALU 的概念

最基本的算术运算是加法运算，其他各种运算都可以通过加法运算来完成，因此二进制数的加法运算就是计算机的核心运算功能。当然，这个二进制运算要由数字电路来完成。按照逢 2 进 1 的二进制加法运算规则，两个 1 位二进制数 A 和 B 相加时，其加法运算电路的真值表如表 1－1 所列。

这种加法运算产生了两个信息，即两数和 S 及进位 C。

显然，作为能进行完整加法运算的加法器，应该能进行被加数、加数、低位进位三个数的相加运算，即一个能进行完整加法运算的加法器的真值表应该如表 1－2 所列。

表 1－1　一位二进制加法的真值表

输　入		输　出	
A	B	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

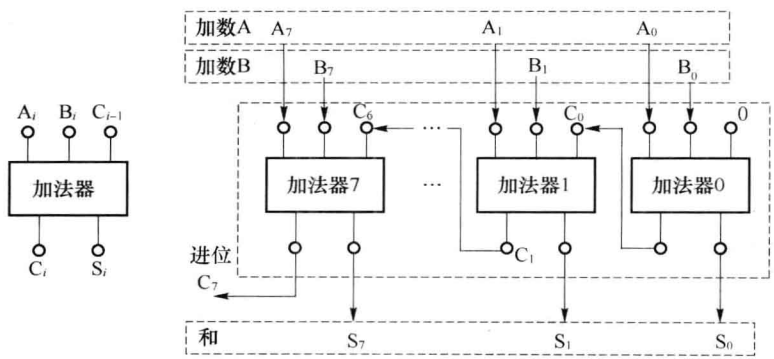
表 1－2　带有进位值的一位二进制数加法真值表

输　入			输　出		输　入			输　出	
A_i	B_i	C_{i-1}	S_i	C_i	A_i	B_i	C_{i-1}	S_i	C_i
0	0	0	0	0	0	0	1	0	1
0	1	0	1	0	0	1	1	0	1
1	0	0	1	0	1	0	1	0	1
1	1	0	0	1	1	1	1	1	1

为了区别，把具有表 1－1 运算功能的加法器叫做半加器，而具有表 1－2 运算功能的加法器叫做全加器。

根据表 1－2 的真值表，可以用数字电路构成一个如图 1－1(a)所示的全加器。而用多个全加器就可以构成如图 1－1(b)所示的多位二进制数加法器。

图 1－1(b)所示的多位二进制数加法器就是计算机的核心部件，如果在此基础上配置一些其他辅助电路就构成计算机 CPU 中的算术逻辑运算单元 ALU。



(a) 1位加法器　　(b) 8位加法器

图 1－1　数字加法器示意图

由于进行数的运算时，还会产生一些与运算过程和结果相关的其他信息，例如运算中是否发生了进位、运算结果是否为零，等等。为了记录这些信息，ALU 中还设有一个专门保存上述信息的存储装置，这个存储装置叫做程序状态寄存器。

ALU 和程序状态寄存器的示意图如图 1－2 所示。

2. 机器数

显然，是由于计算机硬件电路的物理限制，才使得计算机不得不用这种人们既不熟悉也不方便的二进制数，所以这种二进制数也常常叫做机器数。

凡使用机器来计数的装置都有一个共同的特点：它们的字长是有限的，或者说它们的计数范围是有限的，即当计数计到最大值再进行加法计数时，其和会返回(回卷)到 0 重新计数。

例如图 1－3 所示的钟表的表盘，当顺时钟计数(加法计数)计到最大值 11 后，如果再加 1，则从表盘得到的值是 0 而不是 12(当然，真实的表盘写的还是 12，而不是人们不喜欢的 0，但实质是 0)。计算机的计数器和运算器也有同样的特点，例如，8 位二进制计数装置在进行加法和减法计数时的变化情况如图 1－4 所示。

