国外计算机设计自动化技术

复旦大学、上海交通大学、一九三二所
上海科学技术情报研究所等编

上海科学技术文献出版社

国外计算机设计自动化技术

复旦大学、上海交通大学、一九三二所

上海科学技术情报研究所等编

*

上海科学技术文献出版社出版

新华书店上海发行所发行

上海市印刷三厂印刷

*

开本：787×1092 1/16　印张：4　字数：101,000

1979年10月第1版　1979年10月第1次印刷

数：1—25,400

书号：15192·40　定价：0.55元

《科技新书目》130—88

前　言

计算机辅助设计(CAD)和设计自动化（DA）技术目前在国外几乎遍及电子、航空、造船、化工、冶金、建筑等各行业的设计工作中。计算机本身的设计自动化问题早在一九五六年就首先被提了出来。目前设计自动化已成为计算机科学中一门独立的学科，在国外再也找不到一个计算机制造者不采用设计自动化系统。据统计使用DA系统后设计时间平均缩短四分之三，设计费用减少一半左右。

为了促进国内计算机设计自动化工作的开展，努力赶超世界先进水平，我们组织了复旦大学、上海交通大学、一九三二所、上海科技大学、上海师范大学、上海计算所、上海科技情报研究所等单位作了一次全面的国外资料调查，并以报告会形式向上海市有关单位科技人员作了介绍。

现将这次调查内容编写成动态性技术资料。系统地介绍国外在这一领域的历史、现状和最近动向。供从事计算机科研设计人员、大专院校有关专业师生参考。由陈增荣、宋云麟、蔡鸿良、张根度、彭澄廉、仲　毅、徐拾义、何积丰、孙季伦整理改编。

由于我们水平有限，时间紧迫，更缺乏实践经验，一定存在不少问题，望读者批评指正。

编　者

一九七八年六月

目　　录

一、引　论

由于电子计算机运算速度高、处理能力强。目前国外在各种设计工作中广泛采用计算机辅助设计和设计自动化技术，以代替大量重复的手工工作，提高设计效率。

早在五十年代中期，晶体管计算机的出现大大缩小了计算机的体积，可靠性大幅度提高。接踵而来的军用计算机的订货合同迫使计算机制造商寻找设计计算机的新途径。

设计自动化方面的最早文献是一九五六年由 Cray 和 Kisch [1] 在美国西部联合计算机会议上提出的。他们描述了一个含有三个阶段的程序。第一阶段是逻辑方程的检查。目的是检查逻辑的、书写的和时序的错误，进一步还计算每个线路的扇入和扇出。第二阶段是模拟。设计师可以从被模拟的开关底板置入布尔方程的输入值，观察所得的结果。最后在第三阶段里，设计师把他的设计组装成部件，并对机架间的线路连结进行计算，造好导线的原点和终点表格，算出导线的颜色和长度。

接着，有人设想能否叫计算机产生一些布线机所能遵循的指令，直接穿在卡片上？Gardner-Denver 工具公司研制的一台卡片控制布线机，在五十年代末期曾被广泛使用[2]。它不但提高了布线速度，错误也大大减少了。

由于自动布线机的生产效率很高，如果逻辑设计中的错误发现得慢，就可能会引起大量的返工和浪费。如果对逻辑先进行模拟，那么浪费和返工将大大减少。于是逻辑模拟被广泛使用。

随着计算机系统越来越大，模拟所需的人工准备工作量大大增加，而且要模拟整个系统也遇到了困难。另一方面设计师也希望在门级实现之前，就能模拟整个系统，以证实系统设计是否正确。这就需要有较高级的语言。由于语言具有在较高级上描述数字系统的能力，现在它正变成设计、模拟、交流、文件化等方面的基本工具。硬件描述语言现已成为计算机设计师必须掌握的一种基本知识。

从六十年代后期开始，设计自动化发展非常迅速。它已从个别设计领域扩展到多种领域，包括系统的设计和模拟，微程序的设计和模拟，硬件描述语言及其翻译，自动逻辑综合，逻辑模拟，系统的分划，LSI 和印刷板的布局、走线，整机和印刷板的诊断设计，文件设计等。系统的管理也从个别的串行执行程序发展为以数据库和数据管理系统为中心的大型系统。从运行方式来看，原来较多地使用成批处理方式。现在由于对设计的要求越来越高，有许多问题又不能全自动化地解决，人机对话的交互设计使用得越来越广泛，它允许设计师在任何阶段进行干预。

目前设计自动化已成为计算机科学中一门独立的学科，有专门的研究会，每年召开专门的国际性会议。在国外再也找不到一个计算机系统的制造者，他的生产装备不采用设计自动化系统。而且设计自动化系统已经开始作为商品在市场出售（CII-HB公司，它是美国和法国的三家公司联合组成的计算机公司，现在已有DA系统产品）。

