HP Computer Systems
Training Course

HP260 To HP3000 Migration Training

Student Workbook

HELLO 0409, STUDENT.DEMO
: BASIC

>: FILE PRINTER; DEV=LP
> COPY OUTPUT TO "*PRINTER"
> LIST
> COPY OUTPUT TO DISPLAY

d = delete
i = insert
r = replace
// - Control y = cancel modification

HEWLETT PACKARD

Boeblingen General Systems Division
Herrenberger Str.130, 7030 Boeblingen

Printed in Federal Republic of Germany E0686

Copyright

NOTICE

The information contained in this document is subject to change without notice.

HEWLETT-PACKARD PROVIDES THIS MATERIAL "AS IS" AND MAKES NO WARRANTY OF ANY KIND, EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. HEWLETT-PACKARD SHALL NOT BE LIABLE FOR ERRORS CONTAINED HEREIN OR FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES (INCLUDING LOST PROFITS) IN CONNECTION WITH THE FURNISHING PERFORMANCE OR USE OF THIS MATERIAL WHETHER BASED OR WARRANTY, CONTRACT, OR OTHER LEGAL THEORY.

Some states do not allow the exclusion of implied warranties or the limitation or exclusion of liability for incidental or consequential damages, so the above limitation and exclusion may not apply to you. This warranty gives you soecific legal rights, and you may also have other rights which vary from state to state.

Hewlett-Packard assumes no responsibility for the use or reliability of its software on equipment that is not furnished by Hewlett-Packard

HP Computer Museum
www.hpmuseum.net

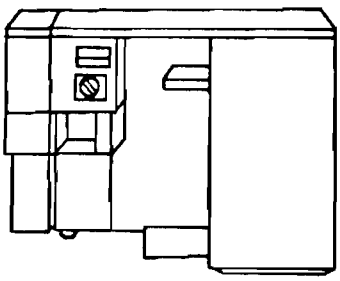
For research and education purposes only.

Preface

This is a special training to support the migration process from a HP260 to the HP3000 environment. On completion of the course the student will be able to upgrade existing HP260 applications to the HP3000 system.

Wesley Sawyer Book
Michael Buckle Book Lev

LASER PRINTED WORKBOOK



This workbook was printed on an HP2680 Laser Printer. The graphics were prepared on an HP graphics terminal using HPDRAW and interfaced to the laser printer by TDP/3000, a Hewlett-Packard text processor.

- OPTION SEGMENT = segment

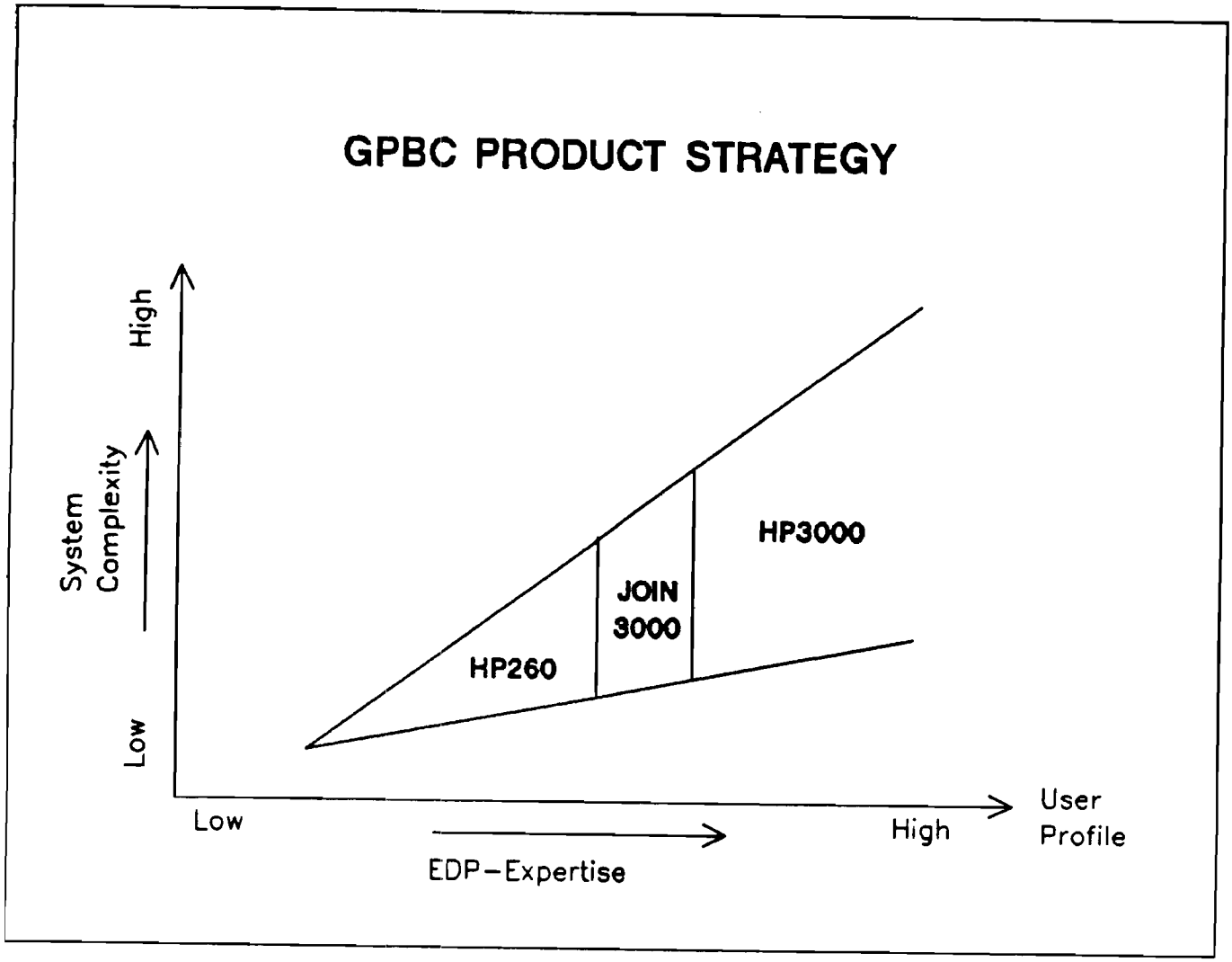
COMPILE MAIN [illegible] placed
[illegible] OB [illegible] to main
at run time

SPLINTR.PUB.SYS

COMPILE [illegible] USL = usl file

-
-

☐ JOIN/3000 Project

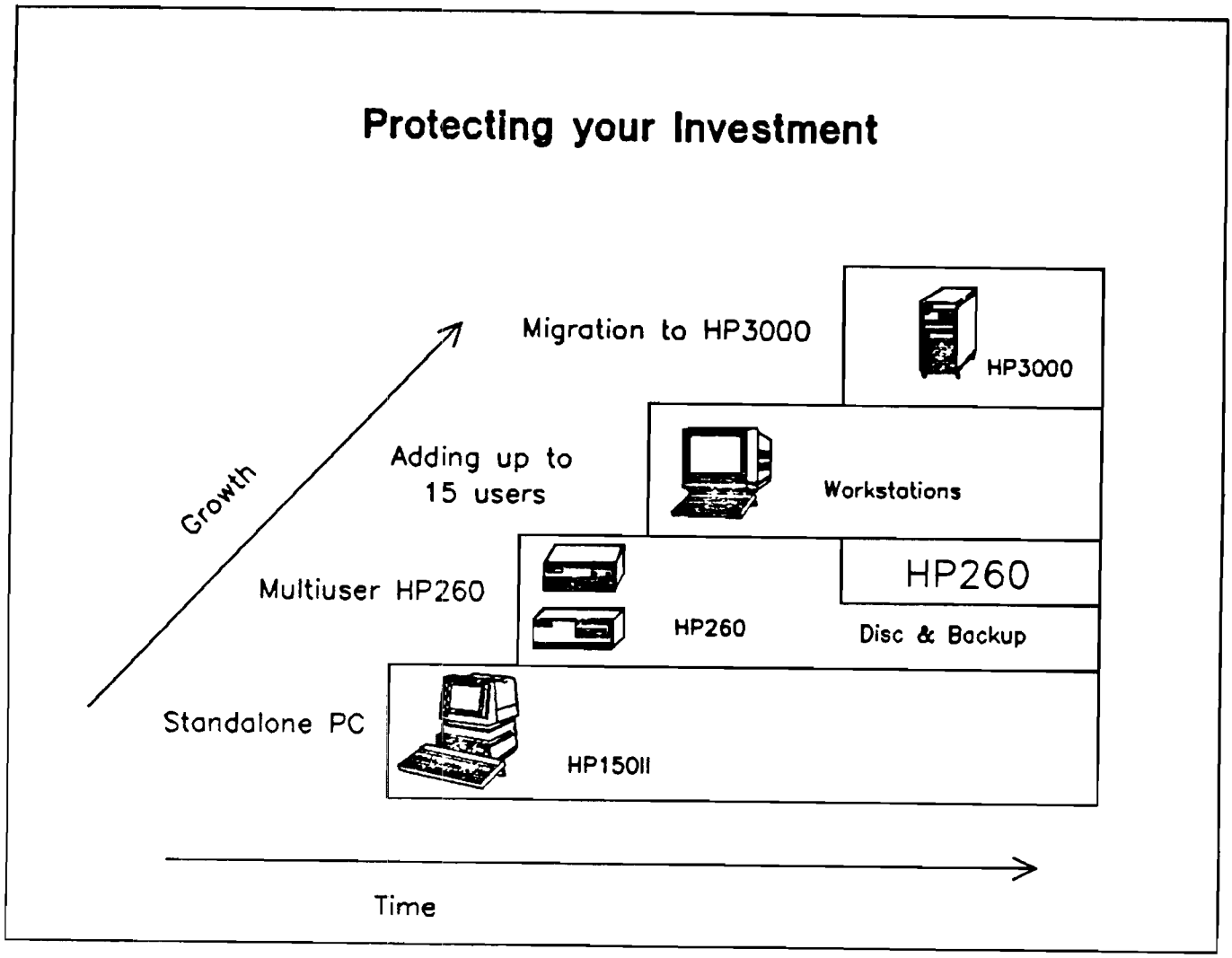


MI1 1.00a

JOIN/3000 Benefits

- Integration of the HP260 into HP3000 family
  - Minimum hardware changes
  - same user Interface
- Protection of Software Investment
- HP3000 Benefits
- Easy Migration

