XDJ-73小型电子计算机

系统程序参考资料

八进调试程序

〔ODT〕

中国科学院
北京电工研究所五室编

XDJ－73机系统程序参考资料

八进制调试程序

（ODT）

中国科学院北京电工研究所五室编

1976.5.

目　　录

一、八进制调试技术的一般知识

八进制调试技术简称ODT（Octal Debugging Technique）。利用它便于和目的程序进行对话和修改目的程序，程序员利用规定好的命令通过电传和目的程序进行对话。所用的编码全部是八进制数。

1。主要功能：ODT的主要功能有以下几个方面：

1•1 对存放目的程序的内存单元进行考查，必要时对其内容进行修改。

1•2 把程序员所指定的内存段的内容由电传穿孔机或高速穿孔机制成二进制纸带输出。

1•3 将内存中存放的内容利用检字机能通过电传打出八进制码。

1•4 在目的程序运行过程中按照操作员的要求插入断点，以便检查和修改程序。

2。存贮要求：ODT本身占用 600_8 个内存单元（384_{10}）以及零页的0004内存单元作为断点联系之用。

它可占用1000到1577内存单元，这是低地址方式，起始地址为1000。也可占用7000到7577内存单元，这是高地址方式，起始地址为7000。不仅如此还可以重新更换其所在内存页的位置。也就是说ODT的源程序带的起始地址除了零页之外可以放在任何一页的开头，而ODT程序就可以存放到接下去的各内存单元中，不过更换内存页的位置应在同一内存区进行。

断点联系用的内存单元为ZPAT，其含义见以后的叙述。ZPAT的绝对地址为0004，其内容在输入ODT后给定，低地址时为1243，高地址时为7243。

ODT程序和目的程序都是二进制纸带，应用二进制引导程序输入到内存中，但应注意，不能将两者重叠起来。

3。启用过程

3.1 ODT的起始地址是START符号。一般多规定为7000（高地址形式）或1000（低地址形式）。

3.2 将起始地址用开关寄存器给出后，按送地址键和启动键。ODT回答一个“回车”和“换行”信号以表示ODT程序已存好，可以运行并等待输入命令。

3.3 如果重新启动ODT，为了避免清掉检查和码，令起始地址为START＋1（即1001或7001），再按送地址键及启动键即可。

二、ODT的使用方法及命令

1。控制命令

1.1 nnnn／ 这条命令中nnnn是八进制码表示的内存单元，斜画是将该内存单元打开，打印其内容（也是用八进制码）。然后程序员再打印所需要的八进制码就可对已打开的内存单元进行修改。所打印的八进制码不超过4个字符就是合法的，否则就是不合法的，非法码将被去掉，ODT并回答？号。只打／或0／则将紧前面给出过的内存单元打开并打印其内容。例如：

