

Intel 8080

汇编语言及其程序设计

(上)



Semiconductor Devices

Intel 8080

汇编语言及其程序设计

(上册)

海元件五厂

14

Shanghai No. 5 Component Factory

前 言

随着科学技术的发展，微型计算机的应用日益广泛。本厂系专业从事微处理机芯片、接口以及RAM、EPROM等元件的生产单位。为了进一步了解Intel 8080微型计算机的性能和应用，特邀请上海计算所宋惠忠同志在厂举办的“微型计算机讲座”中对该机作了讲授。现根据各方面的要求，我们请宋惠忠同志对讲稿进行补充和修改。现先将其中部份定名为“Intel 8080汇编语言及其程序设计”出版供有关用户单位参阅，并向作者宋惠忠同志致以谢意。

上海元件五厂情报资料室

一九八〇年

目 录

第 一 章 概 论	1
§ 1 · 1 微型计算机的软件体系	1
§ 1 · 2 计算机语言与程序设计	2
§ 1 · 3 微计算机开发系统与软件开发	3
第 二 章 8 0 8 0 汇编语言程序设计	10
§ 2 · 1 汇编语言和处理机	10
§ 2 · 2 汇编语言的概念	20
§ 2 · 3 汇编语言文本说明	23
§ 2 · 4 机口指令详述	30
§ 2 · 5 伪指令详述	89
§ 2 · 6 程序设计技术	118
例 1 : 8 位二进制数相加	118
例 2 : 分离出一个 1 6 进制数	120
例 3 : 一个数据字节的拆卸	122
例 4 : 数据求和	125
例 5 : 数据传送	129
例 6 : 检查最大值	132
例 7 : 二进制多位精度运算	137
例 8 : 十进制加减运算	143
例 9 : 二进制乘除法运算	148
例 1 0 : 十进制乘法运算	151
例 1 1 : 乘方表	155

例 1 2 : 一个实用的查表程序.....	159
例 1 3 : 计算字符串长度.....	162
例 1 4 : ASCII 至十进制的转换.....	164
例 1 5 : 图形比较.....	167
例 1 6 : ASCII 至十进制换算子程序.....	171
例 1 7 : ASCII 至十进制转换子程序的调用程序.....	172
例 1 8 : ASCII 至十进制转换子程序的修改形式.....	214
例 1 9 : 字节拆卸子程序.....	174
例 2 0 : 求最大值子程序.....	175
例 2 1 : 多精度加法子程序.....	176
例 2 2 : 段数据变换子程序 (实用程序)	177
例 2 3 : 键扫描子程序 (实用程序)	180
例 2 4 : 定时例行子程序 (实用程序)	184
例 2 5 : 地址寄存口和数据寄存口显示子程序.....	185
例 2 6 : 键输入例行子程序 (1)	189
例 2 7 : 键输入例行子程序 (2)	195
例 2 8 : 16 进制数左移一位子程序.....	196
例 2 9 : 串行输出例行子程序.....	197
例 3 0 : 串行输入例行子程序.....	200
例 3 1 : 寄存口恢复子程序.....	204
例 3 2 : 电传打字机的接口软件程序设计.....	206
例 3 3 : Intel 8080 中断系统程序设计.....	215
• 一个单一中断向量.....	216
• 多级中断向量.....	218
• 保护优先权中断状态.....	220

• 优先权中断系统.....	221
• 扩展中断系统.....	226
• 典型的 8080 中断.....	228
开关基本中断.....	228
定时口中断.....	229
键盘中断.....	235
U A R T 中断.....	237
转换口中断.....	239
D / A 转换口中断.....	240
A / D 转换口中断.....	241
例 3 4 : 软件数字钟.....	243
例 3 5 : 电子警报口.....	247
例 3 6 : 可编程序节拍口.....	250
例 3 7 : 音乐自动演奏.....	255
 第 三 章 微型计算机的汇编程序.....	 262
§ 3 . 1 软件的编制.....	262
§ 3 . 2 汇编语言的语法.....	263
§ 3 . 3 汇编的使用方法.....	272
§ 3 . 4 汇编程序的内容.....	275
附表 : MICRO SELF ASSEMBLER LIST.....	323