☐ JOIN/3000 Project

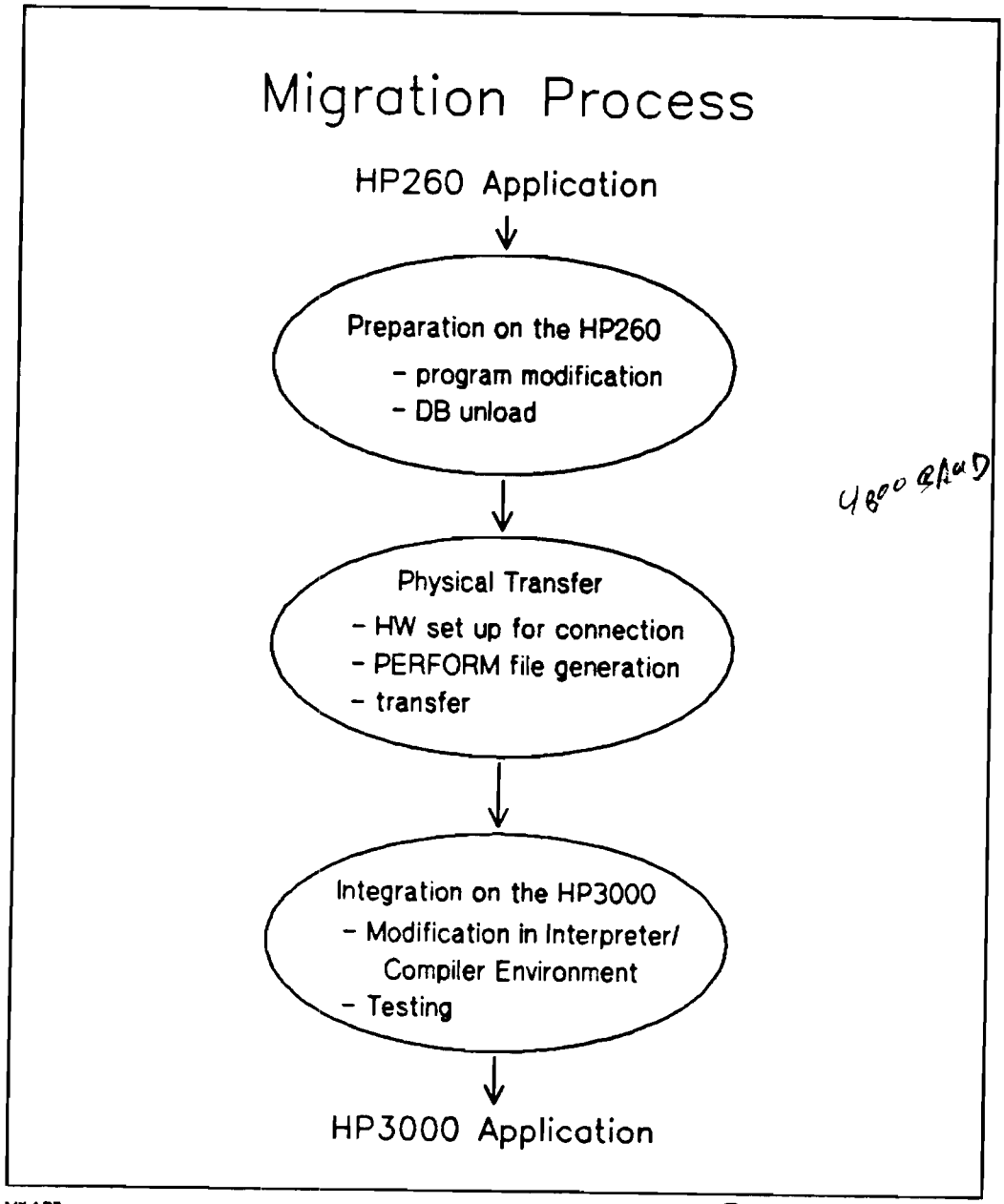


MI1 1.01a

HP3000 Benefits

- Performance
- Number of users
- BASIC compiler
- DB capabilities
- HP3000 Peripherals
- HP3000 Subsystems
- Account Structure
- Backup

☐ JOIN/3000 Project

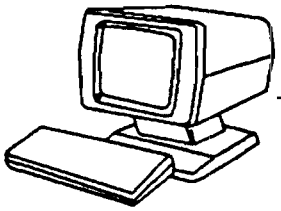


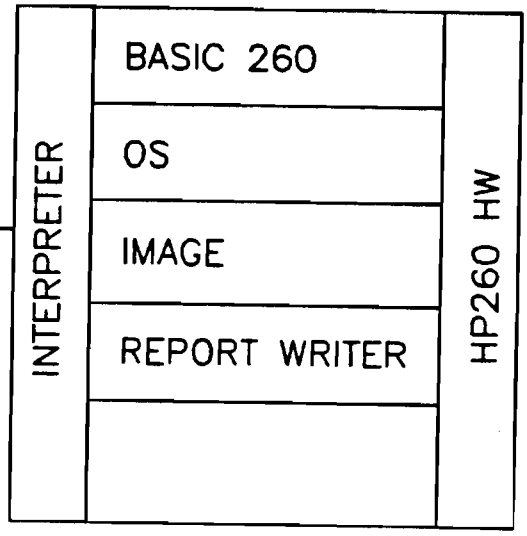
4800 BAUD Recommended

MH 1.03

☐ HP260 and HP3000 System Environments

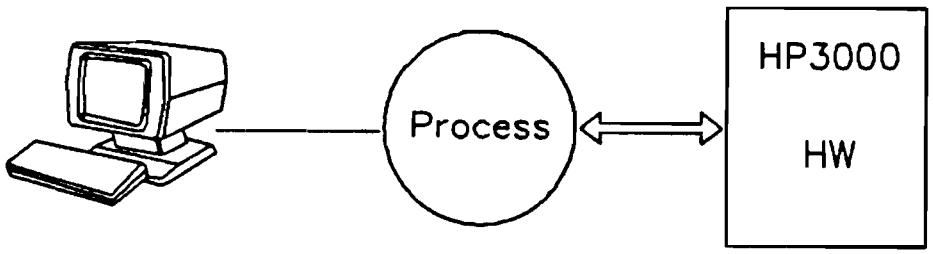
HP260 User Environment





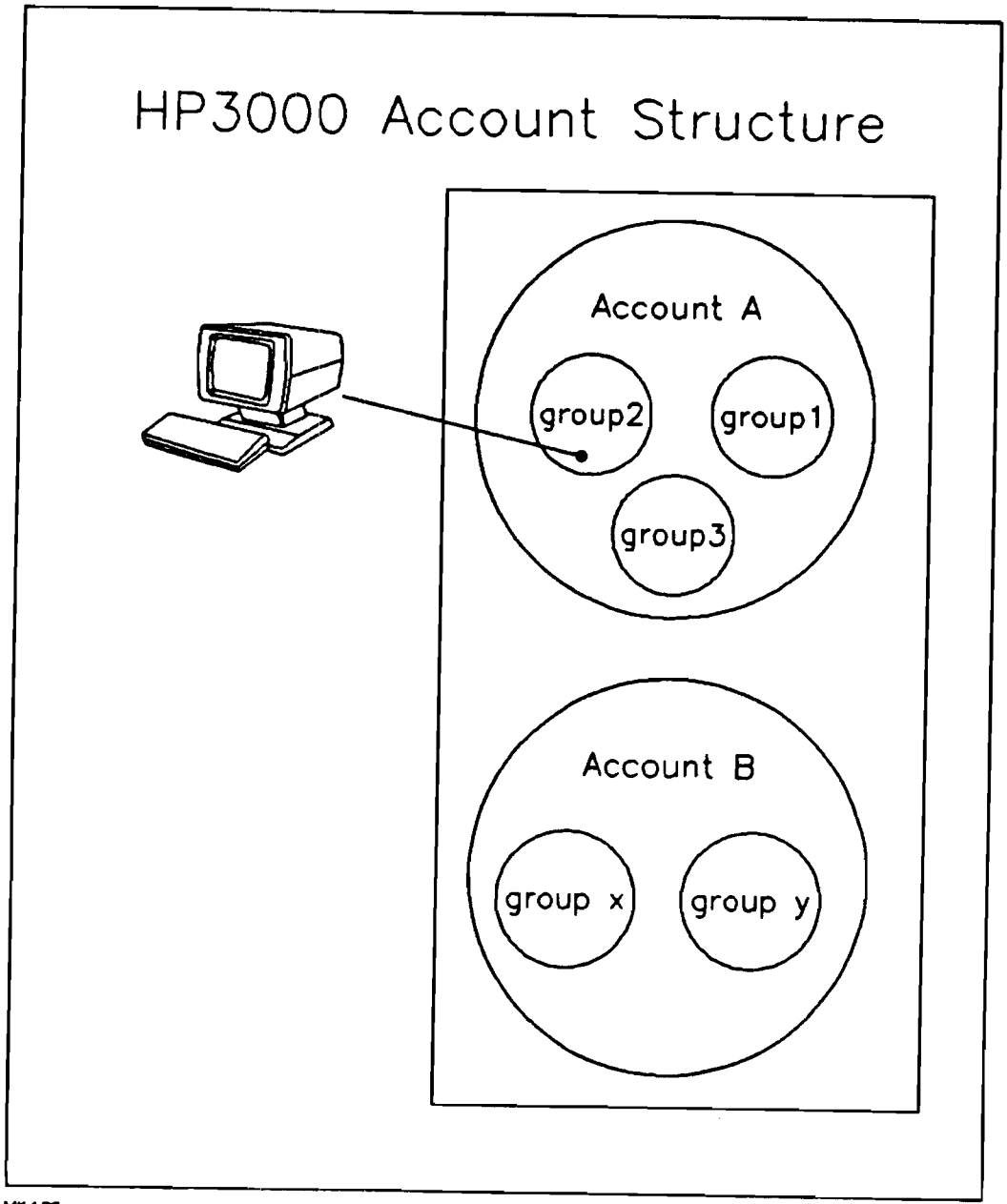
☐ HP260 and HP3000 System Environments

HP3000 Process Structure



Special Processes: Command Interpreter (CI) ⇒ MPE Command
HPBB Interpreter
HPBB Compiler
HPBB program (Compiled Version)

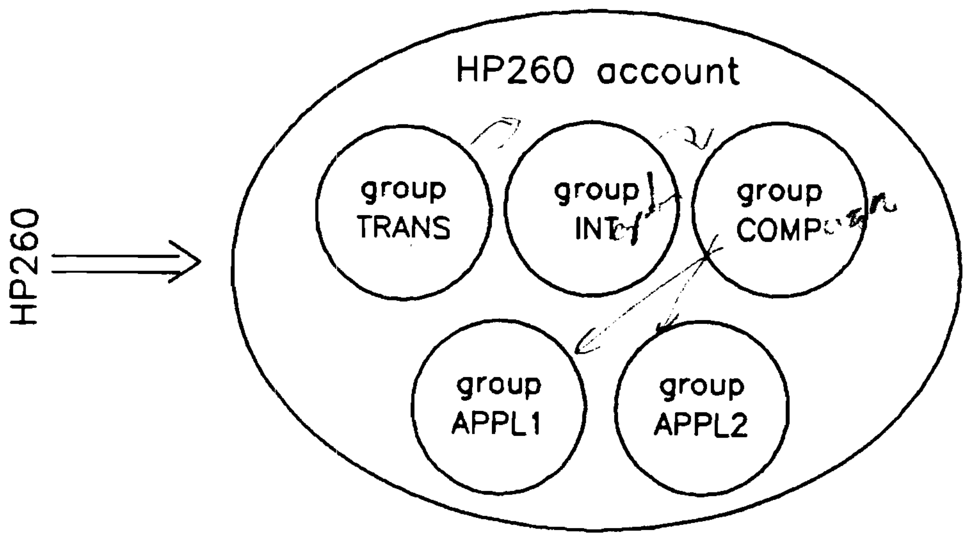
☐ HP260 and HP3000 System Environments



MH 1.05

☐ HP260 and HP3000 System Environments

Special Migration Account HP260



Keep jones + Steven in Pub Code Group.
Data types are only concern when calling different Language (Cobol, Pascal)

☐ Differences between HP260 and HP3000 Environments

Differences HP260/HPBB environment

DATA TYPES

File TYPES

BASIC

IMAGE

FORMS

REPORT WRITER

Memory Management

HPBB enhancements

☐ Differences between HP260 and HP3000 Environments

DATA TYPES

HP260	HP3000	
DIM	DIM	default length 18 char
INTEGER (I)	SHORT INTEGER (I)	16 bit
SHORT (S)	SHORT DECIMAL (K2)	32 bit – 6 BCD digits *
REAL (L)	DECIMAL (K4)	64 bit – 12 BCD digits
	INTEGER (I2)	32 bit
	SHORT REAL (R2)	32 bit FLOATING POINT
	REAL (R4)	64 bit

Fastest ← (INTEGER, SHORT REAL, REAL)

Handled internally with software = SLOW compared to SHORT REAL + REAL (INTRINSIC)