(一)DA 设计的流程

使用 DA 系统的设计一般与对象的规模和使用的工艺有关，但通常取图 1 所示的流程。

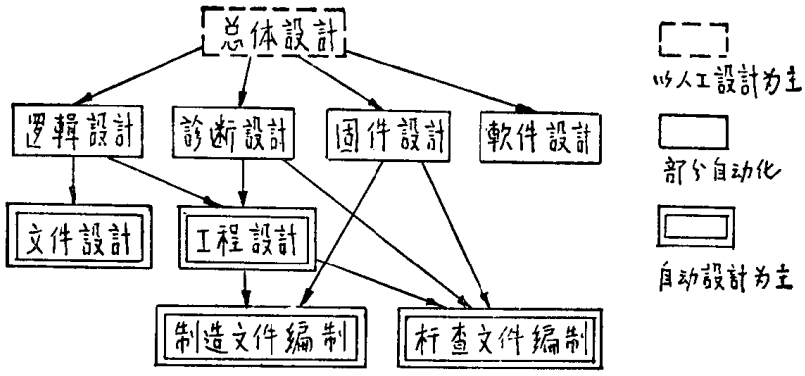

图 1　使用 DA 系统的设计流程

总体设计部分是决定计算机的体系结构、处理速度和价格。目前主要靠人工进行设计，有时为了验证总体结构是否正确往往采用系统级模拟。如系统模拟曾有效地用于先行存贮器[3]和高速缓冲存贮器[4]的方案论证。例如对 IBM360/85 机的缓冲存贮器的性能评价和设计，选取了应用领域有代表性的 26 个作业，将 1.6 亿条指令制成 714 根扫示纸带。其中 19 根有代表性的扫示纸带的命中率在 92%到 99%之间，平均命中率为 97%[5]。

总体设计完成后便进入逻辑设计、固件设计(包括微程序设计)，诊断设计和软件设计。目前这些工作也是以人工设计为主，但部分处理已自动化了。逻辑设计阶段主要是产生决定各指令功能的硬件逻辑图。人们期望能用硬件描述语言描述系统，作为输入直接进行自动逻辑综合。尽管这方面已做了大量工作，但由于它比人工设计需要更多的元件，至今尚未实用。如 IBM 公司的 ALERT (自动翻译系统)，它经过八个阶段的处理[6]，把 APL 语言变换为 DA 系统的输入，对 IBM 1800 的主要部分进行了逻辑设计。结果表明由 ALERT 产生的设计所用的逻辑门数量要比实际的 IBM 1800多 160%，他们正在对 ALERT 进行改进，企图达到仅比实际设计多 33%的目标。也有人认为如果今后进行 LSI 化，使门的价格进一步降低，这种技术将会引起人们进一步的重视。目前逻辑设计阶段主要使用逻辑模拟，检验人工设计的逻辑是否正确。

固件设计在计算机研制中所占的比例正在逐年增加。其主要原因是：随着控制存贮器的高速化和价格降低，出现了采用多重微控制部件的大型计算机；操作系统的固件化；仿真机构和微诊断的采用；外围设备控制器的固件化；微处理机的大量应用。这些因素使固件设计的任务大大增加。现在已经有一些成功的固件设计自动化系统。它是从微程序设计到制造、检查资料完成的连贯的系统。其中包括高级语言（或汇编语言）的描述，固存地址的自动分配，根据频率统计决定最佳机器周期，微程序模拟，自动编制 ROM 写入纸带和 ROM 检查纸带等功能。

诊断设计的任务是生成一组测试码序列。主要用于调试和维修，进行故障定位检测。

在逻辑设计完成后就进入工程设计(也称组装设计)阶段以及制造和检查资料的编制。工程设计就是把实现逻辑功能的具体硬件组织到一个个组装单位中去。由于问题的复杂性，总是分几个阶段处理的。首先把整个逻辑分成印刷线路板和 LSI 片单位，这是分划问题。然后把门分配到 IC 组件上。这是分配过程，有时也称为选择问题。分划分配问题和下面的布局走线问题都可以对应好多个组装级。它们都是讨论相邻两个组装级之间的某种关系。在分配过程结束后就进入布局处理。即确定低级组件在高级组件上的位置，如印刷线路板

在底板上的位置，组件在印刷板上的位置，门在组件上的位置等。布局处理的最终目标是使走线方便或走线总长度极小。工程设计的最后一个环节是走线，即确定组件间、印刷板之间的具体连结。这是整个计算机设计过程中最困难的步骤之一。工程设计及制造检查资料的编制最早成为设计自动化的对象。许多制造厂家正在广泛使用。近年来又以数据库为中心组成集中化系统，而且有逐渐将逻辑设计、固件设计和诊断设计也组织到这综合系统中去的倾向。

在逻辑设计完成后还要进行文件的设计和绘制。这个课题近年来逐步引起了注意。

软件设计自动化是当前研究的重点之一。

(二)DA 系统的组成

集中化设计自动化系统一般由输入、数据库、处理和输出四部分组成[7]。

1. DA 系统的输入

DA 系统的输入形式按要求不同大致可分为逻辑卡片、逻辑图、草图三种形式：

1) 逻辑卡片。这种形式描述电路的逻辑连结。它又分为三种不同的表示方法：逻辑式方式是用表达式表示逻辑功能；元件形式是以基本逻辑元件为单位；网形式是以网（信号线）为单位。由于元件形式具有与逻辑图一一对应的优点，它用得较多。为了减少输入信息量和提高输入可靠性，也有用汇编语言描述宏功能和按功能块输入的系统[8]，以及把逻辑信息和组装信息用设计语言形式输入的系统[9]。

2) 逻辑电路图。它是一种用数字输入器直接读取逻辑电路图，输入到设计自动化系统的方式。输入后图形的构筑、放大、缩小等是靠软件支援的。这种输入方式的优点是图形信息可脱机变换录入磁带，可以减少人工将图形译成逻辑说明时的错误而且修改逻辑图比较容易。数字输入器在国外已作产品生产[10]。现在已有这样的系统，它能够把存入数据库的图形信息自动绘制出与组装信息一致的电路图。

