数字计算机错误检测系统的设计

张国良　赵庆林 编

国防工业出版社

内 容 简 介

本书比较系统而又通俗地介绍了数字计算机中错误检测系统的设计原理和技术。全书共分十五章。第一章论述了错误检测系统在数字计算机中的作用；第二、三章讲述错误检测系统的基础知识，即数学基础和一些编码理论；第十五章阐明了有关整个错误检测系统的一些设计问题，其余各章则分别论述了计算机中各种逻辑部件的错误检测的逻辑设计。

本书可供从事计算机的专业人员，特别是逻辑设计者、维护人员，作为一本有关错误检测系统的入门书来阅读。亦可供高等学校或中等学校的计算机专业的师生参阅。

数字计算机错误检测系统的设计

张国良　赵以林　著

*

国防工业出版社出版

新华书店北京发行所发行　　各地新华书店经售

国防工业出版社印刷厂印装

*

787×1092 1/16　印张 10　228 千字

1980年6月第一版　1980年6月第一次印刷　印数：0,001— 7,500册

统一书号：15034·1934　　定价：1.10元

前　言

随着我国社会主义四个现代化事业的蓬勃发展，电子数字计算机在国防、科学技术，以及国民经济的各个部门的应用，必将愈来愈广泛。这种应用领域的深度、广度的不断扩大，就对数字计算机的可靠性、稳定性提出了越来越高的要求。这就是说，要求投入运行的计算机，应尽量少出故障，而出了故障之后，恢复得愈快愈好。针对这些要求，在数字计算机的设计中，发展了一些新的设计技术，即所谓的RAS技术、容错技术，以及故障诊断技术。这些技术，在国内外的计算机设计中已广泛采用，并且成为评价计算机性能的重要标志。因此，无论对从事数字计算机的设计人员、调试维护人员，或者计算机专业的师生，都有必要了解和熟悉这些技术。而错误检测系统的设计，则是这些技术的基础。这是因为，对于一个运行的计算机系统，首先是要求能及时的发现故障，然后才谈得上对故障的处理问题。错误检测系统正是计算机中的监视线路，它能及时捕捉计算机运行中的错误。

本书通俗而又较系统地介绍了数字计算机中错误检测系统的设计问题。全书共分十五章。第一章概述了错误检测系统的功能。第二、三章介绍了基础知识。第四至第十四章则分别论述了数字计算机中各类逻辑部件的错误检测的逻辑设计。第十五章介绍整个错误检测系统的设计。由于各章之间没有什么很密切的联系，所以，读者也无需按次序阅读。在读完第二、三章之后，读者可根据自己的需要和兴趣任意选读其余各章。

本书绝大部分内容取材于〔美〕F. P. Sellers等人写的《数字计算机中的错误检测逻辑》一书。只是对该书的某些章节重新进行了编写，加进了一些新的内容。

本书中的逻辑图，是采用《中华人民共和国第四机械工业部部标准（二进制逻辑电路图形符号SJ1223-77）》绘制的。而部标准所没有的符号，我们在使用处都作了说明。这一点请读者注意。

由于我们的水平有限，书中错误、缺点一定不少，望读者及时指正。

目　录

第一章　错误检测系统的功能

本章主要讨论错误检测系统在数字计算机中所完成的基本功能。为此，首先对数字计算机系统中存在的故障进行分析；其次，简单地介绍为了对付这些故障而兴起的容错技术和RAS技术。错误检测系统的设计正是这些新兴技术的基础。

1.1　数字计算机系统故障分析

要设计错误检测系统，只有针对现代电子数字计算机系统的故障情况来进行，才能达到预期的效果。因此，就有必要对现代电子数字计算机系统运行中所出现的故障加以分析。

1.1.1　故障的分类

现代电子数字计算机是一个极其复杂的系统。一台大型电子数字计算机，都是由几万片集成电路组件，上千个各种类型的接插件，以及数十种外部设备构成的庞大系统。这样的系统在软件的控制下，可以完成各种各样的功能。系统的正常运行，不仅和构成这个系统的硬件、软件是否正确有关，而且和系统所处的环境，如温度、湿度、电磁干扰等有关。根据国内外电子数字计算机运行中的统计说明，按故障所表现的各种形式，可以对故障进行如下分类：

