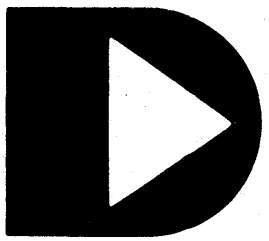
DOS. 2.6 ADDENDUM

User's Guide

Version 2

October, 1980

Document No. 50595



DATAPOINT

DOS. 2.6 ADDENDUM
USER'S GUIDE

Version 2

October, 1980

Document No. 50595

NOTICE

Datapoint strongly recommends that its customers use Datapoint Customer supplies. These disks, diskettes, cassettes, ribbons and other products are certified by Datapoint to meet all Datapoint Hardware specifications for consistent optimum performance.

TABLE OF CONTENTS

CHAPTER 57. ROUTINE ENTRY POINTS

These entry points are contained in a file called DOS/EPT released with DOS. versions 2.6. This is an updated listing of the entry points to replace Chapter 57 of the DOS. 2.6 User's Guide.

Loader Routines

01000	BOOT$	RELOAD THE OPERATING SYSTEM
01003	RUNX$	LOAD AND RUN A FILE BY NUMBER
01006	LOADX$	LOAD A FILE BY NUMBER
01047	GETNCH	GET THE NEXT DISK BUFFER BYTE
01052	DR$	READ A SECTOR INTO THE DISK BUFFER
01055	DW$	WRITE A SECTOR FROM THE DISK BUFFER
01060	DSKWAT	WAIT FOR DISK READY UNLESS: ARC AND NOT 9320: RETURN; NOT ARC AND 9320: RE-NORMALIZE BUFFER POINTER
01173	DWNV$	DW$ WITHOUT WRITE VERIFY 2.3 FORWARD ONLY

Time-critical Sceduling Routines

01033	CS$	CHANGE PROCESS STATE
01036	TP$	TERMINATE PROCESS
01041	SETI$	INITIATE A TIME-CRITICAL PROCESS
01044	CLRI$	TERMINATE A TIME-CRITICAL PROCESS

Generalized Processing Routines

01011	INCHL	INCREMENT HL
01022	DECHL	DECREMENT HL
01140	ERROR$	CLOSE ALL FILES, EXIT CHAIN, AND RELOAD DOS.
01143	BLKTFR	TRANSFER A BLOCK OF MEMORY
01146	TRAP$	SET A DISK ERROR CONDITION TRAP
01151	EXIT$	CLOSE ALL FILES AND RELOAD DOS
01170	WAIT$	DOS WAIT-A-WHILE "NOP" ROUTINE
07400	DOSFNC	DOS FUNCTION LOADER

Symbolic File Handling Routines

01063	PREP$	OPEN OR CREATE A FILE
01066	OPEN$	OPEN AN EXISTING FILE
01071	LOAD$	LOAD A FILE BY NAME
01074	RUN$	LOAD AND RUN A FILE BY NAME

Logical File Handling Routines

01077	CLOSE$	CLOSE A FILE
01102	CHOP$	DELETE SPACE IN A FILE
01105	PROTE$	CHANGE THE PROTECTION ON A FILE
01110	POSIT$	POSITION TO A RECORD WITHIN A FILE
01113	READ$	READ A RECORD INTO THE BUFFER
01116	WRITE$	WRITE A RECORD FROM THE BUFFER
01121	GET$	GET THE NEXT BUFFER BYTE
01124	GETR$	GET AN INDEXED BUFFER BYTE
01127	PUT$	STORE INTO THE NEXT BUFFER POSITION
01132	PUTR$	STORE INTO AN INDEXED BUFFER POSITION
01135	BSP$	BACKSPACE ONE RECORD

Keyboard and Display Routines

01154	DEBUG$	ENTER THE DEBUGGING ROUTINE
01157	KEYIN$	OBTAIN A LINE FROM THE KEYBOARD
01162	DSPLY$	DISPLAY A LINE ON THE SCREEN

Cassette Tape Handling Routines

010000	TPBOF$	POSITION TO THE BEGINNING OF A FILE
010005	TPEOF$	POSITION TO THE END OF A FILE
010012	TRW$	PHYSICALLY REWIND A CASSETTE
010017	TBSP$	PHYSICALLY BACKSPACE ONE RECORD
010024	TWBLK$	WRITE AN UNFORMATTED BLOCK
010031	TR$	READ A NUMERIC CTOS RECORD
010034	TREAD$	TR$ AND WAIT FOR LAST CHARACTER
010037	TW$	WRITE A NUMERIC CTOS RECORD
010042	TWRIT$	TW$ AND WAIT FOR LAST CHARACTER
010045	TFMR$	READ THE NEXT FILE MARKER RECORD
010050	TFMW$	WRITE A FILE MARKER RECORD
010053	TTRAP$	SET A CASSETTE ERROR TRAP
010056	TWAIT$	WAIT FOR I/O COMPLETION
010061	TCHK$	GET I/O STATUS

Command Interpreter Utility Routines

01165	CMDINT	RETURN TO COMMAND INTERPRETER & SCAN MCR$ LINE
013400	DOS$	RETURN TO COMMAND INTERPRETER & DISPLAY SIGNON
013403	NXTCMD	RETURN TO COMMAND INTERPRETER & SAY "READY"
013406	CMDAGN	RETURN TO COMMAND INTERPRETER & GIVE MESSAGE
013411	GETSYM	GET THE NEXT SYMBOL FROM MCR$
013414	GETCH	GET THE NEXT CHARACTER FROM MCR$
013417	GETAEN	GET THE AUTO-EXECUTE PFN
013422	PUTAEN	SET THE AUTO-EXECUTE PFN
013425	GETLFB	OPEN THE USER-SPECIFIED FILE (LFN IN B)

```
013430     PUTCH      STORE THE NONBLANK CHARACTER IN THE "A" REGISTER
013433     PUTCHX     STORE THE CHARACTER IN THE "A" REGISTER
013436     PUTNAM     FORMAT A FILE NAME FROM A DIRECTORY BLOCK
013441     MOVSYM     OBTAIN THE SYMBOL SCANNED OFF BY GETSYM
013444     GETDBA     OBTAIN THE DISK CONTROLLER BUFFER ADDRESS
013447     SCANFS     SCAN OFF A FILE SPECIFICATION
013452     TCWAIT     TEST CONTROLLER MEMORY & WAIT FOR COMMAND
013455     INPTR      BYTE POINTER FOR USE WITH GETCH
```