MI1 1.09

* used to carry over the precision of 260 ie no rounding errors

But these Data Types Query will not support

NO SORT OR FIND

OPTION - SHORT →
Decimal - REAL →

CAN DEFINE DEFAULT DATA TYPE

Suggests REAL as default

ORIGINALS WILL REVERT TO Default

☐ Differences between HP260 and HP3000 Environments

DATA TYPES

Use INTEGER whenever possible
better performance

enormous codespace savings
esp. in FOR LOOP
and SELECT

Arithmetic expressions with only SHORT INTEGER operands may cause SHORT INTEGER overflow even if result would be small enough

substring behaviour slightly different

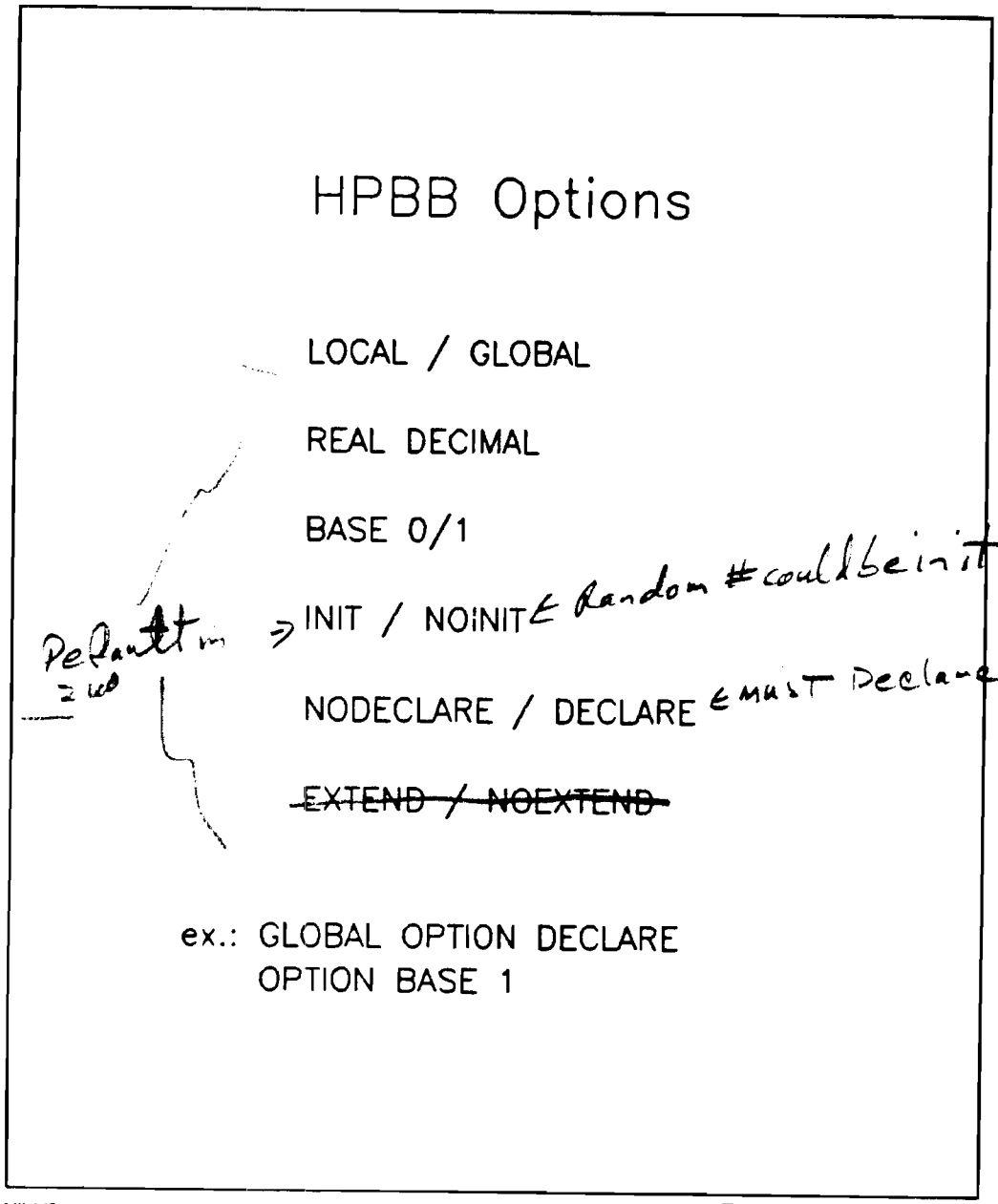
POS with empty 2. parameter returns always 1

expensive string operations

+concatenation
(1;5) substring access
RPT$

MH 1.09a

☐ Differences between HP260 and HP3000 Environments



☐ Differences between HP260 and HP3000 Environments

File Types

HP260	HP3000
DATA	80 FA
	BDATA → DATA
	BINARY → WORK FILE DECLARED AS
PROG	BSAVE
FORM	BFORM
DSET	PRIV
ROOT	PRIV

MUST BE

V FORM from V Plus

PROG = Compiled

☐ Differences between HP 260 and HP 3000 Environments

DATA FILES

ASSIGN returns different errorcodes - 3000 FIND system Error Codes

PRINT ; END works differently

Be cautious with direct word access word pointers may be different if SHORT or REAL used

1 4 bytes → 6 BYTES REAL
2 3 SHORT

3000 WILL NOT RECOGNIZE END OF FILE MARK when the end of File is moved (Sequential)

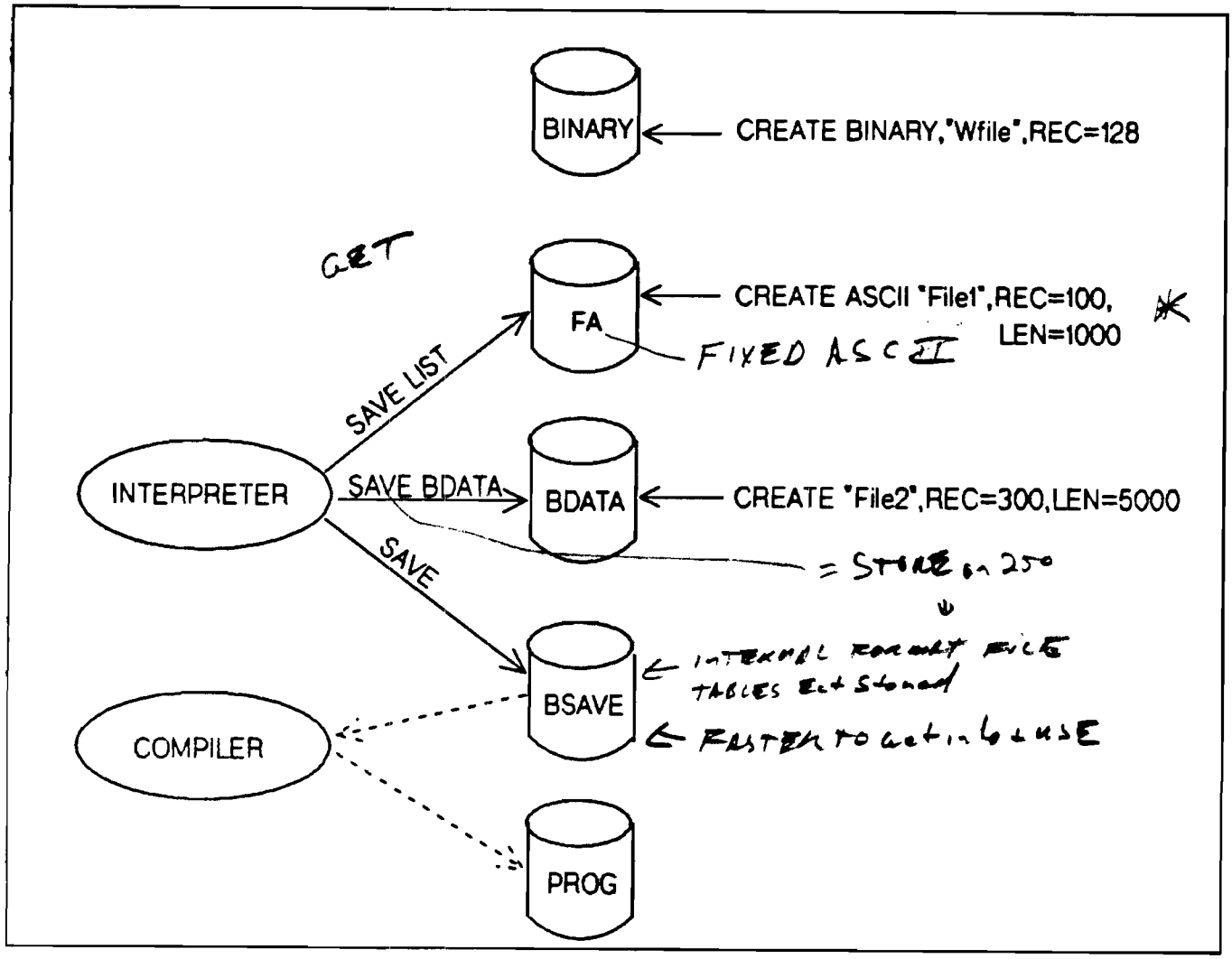
	250	3000
1	RECORD	EOF
2	FILE	EOF

MAT 1.file

MUST HAVE Header on 3000 B DATA FILE

Module 3 - 7

☐ Differences between HP 260 and HP 3000 Environments



MI1 1.12

SAVE LIST → can Serialize

* LIKE DATA on 250 USE TO "CLERK" TABLES - MAKE NEW "MAP"

☐ Differences between HP 260 and HP 3000 Environments

SCREEN AND SOFTKEYS

CURSOR options reduced — NO CURSOR UP ALL

~~usage of RPOS and CPOS restricted~~

LINPUT behaves slightly different — WILL NOT INPUT PROMPT STRING

INPUT Default string.

Effects of interrupting INPUT <varlist> with softkey

Softkey labels only 2 * 8 characters — DO OWN Justification

no videoenhancements in label

8 SoftKeys only

Display string $?

MH 1.12a

WAIT ENABLE INPUT ON SoftKey w. ACCEPT RESPONSE
LOOP for RESPONSE
1 key ☐

↳ NOT Dynamic
ON HALT = Still Dynamic (ie Terminate Something)

☐ Differences between HP 260 and HP 3000 Environments

Untranslatable BASIC Statements

- Disc Handling

  HOLE READ LABEL DEVLOCK CATLINE DIRECT

- Image Utilities

  DBCREATE, DBERASE

- TIO

  REQUEST, ATTACH

- MISCELLANEOUS

  PERFORM, WAIT, SD, LOAD

- No SW Interrupts

- only static loop constructs

- IN DATASET USE/DIM ALL

☐ Differences between HP260 and HP3000 Environments

Uncompilable BASIC Statements

COMMAND GET LINK MERGE
PAUSE SAVE SCRATCH TRACE

IF INTERPRETED THEN PAUSE

Key word to skip IF Compile

Gives Error Message But will not stop at Runtime

Could not be declared Public

☐ Differences between HP260 and HP3000 Environments

DATABASE MANAGEMENT IMAGE/3000 VS. IMAGE/250

IMAGE/3000 does not support:

- Read Locking
- Buffer flushing on DBCOLSE
- Multiple volume for data
- Avoidance of Deadlocks with multiple databases.
  - only one lock per program
- "Format numbers" or "control numbers".
- DBINFO with qualifier 4xx
- Argument must be of exactly the same type as the Item

MH 1.15

12 seconds

☐ Differences between HP260 and HP3000 Environments

NOT IN BUSINESS BASIC

Database utilities are not part of MPE and are not accessible from Business BASIC.

Run DBUTIL.PUB.SYS to do the following:

- CREATE a database
- PURGE or ERASE a database
- CHANGE a password

☐ Differences between HP 260 and HP 3000 Environments

IMAGE

FEATURES	IMAGE/250	IMAGE/3000	TURBO
Item/database	255	255	1023
Sets/database	50	99	199
Items/dataset	127	127	255
Recs/dataset	65534	8388607	2 Billion
Paths (keys)	8	16	16
Char. in name	15	16	16
Passwords	31	63	63
Max. rec length	1024	4096	4096
Security	set	set & Item	set & Item
Sorted chains	no	yes	yes

☐ Differences between HP 260 and HP 3000 Environments

DATABASE STATEMENTS

IMAGE/3000 supports full transaction logging:

BEGIN TRANSACTION

END TRANSACTION

DBMEMO

ON DBERROR

OFF DBERROR

- STATUS ARRAY S(0) ONLY?
- IF ... DBERROR THEN — New Statement

☐ Differences between HP260 and HP3000 Environments

SYNTAX of a DATABASE STATEMENT

using DSET

```
               [USING "DSET"  ]
DBGET dbname   [INTO = STRING ]  ,DATASET=setname
   [,MODE=read_mode][,ITEMS=items]
   [STATUS=array][,KEY=key]
```

Read_mode is either a numeric expression with one of the following values, or the equivalent mnemonic:

Value	Mnemonic	Mode
1	READ	reread
2	SERIAL	serial read
3	SERIALBACK	serial backward
4	DIRECT	direct read
5	CHAIN	chained read
6	CHAINBACK	chained backward
7	CALCULATED	calculated read
8	PRIMARY	primary calc.