一、系统故障

这是一种影响到系统运行的全局性故障。这时，系统出现如下情况：停机或永不停机；系统可以执行程序，但结果总是错误的。这些情况，机房的操作员或程序员都是可能遇到的。系统故障也有固定性故障和偶然性故障之分。如果发生系统故障之后，利用重新启动功能，系统便恢复正常运行，则说明这次系统故障为偶然性的。反之，若重新启动不能复常，需经检修、更换硬件或软件，系统方能恢复正常运行，这时，则称为固定性故障。

二、硬件故障

硬件故障主要指：构成系统的物理器件的工作参数偏离其正常值，或者根本上完全损坏而造成的故障。例如，一个失效了的组件，就引起一个硬件故障。这种硬件故障可能是固定性的，也可能表现为偶然性的。下面还要详细讨论。

三、逻辑故障

逻辑故障亦称逻辑错误。它表现为逻辑部件的输入、输出的逻辑关系不正确。逻辑故障按其持续时间来分，可分为偶然性的和固定性的两类。按其故障范围，可分为独立的和相关的两类。所谓故障范围，主要指逻辑部件中有多少个逻辑变量发生错误。在这些发生错误的逻辑变量中，它们的错误可以是独立的，即一个变量的错误并不影响另一个变量的值；也可以是相关的（或称为分布的），即几个变量的错误是互相关联的。按其数值来分，逻辑故障可分为确定的和不确定的两类。所谓确定的逻辑故障，是指逻辑故障值恒为0或

1；所谓不确定的逻辑故障，则是指其值在 0 和 1 之间来回变动的逻辑故障。

四、软件故障

在系统运行中，软件故障不是指那些由于硬件故障或逻辑故障而引起的，表现于软件的故障，而是指软件本身所蕴含的错误。从某种意义上讲，软件故障都是软件设计和实现中的错误所造成的。在现代计算机的软件生产中，由于软件工程的复杂性，加上缺乏一定的科学方法论基础，以及一整套科学而合理的生产制度，因此可以说，软件的错误几乎是不可避免的。软件的可靠性问题，已成为当前共同关心的一个重要问题。

系统故障、硬件故障、逻辑故障及软件故障，它们之间有着紧密的内在联系。这是由于，系统是由软件和硬件有机结合而成的，而逻辑部件又是由各种单个硬件组成的。因此，一般说来，硬件故障会引起逻辑故障，逻辑故障最后会引起系统故障。但它们之间也有差别。例如，逻辑故障不全是由硬件故障引起的，逻辑设计中时间余量的不合理、电磁干扰的存在等，都会引起逻辑故障。

进行这样的分类，正是为了针对它们的各自特点，采取不同的措施，以求得合理的解决。这一点，在以后各节中还会进一步讲到。

1.1.2 硬件故障分析及一般处理

这一小节特别将硬件故障提出来加以分析，其原因是：尽管本书所讲述的各种错误检测线路，完全是针对数字计算机和数字装置中的逻辑错误而提出的，但在一般情况下，逻辑故障主要是由硬件故障引起的。

在设计错误检测线路时，对系统所使用的元、器件和工艺的失效情况必须有所分析。这一点是很重要的。这是因为，专门设计某种设备去检测那些根本不会出现的错误，显然是一种设备上的浪费；反之，对于应检测的错误而又没有去检测，这在系统设计上，不能说不是一个很大的缺陷。

本书中所设计的错误检测线路，主要是以集成电路作为基本逻辑单元的。无论是从我国目前已投入运行的集成电路计算机的故障统计来看，还是从国外第三代计算机可靠性分析情况来看，集成电路的典型故障都可以分为以下几类：

1．输出固定为 0 或 1；或不高不低；

2．逻辑门的输出，对一个或几个输入没有响应；

3．输出对输入变化的响应很慢（上升或下降时间拖长）；

4．扇出端对有些输入恒是 1，而对另一些输入恒是 0；

5．一个完整的片基失效（电源引线断开或片基损坏）；

6．一个片基上的两个或多个引线短路（交叉处失效）。

就目前的集成电路工艺结构来说，上述硬件故障所引起的逻辑错误，大多数仍属于范围独立的逻辑错误，即多数为单错误，少数为范围相关的逻辑错误。这一点，对错误检测系统及诊断系统的设计，都是一个极为重要的前提条件。