3) 草图形式。常用于印刷板设计。支援软件将草图变换为正规图，并产生制造用纸带。使用这种方法的目的是解决布线不能达到100% 的矛盾。

2. 数据库

最近的设计自动化系统大多以数据库为中心。它通常由本质不同的二种文件组成，即基本信息文件和设计信息文件。前者存贮了设计过程中作为基准的数据。它实际上是一类公共文件。设计信息文件存贮记录各种不同设计对象的信息。以印刷板为例，公共文件包括可使用元件的信息，电路使用规则，板的形状等；设计信息文件则包括板的布局、连结信息和图形信息。

3. 处理内容

在设计自动化系统中处理的内容有多有少。从目前来看，多数实用系统只包括组装设计和设计验证。组装设计通常包括分划、布局、布线等，而设计验证则包括逻辑模拟、电路规则检测、电路延迟时间计算、印刷板设计规则检验等。

4. DA 系统的输出

DA 系统的输出包括两个方面：把设计内容回授给设计人员的资料和制造检查中使用的资料。其中有部分资料是设计和制造都需要的。进而还有产品出厂后维护用的资料。

设计用的代表性资料有：引脚分类明细表，逻辑分类明细表，设计规则检查目录，逻辑电路图，操作时间表和印刷板的版图等。制造检查资料包括：自动画图纸带，钻孔纸带，

自动布线纸带，布线试验纸带和功能试验纸带等。

输出形式有：缩微胶卷图形输出、磁带、纸带、软盘以及图形显示等。

(三)采用 DA 系统的效果

目前在国外差不多各个计算机公司都拥有自己的 DA 系统。由于各公司设计的计算机的性能、规模及组装技术不同，要对应用 DA 系统的效果作出全面精确的评价是困难的。但是 DA 系统的经济技术效果确是相当明显的。

1．缩短设计周期、降低设计成本

美国 GET 公司中 UNIVAC 的一个多层板自动画稿系统。板子有 192 个片子，4 只插座，每只插座 74 芯，尺寸为 11×11½ 英寸。十层板，四层布信号线，三层地线，二层电源线，一层过渡层。其中信号线层平均布线时间是 7 分钟。该公司另一个通用 MOS 电路自动布线程序专为顾客进行 MOS 电路设计。人工设计一个非常复杂的电路费用往往超过一万美元，时间要几个月，最后的设计还很可能存在错误。使用自动布线程序的设计费只要一百美元，时间少于一周，而且产生的版图没有错误。该公司 FANSSIN 逻辑模拟程序对 600 个元件、二万个信号的模拟费为 2.30 元，时间只要 30 秒。日本富士通的 ESS 系统[11]，共分七个子系统。其中数据管理系统对二万个门的文件从创建到设计完成只花 80 分钟，另外，40 秒钟就可以打印一张 A_2 尺寸的逻辑电路图纸。CII-HB 公司的 DA 系统包括逻辑主文件、微程序生成、测试诊断，印刷线路板设计，绘制逻辑图等功能，共 56 万 8 千条指令。研制该系统的人员有 100 人，后留下 35～40 人改进系统。使用此系统后设计成本降低，设计时间为人工的⅓，而且设计错误大大减少。随着计算机系统的飞速发展，缩短设计周期具有十分重要的意义。

2．提高设计人员的创造性

由于计算机替代了人的大量而又繁重的劳动，设计师就有可能把精力集中在那些有意义的工作上。由于设计周期的大幅度缩短，可以对好几个系统进行比较，选择最佳系统。

此外设计内容的可靠性大为提高。DA 系统还能自动编制各种制造、检查资料，而且各种资料之间规格统一，便于管理和标准化。

有人统计，使用 DA 后平均设计时间仅为原来的¼，设计费用减少一半。如果考虑因人工设计出错修改的费用以及制造自动化节省的费用，DA 系统的效果就更为显著。

去年 IIASA（国际应用系统分析学会）作了一次世界范围的调查，包括东欧、西欧、美国和日本等十六个国家[12]，85 个组织，访问了 250 人。调查指出早在 1972 年就认识到 CAD 作为正在发展中的新技术，很可能会对工业的发展产生重大影响，必须投入更多更高级的科技人员到这一领域中去。这应当引起我们的重视。

二、计算机描述语言

计算机描述语言现已成为每个计算机设计师不可缺少的基本知识。在国外，如果一个计算机设计师不懂得 RTL，那简直不可理解。

任何一门科学需要抽象，需要使用符号表示这种抽象。因为这样能更简炼、更清楚、

更精确地表现事物的本质。

计算机设计的抽象最早可以追溯到1938年Shannon关于开关电路的工作。当时他发现布尔方程能以理想化的状态表示数字系统，它能揭示逻辑行为。这时电压的变化不再看成是连续的，它已被抽象为0、1两种状态。1940年又出现了逻辑图。随着计算机复杂性的增长，出现了许多方法。如块图、定时图、次序图、流程图等，以反映系统的总体特征并描述它如何运行。尽管每种记号都能有效地表示设计，但它们都缺乏标准化、简明性，总需要用自然语言的冗长描述来扩充这些文件。由于使用不同的记号妨碍了人们之间的通讯和理解。一个设计师为了搞清他人设计的系统必须花上大量时间，尽管他是熟悉这些设计方法的。计算机的复杂化使这个问题愈益严重。