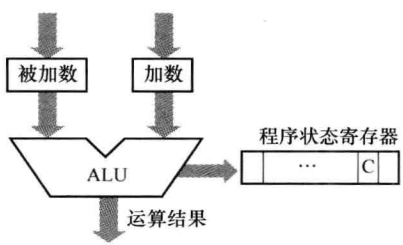
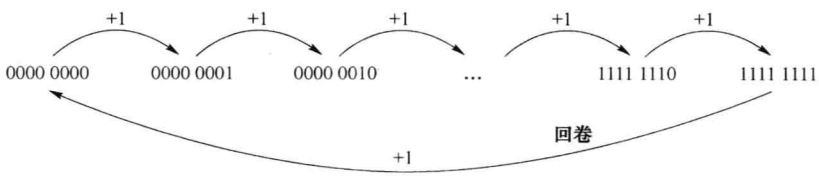
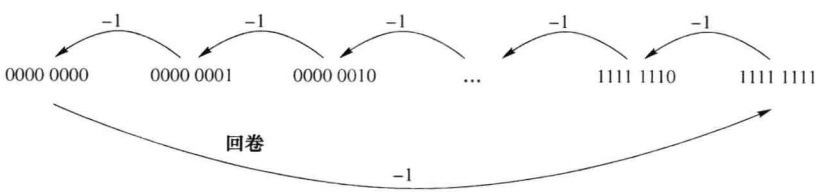


图 1－2　ALU 及程序状态寄存器的示意图

图 1－3　表　盘



(a) 8位二进制加法计数时数的变化情况



(b) 8位二进制减法计数时数的变化情况

图 1－4　8 位二进制计数时数的变化情况

3. 有符号数的机器数表示及其运算

前面谈到的都是无符号数的二进制形式，那么机器数的有符号数又如何表示呢？下面以8位二进制为例来进行说明。前面已经知道，8位二进制无符号数为：

0000 0000 表示十进制数 0；
0000 0001 表示十进制数 1；
0000 0010 表示十进制数 2；
0000 0011 表示十进制数 3；
0000 0100 表示十进制数 4；
⋮
1111 1111 表示十进制数 255。

众所周知，在十进制数制中，0减去某个正数得到的就是与这个正数对应的负数，例如0减去1就是与1对应的负数－1，0减去2就是与2对应的负数－2。那么，从图1-4中可知，8位二进制的0000 0000减0000 0001的结果1111 1111就应该是对应十进制的－1，如果从1111 1111再减0000 0001，即减0000 0010得到的1111 1110就应该是十进制的－2。以此类推，可以一直减到1000 0000(对应十进制的－128)。这就是说，0000 0000～1111 1111这256个数既可以看成正数也可以看成负数；也就是说，0000 0000～1111 1111只是一种表示而已，至于它们究竟是正数还是负数，则是由使用数的人来解释的。但是，作为一种通用运算装置，对于计算机中的有符号数最好还是有一个通用的规则。由于通常希望在一个计数规则中的正数和负数的数目最好基本相等，于是在表达有符号数时，人们就把0000 0000～1111 1111这256个数里那些最高位为1的二进制数看成负数，而把最高位为0的那些二进制数看成正数(0000 0000除外)，这样这256个数就可以0000 0000为界分为如下两部分：

0111 1111 表示十进制数 127 ┐
⋮ │
0000 0011 表示十进制数 3 ├ 正数
0000 0010 表示十进制数 2 │
0000 0001 表示十进制数 1 │
0000 0000 表示十进制数 0 ┘

1111 1111 表示十进制数－1 ┐
1111 1110 表示十进制数－2 │
1111 1101 表示十进制数－3 ├ 负数
⋮ │
1000 0000 表示十进制数－128 ┘

由于所有正数的最高位都为 0，所有负数的最高位都为 1，因此可以用数的最高位来判别有符号数的正负，于是就可以把有符号数的最高位叫做符号位，而把剩下的所有位叫做数值位。同时，为了讨论问题方便，把有符号数所对应的十进制值叫做机器数的真值。