较为麻烦的是偶然错误。它只出现一次，或者经过很长的一段时间才重复出现一次。以下几种情况都可能引起偶然性逻辑错误：

1．电磁噪声耦合到逻辑线路上；

2．电源不符合使用条件；

3．线路元件的工作温度过高；

4．在具有精确时标的逻辑中，线路的延迟特性的随机漂移；

5．偶然使用了一个失效的元件。

在设计错误检测线路时，只可能集中对付那些经常出现的错误类型，要想检测系统中所有的错误是不现实的。设计时，应该在性能和成本之间取得较好的综合平衡。

针对上述的故障分析，在设计计算机系统时，一般来说，可以检测以下几类错误和故障：

1．程序中的错误；

2．操作员的错误动作；

3．数据存贮载体或数据传送中的错误；

4．硬设备中的错误：

a）固定性逻辑错误；

b）偶然性逻辑错误；

c）错误检测线路本身的错误；

d）电源、冷却系统或设备中的机械零件的故障。

对于程序中的错误，现在许多计算机都能进行基本的检查。例如，编译程序可以发现源程序中语法错误和编码错误。另外，非法操作码、非法存贮访问等错误，也可以被发现。但对于程序中算法流程的错误，数字系统却是无能为力的。它总是要由程序员和用户自己去检查。

通过安装检测器，可以发现操作员的错误。例如，若在磁带机的门上安装一个检测器，则当此门不该打开而被打开时，便发出信号。又如，若在磁盘驱动器的盖上安装一个检测器，则可以防止当磁盘正在运行时打开上盖。再一个办法就是，要求操作员发出的全部信息，都应遵循一定的格式。如果系统接到的信息格式不正确，则发出一个说明操作员操作错误的信号。一个有趣的人类工程问题是，究竟给操作员规定多少种手续、多少种格式最为恰当。若太繁杂了，则会引起新的错误；若太简单了，则又会使其不起作用。实践证明，操作员的误动作是不可轻视的，往往因为操作员的某些疏忽而造成很大损失。因此，在有些特殊使用的计算机系统中，就得明确规定：在什么条件下操作员可以干什么。否则，将发出信号。

电源故障、冷却系统故障或某些机械零件故障，都能引起系统故障。在有些系统中，这类故障甚至比逻辑故障还多。电源和冷却系统故障，很容易通过监督装置进行检测。一般系统还设有掉电保护措施，它可以防止突然停电时，不致破坏信息。这样就可以保证：加电后，只要重新启动系统，即可继续执行掉电前被中断的操作。有的设备装有温度敏感器，当冷却系统因故障造成温度过高时，这种温度敏感器就立即发出信号。

关于其他几类故障和错误的处理，将在本书的以后各章中加以讨论，这里就不多写了。

1.2 RAS 技术和容错技术

本节仅对 RAS 技术和容错技术作一个概念性简单介绍。

数字计算机从它诞生的那一天起，人们就为了使它能可靠而稳定地运行，进行了大量的研究工作。早在五十年代初期，就有人从理论上探讨过，用不可靠的元件能否构成一个可靠系统的问题。随着电子计算机的不断发展，这方面的理论和实践也在不断发展，至今，已逐步形成了 RAS 技术和容错技术。

1.2.1 RAS 技术

RAS 是英文 Reliability（可靠性）、Availability（利用率、可利用性）、Serviceability（维修率、可维修性）的字头缩写。RAS 技术是 IBM 公司在发表 IBM370 系统时首先提出来的。之后，便逐步成为计算机系统设计中不可缺少的一门技术。目前，国内外的许多计算机，都采用了 RAS 技术，并且，它已成为评价一台计算机性能好坏的重要标志之一。