The default is SERIAL (mode 2).

ex.: DBGET USING Label1, DATASET = "SET1"

Dynamic

Label1: IN DATASET "SET1" USE Variable List, SKP "#of bytes", Variable List

MH 1.19

IF USING 'DSET' + HEW NO Buffer (Bull?) ...

☐ Differences between HP260 and HP3000 Environments

IN DATASET

Functionality:

Defines record format of dataset found in thread list

Similar to PACKFMT

Incompatibilities with BASIC/250:

No DIM ALL or USE ALL clause

If you use Set name any where you can use the label or a reference to THREAD

or in SEARCH

NUMREC

☐ Differences between HP 260 and HP 3000 Environments

DATA BASE MANAGEMENT

DBFIND requires strict datatype equivalence

DBGET/DBUPDATE less flexible

check DBINFO return information for compatibility

DBLOCK does not allow a second lock with finer granularity

replace several simple DBLOCKs by a single DBLOCK using predicate with multiple clauses to avoid need of MR capability

check PREDICATE syntax

replace statusfield references by doubleword values

expensive statements

USING clause

therefore use only those variables in IN DATASET USE which are really needed

SHORT
32 Bit INTEGER for Status Array

Rec-no = S(2) x 2^16 + (S(3))
↑ Convert

3 → 2
7 → 6
9 → 8

☐ Differences between HP 260 and HP 3000 Environments

SORT

- Specifies sort keys and usage (ASC,DES)
- Refers to THREAD statement for datasets
- Independent, but related, to SEARCH

  (Does an implicit SEARCH ALL if no SEARCH has been executed yet)

WATCH FOR COMMAND LINES IN

☐ Differences between HP 260 and HP 3000 Environments

SEARCH

Specifies selection criteria for records

Starts database retrieval process

Refers to THREAD statement for search order

☐ Differences between HP 260 and HP 3000 Environments

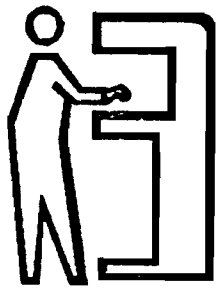
THREAD IS

- Thread list can be 1–10 sets long
- Must alternate between master and detail sets
- Can specify a PATH or default to PATH 1
- Non–executable statement
- Validity checked at run–time

☐ Differences between HP260 and HP3000 Environments

WORKFILE IS

- Must assign the workfile first
- Must be a binary file
- Record size (in words) must be, at least:
  2 * number of data sets in thread
- Pointers are of type INTEGER 32 Bit = 2 WORDS



☐ Differences between HP260 and HP3000 Environments

SEARCH/SORT

Access to workfile in SEARCH expression

SORT restrictions with decimal (0, >=1)

various optimizations and savings
- reduce number of variables in IN DATASET
- put SEARCH/SORT in separate subunits
- remove identical statements (use GOSUB)
- use FILTER and SORT ONLY

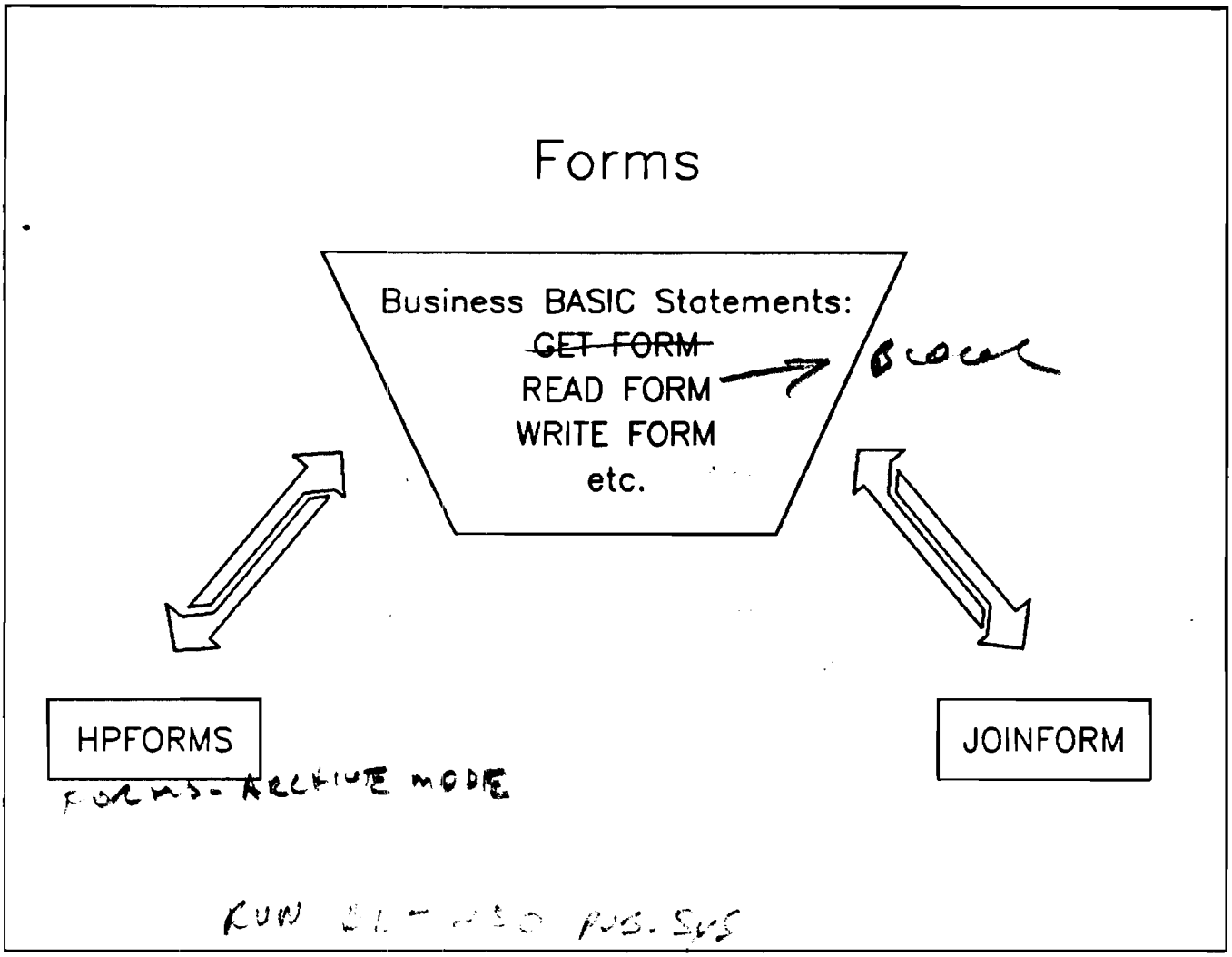
Workfile pointers are 32 bit INTEGER only ←

expensive statements
- SEARCH esp. with long IN DATASET USE
- SORT
- PACK/UNPACK (depending on number of variables)

this is not the case on 16 Bit HP260 - should be normalization?

can PASS FILE NO-
IN DATASET & NO TEST for Errors !
can "MANIPULATE" (Difficult Work)
WATCH DATA TYPE mismatch

☐ Differences between HP 260 and HP 3000 Environments



FORMS-ARCHIVE MODE

RUN B.I-.30 POS.SYS

→ OPTION JOINFORM

→ Form

☐ Differences between HP260 and HP3000 Environments

JOINFORM

Output buffering may affect your program

tricks to put videoenhancements to formfields

DISP may need CR ?