Intel 8080: 汇编语言及其程序设计

第一章 概 论

§ 1.1 微型计算机的软件体系

事情愈来愈明显，微型计算机正沿着当年小型计算机的道路前进。正如小型计算机一样，微型计算机是将硬件和软件融为一体所构成的系统。以前，计算机的总体设计是以硬件为中心，但随着软件重要性的日益提高并变得不可缺少，现在已越来越多地考虑综合地进行硬件和软件的设计，使它们分别担负适当的功能。

微型计算机中，软件的地位可用图 1.1 来说明。

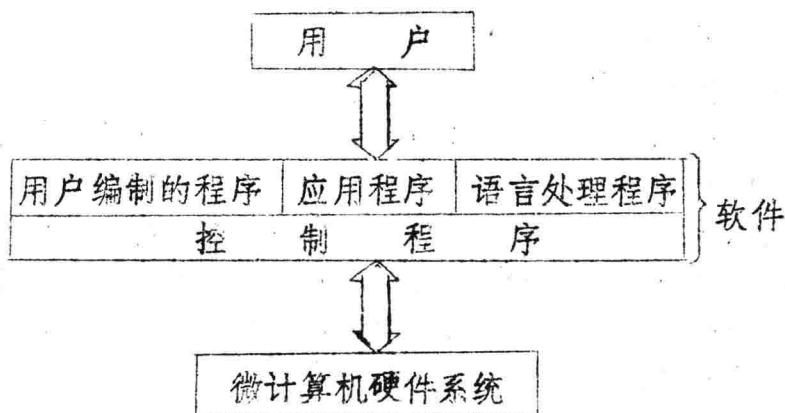


图 1.1

从图 1 可知，用户和微计算机硬件之间是经由软件作为桥梁的。软件的组成可概略地划分如下：

微计算机系统软件：

- 系统控制程序

- 驻留在主存中的系统督监程序；

- 执行及管理程序：

- 宏汇编程序；

- 文本编辑程序；
- 链结程序；
- 定位程序；
- 程序库管理（文件管理），等等。
- 语言处理程序：
 - PL / M-80 语言；
 - FORTHAN-80 语言；
 - BASIC-80 语言；
 - ICIS-COBOL 语言 等等。
- 应用程序：
 - ICE（仿真调试程序）；
 - UPM（通用 PROM 映像程序）；
 - 可靠性测试程序，等等。
- 用户编制程序：
 - 用户从自己的使用目的出发而编制的具体任务的程序，如此等等。

以上诸项，组成了微计算机系统的软件。和小型计算机一样，微计算机系统的性能和操作系统的关系极大。因此微型计算机系统设计者总是站在系统的角度对软件和硬件两者进行合理地功能分担，以期达到好的性能价格比。

§ 1.2 计算机语言与程序设计

微计算机就其实质而言，它只能识别一种语言，那就是通常我们所说的机口语言，亦即机口的指令系统。因此我们就把这种机口能直接理解的机口语言称做目标语言。人所共知，使用目标语言进行程序设计是一件极其困难的事情，甚至有时是不可能的。为便于进行程序设计，人们开发了许许多多种程序设计语言。如果我们对这些程序设计语言作一分类时，又将区分为两种：其一是汇编语言，亦称符号语言；其二是编译语言，亦称高级语言。

汇编语言（Assembler Language）：为了摆脱用机口语言

书写程序造成的巨大困难而研制的程序设计语言。由于汇编语言是用助记符号来等价机口指令的代码，因此人们称汇编语言为符号语言。这种代换导致人们阅读程序简易。值得指出的是汇编语言是质量最高的翻译语言，汇编语言的每一助记符号均单值地对应于一机口指令代码，和高级语言相比，汇编语言源程序较之高级语言源程序经编译后占用的存贮单元数量为少。因此，微计算机的所有核心程序均毫无例外地使用汇编语言书写。汇编语言的局限性在于不同机种之间不能通用。

编译语言 (Compiler Language)：它是比汇编语言更为高级的一种程序设计语言，它现在已经成为一种主要的微计算机程序设计语言。例如：FORTRAN-80，PL/M-80，iCIS-COBOL等等。由于这些语言的语句结构近似于英语语句和公式格式，故极易被人们阅读和书写。这就导致编制程序简便，生产性强。此外，原则上可用于各种机种，通用性强。