Reed很早就认识到这个问题，他开始探索是否有更好的抽象方法。1950年他发现了类似于寄存器传输的触发器输入方程，这些平行的方程合起来能用一个向量方程表示。它可以看成是计算机设计语言的雏型。描述计算机的形式语言理论在最近二十年中发展很快，兴趣也在不断增长，还专门召开了这方面的国际会议。特别是最近十年，研制了许多优秀语言(如AHPL[15]，CDL[16]，DDL[17]，ISP[18]，PMS[18])，用以描述计算机的系统结构和实现指令的算法。

使用计算机设计语言的优越性是很多的，主要有以下五个方面：

(1)作为形式语言它能严格而简明地描述计算机。已有许多语言可用于从系统级直到门级的描述。特别是系统级语言以非常抽象的形式反映出系统最本质的部分。

(2)因为有上述优点，所以可作为一种很好的文件。查阅它比直接查阅图纸方便得多。

(3)由于语言描述的精确性，它为系统级模拟提供了方便的手段。在这之前人们只能进行门级模拟，而现在在门级实现之前就可以评价系统的性能。

(4)由于语言提供了系统描述、文件、模拟等各方面的方便性，它在设计自动化中是很有用的。它为从系统结构设计直到连结表的产生整个过程的自动实现提供了手段。

(5)最重要的是语言已作为通讯的工具。它可用于工程师之间的通讯，如总设计师与各分系统设计师之间的协调。它也可用于计算机之间、计算机各分系统之间的通讯。语言已成为学习、交流的极好形式。有了设计语言模拟器后，一个或几个学生就可以用语言"构造"一个复杂的多处理机。

现有的语言很多，据查阅已发表的语言在六十种以上。

(一)RTL的主要特点

寄存器传输语言(RTL)关心的是抽象，或称为模型化。它有助于对复杂系统的理解。抽象就是关心类似性，而把差别作为无关的细节放在一边。抽象的级越高，就有越多的细节作为无关事情而丢弃，注意力集中在系统的总行为上。例如在中央处理部件的描述中关心的只是：指令系统；每条指令实现的算法；系统寄存器，数据通道，处理部件等等，以及它们之间的连结。

RTL一般包含两类语句：说明语句和运算语句。说明语句揭示计算机的结构，运算语句则指出其功能和算法。

以下以DDL语言为例说明RTL的一般特点。

1. 标识符

各种逻辑硬件如触发器、时钟、硬件的装配(如几个寄存器合成一个寄存器)，逻辑单

元之间的连结、系统等等，数学上抽象为RTL中的名称或标识符。通常定义为

〈ID〉::=〈字母〉|〈ID〉〈字母〉|〈ID〉〈数字〉

于是A，PQ12等是标识符，而1D7则不是。

数组　寄存器，一组类似的连结，寄存器之间的传输或运算，存贮器等都有一定特点，就是硬件的许多项类似。于是可将整个集合取为一个标识符，而其各项用非负整数下标表示。为穿孔打印方便起见，下标放在同一行中加上括弧。通常是方括弧。语法上是〈ID〉[〈非负整数〉]。

具有连续下标的一些项可以简写为〈ID〉[〈非负整数〉:〈非负整数〉]。

有时还使用二维数组或多维数组，如

A[i1:i2, j1:j2]。

二值串　为了描写寄存器的状态或模拟的初始激励，需要给出一串逻辑0或1。DDL中形如nRk, n为二值串，R为类型，k为串的十进长度，如010110可记成10110B6，26O6，22D6等形式。其中B表示二进制，O表示八进制，D表示十进制。

2. 简单的设备说明

通常计算机设计语言要求类型一定要说明。设备说明分简单的和复杂的两种。简单设备说明大致有五种：

端点说明　描述任何一个计算机必须给出连结。端点说明就是为连结运算作准备的。它允许说明多次。如

〈TE〉 E[16]=ESIGN∘EMAG[14:0]

这里E_1，E_2，…，E_{16}说明了两次，另一次在等号右面。其中∘是标识符的连结。于是E等价于ESIGN，E_2等价于$EMAG_{14}$，…，E_{16}等价于EMAG。

记忆单元说明　寄存器通常是触发器的一维阵列，而存贮器则是多维阵列。有些语言对它们不加区分。有些则不然，对存贮器还要加上存取周期、读出时间等参数。为了便于寄存器的装配，允许双重说明，如

〈RE〉 IR[24]=OP[0:5]∘IX[3:1]∘ADR[15]

指出了OP，IX，ADR装配成IR的方式。

时间和延迟说明　事件的定时问题是逻辑设计的中心问题之一。所以任何RTL都必须提供定时的手段。时间说明指出周期波形发生器的存在。延迟说明指出延迟单元的存在。后者常用来说明门延迟，RC网络延迟等。

布尔说明　RTL中描述组合逻辑网络通常采用布尔表达式。布尔说明指出了输入端点与输出端点间的布尔关系，具体的电路实现用自动逻辑综合算法得到。

单元功能说明　在系统设计中有些逻辑部分未知或不关心。在描述系统的结构时这些部分要作为“黑箱”给出输入输出端点的定时与功能特征。

3. 运算语句

通常RTL中有布尔表达式、条件控制、连结和寄存器传输、移位和计数等运算。大多数RTL用条件语句来实现控制。如if C then A←—B就等价于A←—C·BV ¬C·A。

“控制”部件的中心通常是一个寄存器，从它引出的信号组合用来控制运算的执行。有限状态机理论将这个寄存器的触发器状态集合称为机器的状态。当控制处于特定状态时所有要执行的运算分类为同时执行的运算群。状态的名称就是这些运算的执行条件。当然也