Internal DOS Equivalences

```
00004      DOSPFN     PFN FOR USE BY DR$ AND DW$
00005      DOSPDN     PDN FOR USE BY DR$ AND DW$
00026      DOSPTR     BUFPTR USED BY GETNCH
00027      SDFLAG     SUB-DIRECTORY EXISTANCE FLAG
00030      SDNR       SUB-DIRECTORY NUMBERS (1 PER DRIVE)
01375      DISKTYPE   DISK CONTROLLER TYPE CODE
. 0=NONE(ARC AP); 1=MIDS; 2=9370; 3=9374; 4=9320; 5=9350; 6=9380; 7=1842
0          $ARC       ARC AP
1          $9390      9390 DISKS
2          $9370      9370 DISKS
3          $9374      9374 DISKS
4          $9320      9320 DISKS
5          $9350      9350 DISKS
6          $9380      9380 DISKS
7          $1842      1842 DISKS
01376      DISKADR    DISK CONTROLLER ADR
01377      DOSFLAG    DOS FLAG BYTE
01200      DOSFL2     DOS FLAG BYTE #2
01400      MCR$       MONITOR COMMUNICATION REGION
01544      LFT        LOGICAL FILE TABLE
4          TFT        TEMPORARY FILE TABLE
0<4        LF0        LOGICAL FILE #0
1<4        LF1        LOGICAL FILE #1
2<4        LF2        LOGICAL FILE #2
3<4        LF3        LOGICAL FILE #3
```

Logical File Table Description

```
0          PFN        (1) PHYSICAL FILE NUMBER
1          PDN        (1) PHYSICAL DRIVE NUMBER AND PROTECTION
2          LRN        (2) NEXT LRN TO BE DEALT WITH
4          BLRN       (2) FIRST LRN WITHIN CURRENT SEGMENT
6          CSD        (2) CURRENT SEGMENT DESCRIPTOR
8          RIBCYL     (1) PDA (MSB) OF RIB
9          RIBSEC     (1) PDA (LSB) OF RIB
10         MAXLRN     (2) LARGEST LRN REFERENCED
12         LRNLIM     (2) RESERVED FIELD (INITIALLY ZERO)
```

```
14          BUFADR     (1) CURRENT CONTROLLER BUFFER ADDRESS
15          XXXXXX     (1) NOT USED
```

Dos Memory Mapping

```
000000      LDRAD$     SYSTEM LOADER
000053      BOOTDRIV   DRIVE DOS WAS BOOTED FROM
001000      DOSAD$     DOS RESIDENT
004000      OVLAD$     DOS OVERLAYS
005400      DSPAD$     CRT WRITE ROUTINE
005572      KEYAD$     KEYBOARD READ ROUTINE
006000      DEBAD$     DISK DEBUG
07400       FLDAD$     DOS FUNCTIONS PAGE
010000      CASAD$     CASSETTE TAPE DRIVERS
013400      CMDAD$     COMMAND INTERPRETER
017000      COVAD$     COMMAND INTERPRETER OVERLAYS
0160000     BFRTABLE   BASE ADR OF IN-MEMORY 'DISK' BUFFERS
```

DOS FLAG byte #1 (location 01377)

```
1<7         ABTIF      1... .... ABNORMAL PROGRAM COMPLETION
1<6         NETACT     .1.. .... DOS NET FACILITY ACTIVE
1<5         UBOOT      ..1. .... DISK WAS BOOTED FROM DISK
1<4         CHACT      ...1 .... CHAINING ACTIVE
1<3         IS55AVL    .... 1... 5500 INSTRUCTIONS AVAILABLE
1<2         PSACT      .... .1.. PS ACTIVE
1<1         RAMAVL     .... ..1. RAM DISPLAY AVAILABLE
1<0         ROMBOOT    .... ...1 BOOTSTRAP LOADED FROM ROM
```

DOS FLAG byte #2 (location 01200)

```
1<7         $MIDSDSK   1... .... BOOTED FROM/ RUNNING ON  MIDS DISK
1<6         $REMARC    .1.. .... REMOTE CONSOLE FACILITY SUPPORT ACTIVE
1<5         $1800CPU   ..1. .... RUNNING ON AN 1800 CPU
1<4         $UPSACT    ...1 .... UPS IS ACTIVE
1<3         $MEMD      .... 1... MEMORY RESIDENT OVERLAYS DESTROYED
1<2         $MEMU      .... .1.. MEMORY RESIDENT OVERLAYS PRESENT
1<1         $DOSINIT   .... ..1. DOS DISK CONTROLLER INITIALIZATION FLAG
1<0         $ONESHOT   .... ...1 ONE-SHOT BIT FOR STARTUP PROCEDURES
```

DOS Keyboard/Display Routine Control byte equates

```
3           EOS        END OF STRING, NO CR/LF
011         H          HORIZONTAL POSITION FOLLOWS
013         V          VERTICAL POSITION FOLLOWS
015         EOL        END OF LINE, CR/LF
021         ECF        ERASE CURSOR TO END OF FRAME
022         ECL        ERASE CURSOR TO END OF LINE
```

023	R	ROLL SCREEN UP ONE LINE
11	BL	NUMBER OF BOTTOM LINE ON SCREEN
0	TL	NUMBER OF TOP LINE ON SCREEN
79	RC	NUMBER OF RIGHTMOST COLUMN ON SCREEN
0	LC	NUMBER OF LEFTMOST COLUMN ON SCREEN

CHAPTER 59. AIMDEX - THE AIM FILE GENERATOR

AIMDEX takes a standard text file, together with key fields specified on the command line, and generates the AIM index required by programs that allow AIM access.