RAS 技术是研究探讨提高电子计算机可靠性、利用率和维修效率的一门综合性技术。它的基本目标是：

1．尽量减少硬件故障的发生，使其对系统的影响降低到最小限度。

2．当发生硬件故障或逻辑错误时，能够及时发现并加以处理，不使其蔓延下去，而造成更大的损失。

3．当硬件故障或逻辑错误引起系统故障时，能在尽量短的时间内使系统迅速恢复工作。

因此，可以说，RAS 技术要解决的是系统少生“病”、生了“病”则能对症下药、及时治疗的问题。为了达到上述目标，RAS 技术主要包括以下几方面内容：

1．采用高可靠性的元件、器件及设备。为此，必须对元件、器件进行科学地筛选及合理地使用。生产过程的质量控制，也是非常重要的。这是因为，一台大型数字计算机就有几万片组件、上千个接插件、数十万个接点等等，其中任何一个环节的失效，都可能给系统带来严重的后果。

2．设置周密而合理的错误检测系统，以便及时发现错误。关键就在于要“及时”发现。要及时发现，就必须在系统的各部分设置必要的监视线路，以便随时捕捉任何逻辑错误。

3．建立故障诊断系统。对于系统中的固定性硬件故障，诊断系统的作用就是，能迅速地确定故障的位置，以便最快地排除故障，提高维修效率。

评价 RAS 技术效能的主要指标有：平均故障间隔时间 MTBF（Mean Time Between Failures）和平均修复时间 MTTR（Mean Time To Repair）。这里，

$$R = \mathrm{MTBF}$$

$$A = \frac{\mathrm{MTBF}}{\mathrm{MTBF}+\mathrm{MTTR}}$$

$$S = \mathrm{MTTR}$$

需要说明的是，上述三项定义，目前的看法并未完全统一。另外，要准确地测量上述

指标并不容易。一般有两种方法：一是事后的，即用统计推断的办法来求得MTBF和MTTR；另一种是预估的办法。有兴趣的读者可以参考有关资料。

1.2.2 容错技术

容错技术是实现系统能进行容错计算所采用的技术措施。所谓“容错计算”，目前比较公认的定义是：在计算机系统中出现故障的情况下，能正确地执行它的全部程序。

究竟是否正确地执行程序，可用下述四个标准来衡量：

1．程序不为故障所改变或停止；

2．结果数据不包含由故障所带来的错误；

3．程序执行的时间不超过规定的限度；

4．程序可用的存贮容量保持在一个指定的最小限度内。

因此可以说，容错技术要解决的问题，是计算机系统如何能“带病工作”的问题。它和RAS技术有着明鲜的区别。当然，这里并不排除，RAS技术和容错技术在具体设计上会有共同的要求。例如，错误检测系统，无论对RAS技术还是对容错技术，都是不可缺少的。

要使系统能带病工作，一般采取两个办法。一种是从系统中切除有病的功能部件，并且用没有病的其余功能部件替换，以重新组成一个系统，继续执行原有的任务。这种方法称之为降级使用或局部容错技术。这是因为，这个新组成的系统，比起未切除之前的系统来，其功能肯定是减弱了。不过，这种功能的减弱，还能最低限度地执行原有的任务。另一种办法是，在从系统中切除有病的功能部件的同时，加入一个同样的、完好的功能部件，以取代被切除的部件的工作。这时，整个系统功能丝毫不受影响。这种方法称为完全容错技术。特别要强调的是，这种切除或更换部件的过程，必须是在系统内部自动进行的，不允许有外界条件的干预，特别是不允许人来干预，否则，就会失去容错计算最根本的要求。

由此可见，容错技术必须包含下面的内容，才能真正使系统带病工作：

1．故障的自动检测和定位；

2．自修理的能力；

3．自恢复的能力。

必须指出的是，要完全实现上述三点，这是一件极复杂而困难的工作，无论在理论上或实践上都还有许多问题有待研究。

实现容错的基本技术是冗余技术。它包括硬件冗余、软件冗余和时间冗余。硬件和软件冗余就是设置备份硬、软件。硬件备份，可以是静态备份，也可以是动态备份。这种备份可以是系统级的，也可以是功能部件级的，还可以是逻辑元件级的。究竟采用什么级别，这要从系统容错的具体要求、性能、价格等方面通盘考虑来决定。所谓时间冗余，主要指运算的重复执行。这种重复执行，可以是整个程序的，也可以是一个程序段或一条指令的重复执行。