汇编语言和编译语言统称为“源语言”。使用源语言编制的程序称为源程序。源程序是不能直接被计算机运行的，为此必须在计算机上设有能将源程序翻译成目的程序的翻译程序。所谓目的程序就是用机口指令代码表述的程序。

综上所述，程序设计就是用某种语言（目的语言或源语言）就某项任务进行解折，编排流程图，书写程序，并附以调试的全过程。显而易见，程序设计是需要选择语言的。

§1.3 微计算机开发系统与软件生成

微计算机系统并不都是一个十分理想的完善系统。诸微计算机系统之间，在结构上差别很大。我们可以分作四种情况予以说明。

一、最基本的系统

最基本系统的硬件应包括一个微处理机芯片族（8080A+8224+8228），输入/输出接口电路，以及ROM和RAM的组合。通常系统的监督程序是留驻在ROM之中，用户程序存放在RAM之中。有时RAM的容量甚小（只有128字节），用户程序干脆也驻留在

ROM之中，RAM只当作堆栈之用。

二、单板微计算机系统

在OEM市场上，单板计算机的品种甚多，比如，BLC 80/07 (512字节RAM)，BLC 80/10 (1024字节RAM)，BLC 80/11, 12, 14 (1K, 2K, 4K字节RAM)，BLC 80/204 (4K字节RAM)。上述各种型号的单板计算机均有8K字节的EPROM，其中2K字节EPROM是为系统监督程序留用的，其余均是为用户设置的。

三、用户微计算机系统

用户微计算机系统一般系指那些采用OEM产品而构成的相当于低档小型计算机的系统。这类系统通常是以单板计算机为中心，选择配备扩充存贮口单板，串行或并行扩展接口单板，软磁盘控制口单板，以及键盘显示控制口单板等等构成系统的主机，然后再选用某种单密度或双密度软磁盘驱动口，键盘及CRT，以及行式打印机等等外部设备。有的用户系统可允许接若干个终端，组成所谓多用户系统。由于这类用户系统配有功能强大的磁盘操作系统，因而通常配有多种高级语言。有的用户系统还配有模/数和数/模转换口接口，因此利用用户系统配备的实时操作系统（实时多任务执行系统）可进行模拟量的实时处理和实时控制。这样就使微计算机在军事方面的应用潜力，日渐增强。

四、微计算机开发系统

微计算机开发系统不同于一般用户微计算机系统之处就在于它是出于为设计一个新的微计算机系统而建造起来的一个功能完备的微计算机系统。微计算机开发系统不是面向用户的，而是面向于微计算机系统设计工作者的。微计算机开发系统配备种类繁多的语言，但这是为了研究和调试微计算机系统设计工作者提出的源程序。微计算机开发系统提供文本编辑，宏汇编，编译语言，链结程序，定位程序，硬件仿真程序，PROM映像程序，调试程序，实用程序。此外还有文件管理程序等等。微计算机开发系统对微计算机系统设计工作者提供两种产品：其一是源程序列表清单（它包含源程序表及目标程序表两

者)，其二是驻留有目标程序的 PROM 或 EPROM 组件，亦称固化的软件。从这一点可以看出，微计算机开发系统和用户系统是完全不同的。用户系统的产品是源程序的运行结果。

上述四种系统的结构是互有区别的，以前两种系统最为简捷。但从应用的场合来看，前两种系统最广大。这就导致前两种系统强烈地依赖于微计算机开发系统进行软件生成的必要性了。