FORMFILES

Several forms in one file C

OPENFORM, Formfile: Formname

READFORM

JOINEDIT

WRITE -

C

☐ Differences between HP260 and HP3000 Environments

HPBB REPORT WRITER

100% compatible to HP260

- default page length HP260 66.2.2
- default page length HPBB 60.0.0

HP3000 Enhancements

- TOTAL
- output count
- SUPPRESS PAGE HEADER/TRAILOR
- BREAK IF

☐ Differences between HP260 and HP3000 Environments

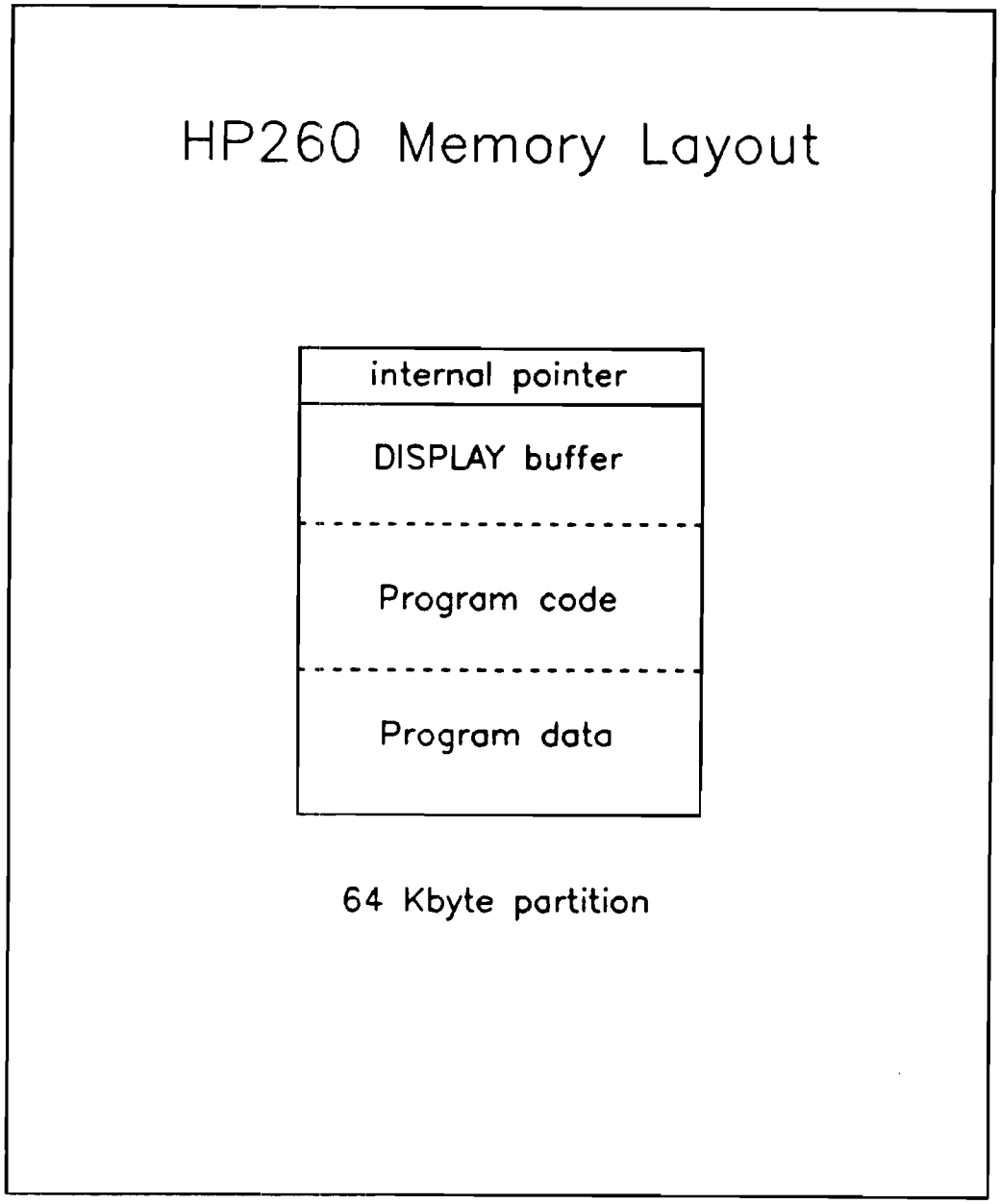
REPORT WRITER

Reports on screen should be <=79 characters

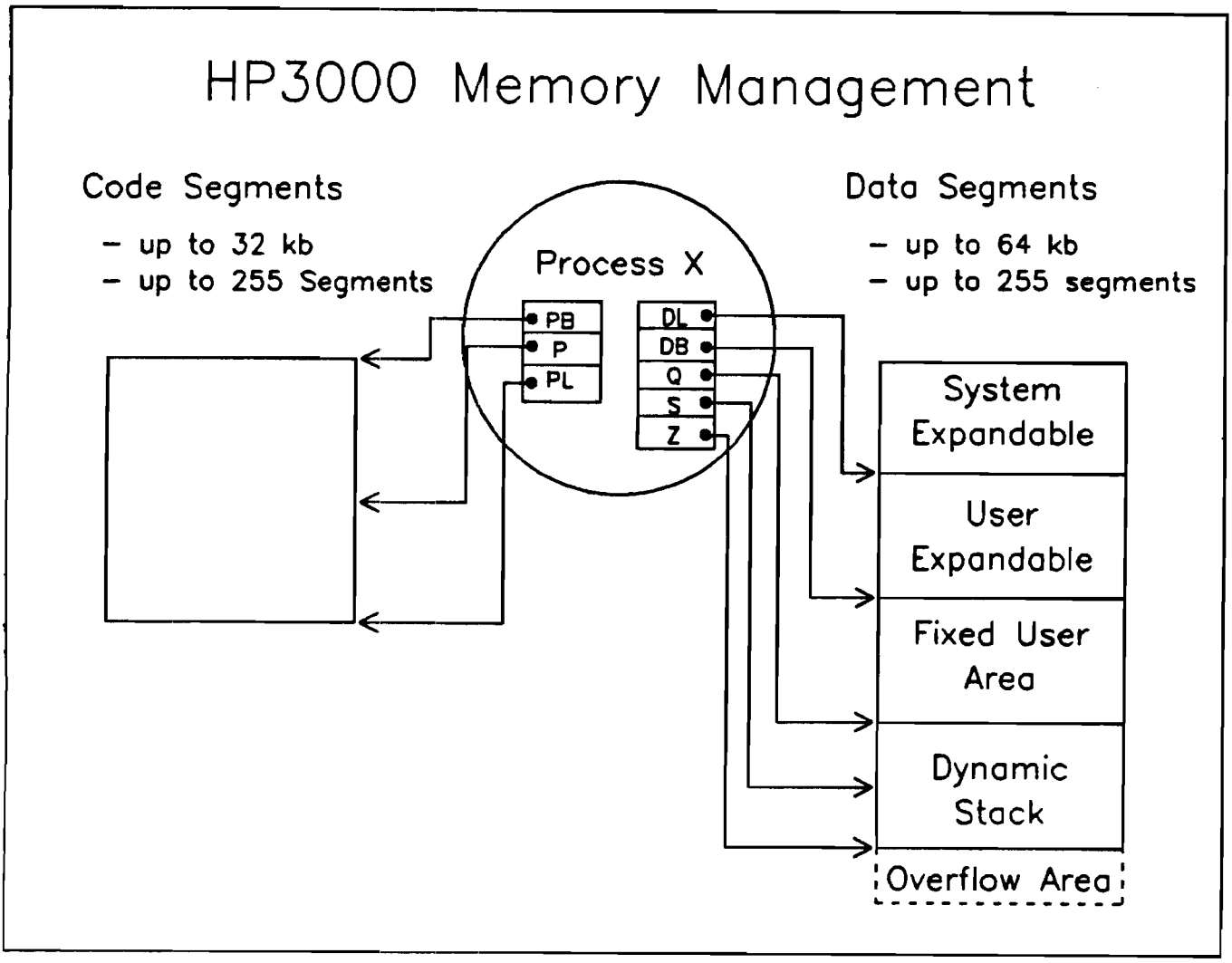
reduce number of expensive statements
- DETAIL LINE 1 → use in GOSUB
- TRIGGER BREAK

put report in separate subunit

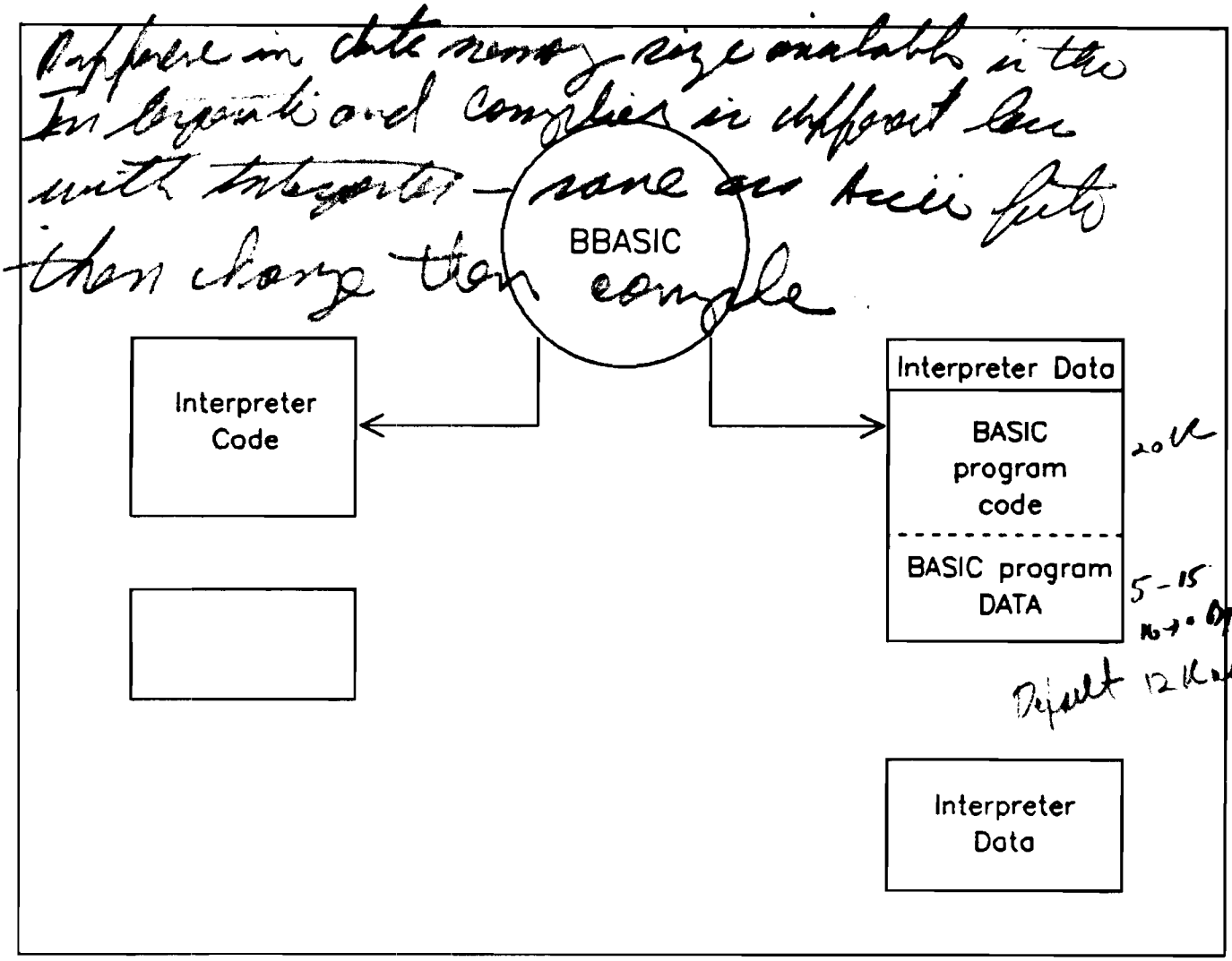
☐ Differences between HP260 and HP3000 Environments



☐ Differences between HP260 and HP3000 Environments



☐ Differences between HP 260 and HP 3000 Environments



☐ Differences between HP 260 and HP 3000 Environments

INTRINSICS and EXTERNALS

The INTRINSICS and EXTERNALS statements provide access to subroutines and functions written in:

- HP Business BASIC
- Pascal/3000
- SPL/3000

If the INTRINSIC or EXTERNAL statement is:

- Declared to be GLOBAL, it can be called from any program unit.
- Otherwise, it is available only in the subprogram unit that declares it.

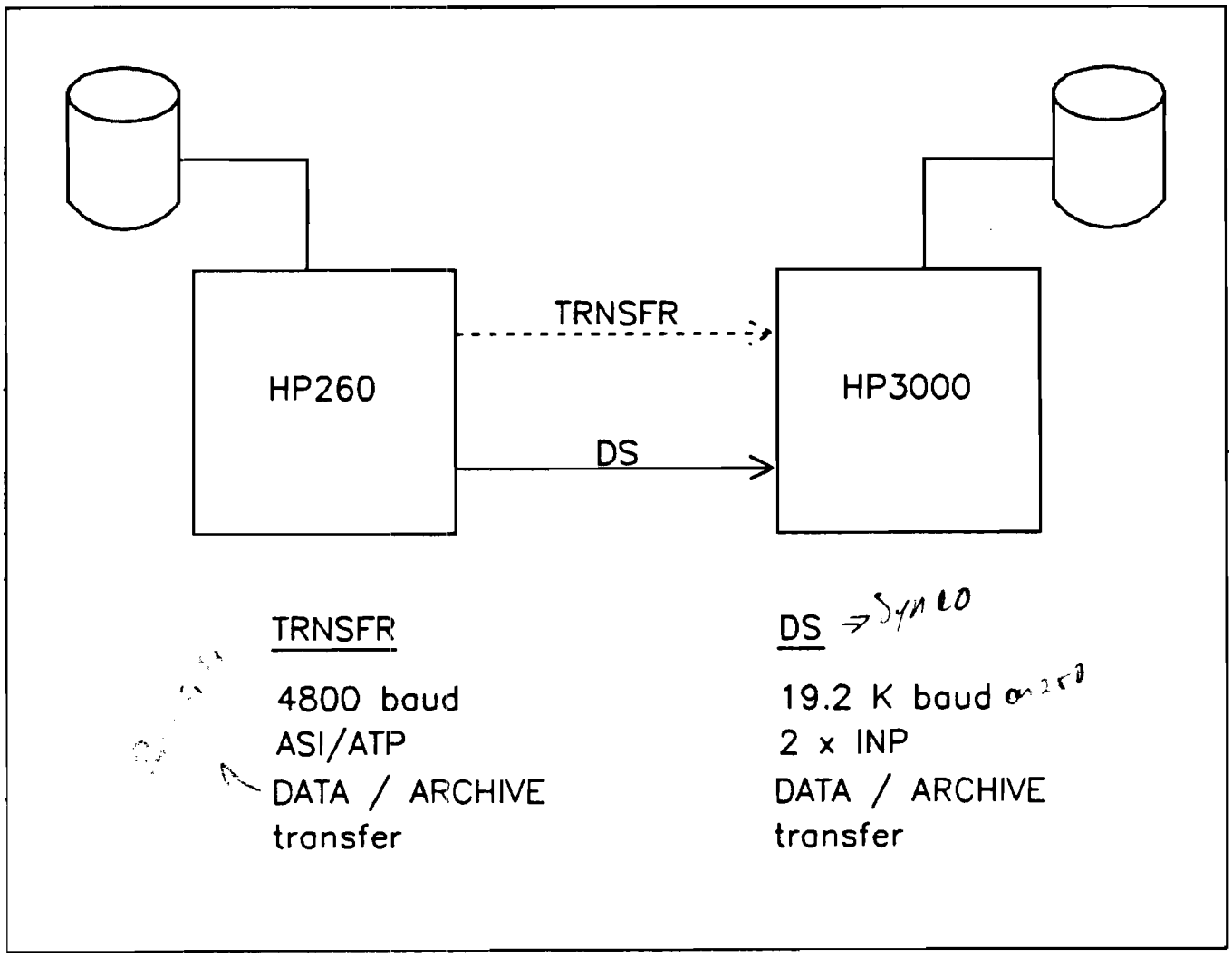
EX.: GLOBAL INTRINSIC <Intrinsic name>

→ COPTION SEGMENT <name>

ALLOCATES the routine in memory

USL FILE SUBROUTINES

☐ Join/3000 Tools



B .08

☐ JOIN/3000 Tools

BBCT250.PUB.SYS

OPTION	NO INPUT LOOPS/INPUT LOOPS DECIMAL/REAL JOINFORM
DATA	<input file>,<output file>
DATASET	<input file>,<data base>
FORM	<input file>,<output file>[,<report file>]
SCHEMA	<input file>[,<output file>]
PROG	<input file>[,<output file>[,<report file>]]
CONVERT	<input file>[,<output file>[,<report file>]]

☐ JOIN/3000 Tools

BBCT250 Options

NO INPUTLOOPS (default)
INPUTLOOPS

affected Statements: INPUT, LINPUT, TINPUT
ACCEPT, WAIT

Ex.:

BASIC 260	HPBB
10 WAIT	100 LOOP !** SYNTAX CHANGE
	101 ACCEPT !** ADDED LINE
	102 EXIT IF RESPONSE=1 !** ADDED LINE
	103 END LOOP !** ADDED LINE

CONVERTING FORMS

specify OPTION JOINFORM for BBCT250

converting HP260 FORMs into JOINFORMs

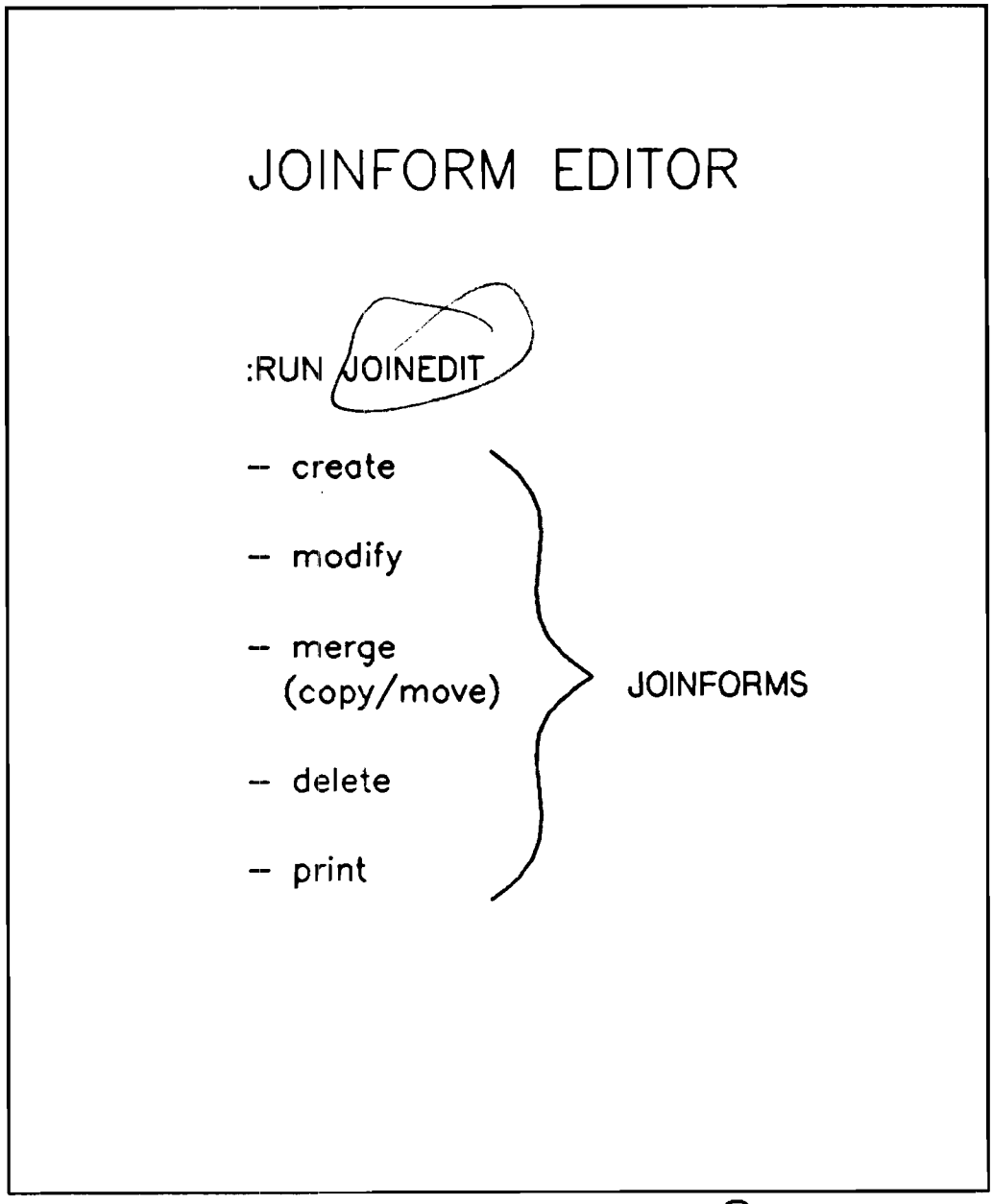
- single FORM transfer
- batch mode transfer

Example batch file (ASCII):

```
1   !RUN BBCT250.PUB.SYS
2   OPTION JOINFORM
3   FORM form1.group.acct,jform1
4   FORM form2.group.acct,jform2
```

```
100  OPENFORM  form1:jform1
```

☐ JOIN/3000 Tools



MH L34a

☐ JOIN/3000 Tools

HP3000 Helpfile

HPBB program

Eliminate BBCT250 comments

Insert General used Options

use Editor Statements
FIND
CHANGE "String1" TO "String2" IN ALL

Locate critical Statements

Report every modification

☐ Upgrade Process

Upgrade Process

HP260:

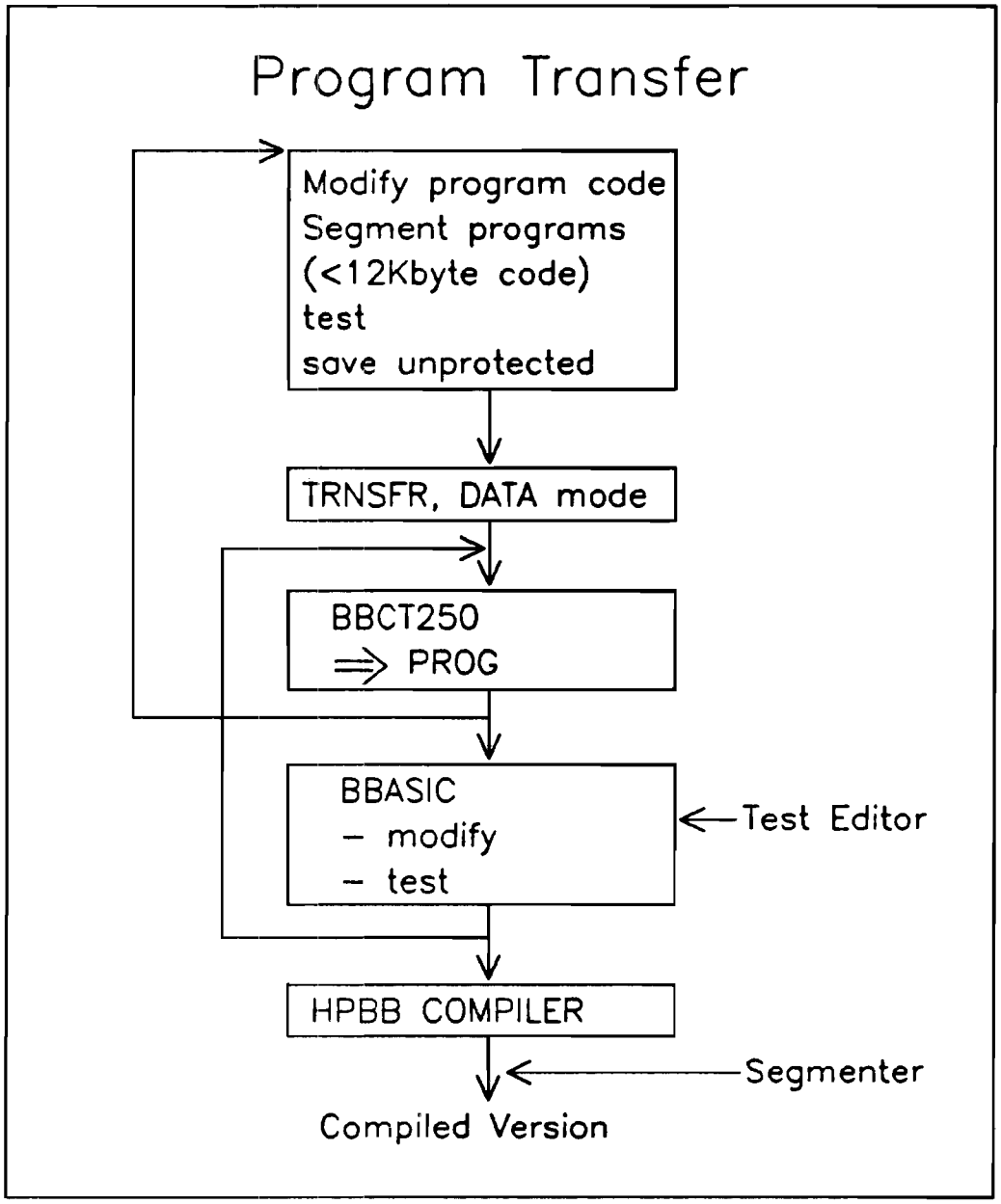
- Backup everything
- Modify programs
- Test programs
- Save programs, unprotected
- Unload Datasets, set by set
- Set up PERFORM file

physical Transfer

HP3000:

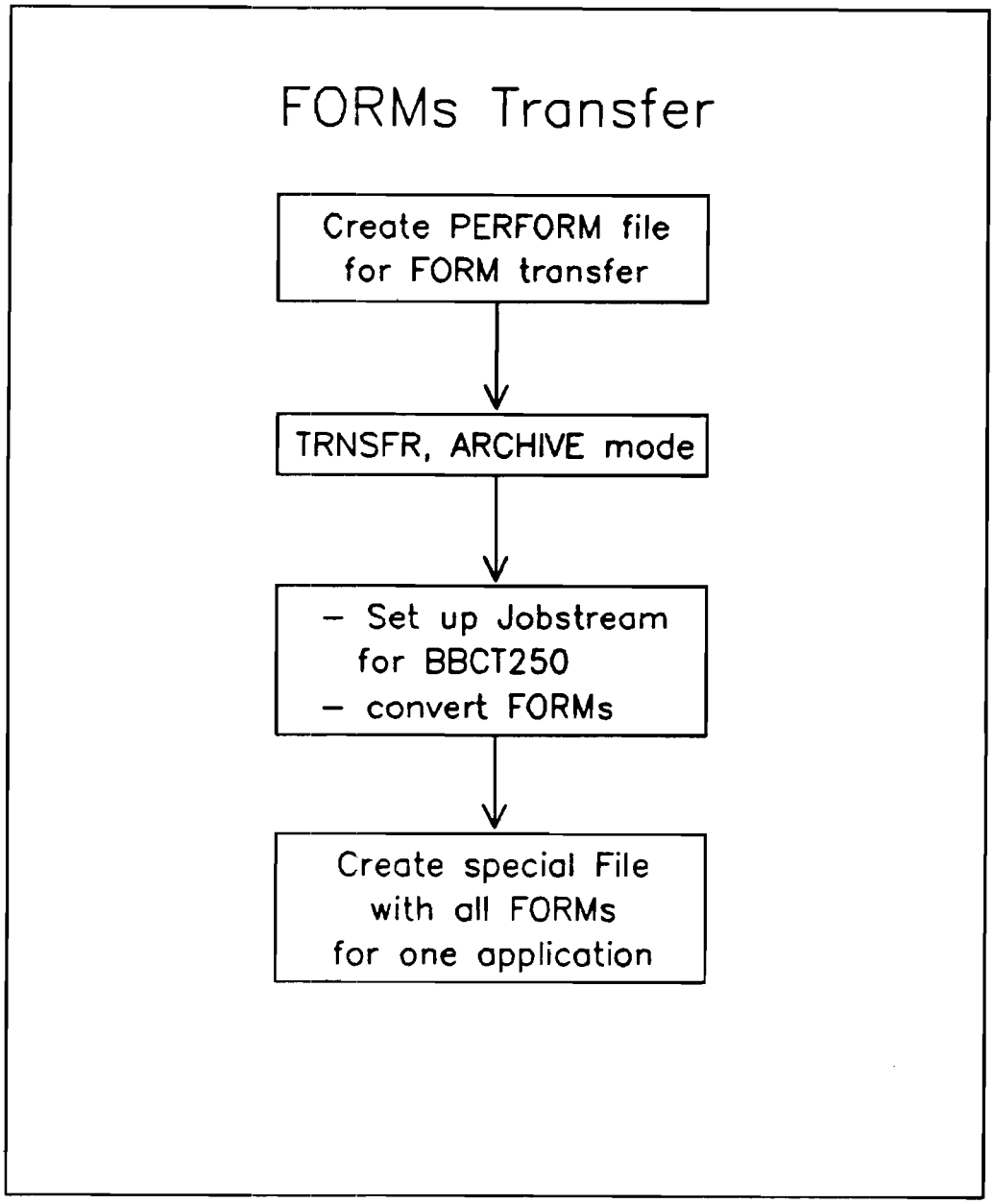
- Convert Programs
- Convert FORMs
- Convert SCHEMA
- Create (Load) Datasets
- Test