衡量容错技术效益的主要参数是：可靠性、残存运行率和可用性。有两种容错预测方法：一种是分析法，即通过建立容错系统的数学模型来预测；一种是实验法，即在模型机

或样机中，插入模拟故障，然后靠统计数据来估算容错度量。

总之，容错技术比之 RAS 技术，对错误检测系统要提出更严格的要求、更高的标准。

1.3 错误检测系统的某些功能

本节就某些方面的具体要求，叙述错误检测系统的某些功能。

1.3.1 保护用户

从保护用户的角度出发，错误检测系统应该具有以下功能：

1. 当发现错误时，应发出信号，以便通知用户：正在运行的程序的结果可能是错误的；

2. 在系统确实不能再继续运行下去的时候，及时告诉用户。

实践证明，无论对于科学计算、实时控制或商业数据处理的用户，上述两类保护功能都是很重要的。用户在使用计算机时，所关心的就是，最后的处理结果是否正确。如果一个计算机系统在运行过程中，发生了错误又不能给出信号，使用户对运算结果是否正确总是不能完全相信，那末，这显然是很不合适的。

在商业数据处理中，若涉及到资金流通、经济核算等重要问题，则上述第一类保护功能尤为重要。同样，诸如在铁路系统采用计算机进行自动调度等工作，这种保护功能也是极重要的。否则，可能造成极大的损失。

在用一台过程控制计算机监督控制某一过程时，更需要这种保护功能。这时，由于操作人员不能直接观察过程的进行情况，因此，如果计算机不能提供其是否正常运行的信息，那末就会使操作人员在判断过程是否处于正常运行时，遇到极大困难。如果计算机具有自动发现错误的能力，那末，采取其他的外部监视措施的必要性就会减少。

在计算机使用愈来愈普遍的趋势下，计算机会成为用户每天必不可少的工具。如果在计算失常几小时之后，才发现它工作不正常，则会给用户造成极大的浪费。若有一个适当的错误检测系统，则可以保护用户避免这些问题。

关于保护用户，还有一个附带问题是，系统在发现用户程序或数据的错误之后，能否给用户提供适当的信息，使用户能很快纠正错误数据，并且重新启动程序。一般来说，用户希望系统能指出哪些数据有错误及其实际值，并且想知道，当发现错误时，计算机正在执行哪一段程序。

从保护用户的功能出发，比较一下检测错误的程序技术和硬设备技术的特点是重要的。硬设备技术，不仅能完成程序技术所实现的功能，而且能完成后者所不能实现的功能。由于用硬设备检测的错误，较之程序技术更接近于错误发生点，所以，它能够提供有关故障的更准确的信息，而且还能发现一些程序技术所难以发现的偶然错误。由于这些优点，硬设备技术才得到广泛采用。

1.3.2 重新启动

在大多数计算机系统中，偶然错误在所发生的错误中占相当大的百分比。通常，系统在发生偶然错误之后，还是可以继续运行的。但是，对用户来说，由于这种偶然错误可能破

坏了他的程序或数据，因此，他必须在纠正或重新准备好他的程序和数据之后，才能重新启动系统。这样的中断，往往会给用户造成很大的麻烦。

可以设计这样一种系统，它能自动地重新启动被偶然错误所中断的操作。为此，它应满足以下几点要求：

1．在适当的时间间隔内，必须保存有关程序状态的信息，以作为程序重新启动的起点。

2．必须具备一些程序，使其能执行诸如信息的存贮和输出等功能。

3．必须有一个错误检测系统，使其能进行重新启动的过程。

4．还必须有一个控制系统，使其能把计算机的状态恢复到程序的启动点，并采取适当的纠正错误数据的步骤，然后启动系统。

自动的重新启动系统，对用户来说，具有极大的价值，它既可以消除偶然错误所引起的中断，也可以大大减少用户的时间浪费。尤其是对一些运行时间较长的题目，如果在运行即将结束时，发生系统的偶然错误，那末，这将使其前功尽弃。但是，若采用这种自动重新启动系统，则可以尽可能避免这种后果。