AIMDEX will run standalone or under ARC on any processor with the 5500 instuction set and at least 48K of memory. Any additional memory will be utilized by AIMDEX to provide faster execution. Note that on smaller memory configurations, AIMDEX will create a temporary scratch file for use in generating the index. AIMDEX requires DOS.D 2.5 or later to run.

59.1 AIMDEX Command Line

59.1.1 Command Line Format

The format of the AIMDEX command line is

AIMDEX <txtfile>[,<aimfile>];[<options>,]<key fields>

Items in angle brackets (<>) are values to be filled in. Items in square brackets ([]) are optional.

<txtfile> is the text file to be AIMed.
- The file name is required.
- The extension defaults to /TXT.

<aimfile> is the AIM output file.
- The file name defaults to that of the text file.
- The extension defaults to /AID.
- The drive defaults to that of the text file.

<options> is a list of options.
- The valid options are
  ? or H to print a short help message,
  D=<c> to change the don't care character from "?",
  F to direct the information output to a print file,
  I to display the header sector information,
  L to direct the information output to a local printer,
  O to overwrite an existing invalid /AID file,
  P<nnn><=|#>"<cccccccc>" to select primary records,

R to re-AIM using the information from the header sector,
S to direct the information output to a servo printer,
U to treat all lower case characters as upper case,
U=<lll>-<uuu> is same as U, but change upper case range.

- Options may run together, or can be separated by a comma and/or space(s).

<key fields> is a list of fields to key on.

- The format of each key field is n-m or Xn-m, where the letter X preceding a key field specification indicates that the field is an "excluded field", n is the position of the first character of the key field, and m is the position of the last character of the field.
- A single character field can be specificed by omitting the -m.
- Multiple key fields must be separated by a comma and/or space(s).

Options and key fields can be intermixed. They can be continued onto the next line by replacing one of the separating commas with a colon (:). Any characters on the command line following the colon will be ignored, and command line processing will continue with the first character of the next line entered. (The command continuation line will be prompted for by ': '.)

59.1.2 Text Input File

The input file required by AIMDEX must be of standard text file format. The only data characters accepted are 01 through 0176 ("~"). Characters in the range 0200 thru 0377 are converted to 0174 ("¦") prior to processing. (Note: The data file is not modified in any fashion; the conversion is internal to AIMDEX only. The conversion is done to conform to DATASHARE text file processing format.) The control bytes accepted are 003 (end of sector), 011 and the following byte (space compression), 015 (end of record), and 032 (deleted). If invalid text is encountered, AIMDEX will abort.

Records are internally padded with blanks if necessary (if a line ends before the last key field, the remaining characters in the line are assumed to be blanks). There is no maximum limit on the line length, but characters beyond column 255 are ignored by AIMDEX. The text file may be empty, and the maximum file size is limited only by DOS.

59.1.3 AIM Output File

The AIM index file (/AID) is the only output file produced by the AIMDEX utility.

59.1.4 Command Line Options

59.1.4.1 Help

The ? or H option will cause the display of a short help message that includes a brief summary of the command line format and available options.

59.1.4.2 "Don't Care" Character

The D=<c> option changes the default "don't care" character. If an AIM access has keys using this character, the positions in the record corresponding to the "don't care" characters in the key can contain any character. The "don't care" character can be any character from 041 (!) through 0176 (~) (this includes all alphanumeric and special characters that can be entered from the keyboard except space and delete). It defaults to 077 (?).

```
AIMDEX DATA;1-40,D=@
```

The above command line will result in a default "don't care" character of "@".

The "don't care" character selected has no effect on either AIM index generation speed or the AIM access time. This character shouldn't be changed from the default unless it could appear in a key; if it could be a character in a key, it should be changed. The "don't care" character selected should be one which does not occur (or occurs infrequently) in the data file.

59.1.4.3 Information Display

The I option can be used to display the information contained in the header sector of the /AID file. This provides a means of examining the key fields and options selected when the file was initially AIM'd.

```
AIMDEX DATA;I
```

The above command line will cause the key field specifications and the options to be displayed on the processor screen.

This information can also be directed to either a print file, local printer, or servo printer. This is accomplished using the F, L, or S options respectively. Use of either the F, L, or S option implies the I option. These options are mutually exclusive.

59.1.4.4 Overwrite Invalid /AID

When AIMDEX detects that the /AID file to be generated already exists, it verifies the content and format of the /AID file to ensure that it was created by AIMDEX and is valid. If the header sector is found to be invalid, AIMDEX will abort. The O option can be used to override this abort and force AIMDEX to overwrite an existing invalid /AID file. Note that interrupting AIMDEX while it is processing a file will result in the corresponding /AID file having an invalid header sector.

```
AIMDEX DATA;1-40,O
```

The above command line will cause AIMDEX to ignore an invalid header sector and overwrite any existing DATA/AID file.

59.1.4.5 Primary Record Select

The P option is used to select primary records. If the P option is given, only those records that meet the specified conditions are AIMed; all others are ignored.

The format of the P option is P<nnn><=¦#>"<cccccccc>". <nnn> is the starting column of a primary record selection string, <=¦#> means that either "=" or "#" can be specified, and <cccccccc> is the primary record selection string. All records are read and matched against the string starting in the specified column. If "=" was specified in the P option, the record is selected if the specified field matches the primary record selection string; if "#" was specified, the record is selected if the string doesn't match.

```
AIMDEX DATA;1-40,P60="AbC"
```

The above command line will cause AIMDEX to select, for AIMing, only those records which contain the string "AbC" starting

in column 60 of the record.