☐ Upgrade Process



MM 1.37

☐ Upgrade Process



MH 1.38

☐ Upgrade Process

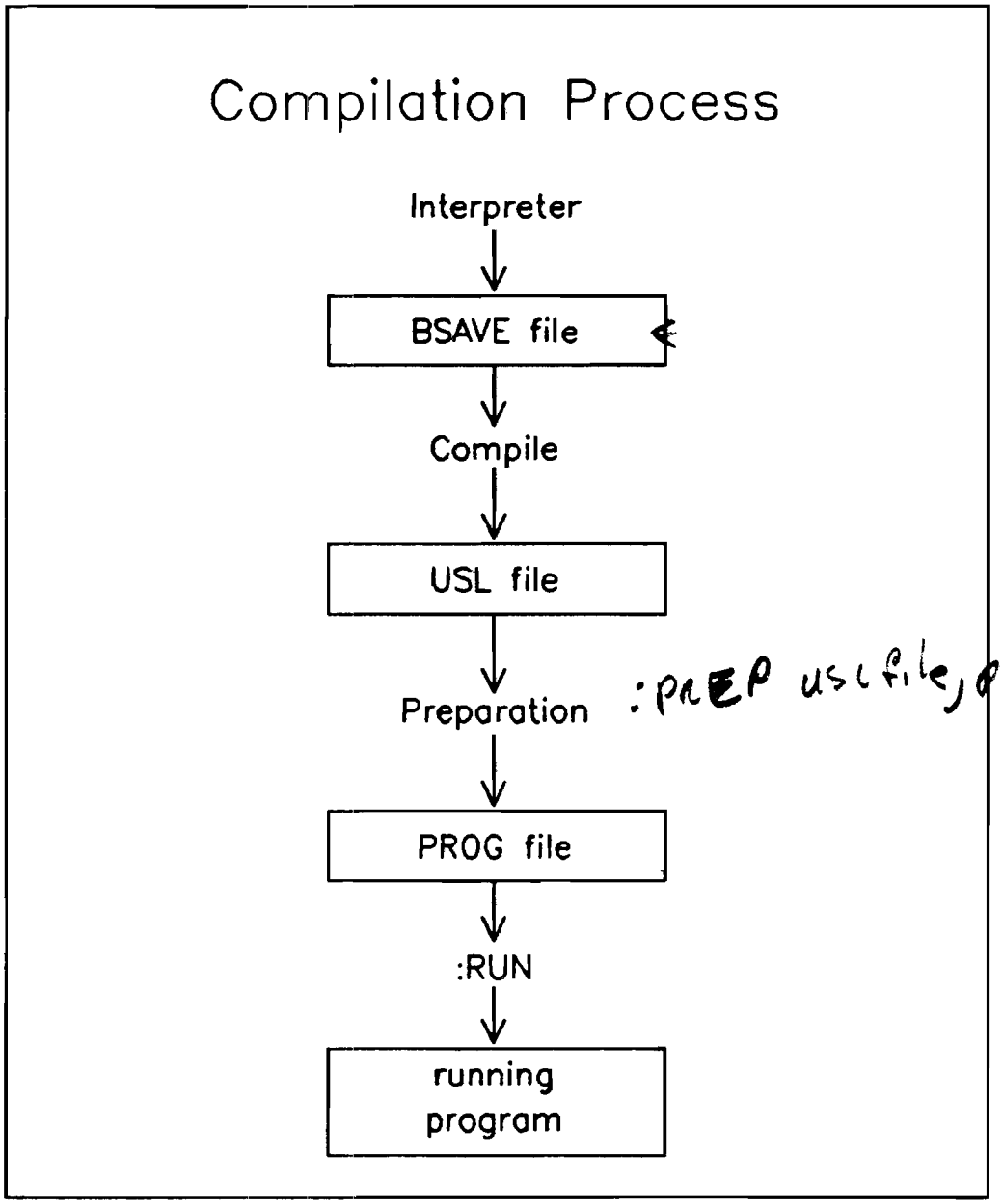
DB Transfer

Backup
Create Schema file
DBUNLD, set by set

TRNSFR
Schema: DATA mode
Dset bkup's: ARCHIVE mode

- Create Root
  RUN
  :BBCT250
  Schema
  Check Schema
  :FILE DBSTEXT=<Schemafile>
  :FILE DBSLIST;DEV=PP
  :RUN DBSCHEMA;PARM=1
- Create DB
  RUN DBUTIL
  >>CREATE <DBname>
- Load Database
  :RUN BBCT250
  DATASET <Bkupname>,<DBname>

☐ Application Tuning



☐ Application Tuning

Compilation Process Commands

Step(s)	Command form Interpreter	Command form Operating System	Effect on $OLDPASS
Compilation	COMPILE [infile] [{:} USL[=]uslfile [{:} LIST[=]listfile]]	:BBASICOMP infile [,[uslfile]] [,[listfile]]]	If uslfile is not specified and command succeeds, then $OLDPASS is the USL file
Compilation Preparation	COMPPREP [infile] [{:} PROG[=]progfile [{:} LIST[=]listfile]]	:BBASICPREP infile [,[progfile] [,[listfile]]]	If uslfile is not specified and command succeeds, then $OLDPASS is the PROG file

☐ Application Tuning

Compilation Process Commands (cont'd)

Step(s)	Command form Interpreter	Command form Operating System	Effect on $OLDPASS
Compilation Preparation Running	COMPGO [infile] [{;} LIST[=]listfile]	:BBASICGO infile [, listfile]	If uslfile is not specified and command succeeds, then $OLDPASS is the PROG file
Preparation	Not possible	:PREP uslfile, progfile	None
Preparation Running	Not possible	:PREPRUN uslfile	If preparation succeeds, #OLDPASS is the PROG file
Running	SYSTEMRUN progfile	:RUN progfile	None

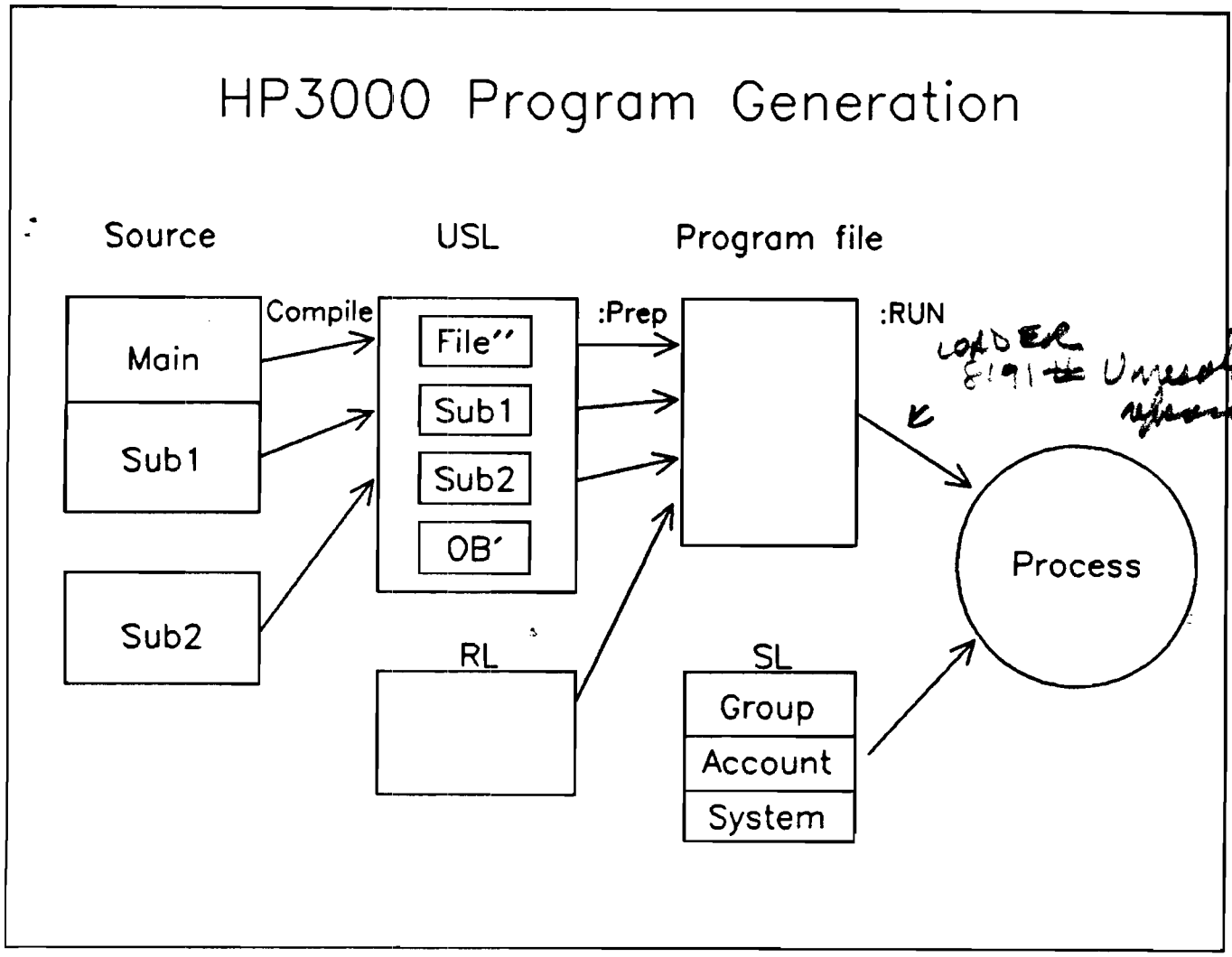
SAVE $OLDPASS

☐ Application Tuning

Important COPTIONS

		reduces code size	better performance
RANGE CHECK/ NO RANGE CHECK	no array range check during runtime	*	*
REDIM NO REDIM	no array redimension during runtime	*	*
SET ERRL NO SET ERRL	ERRL always 0	*	
ERROR HANDLING NO ERROR HANDLING	disables ON ERROR and ON DBERROR	*	*
HALT CHECKING NO HALT CHECKING	disables ON HALT	*	
SEGMENT=<name>	following code in new Segment		
USLINIT	only Global Option in MAIN purges all USL files		
NEWCOM NO NEWCOM	affects COM usage	*	

☐ Application Tuning



MI1 1.44

LOADER PROBLEM

LOADER ERROR 71 -

DL Space

50 + 90 unresolved reference segment per segment

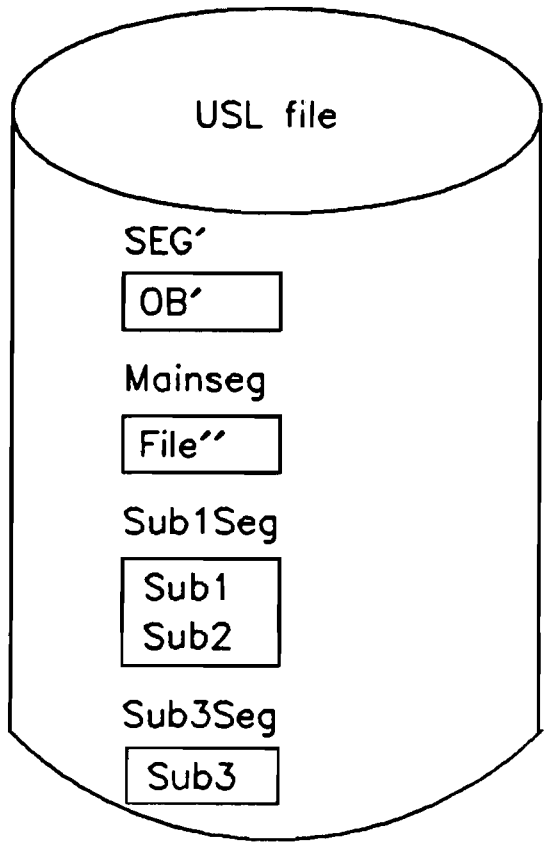
Dynamic RCT

Segment exceeds limit

FIX move SL to USL then Resolved at prep

☐ Application Tuning

```
!MAIN

COPTION SEGMENT "Mainseg"
EXTERNAL SUB1, SUB2, SUB3
  .
  .
  .
END

!SUB1
SUB SUB1
COPTION SEGMENT "Sub1Seg"
  .
  .
  .
SUB END

!SUB2
SUB SUB2

END SUB

SUB SUB3
COPTION SEGMENT "Sub3Seg"

SUB END
```



☐ Application Tuning

Performance Tuning

- use COPTIONS
- Operants should have same Datatype
- Don't use DECIMAL
- no selfmodifying code
  (LOAD, SAVE, DELETE)
- compile
- comments decreases speed of interpreted version
- GOSUB faster then CALL

☐ Application Tuning

Memory Space Tuning

- use LOCAL instead GLOBAL intrinsics/externals
- put constants in DATA statement, not in the program
- Specify array dimensions
- use COPTIONS
- use Segmenter to delete dummy blocks
- generate a special SL

DATA I,J,K
READ A,B,C
NOT A=1 B=2 C=3

Dynamic Array space expensive

Outer Blocks

COPTION · SUBPROGRAM

GLOBAL ALLOWS CALLS from other SUBS

OUTER BLOCK = Jump code to start of subs (Segment)
ONE OUTER BLOCK ACTIVE for one use

HOP. PAUSE CONTINUE

IFDL

INFO → state of Interpreter

INFO$

TRUNC

INTRINSIC

ITM ←

LEN

LEX

Lock ← Pick

LTRIM$

MARGIN

MAX MIN MOD

MAXLEN

PALENVM

POSITION

PRESS KEY

ROUND

RTRIM$

Screen NULL

SGN

TINPUT

TRUNC

UPDATE

FILTER

SEARCH

NAME.