许还会有其它的附加条件。

4．复杂的设备说明

包含设备说明和运算语句的设备说明称为复杂的设备说明。如运算器说明、状态说明、自动机说明等。

总而言之，RTL是{一组符号；一组规则；一组语义}。符号就是字母表，规则构成句法，语义则将RTL的语句与计算机的结构对应。

(二)如何设计一个语言

1．确定语言的基本要求

计算机模型化的基本要求 设计语言是为了描述对象。一个好的计算机设计语言应该能方便、简明、精确地描述极大多数计算机。这取决于如何将计算机模型化。模型的好坏基本上确定了语言的性能。所以模型化这一步是十分重要的。可以说语言就是模型化的工具。

计算机的模型化可有种种方式 最简单的方式是用一组方程来描述，相当于上面所说的布尔说明。它将输入映照到输出。通常输入变量是时间的函数。如果只有这种说明手段，定时信息则必须包含在方程中。这类描述对小的系统是有用的。然而当系统复杂庞大时，描述系统所需要的方程太多了。这类逻辑方程描述本质上是系统的结构描述，而时间的变化反映出系统控制的要求。将时间放在结构描述中的失败，使我们认识到控制部分应该与结构部分分开。

第二种模型化方法是使用微程序。在这类模型中基本的系统操作（即微命令）已经定义。这些微命令也许是寄存器的移位、寄存器之间的传输，或是对系统中的某些变量执行逻辑运算。微命令的组合构成微程序，它执行更为复杂的系统功能。这类模型在固件自动设计中是很有用的，它本质上是描述系统的行为或控制。系统的结构则隐含在微命令中。这就使详细地描述系统的结构变得困难。

第三种模型化方法是用算法语言来描述系统的功能。在这类模型中，一个复杂的多重操作可以定义为基本的功能模块。因此这种模型化方法适宜于在较高的级上来描述系统。但这同时也意味着，每个功能模块的结构细节是不清楚的。

早期的设计语言是简单地将第一、三种模型化方法的描述能力加到微程序的模型化方法上。由于结构与控制不分开，产生两个问题。一是二义性，二是描述语言复杂。这就要求研究统一的模型化方法，它能分别描述结构和控制部分，又允许功能描述，同时它还要满足当代计算机的其他要求。

已有语言的优缺点分析 方便、简明、精确是设计语言的重要而又相互制约的三个要求。设计一个好的语言是很困难的。一个行之有效的办法是对现有的语言进行分析，找出一个合理的方案。

IEEE会员Schorr在1964年提出了一个语言[19]，他将计算机描述为寄存器之间的传输。语言的结构不复杂，对简单的系统使用得相当满意。但因不能方便地将系统划分为子系统，描述大的系统就复杂了。CDL[16]基本上是第三种模型化方法，在描述行为时是满意的，但缺乏系统结构和控制的细节，用于自动逻辑设计是不方便的。DDL[17]能满足大多数需要，但有点复杂，目标程序长。且DDL描述的翻译需对整个系统多遍扫描。在分时计算机中，这是不希望的。因为对系统作出修改时，整个系统要重新翻译。

在评价了若干种主要语言后，就可以给出所需语言的设计目标。

确定要设计语言的目标　通常现代好的语言必须具有下述的基本特点：简洁、无二义性；允许分别定义系统的结构和控制；由于工业中倾向于使用组件，必须允许以逻辑方程形式或功能形式来描述这些组件；允许方便地将系统分解为子系统，即各个部分可以独立设计，最后很容易合起来构成完整的系统；能够直接描述任意复杂的并行控制操作；语言应该简单，但允许宏定义(Macro)的方便性，以构造复杂运算。

2．确定系统模型

设计目标确定后需要确定计算机模型化的细节。不同的语言由于要强调的问题不同，这些细节是不一样的。从大的方面来讲 AHPL 就将整个系统分为控制部分、数据寄存器和逻辑部分。从小的来看，寄存器级的语言要满足 RTL 所述的特点。这个过程实际上也是语言的设计过程，它要考虑到语言如何实现有关的模型化细节。

(三)计算机描述语言的分类

计算机描述语言有多种分类方法。

按设计对象分有硬件描述语言、固件(微程序)描述语言、软件（如操作系统）描述语言、工程(组装)描述语言、总体(系统)描述语言等。

按设计师所关心的级分类有总体级、程序级(为固件和系统软件作准备)、指令系统级、算法级（建立实现指令系统的控制算法）、寄存器级、门级、电路级（电子设计）、制造级(即工程组装)。通常一个语言应能适用于若干个级。

按源程序的结构分类有面向激发语言、面向事件语言和面向进程的语言。

激发是最小的工作单元，通过“激发”的执行完成了工作。它可以对应指令中特殊的一拍，也可以在较高的级上对应一件完整作业的执行。“激发”所引起的工作的完成就是“事件”。一组相互有关的激发构成一个进程。

面向激发的语言是要描述一组激发。每个激发的描述包括：激发执行的条件；它所激发的运算及其执行时间。CSL[20]就是这类语言。

面向事件的语言是将运算独立出去，变成事件子程序。激发的描述则采用表结构，指出激发的执行时间和事件子程序的名称。执行的次序反映在事件表的编排中。这类语言有 GASP[21]，SMPL[22]。

面向进程的语言如 ASPOL[23]，SIMULA[24]，SOL[25]等。它们要求模拟程序按一组进程来组织。进程描述定义了这类进程的属性和激发。它采用子程序或过程的形式。这里的过程是多重进入的。这是因为进程执行到某点时，有些激发的执行条件尚未产生，这时必须退出，待条件满足时再从这里继续做下去。