仔细观察就会发现，在 8 位有符号数中，所有负数与其对应的正数相加，其结果都为 10000 0000(对应十进制的 256)，于是就把这个值叫做 8 位二进制数的模。这也意味着 8 位有符号数的负数与其对应的正数是以 10000 0000 为模互补的，所以上述这种有符号数的表示形式叫做有符号数的补码形式。

虽然上面有符号数是用 8 位二进制数来介绍的，但其规律同样可以推广到其他字长的计数规则中。

在十进制计数中，人们熟悉的只是用真值来表示负数的方法，而不熟悉上述这种补码表示方式，为了方便从真值求取负数的补码，又引入了反码的概念。所谓反码，就是把一个二进制数按位取反所得到的数码。例如，0001 0101 的反码为 1110 1010。有了这个反码的概念后，再求取负数的补码形式就方便得多。具体方法为：先把一个负数的真值的绝对值用二进制形式表示出来，然后把这个二进制数按位取反，最后再加 1，得到的就是该负数的补码形式。例如，一个有符号数的真值为－33，为了求得它的 8 位二进制补码，就先写出其绝对值 33 的二进制形式 0011 0011，然后按位取反得 1100 1100，最后再加 1，得到的就是－33 的二进制补码形式 1100 1101。

采用上述补码的最大优点是，计算机的运算器在做二进制的加法运算时，无须对两个加数是否为有符号数进行区分，只要简单地把这两个数相加就可以了，至于是有符号数运算还是无符号数运算则由用户来解释。所以，要注意，有符号数的最高位(符号位)也是参与运算的。

计算机之所以采用二进制的补码形式，就是为了把有符号数与无符号数的运算规则统一起来。因为减法运算就是一个加负数的运算，这样，运算器只要具有加法运算功能就可以了，而没有必要再另外设计减法器。

例如，1001 0011(如果看成无符号数，则该数为 147；如果看成有符号数，则该数为－109)和 0011 1111(如果看成无符号数，则该数为 63；如果看成有符号数，则该数仍为 63)相加的结果为 1101 0010(如果看成无符号数，则该数为 210；如果看成有符号数，则该数为－46)。可以看到，所有结果都是正确的。

从上面的介绍中已经知道，在有符号数中，二进制形式中的最高位为符号位，也就是说，有符号数的数值位比数据的整个位数少了 1 位。例如在 8 位有符号数中，它们的数值位只有 7 位，这样就使得 8 位有符号正数的数值范围为 0～127，负数的数值范围为－128～－1，因此，在 8 位有符号数的运算过程中，其运算结果的数值不得超过上述范围，否则得到的是一个错误结果，这种现象叫做溢出。为了在出现溢出时能为用户提供相应的提示信息，运算器应该具有判断溢出的能力。图 1－5 给出了计算机运算器用双进位法来判断溢出的示意图。图中 C_2 是运算结果最高位 D_{n-1} 向进位位产生的进位信号，C_1 是运算结果数值位的最高位 D_{n-2} 向 D_{n-1}