针对固定性错误，要设计一个自动重新启动系统也是可能的。但如果要求百分之百地覆盖固定性错误，则就比较复杂了。

自动重新启动对错误检测系统的基本要求是：

1．在重新启动点的数据仍然有效的期间内，要及时发现错误。

2．因错误而破坏了的数据，还没有超过重新启动系统所能纠正的数据之前，必须停止因错误而引起的任何动作。

3．要给重新启动控制系统提供足够的、有关错误性质的信息，使其能确定执行哪一个重新启动过程（如果有一个以上的可能性过程的话）。

举一个特例来说明一下。我们考虑这样一个自动重新启动系统，它保存的程序状态数据能够保证，并且，在错误发生于三条指令之内，就可以重新启动这一程序。显然，如果对于三条指令之内的错误，在执行了三条以上指令之后才被发现，则当然就不能重新启动了。上述要求的第一条，就是针对这种情况提出的。

为了重新构造被错误所破坏了的数据，重新启动系统，一般都包含有限个数的后援数据。如果由于错误的传播，破坏了更多的数据，以致使后援数据不足以重新构造被破坏了的数据，那末，也不可能自动重新启动程序。上述的第二条要求，就是以限定被损坏的数据的个数为基础提出的。

在很多情况下，要根据错误在计算机中发生的范围，重新启动控制系统才能选取不同的过程。为此，错误检测系统必须能够提供足够的信息，以确定这个范围。上述第三条要求，就是据此提出来的。

1.3.3 提高维护效率

通常，计算机系统的维护费用是很高的。这些费用主要消耗在培训人员、编写文件、提供各种维护仪器和工具等方面。事实上，系统愈庞大、复杂，所花的费用愈高。

对维护效率而言，排除机器故障的时间愈短愈好。这是因为，停机时间愈长，对用户

造成的损失愈大。在某些应用中，例如在一些实时控制系统中，过长的维修时间甚至是不允许的。

维护费用昂贵和维护时间过长，一般都是由于不能迅速而准确地判断故障的原因所引起的。如果针对维护而设置一个较好的错误检测系统，能给维护人员提供较多的信息，以帮助他们迅速分析、判断故障原因，那末，这不仅能大大缩短维护时间，而且也能降低对维护人员技术水平的过高要求。这样的错误检测系统应具有如下能力：

1．最好能指出故障所在的部位。当然，要精确地对故障定位，这是诊断系统应实现的功能。但是，错误检测系统应能显示出哪一部分设备失效。

2．在出错时，能保存机器的状态，以便维护人员利用这些状态信息，分析故障原因。

3．能针对一些特殊错误，启动自动诊断程序运行。

如果在系统的设计中，从可更换部件（如插件）的逻辑划分，到错误检测系统及诊断系统的设计，都作了通盘的考虑，那末，就有可能做到：一旦系统发现错误，就能告知维护人员或操作人员应该更换哪些部件。

还要特别指出的是，如果没有错误检测设备，则要诊断偶然错误是很困难的。

在这里，我们将讨论一个简单的例子。从这个例子中可以体会到，如果使逻辑划分、错误检测及诊断系统三者之间合理地协调，那末，就可以使维护工作得到最佳的效果。下面我们就来分析一下，信息从一个寄存器传送到另一个寄存器的问题。实践证明，可以采用几种不同的方法，对信息的传送进行校验。方法之一就是，在发送寄存器被清除之前就完成校验，甚至可以用错误信号，禁止清除发送寄存器和接收寄存器。这时，根据这两个寄存器的内容，维护人员就可以很容易地判定在传送中哪几位发生了错误。如果逻辑划分能够保证，使给出错误信号的检测线路所覆盖的逻辑线路处在同一个可更换的部件内，那末，就会给故障的诊断提供很大的方便。这是一个很重要的设计思想。假如在设计中，把所有的错误信号都“或”在一起，而最后所给出的错误指示仅仅只有一个，那末，这对分析故障是很不利的。