(四)国外的语言概况

计算机描述语言的研究和使用方面最活跃的是美国，其次是日本。这两个国家已提出了许多语言，计有五十种以上。其他国家如加拿大、英国、法国、西德、意大利等也作了不少的工作。不少大学、公司也有自己专用的语言。如法国 CII 公司研制了微程序描述语言 COSEQ 和用于下一事件三值模拟的 SISEQ 语言。在他们出售的 DA 系统中就使用这两种语言。加拿大在这方面也是比较先进的。CDL 的 IBM360 程序系统最早是 1969 年加拿大的 Toronto 大学实现的。加拿大的 Montreal 大学也研制了 DIGISM 模拟语言。加拿大的 Carleton 大学将 DDL 的子集搞了一个语言。关于国外语言的详细情况请参阅〔26—32〕。

三、计算机描述语言的翻译

计算机程序以某种形式接受输入符号串，应用一遍或多遍翻译将此串变换为输出串，这个程序就称为翻译器。计算机程序语言的一些翻译器称为编译器，它们接受以抽象语言写的程序(如 Fortran, ALGOL, COBOL 等)并翻译为目标计算机的机器语言或符号指令串。编译器方便了计算机的使用，简化了它们的程序，提高了专家的效率。于是计算机能广泛地用于教学、研究、工业辅助等各个领域。

类似的翻译器对研究设计计算机也是有价值的。这种翻译器以计算机的抽象描述为输入，把它变换为更明确地叙述系统的逻辑设计的文件，即将语法结构变换为按语言的语义组成的逻辑。整个翻译过程通常由若干个作业组成。这些作业与语法和语义都有关。所以在翻译器完成之前，必须完整地定义语言的语法和语义，各种允许的语法结构都必须搞清楚。正因为如此准备翻译器在语言设计中是非常重要的。可以说在翻译器设计好前，语言的定义往往会是不完全的。

寄存器传输级语言的翻译器已有很多。[22]的第二章末给出了翻译器的参考书目。翻译过程的细节显然与语言的语法有关。所以不同语言的翻译器是很不相同的。如有些语言不关心结构部分的硬件实现。有些语言则对此非常关心，它要求将结构部分直接翻译成布尔方程形式，而对功能说明部分则要求翻译器充满了各种标准设计。尽管如此翻译器还是有许多共同问题要处理。关于编译技术的一般理论请参阅[33]。这里只介绍计算机设计语言的编译所特有的几个问题。

1．输入处理

输入处理的主要目的是分析描述系统的语句，用比面向设计形式更简单的方式叙述。这一步中还要进行语法分析，检查输入的语句是否违反语法规则。然后将被说明设备的参数编成表。在留下的描述系统的语句中给出表的指示器以引用这些设备。

设备表通常包含以下信息：名称；设备类型；设备的维数；形式参数的编号；设备说明体部的指示器。

在翻译器的这一部分中还要处理等价说明。如寄存器说明 IR[16]=OP[0:3]•IX[2]•ADR[10]中 OP，IX，ADR 等标识符对用户很有用，但进入翻译器时就失去了意义，必须消去。这就要修改有关的语句。如 MAR←—ADR 就要改为 MAR←—IR[7:16]。

2．数据流分析

传统的逻辑运算如“与、或、非、与非、或非、异或、等价”等不需要翻译，因为它们与硬件之间有直接的对应关系。数据流分析的目的是处理所有的运算语句，将它变换为以传统逻辑运算构成的逻辑方程。逻辑极小化和覆盖布尔方程则是自动逻辑综合算法的任务。为了提高自动逻辑综合算法的有效性，有些翻译器对逻辑方程中可使用的逻辑运算作某些限制，因为对“异或”等逻辑运算尚未有好的综合算法。

数据流分析中要研究的问题是如何处理移位、计数、算术运算和关系运算符。移位当然可以直接与硬件对应，也看成传统的逻辑运算，但这样使自动逻辑综合变得更困难。通

常移位直接表示成寄存器传输。计数、算术运算和关系运算符的翻译很复杂，通常采用标准库的方法，即用由传统逻辑运算组成的过程来实现这些运算。如加法，则进位链等都将反映进去。显然实现这些运算的方法不是唯一的。可以将各种实现方法的性能指标列入，供翻译时按需要选取。

3．控制分析

控制部分的翻译是翻译器中最复杂的部分。它通常由若干个有限状态机组成。它的输入是条件语句中条件部分的自变量，其值取自结构部分。自动机的输出是控制变量值。因为平行操作属于不同的自动机，所以控制分析程序应能在平行操作的起点引出新的有限状态机，并在并行处理结束时终结有关的自动机。

自动机的理论牵涉面很广，与编译有关的部分可参阅〔33〕。

在分析出各个自动机后，控制分析程序要处理状态简化、状态分配、自动机输入译码器、输出译码器的设计等问题。讨论代码分配是为使输入译码器的实现有极小量的逻辑，已有这方面的算法[34]。输入译码器、输出译码器的设计是多重输出组合线路的自动逻辑综合问题。

4．模块结构和增量翻译

由于使用 DA 的计算机设计比较多地在大型分时宿主机上实现，希望有适合此场合的语言和相应的编译器〔35〕。提出的语言和〔36〕设计的翻译器便是讨论这一问题的。他们提出的语言是模块结构的，各个部分能独立地翻译。这对设计大型系统是很方便的。由于可以各部分独立翻译，只要使用较少的核心存贮。由于各部分独立，就可以采用增量翻译方法。当修改系统的描述时不必重新翻译整个系统而只需翻译修改部分。