微计算机开发系统的软件生成过程如图 1·2 所示。在图 1·2

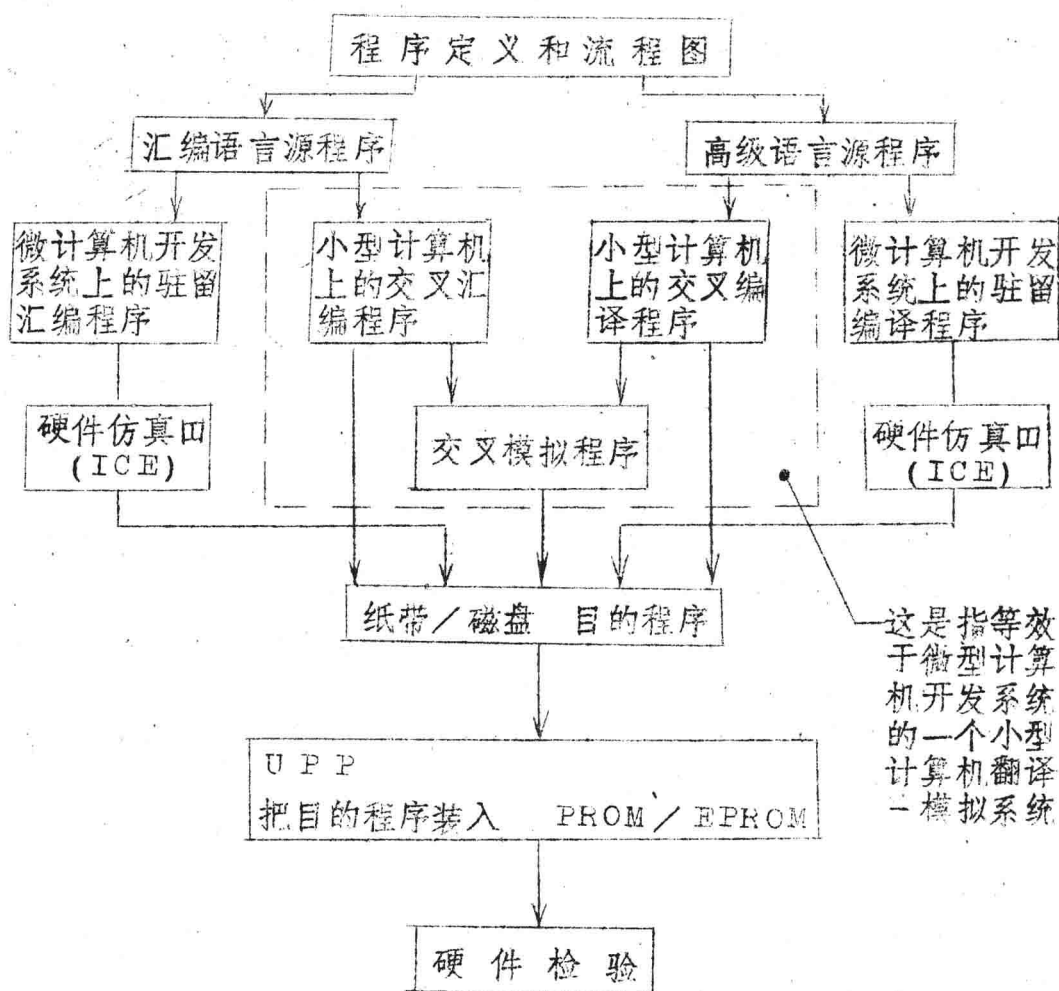


图 1·2 微计算机开发系统软件生成示意图

中所示意出一个等效于微型计算机开发系统的小型计算机翻译-模拟系统。

对于要进行开发的程序而言，它的程序定义和流程图是已知的。用何种语言书写源程序这是进行软件开发所遇到的第一次选择。

假定我们具有汇编语言方面的经验。我们便可以书写汇编语言源程序。对于已经写好的汇编源程序而言就面临着选择何种环境的汇编程序的问题。如图 1.2 所示，汇编程序可能存在于两种环境：驻留汇编 (Resident-assembler) 和交叉汇编 (Cross-assembler)。在微计算机开发系统上提供了驻留汇编程序，这是最理想的环境。因此，国外使用微计算机开发系统极其普遍。汇编程序提供强有力的文本编辑程序 (Edit) 来帮助程序员改正源程序的输入差错。文本编辑程序可提供下列各种功能：