产生的进位信号，如果这两个信号相同，则无溢出，否则有溢出。即：

$$OF = C_1 \oplus C_2$$

运算器用上述方法来判断运算是否有溢出，其溢出信息被存入程序状态寄存器。

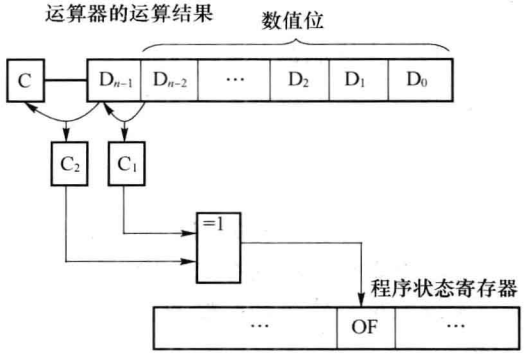


图 1-5　双进位位法判断溢出的示意图

4. 二进制数据的十六进制表示

显然，数的二进制表示方式有一个缺点：一个很小的数的二进制表示不仅很长，而且不便于阅读、书写和记忆。为此，常常把一个较长的二进制数分成若干组，每组 4 位二进制数码，然后把每组所表示的数用一位数码来表示。由于 4 位二进制数 0000～1111 对应的十进制数为 0～15，是两位数，所以又引入了 6 个大写英文字符 A、B、C、D、E、F 作为数码来分别表示 10、11、12、13、14 和 15，从而形成了二进制数的十六进制表示方式。4 位二进制数对应的十六进制数码见表 1-3。

表 1-3　4 位二进制数对应的十六进制数和十进制数

二进制	十六进制	十进制	二进制	十六进制	十进制
0000	0	0	1000	8	8
0001	1	1	1001	9	9
0010	2	2	1010	A	10
0011	3	3	1011	B	11
0100	4	4	1100	C	12
0101	5	5	1101	D	13
0110	6	6	1110	E	14
0111	7	7	1111	F	15

例如，上面 8 位二进制的 256 个数，用十六进制方式表达出来就是：

0x00 表示二进制数 0000 0000，十进制数 0；
0x01 表示二进制数 0000 0001，十进制数 1；
0x02 表示二进制数 0000 0010，十进制数 2；
0x03 表示二进制数 0000 0011，十进制数 3；
⋮
0xFF 表示二进制数 1111 1111，十进制数 255。

为了区别，在十六进制数的前面要加上前缀 0x。

1.2.3 数据的存储

人们在进行数的运算时，为了记录运算所需的初始数据、中间运算结果和最终结果，一般需要一张纸，这张纸实质上就是一个存储数据的存储装置。同样，计算机在进行运算时也需要相应的数据存储装置，这种装置就叫做计算机的存储器。

1. 存储器的基本电路

原则上说，任何能接收并且能保存数字信号的装置都可以作为计算机的存储装置。而数字电路中的双稳态电路就是这样一种可以存储二进制信息的装置。

例如图 1-6 所示的一个双稳态电路，在选择信号使开关 K1 和 K2 接通时，就会使 D 端的输入传送到 A 端。即当 D 端输入信号为高电平“1”(或低电平“0”)，则会使双稳态电路的 A 端为“1”(或“0”)，B 端为“0”(或“1”)。通过数字电子电路中所学的知识知道，由于 Q1 和 Q2 的交叉耦合作用，即使以后开关 K1 和 K2 断开，电路 A 端和 B 端的电平也不会发生变化。也就是说，电路可以把输入的信息保存下来，并且在其后的某个时刻如果再度接通开关 K1 和 K2，则会在 D 端测得 A 端的“1”(或“0”)信号。这就是说，这种电路可以输入(写)和保存(存储)电平信号，而且在需要时还可以输出(读)该信号。因此，图 1-6 所示的电路就是用来存储一位二进制数字的基本存储电路。

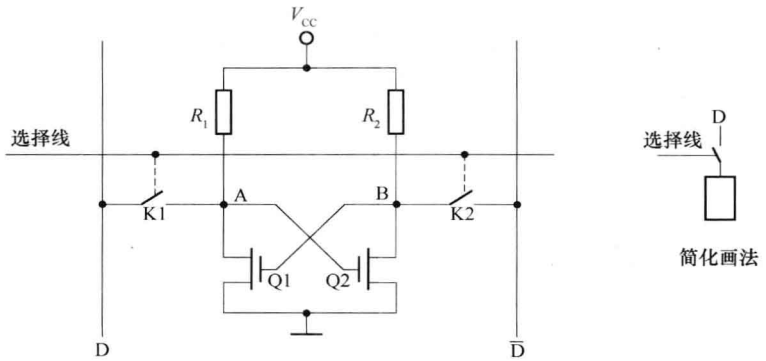


图 1-6 双稳态电路