400／6046　　400内存已打开

400／6046　2468?　有“8”为非法码

400／6046　12345?　有5位数字为非法码

／6046　　400内存单元的内容未变

凡下面有长划的字符表示由ODT程序指挥打印出来的，下同。

1.2 ↵ 将已打开的内存关掉。

400/6046 ↵ 400内存不改变内容关掉

400/6046 2345 ↵ 400内存改变内容，关掉

/2345 复查400内存内容已用2345

换掉6046

400/6046 401/6031 2346 ↵

400内存内容不改变就关掉

401内存内容更换再关掉

400/6046 401/2346 ↵

复查400及401的内容

1.3 换行"↓" 这个命令关掉刚被打开的内存单元将其下一个内存单元打开。例如

400/6046 ↓ 将400单元不改变内容关掉，

0401/6031 1234 ↓ 打开401，用户打印需要更换

0402/5201 ↵ 的内容，将0401关掉。

再打开0402，把402不改变

内容即关掉。

注意：在本例中 401单元的内容已换成1234。

1.4 ↑ 这项命令将刚被打开的内存关闭，然后将其内容作为访内指令来解释，将其所访单元打开并打印其内容。例如

404/3270 ↑ 关掉404不改变其内容

0470/0212 0000 ↓ 3270表示DCA，现行页，页地址

404/3270 ↑ 70所以打开0470单元，再打印

0470/0000 更换内容。

最后两行是复查一次，上项命令执

行无误。

1.5 ← 这个命令的作用是将已打开的内存单元关闭，将其内容作为地址，把与此地址相对应的内存单元打开并打印其内容。例如：

365/5670 ↑　　关闭365打开270

0270/0426 ←　　关270打开0246

0426/5201

1.6 非法符号　遇到非法符号时，ODT 回答“？”号并发出回车换行信号，不接受非法符号，准备接收新的命令。例如：

4：？ ↓
4 U ？ ↓　　} ODT不打开任何内符，准备接受新命令

406/4761 67K ？ ↓　　ODT去掉修改内容（因其非法）

关掉406

406/4761 ↓　　复查406

1.7 断点操作　这是ODT 最有用的操作之一。ODT 的作用好象一个监控程序，它密切地监视着目的程序，如果程序员需要在目的程序运行到某一点时断开，那么他首先应给出一个命令 nnnn B，其中 nnnn 表示目的程序应在该内存单元处断开，而 ODT 在接到命令 B 后，将 nnnn 这个地址存放到某一个 ODT 可以对之进行操作的特殊内存单元处。然后发出回车换行信号等待打出新的命令。这时程序员给出 mmmm G命令。在接到这项命令后，仍先执行ODT程序，作一些准备工作，这包括：

(1) 将存放 AC 内容的内存单元清掉。

(2) 将断点处内存单元 nnnn 的指令放在一个特定内存单元中暂时存放。

(3) 在nnnn内存单元中存放一条无条件转移指令 JMP I 4 ，以便当目的程序执行到nnnn断点处时将程序转到0004单元处。

(4) 在零页的0004内存单元中早已存放为执行断点操作的第一条指令。

作好这些准备工作后，开始从mmmm内存单元执行目的程序，一直到遇到断点内存地址nnnn为止。如果没有预先安排断点，那么到目的程序执行完为止。

如果只打G则作为错误处理，但不打出“？”，程序转移到0000地址处。

当目的程序执行到断点地址nnnn时，通过 JMP I 4 指令转去执行ODT程序中处理断点的操作。首先将当时AC和L的内容保存到适当的内存单元中。注意这时nnnn中原来目的程序的指令并未执行，因此ODT程序应做好一系列准备工作，以便一旦处理完断点操作时，再继续作完尚未执行完的目的程序。这包括：

(1) 恢复目的程序中nnnn中原来的指令。

(2) 根据nnnn中原来指令的性质在ODT程序的某一点安排好一条指令，这条指令的功能相当于执行nnnn中原来目的程序的指令，以便在处理完断点操作时，根据C命令继续执行目的程序。请注意这时nnnn中原来目的程序的指令是由ODT程序在这个特定点执行的。

(3) 准备好转去执行nnnn＋1条指令的条件。

做好这些准备工作后再将断点地址打印出来，打一个“(”符号，再将保存着的AC的内容打印出来。之后程序员就可按照前面叙述的办法检验并修改目的程序的任一内存单元了。也可以用A命令去检验

和修改AC和L的内容（详见下述）。程序员也可以用命令C使目的程继续进行（详见C命令项中的叙述）。

如果只打印一个B，前面不打nnnn，则ODT将原来规定的断点挪开，给出回车换行信号，继续等待接受新的命令。原来给出的断点就不起作用了。

如果在给出一个断点之后，当目的程序执行到规定断点，由ODT程序进行操作时，再打出一个新的断点mmmm B，则断点从原来的nnnn处换到mmmm处。因为在同一时间内只能有一个断点有效，因此请求新断点将使原来的断点失效。