为了提高维护效率，在设计错误检测线路时，还应考虑下列功能：

1．在错误被发现之前，不能让它在系统中传播得太远。这涉及到错误检测线路的合理设置问题。这个问题在第五章中将有所讨论。

2．在错误被发现之后，应尽快停止机器操作，以便于利用寄存器所保持的现场来分析故障范围。进一步还应设法在停机之前，能把错误信息保存在一个适当的存贮装置中。

在采用集成电路的系统中，大量的逻辑线路已装在一块插件上。如果把检测线路也装在这块插件上，甚至在每块插件上装一个标志，那末，这就为维修更换部件提供了很大的方便。

总之，用硬设备实现错误检测，只要设计得合理，就可以有效地提高维护效率。

1.3.4 部件的重新组合

所谓部件的重新组合是指：把有故障的逻辑部件从系统中切除掉，并用同样的备份部件代替它，或者降级使用系统。这种使系统重新恢复工作的过程，就称为部件的重新组合。这是一种获得高可靠性的基本方法。在前节中已经提到过，实现这样的系统的先决条件就

是，要有一个适当的错误检测系统，而且这种检测系统应当满足如下条件：

1．发现错误之后，应给出信号，以便能进行重新组合的处理，并能重新启动被中断的程序。

2．检测系统能很容易地确定哪一个逻辑部件应该被切除。通常，对于两个逻辑部件交界面处的错误是很难区分的。因此，在设计中要特别注意这种情况。为了重新组合，就必须把这种界面缩小到最低限度。面留在界面中的逻辑则应设计成使诊断程序能较容易地确定有错误的硬件属于哪一个逻辑部件。为了能进行部件切除，错误检测系统必须有能力把错误隔离在系统的可切除的部件之内。

第二章　数学基础

本章介绍几个数学概念。这些概念在分析逻辑错误时是十分有用的。

在这里，主要讨论布尔差（亦称布尔差分）的概念及其应用。读者必须具有布尔代数的基本知识。关于布尔代数，读者可以在许多著作中，特别是在有关数字计算机原理、逻辑设计等书籍中找到，因此，本书就完全略去了。

2.1　布尔差的概念

设逻辑网络 L 有 n 个输入 $x_1, x_2, \cdots, x_n$；其输出为 F（如图 2.1 所示）。其中，x_i（$i = 1, 2, \cdots, n$）取值 0 或 1，称为布尔变量；F 称为 x_i 的布尔函数或逻辑函数，记为

$$F(x) = F(x_1, x_2, \cdots, x_n)$$

如果其中有一个变量 x_i 发生了错误，即 x_i 变成了 $\bar{x}_i$，则此时的布尔函数就变为

$$F(x) = F(x_1, x_2, \cdots, \bar{x}_i, \cdots, x_n)$$

为了分析 x_i 变为 $\bar{x}_i$ 时，$F(x)$ 究竟有什么变化，这种变化发生的条件是什么，我们定义如下函数。

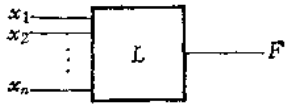

图2.1　逻辑网络 L

定义 2.1

$$\text{令} \qquad \frac{dF(x)}{dx_i} = F(x_1, x_2, \cdots, x_i \cdots, x_n) \veebar F(x_1, x_2, \cdots, \bar{x}_i, \cdots, x_n) \tag{2.1}$$

式中，函数 $\frac{dF(x)}{dx_i}$ 称为 $F(x)$ 对变量 x_i 的布尔差（或称布尔差分）；符号 $\veebar$ 表示“异或”运算，也称为“半加”运算。必须注意，符号 $\frac{d}{dx_i}$ 不是导数，而是一个布尔差算子。其所以采用这个符号，是因为它应用起来比较方便。

定义 2.2

令　当 $x_i = 1$ 时，　$F_i^1(x) = F(x_1, \cdots, 1, \cdots, x_n)$

及当 $x_i = 0$ 时，$F_i^0(x) = F(x_1, \cdots, 0, \cdots, x_n')$，则

$$\frac{dF(x)}{dx_i} = F_i^1(x) \veebar F_i^0(x)$$

定义 2.1 和定义 2.2 是等价的。证明如下：

$$\begin{aligned}
\frac{dF(x)}{dx_i} &= F(x_1, \cdots, x_i, \cdots, x_n) \veebar F(x_1, \cdots, \bar{x}_i, \cdots, x_n) \\
&= x_i F(x_1, \cdots, 1, \cdots, x_n) \veebar \bar{x}_i F(x_1, \cdots, 0, \cdots, x_n) \veebar \bar{x}_i F(x_1, \cdots, 1, \cdots, x_n) \\
&\veebar x_i F(x_1, \cdots, 0, \cdots, x_n) = (x_i \veebar \bar{x}_i) F(x_1, \cdots, 1, \cdots, x_n) \\
&\veebar (x_i \veebar \bar{x}_i) F(x_1, \cdots, 0, \cdots, x_n) = F(x_1, \cdots, 1, \cdots, x_n) \veebar F(x_1, \cdots, 0, \cdots, x_n)
\end{aligned}$$