Any printable character can be included in the P option selection string, including blanks, commas, and quotes ("). To include a quote in the string, precede it with another quote (two quotes in a row are treated like a single quote instead of string delimiters): the option P32="""" selects only those records that have a quote in column 32. The primary record selection string is limited to 8 characters, and must end at or before column 255.

59.1.4.6 Re-AIM

The R option causes AIMDEX to re-AIM the file using the key specifications and options from the header sector. This allows the user to re-AIM the file using the same key specifications and options as given on the original AIMDEX command line. If this option is used, the header sector must be valid.

```
AIMDEX DATA;R
```

The above command line will cause AIMDEX to use the information from the header sector of DATA/AID. The D, U, and P options can be used in conjunction with the R option if it is desired to override selected parts of the information from the header sector.

```
AIMDEX DATA;R,D=@
```

The above command line will cause AIMDEX to re-AIM the file using the information from the header sector. The "don't care" character from the header sector will be overridden by the "don't care" character specified on the command line (i.e. The "don't care" character will be '@' in the above command line).

If key specifications are given on the command line in conjunction with the R option, the key specifications from the header sector will be overridden by the key specifications on the command line.

```
AIMDEX DATA;R,1-10
```

The above command line will cause AIMDEX to re-AIM the file using a single key specification of 1-10. The setting of the D, P, and U options will be taken from the header sector of DATA/AID.

59.1.4.7 Upper Case

The U option causes all lower case characters to be treated as upper case. The normal upper case range is 0101 thru 0132 ("A" thru "Z"). If this option is given, all characters in the range 0141 thru 0172 ("a" thru "z") will be AIMed as if they were the upper case equivalent. This causes the character case to be ignored when AIMDEXing and when looking up information through the AIM file.

```
AIMDEX DATA;1-40,U
```

The above command line will cause AIMDEX to treat the letters "a" thru "z" as if they were "A" thru "Z".

The upper case range can be changed by giving arguments to the U option: U=<lll>-<uuu>, where <lll> is the lower limit, and <uuu> is the upper limit, which are both interpreted as octal numbers (this is intended primarily for foreign character sets). The normal range is U=0101-0132 ("A"-"Z"). The minimum value for <lll> is 0100, and the maximum value for <uuu> is 0136.

```
AIMDEX DATA;1-40,U=0130-0132
```

The above command line causes AIMDEX to treat all characters in the range "x" thru "z" as upper case ("x", "y", and "z" will be treated like "X", "Y", and "Z"). Note that the leading zero is not required when specifying the upper case range numbers; arguments to the U option (and nowhere else) are automatically interpreted as octal numbers.

Lower case characters that are in the range specified to be converted are converted to upper case by ANDing them with 0337 (binary 11 011 111). This will convert 0141 to 0101 ("a" to "A"),..., and 0172 to 0132 ("z" to "Z").

The U option should be used when keys in the text file could be either case. For instance, a name might be entered as "Sam McDonald" or "SAM MCDONALD". If the U option is given, both names are equivalent; if the U option isn't given, the names are different.

Note that the range specified in the U option must be the same as the shift inversion limits specified when the DATASHARE VI interpreter is configured. Use of limits which are different from the DATASHARE VI shift inversion limits will cause an OPEN error. Consult the DATASHARE VI User's Guide (Model Code # 50536) for details on how to change the shift inversion limits using the

DATASHARE VI Configurator.

59.1.5 Key Fields

A key field is a field that each input record is AIMed on. For each line of text processed, text within the specified key fields is extracted and processed by AIMDEX. The resulting information is stored in the AIM file.

If an X is placed before a field specification, the field is marked as an "excluded field". Excluded fields are still counted as key fields and can be used in searches like any other key, but the information they contain is not indexed, and thus does not affect the contents of the AIM file.

Modifying the contents of a key field normally requires a DELETE/WRITE sequence in order to retain AIM access capability to that record. Excessive DELETE/WRITEs tend to degrade the performance of AIM in a manner similar to that of ISAM. This problem can be avoided by defining the key field as an excluded field. This allows modification of the field contents by an UPDATE, thereby reducing the degradation of access speed to the AIM file. Since an excluded field is not indexed, changing the contents of the key field does not prevent future accesses from finding the record.

***NOTE: UPDATEing of space compressed records or fields is not supported.**

An excluded field can be used in accessing the file provided at least one included field is also specified for the search. Since excluded fields are not indexed, a search cannot be composed entirely of excluded fields.

Up to 64 key fields can be specified. If multiple key fields are specified, they must be listed in ascending order (key field 2 can't occur in the data record before key field 1, etc.). Each field can be from 1 thru 124 characters in length, with the limitation that the last key field must not extend beyond column 255 of the record.

The order in which the key fields are specified is important. The first field specified is field 1, the second is field 2, etc. The field number must be known in order to access any keys indexed in that field (the field number is the same as the key number used when accessing the AIM file).

Key fields are not allowed to overlap; however, one key field can be a subfield (completely contained within) of another. Subfields must always be specified after the master field. A legal key field specification could be:

```
key fields     1-5,6-10,15-40,15-30,16-20,26-30,36-40
field number    1   2     3     4     5     6     7

text data      12345678901234567890123456789012345678901
key fields     ¦-1-¦¦-2-¦    ¦------------3-----------¦
                             ¦-------4------¦     ¦-7-¦
                              ¦-5-¦     ¦-6-¦
```

Note the subfields (4, 5, 6, and 7) in this example. Subfield 4 is specified after its master field 3. Field 4 in turn is the master field to subfields 5 and 6. All fields, including subfields, are specified in ascending order.