在下列情况下，断点就是不合法的：

TAD
DCA
JMS
} 断点在这些地方是合法的

FADD 断点在这里是不合法的（即当浮点指令FADD作为JMS的自变量时不能使用断点操作）。

1.8 A 前已说明当目的程序执行到断点地址时，即转去执行ODT的断点操作。这时AC和L的内容C(AC)和C(L)都保存起来以便在处理完断点操作时再恢复它。当ODT执行断点操作时如打出一个A命令，则将保存AC内容的内存单元打开并打印其内容，于是这个寄存器的内容可以用前述方法进行修改，经过修改的内容也可以用C命令恢复到AC中。

在打出A命令后，如再打一个↓命令，则将存放AC内容的内存单元关掉，将存放L内容的内存单元打开，并打印其内容，以便进行修改，修改的方法同前。经过修改的内容也可以用C命令恢复到L中。

1.9 C 当ODT程序转去执行断点操作时，如打出C命令

ODT 将把原来规定的断点再安排好，恢复 AC 和 L 的内容，执行被断点中断的指令（即目的程序的第 nnnn 条指令），并使用户程序继续执行下去，直到再送到新请求的断点时为止。

有时程序员可能希望将断点放在其程序循环的某一地址处。因为循环可进行成千次，必须有一种措施避免在每次遇到断点时就断开，这时可以发出 nnn C 命令，这个命令在第一次遇到断点后发出，于是只有当程序循环 nnn 次后才再次断开，由 ODT 进行断点操作。

1.10 M 在解释这条命令之前先介绍一下检索功能。为了调试目的程序有时需要了解某一指令的全部（例如停机指令 HLT = 7402 就需要知道全部指令字的内容即 7402）或部分（如存数指令 DCA 一般只需了解第一个八进制码 3 即可）在目的程序中出现的地点，以便进一步处理，这种功能叫做检索。为了进行检索在 ODT 程序中放有检索变量，它们是检索屏蔽值，检索下限和检索上限。检索屏蔽值（SEARCH MASK）用来将被检索的指令字屏蔽出来，通常为 7777，它可以将整个指令字屏蔽出来，如为 7000 则可将指令字第一个八进制码屏蔽出来。检索下限是开始检索的地址，检索上限是检索的终了地址。

给出 M 指令后，ODT 将存放检索屏蔽值的内存单元号打印并将其内容（即屏蔽值本身）打印出来。屏蔽值可任意将之更换，方法是将存放屏蔽值的寄存器打开，ODT 打印其内容之后程序员可将其欲改换之值打印出来，再将该内存单元关掉即可。

如果在 M 之后打↓符号—— 就将检索下限寄存器打开。因为存放屏蔽值的内存单元的后面就是存放检索下限的内存单元。所以在 M 之后打一个↓号，就将屏蔽值寄存器关掉并打开存放检索下限的内存单元，打印其内容以便修改。最初检索下限是 0001，改变此值的方法

和改变屏蔽值一样。

接下去再打一个↓号，就将检索下限内存单元关掉，但打开检索上限内存单元，并打印其内容以便修改。最初检索上限或为 7000（高地址形式）或为1000（低地址形式）。改变检索上限值的方法与改变屏蔽值一样。

1.11 nnnn W 检字

这个命令使ODT在程序员规定的内容范围内检索出其内容等于给定值的内存单元。

检索不改变被检内存单元的内容。例如下例就是要求检索出所有包含ISZ指令的内存单元。检索范围为3000到4000。

M 7777 7000 ↓	将屏蔽值换成7000，打开检索下限内存更
7473/0001 3000 ↓	换其值为3000，打开检索上限内存更换
7474/7000 4000 ↓	其值为4000，关掉内存单元
2000 W	打出检索命令。
2000/2467	在本检索范围内含有
3057/2501	4个ISZ指令。
3124/2032	
4000/2152	

1.12 T 穿制带头命令

打出T命令并将穿孔机接通时，ODT即穿制0200_8码的带头，当打对带头时将穿孔机断开，按“停机键”。在打印任何命令以前必须把穿孔机断开，因为和穿孔机接通时，任何从键盘上打出的符号都