根据上述定义，还可以推导出布尔差的一般表示式。假定

$$F(x)=F_1(x)\cdots F_t(x)G_1(x)\cdots G_s(x)$$

并且 $F_1(x),\cdots,F_t(x)$ 与 x_i 无关，则

$$\frac{dF(x)}{dx_i}=\frac{d}{dx_i}[F_1(x)\cdots F_t(x)G_1(x)\cdots G_s(x)]$$

$$=\prod_{r=1}^{t}F_r(x)\left(\prod_{q=1}^{s}G_q^1(x)+\prod_{q=1}^{s}G_q^0(x)\right)\sum_{q=1}^{s}\frac{dG_q(x)}{dx_i}$$

假定 $F(x)=F_1(x)+F_2(x)+\cdots+F_t(x)+G_1(x)+G_2(x)+\cdots+G_s(x)$

并且 $F_1(x),\ F_2(x),\cdots,F_t(x)$ 与 x_i 无关，则

$$\frac{dF(x)}{dx_i}=\frac{d}{dx_i}[F_1(x)+F_2(x)+\cdots+F_t(x)+G_1(x)+G_2(x)+\cdots+G_s(x)]$$

$$=\prod_{r=1}^{t}\overline{F_r(x)}\left(\prod_{q=1}^{s}\overline{G_q^1(x)}+\prod_{q=1}^{s}\overline{G_q^0(x)}\right)\sum_{q=1}^{s}\frac{dG_q(x)}{dx_i}$$

根据上述定义，还可以推导出如下的一些布尔差的重要运算特性：

$$\frac{dF(x)}{dx_i}=\frac{dF(x)}{d\bar{x}_i} \tag{2.2}$$

$$\frac{dF(x)}{dx_i}=\frac{d\overline{F(x)}}{dx_i} \tag{2.3}$$

$$\frac{d}{dx_i}\left(\frac{dF(x)}{d\bar{x}_i}\right)=\frac{d}{dx_i}\left(\frac{dF(x)}{dx_i}\right) \tag{2.4}$$

$$\frac{d[F(x)G(x)]}{dx_i}=F(x)\frac{dG(x)}{dx_i}\veebar G(x)\frac{dF(x)}{dx_i}\veebar\frac{dF(x)}{dx_i}\ \frac{dG(x)}{dx_i} \tag{2.5}$$

$$\frac{d[F(x)+G(x)]}{dx_i}=\overline{F(x)}\frac{dG(x)}{dx_i}\veebar\overline{G(x)}\frac{dF(x)}{dx_i}\veebar\frac{dF(x)}{dx_i}\ \frac{dG(x)}{dx_i} \tag{2.6}$$

$$\frac{d[F(x)\veebar G(x)]}{dx_i}=\frac{dF(x)}{dx_i}\veebar\frac{dG(x)}{dx_i} \tag{2.7}$$

由于一些读者可能不太熟悉“异或”运算的特性，因此，下面未加证明地列出了它的一些最常用的性质：

$$F\veebar G=F\overline{G}+\overline{F}G \tag{2.8}$$

$$F\veebar G=G\veebar F \tag{2.9}$$

$$F\veebar F=0 \tag{2.10}$$

$$F\veebar 1=\overline{F} \tag{2.11}$$

$$FG\veebar FH=F(G\veebar H) \tag{2.12}$$

$$F+G=F\veebar G\veebar FG \tag{2.13}$$

$$F\veebar\overline{G}=F\veebar G=F\veebar\overline{G} \tag{2.14}$$

$$F\veebar\overline{F}=1 \tag{2.15}$$

如果 $F\veebar G=H$，则

$$H\veebar G=F$$

或

$$H\veebar F=G \tag{2.16}$$