The following key field specification is illegal because the two fields aren't listed in ascending order:

```
key fields    30-40,10-20
field number    1     2

text data     12345678901234567890123456789012345678901
key fields              ¦----2----¦         ¦----1----¦
```

The following key field specification is illegal because the two fields overlap:

```
key fields    10-30,15-35
field number    1     2

text data     12345678901234567890123456789012345678901
key fields              ¦---------1---------¦
                             ¦---------2---------¦
```

The following key field specification is illegal because the subfield is given before the master field:

```
key fields    10-20,10-30
field number    1     2

text data     12345678901234567890123456789012345678901
key fields              ¦----1----¦
                        ¦---------2---------¦
```

The following key field specification is legal, but field 2 serves no purpose because it exactly redefines field 1:

```
key fields     10-20,10-20
field number     1     2

text data      12345678901234567890123456789012345678901234567890
key fields                  ¦----1----¦
                            ¦----2----¦
```

59.2 Key Field Selection

It is important to distinguish between a key field and a key. A key field is a range of columns, and a key is the characters within those columns in a line, or the string compared to those characters. Consider this example:

```
key field       1-5      10-14
field number   ¦-1-¦     ¦-2-¦
text line 1    123456789ABCDE
text line 2    HARRY TOLD JOE
```

Key field 1 is the range of columns from 1 through 5, and key field 2 is the range of columns from 10 through 14. Key 1 from text line 1 is "12345" (the characters from columns 1 through 5 in line 1). Key 2 from text line 1 is "ABCDE". Key 1 from text line 2 is "HARRY". Key 2 from text line 2 is "D JOE".

Key fields can often be selected to speed up AIM access. The more information AIM has to go on when looking up a record the faster it will be. There are several ways to increase the amount of information available for a search.

The amount of information generated is dependent on the length of the keys. Generally, longer keys generate more information. It is important to note, however, that this isn't always the case. **Making longer keys by including repetitive text (such as many trailing blanks, or including the same string in all keys) doesn't increase information content, and therefore won't speed up AIM access.**

Another way to speed up AIM searching is to design the data file with the most frequently used keys at the beginning of the record. This way less comparing will have to be done to determine that a key doesn't match what is being searched for. For example, if the primary key to a file is a user number, the number should

be the first data item in each record, and would be associated with key field 1.

One more way to speed up AIM access is to define seldom used fields as excluded fields (preceding the key field definition with "X" on the command line). This prohibits the excluded field from being used alone to access the data file, but it will speed up accesses that don't require that field.

When selecting key fields, there are three important points to remember: all key fields must end on or before column 255, a key field cannot be longer than 124 columns, and key fields cannot overlap.

APPENDIX H. SAMPLE AIM APPLICATIONS

This chapter presents two typical AIM applications. These examples emphasize some important syntactical points, cover some important design areas, and give an idea of the relationship between key field contents and AIM lookup. Example key fields in this chapter are enclosed in double quotes ("). A question mark (?) is used to indicate a "don't care" character within a key, and a period (.) is used to indicate a space. Unless otherwise specified, each character of each field is blank. The key specification format used in these examples is the format required by the DATABUS language. See DBCMPLUS User's Guide, Model Code # 50321, for details on the key specification format.

H.1 Job Database

H.1.1 Application Description

An employment agency has a database containing information on all job openings it needs to fill. When a new applicant is interviewed, his qualifications and desires are matched against all entries in the database to find suitable possible employers.

The database contains a record for each of the thousands of jobs. Each job record is of the format:

Field	Len	Columns	Descriptions
1	4	1-4	Office/plant region
2	1	5-5	"T" = in town
3	1	6-6	"S" = in state
4	5	7-11	Salary per year (each "$" = $10,000)
5	4	12-15	Salary in addition to key 4 (each "A" = $2000)
6	8	16-23	Hours per week (each "H" = 5 hours)
7	10	24-33	Job life (each "L" = 3 months)
8	2	34-35	Type of work
9	10	36-45	Experience required (each "X" = 6 months)
10	6	46-51	Education required (each "E" = more education)
11	4	52-55	Employer number (index into employer file)
12	120	56-175	Job description
13	75	176-250	Reserved for future expansion

The database uses AIM access instead of ISAM for two major reasons. AIM allows multiple key access, and most record lookups will rely heavily on the contents of multiple key fields. AIM also allows very flexible partial key lookups, and this feature will be used quite frequently, since often the contents of part of a key field won't matter.

The chain file used to AIM the database is:

```
AIMDEX JOBS:
1-4:   KEY01 = OFFICE/PLANT REGION
5-5:   KEY02 = "T" IF IN TOWN
6-6:   KEY03 = "S" IF IN STATE
:
7-11:  KEY04 = "$" FOR EACH $10,000 OF ANNUAL SALARY
12-15: KEY05 = "A" FOR EACH $2000 OF SALARY IN ADDITION TO KEY 4
:
16-23: KEY06 = "H" FOR EVERY 5 HOURS PER WEEK
24-33: KEY07 = "L" FOR EVERY 3 MONTHS OF JOB LIFE
:
34-35: KEY08 = TYPE OF WORK (KEY INTO WORK DEFINITION FILE)
:
36-45: KEY09 = "X" FOR EVERY 6 MONTHS EXPERIENCE REQUIRED
:
46-51: KEY10 = EDUCATION REQUIRED
:                "E     " = READING AND WRITING
:                "EE    " = HIGH SCHOOL DIPLOMA
:                "EEE   " = SOME COLLEGE
:                "EEEE  " = BACHELOR'S DEGREE
:                "EEEEE " = MASTER'S DEGREE
:                "EEEEEE" = DOCTORATE
:
52-55: KEY11 = EMPLOYER NUMBER (INDEX INTO EMPLOYER FILE)
<blank line>
```

H.1.2 Syntax and Design Points

There are several things to note in the syntax of the AIMDEX chain file. Note the use of comments after the terminating colon (the colon allows continuation of the command onto the next line, and anything after it is ignored). Documentation of this sort is an important tool in maintaining a database. It is also a very good reference when writing programs to access the database.