将穿孔输出。发出任何新的命令之前，应重给起始地址并按启动键。

1.13　nnnn；mmmm P　穿孔输出二进制带

打出这个命令ODT将从内存单元nnnn开始到mmmm为止打出这部分内存的内容。这时计算机将自动停机，显示AC内容为7402，以便用户把穿孔机接通。按“继续键”即开始输出，这个命令不穿制检查和码就令穿孔结束，以便继续从其它内存段输出数据，检查和码用单独命令去穿制。注意在给出其它命令之前应先将穿孔机断开。

1.14　E　穿制检查和码和带尾命令

给出E命令时计算机也自动停机以便将穿孔机接通。按“继续键”就穿制检查和码以及0200_8的带尾码。当穿制带尾时，将穿孔机断开按“停机”键，如再继续使用的话应重新给定ODT的起始地址并按启动键。

2．其它控制技术

2.1　电传标志触发器的处理　有时待调试的程序需要将电传标志触发器置1，以便能继续输出，这时程序的输出指令如下安排：

```
TSF
JMP .-1
TLS
```

因为在ODT程序执行过程中一般是将电传标志触发器置0，这样上列程序将在JMP.-1处循环。为了避免这种情况，就需要在发出G或C命令转为执行目的程序时，使电传标志触发器置1。这种工作由改变内存单元XCOIST－3（这个地址通常为7341或1341）的内容为NOP(7000)来完成。例如：

```
7341/6042  7000 ↓     (高地址形式)
1341/6042  7000 ↓     (低地址形式)
```

2.2 现行内存单元

当给出命令G，C，B，T，E，及P等之后，现行内存单元的地址或上一个检验过的内存单元的地址被ODT保存下来。这样的内存单元，为现行内存单元。对于现行内存单元只须给出／命令，无须再打出该单元的地址码就可以打开它并检验其内容。

2.3 用ODT命令编制的符号带 ODT也可以正确地辨认用ODT命令编制的符号带。例如一个符号带上如果穿制下列符号

1021/1157 ↑ 7775

则当将这个符号带输入给ODT程序时，它就将1021内存单元打开并打印其内容，并将其内容换成1157，然后将157内存单元打开并更换其内容为7775。类此其它命令如B，C，P等也可以按这种方法进行。

2.4 用高速穿孔机穿制二进制纸带的步骤如下：

(1) 发出穿孔命令 nnnn；mmmm P，然后计算机自动停机。

(2) 令SR＝7231（或1231）按“送地址键”。

(3) 令SR＝6026按“存数键”。

(4) 令SR＝6021按“存数键”。

(5) 令SR＝7225（或1225）按“送地址”键及“启动”键，于是穿制带头（0200_8）码。

(6) 当带头穿到一定长度时，按“停机”键。

(7) 令SR＝7203（或1203）按“送地址”键及启动键，于是输出规定内存段的内容。

(8) 如果还要输出其它内存段的内容于同一条带上，则应该重复1，2，3，4，和7等步骤。

(9) 如不再穿制其它内存段的内容，则按停机键并令SR＝

7222（或1222），按送地址键和启动键，就可穿制检查和码以及带尾（也是0200_8码）。

(10) 当已经穿制足够的带尾时，按停机键，然后拿开纸带。

(11) 如果继续使用ODT程序，则应将第3第4步改变了的内容恢复过来。

(a) 令SR= 7231（或1231），按送地址键。

(b) 令SR= 6046，按存数键。

(c) 令SR= 6041，按存数键。

(12) 令SA= 7000（或1000）按送地址键及启动键，则ODT又重新处于待机状态。

2.5 关于中断 当遇到断点时，ODT执行一条IOF指令（但当执行nnnn C命令时不执行IOF指令），这是为了在ODT程序打印断点信息时不发生中断，以保护信息的输出。