四、系 统 模 拟

在设计自动化技术的总结[6]中，Breuer 归纳了四种基本级模拟：电路级、门（组件、功能）级、寄存器级和系统级。电路级模拟实际上是对电路进行直流分析、交流分析和瞬态分析，以评价电路的性能。它和其他三个级不同，它使用的是连续变化模型，而其他三级的模拟使用离散变化模型。电路分析涉及的领域主要有图论、非线性规划、大型常微分方程组的求解等。主要应用于组件设计，特别是 LSI 的设计。

门级模拟又称数字逻辑模拟。通常包括组件、功能级的模拟。当前使用非常广泛。它主要用于设计检验和竞争、尖脉冲、冒险的分析。数字逻辑模拟的方法又可推广到故障诊断领域，形成故障模拟方法。

寄存器传输级和系统级模拟都用于估价系统的性能。前者也可用于设计检验。在门级实现之前验证系统的设计是否正确。这两级的模拟使用得不太多。但是就不多的模拟器来看，效果是显著的。最近美国成立的计算机结构评审委员会用这两级的模拟评价出最优秀的一些计算机。最好的是 PDP11，第二名是 IBM370。

寄存器传输级的模拟就是使用寄存器传输语言的模拟。它将实际系统简化为寄存器之间的传输。系统级模拟是更高一级的模拟。它的模型是借助于抽象导出的。它主要用于计

算机系统的总体结构的确定。这里系统是指硬件、固件和软件各部分的有机集合。

(一)寄存器传输级的模拟

RTL 的模拟器主要用来评价新系统的各种设计方案，在设计周期的初期从机器的结构来观察系统的行为。

需要评价的性能主要包括：

(1)执行各种指令的速度及其它的定时分析。

(2)非常规指令的影响和作用，如存贮器搜索指令、开方指令、数“1”指令等。

(3)对重新组合或容错设计，评价各种调度模型的性能。

(4)验证系统的正确性。

寄存器级模拟的用途是很广泛的。其他如确定数据通道的宽度，甚至亦可用于研制计算机的通用软件。

1. 模拟器的设计

模拟器的设计主要应考虑精确性和有效性。模型的精确性是首要条件。只有模型精确，模拟结果才有价值。模拟器的有效性是使用上的最重要的性能。因为一条指令的实现通常要用到宿主计算机的几百到几千条指令。弄得不好，模拟速度太慢，模拟器就不会有人使用。

解决这个问题主要从以下三方面入手：

(1)采用先进的一些模拟方法。

(2)充分利用宿主计算机的能力。

(3)合理设计模拟器的输出。

模拟器的设计中其他要考虑的因素有：建立方便统一的数据输入输出格式；对初始化、起动和结束模拟过程，对模型的监督等要提供人工控制的方便性；设计有效的模拟结构。

2. 模拟器的组成

模拟器主要由三部分组成。第一部分是翻译器，将系统的 RTL 描述变换为执行模拟所需要的形式，通常是表形式的数据库。这一部分包括输入处理、扫描和语法分析、错误检测、表的生成等。模拟器的第二部分是数据库的管理。它包括存贮定位、分类和调度、搜索和更新等。第三部分是执行器。在编译模拟器中它是编译好的代码。在表驱动模拟器中执行器解释执行表中所包含的信息。在解释模拟器中，执行器扫描 RTL 源程序并执行有关功能。

3. 编译器驱动模拟器

第一个寄存器级模拟器[37]是编译方式的。编译模拟器的目标是将 RTL 级的系统描述翻译为适合于当前数字计算机的目标语言。不管编译产生的是汇编语言还是如 Fortran 之类的高级语言，基本问题是相同的。编译器必须简化定时说明，要在 RTL 中找出并发的行为并同步化，变为非并发的时序的单指令流。编译器必须保留必要的存区作为工作单元。对初始化、起动、结束模拟过程、记录中间结果、监督循环等编译器要产生适当的指令。它还必须能解释控制语句，使用户得到系统的有用信息。

4. 表驱动模拟器

现代的模拟器大多是表驱动的[38'39]。表驱动模拟器也有翻译阶段。但它产生的是反映 RTL 的系统描述的一组表。然后支援程序考察这些表，执行模拟模型。

这组表应包括如下分量：

（1）系统中包含的存贮器、寄存器等。

（2）系统的内部连结，要有效地指出每个分量的扇入、扇出。

（3）将不能记忆的，执行加法、逻辑等运算的分量翻译成数据。

（4）能表示系统的状态，它指出所有同时执行的运算。

（5）有效事件，通常事件表指出了控制信息，它表示系统将从一个状态转到另一状态。

在表构造完毕后就执行由表所描述的模型。这是由若干个程序来完成的，它们解释执行表中的数据。显然表作为数据库就要考虑到它的通用性，如也能适用于自动逻辑综合。

寄存器级模拟的详细资料参阅〔22〕的第三章。

（二）系统级模拟

1. *系统模拟器的结构*

系统级模拟是将计算机看成离散系统。关于模拟器的结构，涉及的主要概念是激发、事件和进程。于是有三类模拟程序。当前的系统模拟程序主要是面向事件和面向进程的。

面向事件的模拟器[21,22] 这类模拟器包括一组事件子程序，初始化部分和模拟执行系统。初始化部分处理模拟器的输入激励。模拟执行系统的任务是编排事件表和执行事件子程序。在编写事件子程序时一定要考虑到进程的描述。因为后者是属于整个系统的。曾经有一种倾向，是将所有同时发生的运算收集起来，包括逻辑上无关的激发。结果大的模拟程序将使用大量的标识符，容易相互冲突出错，给模拟结果的分析带来困难。所以尽管是使用面向事件的语言，也要按照面向进程的观点来组织模拟模型。特别是要注意事件子程序的分解，即将属不同进程的激发放入不同的事件子程序中。