The lines containing a colon (:) as the first non-blank character are for documentation purposes only. This format can be

used to continue a comment from the previous line, or to separate major field groups (salary, education, hours, etc.).

Note that the last line of the chain file is blank. This is required because the colon used on the previous line (before the comment) will cause AIMDEX to expect another command continuation line. Without the blank line CHAIN would abort because there would be no input to AIMDEX's request.

This example demonstrates several important design points. Although the currently defined information takes less than a sector, a field was added to the end of the record to fill it out to a complete sector. This allows for future expansion without reformatting the entire database.

Note that a different character is used in each key field. Although this isn't necessary, there are two good reasons for doing it. First, it gives a wider range of characters in the record, and gives AIM more information to go on when searching, and so speeds up AIM access. Second, it makes it easier to pick out the different fields when manually scanning the database.

***NOTE: This example provides a means to perform range checking during record retrieval. In general, the use of repetitive data in key fields should be avoided wherever possible as repetitive data tends to degrade the performance of the AIM access.**

H.1.3 Record Contents

When the employment agency is notified of a new job opening they enter it into their job database, using a program that does AIM WRITES. They perodically reAIM the file (using AIMDEX), because AIM WRITES add sectors to the AIM file that slow down AIM access.

One job in the database is a temporary part time job as a receptionist for a local doctor. The job pays $6.45 an hour for 15 hours a week, and will last a year (until the doctor's regular receptionist is ready to start leaving her infant child at the day care center). All applicants must at least have a high school diploma and some experience with this type of work.

The doctor's office is near the local college, and the employment agency's code for its location is 0006. Therefore, the location key fields would be:

Key field 1 = "0006"
Key field 2 = "T" (the office is in town)
Key field 3 = "S" (the office is in state)

The job pays $6.45 per hour for 15 hours a week, which works out to an annual salary of $5031. The salary key fields would be:

Key field 4 = "....." (0 * 10,000)
Key field 5 = "AAA." (3 * $2000 = $6000)

Since the annual salary is less than $10,000, key field 4 is blank. The salaries closest to $5031 that can be represented are $4000 and $6000. $6000 is used because it is the smallest representable salary not less than the actual salary.

The other fields would be:

Key field 6 = "HHH....." (3 * 5 hours = 15 hours)
Key field 7 = "LLLL....." (4 * 3 months = 12 months)
Key field 8 = "03" (code for clerical/secretarial)
Key field 9 = "X....." (some experience required)
Key field 10 = "EE...." (high school diploma required)
Key field 11 = "0576" (employer number for the doctor)
Field 12 = "Receptionist for doctor, $6.45/hr, 15 hr/wk, 1 yr"

Another position in the database is for a permanent, full time electrical engineer. The position pays $25,000 a year. A bachelor's degree and at least 2 years experience are required. The job would require a move to another town within the state. The entry for this job would be:

Key field 1 = "0072" (code for the town)
Key field 2 = "." (out of town)
Key field 3 = "S" (in state)
Key field 4 = "$$..." (2 * $10,000 = $20,000)
Key field 5 = "AAA." (3 * $2000 = $6000)
Key field 6 = "HHHHHHHH" (8 * 5 hours = 40 hours/week)
Key field 7 = "LLLLLLLLL" (10 * 3 months = permanent)
Key field 8 = "06" (code for technical/engineering)
Key field 9 = "XXXX....." (4 * 6 months = 2 years experience)
Key field 10 = "EEEE.." (bachelor's degree)
Key field 11 = "0328" (employer number for the firm)
Field 12 = "Electrical engineer, $25,000/yr, out of town firm"

H.1.4 Record Lookup

When a new applicant is interviewed, his qualifications and desires are used to search the database for suitable jobs. The applicant's qualifications are matched to the education, experience, and type of work fields. His desires are matched against the job location, salary, hours, and job life fields. If a field is inapplicable, unknown, or doesn't matter, it can be filled with the "don't care" character, or it can be omitted from the list of keys given to the AIM lookup routine.

Consider the applicant Kathy. She's a senior at the local college and needs a part time job. She worked as a secretary in the dean's office during her junior year, and would like a similar position now. She wants to work at least 10 hours a week, but not more than 20. She isn't too particular about the money, but she needs at least $200 per month.

Given this information, it's a simple matter to supply the keys to the database search program. Kathy needs a job near the college, so the location keys would be:

```
Key 1 = "01X0006" (code for college area jobs)
Key 2 = "02XT"
Key 3 = "03XS"
```

Note that keys 2 and 3 don't matter, since key 1 is specific enough in this case to define the job location. "T" and "S" are therefore redundant, but are specified because the more information given to AIM the better.

$200 per month translates to a yearly salary of $2400. Since each "$" in the primary salary field represents $10,000, and each "A" in the additional salary field represents $2000, the salary keys would be:

```
Key 4 = "04X?????" or null (don't care how many tens of thousands)
Key 5 = "05LA" (at least $2000)
```

The salaries closest to $2400 that can be represented in the database are $2000 and $4000. If the keys for $4000 were given, they would rule out all salaries greater then $2000 but less than $4000. Since many salaries in this range are acceptable, the keys for $2000 would be given, and unacceptable salaries would be ruled out by other means (the actual salary could be part of the job description field, or an additional field could be added).

Since Kathy wants to work at least 10 hours a week, but not

more than 20, the hours key would be:

Key 6 = "06XHH??...." (at least 10 hours, not more than 20)

The first 2 characters of the key, "HH", indicate that at least 10 hours per week are required. The next 2 characters, "??", indicate that 15 and 20 hours are acceptable, but not required. The remaining characters in the key are blank ("." represents " " in keys in these examples), indicating that any job that requires that many hours is unacceptable (the matched job must have blanks in those positions). As in the salary, the actual hours per week could be part of the job description field, or an additional field could be added for that purpose.