- 插入字符 (INS.CHAR.) 在语句中的任意位置插入一个乃至几个字符；

- 插入行语句 (INS.LINE) 在程序任意行之后或之前插入一行乃至几行语句；

- 勾消字符 (DEL.CHAR.) 可勾消语句中一个乃至几个字符；

- 勾消行语句 (DEL.LINE) 在程序任意行之后或之前勾消一行乃至几行语句；

- 显示向上推一行 (ROLL UP) 为检查某一行语句，可将源程序语句行向上推一行；

- 显示向下推一行 (ROLL DOWN) 为检查某一行语句，可将源程序语句行向下推一行；

- 显示前一页面 (PREV PAGE) 将源程序文本的前一页面显示出来；

- 显示后一页面 (NEXT PAGE) 将源程序文本的后一页面显示出来；

- 打印输出 (PRINT) 将 CRT 上的显示内容在打印机上打印输出。

文本编辑程序帮助程序员改正源程序一级的差错。把经过改正的, 认为是“正确的程序”交付汇编程序汇编处理。如果确实正确无误的话, 则汇编回答: “全部通过 (Assembler All Passed)”。假如仍存在未被发现的差错, 汇编程序也绝不会马虎放过, 它会指出差错的数目, 并可通过打印源程序表的办法, 发现错误的具体性质和地方, 便于程序员识别和改正。经验表明, 软件开发的绝大部分时间 (占总时间的 70 ~ 80 %) 都花费在这个阶段。

经过文本编辑程序调试过的源程序被汇编成目标程序。该目标程序可再进一步通过 ICE-80 仿真口进行仿真测试。在仿真口上运行目标程序, 无论从什么角度来评价都是极严格的, 也是相当逼真的。

ICE-80 仿真口实际上是由硬件和软件构成的专用于研制 8080 的微计算机系统。它提供如下的功能:

- 全符号化的调试功能 ICE-80 命令中的任何数据均可以用符号来表示。用户符号表随同目标文件在汇编时产生, 并随同被仿真的用户程序一块输往内存。用户可以任意增设在系统调试中所需的内存地址、常数变量的符号。通过访问这些符号化的内存地址, 可以对这些内存单元进行检查、修改或断点停机。ICE-80 对 8080 的全部寄存器、标志和指定的总线都有定义。

- 实时仿真功能 ICE-80 的操作分为三种基本方式: 仿真、单步和查询。在仿真操作方式下, ICE-80 控制用户硬件, 并定时地执行用户程序。在每一机口周期的末尾, 将用户系统有关的状态信息保存起来, 以备在查询操作方式时被取出显示。ICE-80 能控制用户程序以单步方式进行操作。值得注意的是当在仿真操作方式时, 若碰到一个断点, ICE-80 就自动转入查询操作方式。这时的内存地址、数据总线的内容, 断点前最后操作的 4 4 个机口周期中的状态信息, CPU 总线、寄存器、标志和堆栈的内容, 断点地址、当前子程序嵌套、符号名字和符号表中的值, 以及自程序启动后所经历的全部时钟数目均都可显示出来。

- 资源出借功能 通过 ICE-80 的映像功能, 微机开发系统的内存和 I/O 设备可“出借”给用户系统。这样, 用户可以把微计

算机开发系统的设备当作用户样机系统尚未构成的设备。这种资源出借的功能，对孤立故障，排除差错十分有利。例如，当程序在用户系统的内存里运行出了差错，设计者就可将程序转到开发系统的内存中运行，从而确定了是软件还是硬件发生了故障。同样可以采用开发系统的 I/O 设备来孤立用户系统中 I/O 设备运行时出现的故障。

经过 ICE-80 仿真口运行测试而通过的合格程序（目标程序）就可以借助于通用 PROM 映像程序（MAPPER）将目标程序写入 EPROM 或 PROM 之中，或者就干脆以纸带文件或磁盘文件的方式输出或存贮起来。

我们如果使用高级语言书写程序，并在微计算机开发系统上研制，那么情况大体如前，不再赘述。

如果我们并不具备微计算机开发系统这种驻留环境，而且要通过小型计算机上的交叉翻译程序和交叉模拟程序的话，在图 1·2 中如虚线所示之情形。软件开发的过程也不尽相同。

综上所述，微计算机软件的开发可区分成下列几个阶段：

1. 问题定义和程序流程图：问题的定义包括任务的正式要求，输入/输出的规定，处理的要求，系统的指标（执行时间、精度、响应时间）以及设计出满足任务要求的流程图。

2. 程序设计 选择计算机语言，按程序设计手册的规定完满地实现流程图的要求。

3. 调试：这个阶段是发现和改正程序设计的错误。调试是软件开发相当重要和相当费时的阶段。值得提出的是，精通程序设计技术固然对降低错误关系密切，但程序设计当中的逻辑错误，非得在任务要求方面下功夫才行。