面向进程的模拟器[23-25] 面向进程的语言如 ASPOL[23] 与 ALGOL 极相似，但更复杂些。进程是执行一组逻辑上相关的激发。显然在同一时刻也许有许多进程在系统中执行。同类进程有相同的属性，可用同一组规则描述。这种进程的属性、行为、规则的描述称为进程描述。在模拟模型中它往往取过程形式。显然同时执行的进程必须以时序的方式模拟，所以进程可以有四种状态：执行、准备、中止和等待。在任何时候模型中的进程只有一个处于执行状态。其他要执行的进程处于准备状态。如果当前执行的进程中止时根据优先权从处于准备状态的进程中选一个执行。执行的进入点就是上次的中止点。于是所有进程以拟平行方式进行运行。

2. *系统的模型化*

观察的级和多级模型 如何描述系统，依赖于观察系统的级以及系统的本质。例如整个计算机系统可看为一组作业，每个作业也许要执行若干个程序。每个程序又由若干个子程序组成。一直下去可以通过指令级直到逻辑单元。建立系统模型的主要困难在于确定要详细说明的级。当系统庞大时问题就更困难了。因为我们的兴趣也许集中在系统的某一部分，于是这一部分就要在较低的级上详细说明，而其他部分有的可以简化，有的甚至可以省去。例如设计总线结构时也许就不关心盘的工作细节。从而可以看到必须研究多级模型，以使设计师能独立地对系统的各个部分在不同的级上建立模型。现在已有这样的多级系统[40]。他们定义了 LOGOS 系统描述语言及其模拟系统。

软件的模拟 系统级模拟包括要能对软件进行模拟。计算机的操作系统往往是很庞大

的。为模拟真实系统的功能和行为，必须抽象化。抽象的方法通常有两种：简化和综合。简化就是将系统骨胳化，数据结构等皆可简化，只保留系统的基本结构。综合是将计算机系统的一些单元如过程或数据结构组合成较大的模型单元。这时真实系统的某些部分也许不存在了，但系统的行为特征是相同的。综合的动机是缩小模型的规模，然而它通常受模拟语言的限制。一般而言简化的方法更成功些，特别是基于有向图的模型化方法，它在计算机辅助系统设计中使用得相当广泛。

系统级模拟的详细资料参阅〔22〕的第一章。

五、自动逻辑综合

布尔表达式可以采用立体表示法。如 $x_1 \cdot \bar{x}_2 + x_1 \cdot x_3$ 可以表示为{10x, 1x1}。将立体与规范型的布尔积项对应最早是 Urbano 和 Muller 提出的[41]。Roth 在〔42〕中比较系统地将立体表示用于自动逻辑综合。这个记号使计算机对布尔表达式的运算变得方便。

若有 n 个自变量，则规范型的每个布尔积项与 n-立体中的一个顶点对应，若函数 f 的值为 1，对应的顶点称为 f 的必要顶点(或名真顶点)。若函数 f 的值为 0，则称为假顶点。若 f 没有定义，则这些顶点为无关顶点。

立体有维数的差别。与规范型布尔积项对应的立体称为 0-立体。若布尔积项中有 r 个变量可取任意值(此值记为 x)，则此立体为 r-立体。立体集合称为复合体。

逻辑运算和集合运算是大家熟悉的二类运算。由于出现了立体，就需要引进新的一类运算：立体运算。立体运算是对 0-复合体的集合运算。通常若 C 是立体集合，则 C 包含(亦称覆盖)的所有 0-立体称为 C 的 0-复合体，记为 $K^\circ(C)$。两个立体集合的关系有两种：一种是集合关系，一种是立体关系，即 0-复合体之间的关系。若 C 和 C′ 是两个立体集合，则 C′ 覆盖 C 就是指 $K^\circ(C) \subseteq K^\circ(C')$，亦记为 $C \Rightarrow C'$，读作 C 蕴涵于 C′，自动逻辑综合中经常遇到的 * 积和相容运算都是立体运算。

立体对应与门，立体集合则对应函数 f 的与非两级实现。f 的实现价格与总输入头数和门的个数有关。令立体集合中有 q 个(n−1)—立体，它们是不需要与门的，于是函数 f 的实现价格通常可取为

$$\$C = \sum_{c \in C} \$c + |c| - q$$

其中 $|C|$ 是立体集合 C 中立体的个数，$\$c$ 是立体 c 对应门的输入头数。

显然立体越大，包含的 0-立体越多，而其价格却越小。令 f 的真顶点组成立体集合 C_0，f 的无关顶点组成立体集合 DC。f 的本原蕴涵就是在 $C_0 \cup DC$ 中的最大立体，显然它应覆盖 f 的部分真顶点，否则这个立体是无意义的。容易证明若 Z 是 f 的所有本原蕴涵组成的集合，C 是 f 的极小价格覆盖，则 C 是 Z 的子集。这就是说 f 的极小价格与非两级实现，一定是由本原蕴涵对应的门组成的。自动逻辑综合就是要生成本原蕴涵集合 Z，找极小价格覆盖 C 或构造无冗余的覆盖。

(一)单输出开关函数的综合

1. 本原蕴涵集合 Z 的生成方法