Kathy needs the job for at least 9 months while she finishes her last year at school. After she finishes school she'll find a new job in her chosen profession. Given a period of 9 months in which to find a new job, the job life key would be:

Key 7 = "07XLLL???...." (at least 9 months, not more than 18)

Since Kathy worked the previous year as a secretary, she has a year of experience at secretarial work, and would like a similar position now. The type of work and experience keys would be:

Key 8 = "08X03" (code for secretarial positions)
Key 9 = "09X??........" (not more than 1 year experience)

Note the experience key. The important part here is the blanks at the end of the key, not the "??" at the beginning. Kathy doesn't need a job that REQUIRES a year of experience, but a job that DOESN'T require MORE than a year. Key 9 could also have been specified as "09R........".

Kathy is attending college, so her education level is "some college", so the job she gets can't require a college degree. She wants a somewhat challenging job, both for the experience and so she isn't bored by the job, so she wants a job that requires at least a high shool diploma. The education key to find a job matching these conditions would be:

Key 10 = "10XEE?..." (need high school , but not college degree)

The employer doesn't matter, so the employer key would be:

Key 11 = "11X????" or null

The employer key will usually not matter, but sometimes it might.

The main reason for including it as an AIMed key is not for the job search program, but for a program that lists available job sorted by employer.

Given these keys, Kathy's job would match the doctor's receptionist, but not the electrical engineer.

H.2 Software Database

H.2.1 Application Description

Wizbang Software Products, Inc. writes and sells software for Datapoint processors. They have a database containing information on all their software. The database is used by sales and management personnel for various types of product inquiry and list generation.

The database contains a record for each product. Each record is of the format:

Field	Len	Columns	Description
1	8	1-8	Product Name (cccccccc)
2	7	9-15	Version number (x.x.x)
3	6	16-21	Release date (yymmdd)
			Support software ("*" if required)
4	1	22-22	Datapoint BASICPLS 2.1
5	1	23-23	Datapoint DATASHARE VI 1.1
6	1	24-24	Datapoint CHAINPLS 3.1
7	1	25-25	Wizbang extended utility package
8	1	26-26	Reserved for future product
9	1	27-27	Reserved for future product
10	1	28-28	Reserved for future product
11	1	29-29	Reserved for future product
12	1	30-30	Reserved for future product
13	1	31-31	Other product not in above list
14	4	32-35	Purchase price (nnnn)
15	3	36-38	Monthly maintenance fee (nnn)
16	4	39-42	Support programmer (nnnn)
17	4	43-46	Original programmer (nnnn)
18	80	47-126	Product description

AIM access is used for this database instead of ISAM for two main reasons. First, this database will be queried primarily by single key lookups, but at one time or another many of the keys

will be used in a lookup. This would require an index file for each of the accessed keys if ISAM was used, but with AIM only one index file is required. The second reason is the flexibility AIM gives when looking up records. Since AIM allows locating a record when the given key is anywhere within the specified key field, keyword lookups can be used to locate a product based on its description.

The chain file used to AIM the database is:

```
AIMDEX PRODUCT;:
1-8:    KEY 1 = PRODUCT NAME (CCCCCCCC)
9-15:   KEY 2 = VERSION NUMBER (X.X.X)
16-21:  KEY 3 = RELEASE DATE (YYMMDD)
22:     KEY 4 = "*" IF DATAPOINT'S BASICPLS 2.1 IS REQUIRED
23:     KEY 5 = "*" IF DATAPOINT'S DATASHARE VI 1.1 IS REQUIRED
24:     KEY 6 = "*" IF DATAPOINT'S CHAINPLS 3.1 IS REQUIRED
25:     KEY 7 = "*" IF WIZBANG EXTENDED UTILITY PACKAGE IS REQUIRED
26:     KEY 8 = RESERVED FOR FUTURE SUPPORT PRODUCT
27:     KEY 9 = RESERVED FOR FUTURE SUPPORT PRODUCT
28:     KEY 10 = RESERVED FOR FUTURE SUPPORT PRODUCT
29:     KEY 11 = RESERVED FOR FUTURE SUPPORT PRODUCT
30:     KEY 12 = RESERVED FOR FUTURE SUPPORT PRODUCT
31:     KEY 13 = "*" IF PRODUCT NOT LISTED ABOVE IS REQUIRED
32-35:  KEY 14 = PURCHASE PRICE (NNNN)
36-38:  KEY 15 = MONTLY MAINTENANCE FEE (NNN)
39-42:  KEY 16 = SUPPORT PROGRAMMER (NNNN)
43-46:  KEY 17 = ORIGINAL PROGRAMMER (NNNN)
47-126: KEY 18 = PRODUCT DESCRIPTION
<blank line>
```

H.2.2 Syntax and Design Points

Notice the colon following the semicolon in the first line of the chain file. This allows the first key field to be specified on the next line, even when no options are given. This is done only for clarity (so each key field spec gets its own line).

Note the single column key fields in this example. Here they are specified as a single number, while in the previous example both columns were given. The two methods of specifying a single column key field are equivalent.

Like the first example database, this one has been designed to allow for future expansion without reformatting the entire database.

A design factor that speeds up AIM access is the location of the fields. The most frequently accessed fields (especially key fields) are located at the beginning of the record. Since the product name is the key that will usually be used to locate a record, it is the very first data item. Most other key fields follow in order of decreasing probability of access (the ordering of some fields is for logical reasons). Following all key fields are the fields that contain information that the database isn't AIMed on. If there had been more than one of these fields in this example, they also would have been arranged in order of decreasing probability of access.

APPENDIX I. AIMDEX RUNTIME DISPLAY

I.1 Command Line Scanning

One or two lines are displayed if a fatal error is discovered while scanning the command line. The first line, if displayed, consists only of an up arrow (^) under the position in the command line where the error was discovered. The second line is one of the following:

ILLEGAL "DON'T CARE" CHARACTER
This error occurs if the character argument to the D option is less than 041 (!) or greater than 0176 (~).