4. 测试：这个阶段是程序的批准生效阶段。测试可确保程序能正确地执行所要求的任务。

很显然，软件开发除上述四个阶段之外，还必须包括下面几个阶段：

5. 建立文件：这个阶段是提出程序设计文件的阶段。因为无论什么人，都可以通过文件而了解程序的目的和过程，从而把握住程序

的应用领域。

6. 维持和成长：这个阶段是根据应用场合的变化了的条件和经验，修改和完善原程序设计，使之成长而趋完备，功能全面体现。

7. 发展和重新设计：这个阶段是考虑到那些超出原来问题定义的范围而提出的新任务而出现的阶段。这不是一般地总结经验的阶段，而且在变化了为新条件和新任务面前，用新的设计来代替旧设计的阶段。

第二章 8080汇编语言程序设计

§ 2.1. 汇编语言和处理机

汇编语言都是针对于具体处理机的，因为几乎汇编语言源程序的每一行源代码，都能直接被翻译成一条具体的机口指令。因此，汇编语言程序设计人员必须熟悉汇编语言和处理机两个方面。

尽人皆知，在使用汇编程序之前，首先要准备好汇编语言源程序。汇编语言源程序是由程序设计人员所书写的，由若干个汇编程序语句所组成的。一个标准的汇编程序语句则是由标号段、操作码段、操作数段，以及注释段等所组成。

在使用汇编程序（驻留型或交叉型）之时，应先把汇编语言源程序变为可由机口读入的形式。再由微计算机开发系统的文本编辑程序，将源程序变为磁盘文件的形式保存下来，这样就可以把所形成的源程序文件提交汇编程序处理。汇编程序的任务是把符号语言翻译成8080微处理机所能执行的目的代码。汇编程序的输出可包括三个文件：(1)目的文件，其中包括已翻译成目的代码的用户程序；(2)编目文件（程序表格），它打印出源代码，目的代码以及符号表；(3)符号查询文件（Symbol-cross-reference file），这是一张记录符号查询信息的表格。汇编程序的处理过程如图2.1所示。

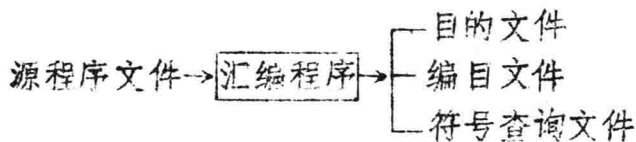


图 2.1 汇编程序处理过程中各文件的关系

还必须指出，对于应用微型计算机的大部分场合，最后大概都得把目的程序装入某种形式的只读存贮口中（但不是全部如此）。然而，应当记住，微计算机开发系统是带有随机存贮口的8080微计算机系统，在大多数场合，程序员可把目的程序装入开发系统，并在其上执

行这些程序，以便测试与调整。这就允许在用户系统的样机未建造之前，就可对其工作程序进行测试与调整。

目的代码：8080汇编程序的特点，是目的代码可根据要求采用浮动形式。这就使程序设计人员不必担心在应用系统中可能需混合使用只读存贮器和随机存贮器。在必要时，当设计结束后，个别的程序段可以浮动。同样，一个冗长的程序可以被分成若干相互分离的汇编好的模块。这些模块，不但容易编码，而且也易于测试。

编目文件（程序表格）：程序表格提供了一份源程序与目的代码的永久性记录。汇编程序同时还在程序表格中提供了一些常见的程序设计错误的诊断知识。例如，假定给一条只能使用8位数的指令规定了一个16位数，那么汇编程序会通知：此值已超出允许的范围。

符号查询文件：符号查询表文件是汇编程序所提供的另一种诊断手段。例如，程序在运转一个取名为DATA的数据段，如测试结果表明在处理该数据时有程序逻辑错误的话，则符号查询表将能简化该错误的排除过程。因为符号查询表指出凡是使用DATA符号的每一条指令。

对8080A微处理器硬件组成情况，是关心程序设计的人所必不可缺的背景资料。对程序设计人员来讲，处理机是由下列各部分构成的：

- 内存；
- 程序计数器；
- 工作寄存器；
- 条件标志寄存器；
- 堆栈与堆栈指示器；
- 输入/输出口；
- 指令系统。

内存：内存实际上并不属于处理机，但它对程序设计关系密切，因此必须讨论一下。由于微处理器所使用的程序都放在内存当中，故所有应用微处理机的场合都需要一定的内存。常用的内存有两类：只读存贮器（ROM）和随机存贮器（RAM）。