然而当接受“G”或“C”命令，又转去执行目的程序时，ODT无法将容许中断触发器置“1”。

2.6 八进制输出 令屏蔽值为0，发出W命令就能由电传将检索范围内的内容打印出来。

2.7 间接访址 如遇到一个间接编址的指令，可以用↑及←符号打开所需内存单元。

3. 出错处理 在ODT中只有命令和4位八进制字符是合法的。其余均为非法字符，对于非法字符ODT回答“？”号并将该字符或全行取消。单独发出G命令也作为出错处理。这时ODT不回答问号但将程序转到0地址。发出任何穿孔命令而穿孔机处于接通状态时如再打出任何符号均将穿制到二进制带上，这意味着纸带不能正确输入和运行，也是出错的例子之一。

三、 应 用 和 举 例

1。没有打印符号的电传操作的表示方法：

(CR) =回车

(LF) =换行

(H) =计算机执行停机指令(HLT)

(CONT) =计算机继续键

(PON) =穿孔机接通

(POF) =穿孔机断开

(LEAD) =产生带头

(BIN) =穿制二进制带

(CKSMT) =穿制检查和码和带尾

2。举例用的目的程序

```
                /这是一个利用预先存好的两个八进制数进行+、-、
                /×、÷的算例。唯一的输入为运算符*+-/四种符
                /号，操作数或由操作面板或由ODT给定，结果放在
                /ANSR单元中
        0360 * 360
0360    0000 TYPE,0
0361    6041        TSF
0362    5361        JMP   ,-1
0363    6046        TLS
0364    7200        CLA
0365    5760        JMP I TYPE
        0400 * 400
```

```
0400  6046          TLS        /起动电传打字机
0401  6031  READ,   KSF
0402  5201          JMP .-1
0403  6036          KRB        /将字符读入AC并存贮
0404  3270          DCA TEMP
0405  1270          TAD TEMP
0406  4671          JMS I TYBEJ/进行回答
                    /核对输入字符并根据输入的运算子转到适当子程
                    /序
0407  1270  LEGL,   TAD TEMP /取运算子
0410  1272          TAD  M257/是"/"吗?
0411  7450          SNA
0412  5253          JMP DIVD /是,转除法子程序
0413  1273          TAD  C2  /否,是"-"号吗?
0414  7450          SNA
0415  5234          JMP SVBT /是,转减法子程序
0416  1273          TAD  C2  /否,是"+"吗?
0417  7450          SNA
0420  5227          JMP  ADD /是,转加法子程序
0421  7001          IAC        /否,是"*"吗?
0422  7650          SNA  CLA
0423  5242          JMP  MULT/是,转乘法子程序
0424  1274          TAD  C277/否,这不是合法字符
0425  4671          JMS  I TYPEJ /打印"?"号
0426  5201          JMP  READ /转去取另外的字符
```

```
                /将STOR 2中的数与STOR 1中的数相加将结果存
                /入ANSR中，按继续键取另外的运算子
0427 1275 ADD,  TAD STOR1
0430 1276       TAD STOR2
0431 3377       DCA ANSR
0432 7402       HLT
0433 5201       JMP READ
                /将STOR1中的数减去STOR2中的数结果存于
                / ANSR按继续键取另外的运算子
0434 1276 SUBT, TAD STOR2/取STOR2中的数并求补
0435 7041       CIA
0436 1275       TAD STOR1
0437 3277       DCA ANSR
0440 7402       HLT
0441 5201       JMP READ
                /将STOR1的数与STOR2中的数相乘结果存入
                / ANSR作法是将STOR1中的数加N次，N=STOR2
                /中的数然后按继续键取另外的运算子
0442 1276 MULT, TAD STOR2/将STOR2中数形成补码作
0443            CIA       /为重复相加的次数
0444            DCA CNTR
0445            TAD STOR1
0446            ISZ CNTR
0447            JMP .-2
0450            DCA ANSR
```