ILLEGAL KEY FIELD SPECIFICATION
This error occurs if one of the columns of a key field isn't of valid numeric format, if the second column of a key field is less than the first, or if a key field width is greater than 124 columns.

ILLEGAL OPTION
This error occurs if an invalid option character is seen. A digit or X as the start of a key field, and ?, H, D, F, I, L, O, P, R, S, and U are the only valid options. Any other character on the command line after the semicolon (;) will produce this error.

ILLEGAL OPTION SYNTAX
This error occurs if a syntax error is encountered when scanning an option, such as a missing "=" after the D option, missing closing quote on the P option, etc.

ILLEGAL UPPER CASE RANGE
Illegal arguments were given to the U option (U=<lll>-<uuu>). <lll> or <uuu> was less than 0100 or greater then 0137, or <uuu> was less than <lll>.

KEY FIELDS OVERLAP
This error occurs when key fields overlap. Note, however, that subfields are allowed.

KEY FIELDS NOT IN ASCENDING ORDER
Key fields must be specified in ascending order. This error occurs if a key field starts in a column less than the starting

column of the preceding key field.

NO NON-EXCLUDED KEY FIELDS GIVEN
The purpose of AIMDEX is to create an associative index for the specified key fields for rapid lookup of records containing given keys in those fields. Therefore it wouldn't make sense to AIMDEX a file with no non-excluded key fields.

NUMBER EXPECTED
This error occurs if a number isn't found where expected.

NUMBER TOO LARGE
This error occurs if a key field column is greater than 255.

NUMBER TOO SMALL
This error occurs if a key field column is 0.

OPTION ALREADY SPECIFIED
This error occurs if the same option is given more than once.

PRIMARY RECORD SELECTION STRING TOO LONG
This error occurs if the primary record selection string (specified with the P option) ends after column 255.

TOO MANY KEY FIELDS
This error occurs if more than 64 key fields are given.

TOO MANY OR ILLEGAL FILE SPECS
This error occurs if more than 2 file specs are given, or if the portion of the command line before the semicolon (;) is unrecognizable.

INVALID HEADER SECTOR IN AIM FILE; FILE NOT MODIFIED
This error occurs if the header sector of an existing AIM file is invalid and the O option was not given on the command line. An invalid header sector can be caused by aborting AIMDEX prior to completion.

THE L, S, AND F OPTIONS ARE MUTUALLY EXCLUSIVE
This error occurs if any of these options are used in conjunction with each other.

THE O AND R OPTIONS ARE MUTUALLY EXCLUSIVE
This error occurs if the O and R options are used together on the command line.

AIM FILE NOT FOUND; CAN'T RE-AIM
This error occurs when the R option is used and the AIM file

specified on the command line can not be located.

FILE WAS AIMED WITH WRONG VERSION OF AIMDEX; CAN'T RE-AIM
This error occurs when the R option is used and the AIM file specified on the command line was created by an incompatible version of AIMDEX.

I.2 File Opening

After the command line has been scanned and verified, the text input file is opened. The message displayed for this is:

TEXT FILE name/ext:drive message

name/ext:drive is the text file name from the command line. message is one of:

NO SUCH DRIVE
An illegal volid was given, or the specified drive is off line. This is a fatal error and aborts AIMDEX.

NOT FOUND
The file could not be found on the specified drive, or on any drive if no drive was specified. This is a fatal error and aborts AIMDEX.

NAME REQUIRED
No file name was given on the command line. This is a fatal error and aborts AIMDEX.

OPENED
The specified file was found and successfully opened.

After the text input file is opened the AIM output file is opened. The message displayed for this is:

AIM FILE name/ext:drive message

name/ext:drive is the AIM file name from the command line. message is one of:

NO SUCH DRIVE
Same as for the input file.

SAME AS TEXT FILE
The AIM output file is the same as the text input file. This is a fatal error and aborts AIMDEX.

PROTECTED
The specified file is write protected. This is a fatal error and aborts AIMDEX.

OVERWRITTEN
The specified file already existed, and is overwritten by the new AIM file.

CREATED
The specified file did not already exist, and is created.

I.3 Text File Processing

The normal progress display is:

CURRENT TEXT SECTOR: sssss

where sssss is the number of the current sector of the text file.

When enough information has been compiled to write an AIM block, the message

WRITING BLOCK

is appended to the normal progress display. While this message is displayed the index data is put into the correct format and written to the AIM file. When this process is finished, the message is erased and processing continues as before.

The normal termination message is:

AIMED nnnnnn LINES IN sssss SECTORS

where nnnnnn is the total number of lines in the text file and sssss is the total number of sectors in the text file.

The following error(s) are detected while AIMDEXing the input file. AIM aborts if any of these error occur. These messages all give the line number the error was detected in.

ILLEGAL TEXT FILE FORMAT: LINE nnnnnn
This error occurs if AIMDEX detects something invalid about the

text file, such as: no 3 before the end of a sector, a missing end of file mark, or a binary 0 character in the file that is not part of the end of file mark.

Manual Name_____

Manual Number_____

READER'S COMMENTS

Did you find errors in this manual? If so, specify by page.

_____

_____

_____

_____

_____

Did you find this manual understandable, usable, and well-organized? Please make suggestions for improvement.

_____

_____

_____

_____

_____

_____

Name_____ Date_____

Organization_____

Street_____

City_____ State_____ Zip Code_____

All comments and suggestions become the property of Datapoint.

Fold Here

Fold Here and Staple

FIRST CLASS
Permit
5774
San Antonio
Texas

BUSINESS REPLY MAIL
No Postage Necessary if mailed in the United States

Postage will be paid by:

DATAPOINT CORPORATION
DIRECTOR OF SOFTWARE SUPPORT
8550 Datapoint Drive, Mail Station# N60
San Antonio, Texas 78284