RESPONSE

Ctrl y

Ctrl E step

S

Q

Disp ; ← NO CR

STEP

Lab1 - HP260 to HPBB Migration Training

In this lab you will use the interpreter and learn the use of a few of the commands and statements. You may wish refer to the Reference manual from time to time so be sure it is handy.

1) Sign-on using the user and account assigned to you.

```
hello userX.hpbb,groupX
```

2) Invoke the interpreter using the the (UDC) command: BBASIC

3) Make a copy of the program "LAB1.PUB.HPBB" using the command COPYFILE "LAB1.PUB.HPBB" TO "LAB1".
Verify that the file was created by using the CAT command.
Which type of program file is it? (BDATA, ASCII or BSAVE)

4) Use the GET command to load the program. LIST the program to get an idea of what it does. What possible error will statement 30 trap? What parameter could you add to the CREATE statement to make statements 30 and 50 unnecessary?

5) RUN the program.

6) Write another program which reads the contents of the file "XYZ".

```
! lab1
  10 !  Lab 1 of the HPBB training                          M.B.
  20          DIM A$[30]
  30          ON ERROR GOTO 50
  40          CREATE ASCII "XYZ",FILESIZE=10,RECSIZE=-80
  50          SEND OUTPUT TO "XYZ"
  60          OFF ERROR
  70 !
  80 Start:   INPUT "Enter employee's name (first and last): ";A$
  90          DISP "Current length of emp. name= ";LEN(A$)
 100          DISP "Maximum Length of emp. name= ";MAXLEN(A$)
 110          Numblank=SCAN(A$," ")
 120          DISP "First name has ";Numblank-1;" blank characters in it"
 130 !
 140 Soc:     INPUT "Social Security Number: ";Ss$
 150          Ss$=TRIM$(Ss$)
 160          ON ERROR GOTO Err1! Ss$ contains no number
 170          IF VAL(Ss$)<=0 THEN GOTO Err1! bad entry?
 180          DISP "THANK YOU VERY MUCH"
 190 !
 200          GOSUB Printit
 210          END
 220 Err1: ! invalid social security number
 230          DISP "INVALID SOCIAL SECURITY NUMBER. PLEASE RE-ENTER"
 240          GOTO Soc
 250 !
 260 Printit: !
 270          OFF ERROR
 280          PRINT "EMPLOYEE: ";A$
 290          PRINT "SOCIAL SECURITY #: ";Ss$
 300          RETURN
```

```
 10          ASSIGN #1 TO "xyz"
 20          DIM A$[80]
 30          ON ERROR GOTO Ex
 40          LOOP
 50             READ #1;A$
 60             DISP A$
 70          ENDLOOP
 80 Ex:      DISP "eof found"
 90          ASSIGN * TO #1
100          END
```

Lab 2
=====

Complete the following program listing!

The purpose of the program is to generate a report about the following items:

Customer name
Order_no
Product_no
Product_description
Price

The sequence in the report should be determined by the name of the customer and the Product no.
The report should only contain orders where the price is higher then 100$.

The datas are found in the SAD database. The SCHEMA listing is in the appendix of the lab- description.

```
10  !  HPBB Lab2              IMAGE example
20  DIM Dbase$[15]
30  INTEGER Index,R2,R3
40  DIM Name$[30],Order_no$[10],Prod_desc$[30]
50  _____________________     Product_no
60  _____________________     Price
70  Dbase$="  SAD.PUB.SYS"
80  DBOPEN Dbase$,PASSWORD="MANAGER"
90  CREATE _____________     "Workfile"
100 ASSIGN #1 TO "Workfile"
110 _____________ Base$
120 Set2:  IN DATASET "Product" USE _____________, Prod_desc$
130 Set3:  IN DATASET "Customer" USE Order_no$,Name$,SKP ____ ,&
                      Product_no,Price
140 WORKFILE IS #1
150 Threadlist: THREAD IS ____________________
160 SEARCH USING Threadlist;_____________________
170 SORT USING _____________; Name$, _____________
180 For Index = 1 TO NUMREC(1)
190       READ #1;R3,R2
200       DBGET Dbase$ USING Set3, DATASET="CUSTOMER",MODE=4,&
                Key=R3
210       DBGET Dbase$ USING Set2,DATASET="PRODUCT",MODE=4,KEY=R2
220       PRINT Name$,Order_no$,Product_no,Prod_desc$,Price
230 NEXT Index
240 END
```

```
$CONTROL    LIST,TABLE,ROOT
$TITLE      "Sales Data Base"
BEGIN DATA BASE SAD.PUB.HPBB
PASSWORDS:
                10          SALESMAN;
                15          MANAGER;
                 3          SEC;
ITEMS:
      ADDRESS,      2X30;
      CITY,          X16;
      NAME,          X30;
      ORDER-NO,      X10;
      PRICE,          K4;
      PRODUCT-NO,      I;
      PROD-DESC,     X30;
      SALESPERSON,    X4;
      STATE,          X6;
      ZIP-CODE,       X8;
SETS:
  NAME:       ORDER,A(3/10,15);
  ENTRY:      ORDER-NO(1);
  CAPACITY:   500;

  NAME:       PRODUCT,M(3,10/15);
  ENTRY:      PRODUCT-NO(1),
              PROD-DESC;
  CAPACITY:   100;

  NAME:       CUSTOMER,D(3/10,15);
  ENTRY:      ORDER-NO(ORDER),
              NAME,
              ADDRESS,
              CITY,
              STATE,
              ZIP-CODE,
              PRODUCT-NO(PRODUCT),
              PRICE,
              SALESPERSON;
  CAPACITY:   500;
END.
```

Solution of the HPBB Lab2
==========


```
10  !  HPBB Lab2                IMAGE example
20  DIM DbaseS[15]
30  INTEGER Index,R2,R3
40  DIM NameS[30],Order_noS[10],Prod_descS[30]
50  SHORT INTEGER              Product_no
60  DECIMAL                    Price
70  DbaseS="  SAD.PUB.SYS"
80  DBOPEN DbaseS,PASSWORD="MANAGER"
90  CREATE BINARY              "Workfile"
100 ASSIGN #1 TO "Workfile"
110 DBASE IS     BaseS
120 Set2:  IN DATASET "Product" USE Product_no  , Prod_descS
130 Set3:  IN DATASET "Customer" USE Order_noS,NameS,SRP 90  ,&
                    Product_no,Price
140 WORKFILE IS #1
150 Threadlist: THREAD IS Set3, Set2
160 SEARCH USING Threadlist; Price > 100
170 SORT USING Threadlist ; NameS, Product_no
180 For Index = 1 TO NUMREC(1)
190       READ #1;R3,R2
200       DBGET DbaseS USING Set3, DATASET="CUSTOMER",MODE=4,&
              Key=R3
210       DBGET DbaseS USING Set2,DATASET="PRODUCT",MODE=4,KEY=R2
220       PRINT NameS,Order_noS,Product_no,Prod_descS,Price
230 NEXT Index
240 END
```

REC — for doSpec=5

f0 = words

16 bit word

Lab3 HP260 to HPBB Migration Training

In the lab a conversion of a HP260 application to the HP3000 BB environment should be done.
This includes a database, three programs and two forms.

```
QUALTY        ROOT
QUALTY  01    DSET
QUALTY  02    DSET
QUALTY  03    DSET

UPDPAR        PROG   !  Program to add or delete parts from the DB
QASORT        PROG   !  Program to generate a sorted report
SUBPR         PROG   !  Subprogram to get the password for the DB

QAF1          FORM   !  FORM used for adding a part
QAF2          FORM   !  FORM used for deleting a part
```

The software will be on the MSI of the HP260.
In your Lab handouts you will find a schema- listing of the QUALTY DB and listings of the programs.

1) Prepare the transfer on the HP260

   - generate SCHEMA file
   - unload the database (set by set)
   - check the program size
   - check for not convertible statements
   - SAVE the program files

2) Transfer the application with DS or TRNSFR into the group TRANS of the HPBB account.

3) Integrate the application into the HPBB interpreter environment.

   - Convert the software with BBCT250.pub.sys
   - Create and load the database
   - Modify the programs

```
      << Schema definition for data base QUALTY >>
      << Password modification count: 2 >>
      << Data base modification count: 3 >>

BEGIN DATA BASE QUALTY,SALES;

PASSWORDS:                                  << Password definition >>
    1   ME;
    2   YOU;
    3   OTHERS;

ITEMS:                                      << Item definition >>
    AMT,                    L;
    ARDATE,                 L;
    PART-DS,                X30;
    PART-NO,                X10;
    PROJCT,                 X10;
    VEND-AD,              2 X30;
    VEND-NM,                X30;
    VEND-NO,                I;
    YIELD,                  I;

SETS:                                       << Set definition >>

NAME:  PARTS, M (2,3/1),SALES;
ENTRY: PART-NO (1),
       PART-DS;
CAPACITY: 19;                               << Entries: 8 >>

NAME:  VENDOR, M (2,3/1);
ENTRY: VEND-NO (1),
       VEND-NM,
       VEND-AD;
CAPACITY: 19;                               << Entries: 6 >>

NAME:  SHIPMT, D (2,3/1),SALES;
ENTRY: PART-NO (PARTS),
       VEND-NO (VENDOR),
       AMT,
       PROJCT,
       ARDATE,
       YIELD;
CAPACITY: 101;                              << Entries: 13 >>

END.
```

```
1020 !
1030 ! RE-STORE "UPDPAR"
1040      DIM Lst$[2],Buff$[100],Clr$[4],Ans$[30]
1050      INTEGER S(9)
1060      Base$="  QUALTY"
1070      Pass$="ME"                    ! THIS LINE SHOULD BE REPLACED BY THE NEXT ONE
1080      ! LOAD SUB "SUBPR"
1090      ! CALL GETPASSWD(PASS$,VALID)
1100      ! IF NOT VALID THEN
1110      !    DISP "NO VALID PASSWORD FOR QUALTY DB"
1120      !    BEEP
1130      !    END
1140      ! END IF
1150      Clr$="  "
1160      Lst$="@;"
1170      DBOPEN (Base$,Pass$,3,S(*))
1180      IF S(0) THEN Dberr
1190      DBASE IS Base$
1200      IN DATA SET "PARTS" USE ALL
1210 Kys:DISP Clr$;
1220      DELETE FORM
1230      OFF KEY #
1240      ON KEY #1:"ADD PART" GOTO Addp
1250      ON KEY #3:"DELETE PART" GOTO Delp
1260      ON KEY #8:"EXIT" GOTO Endd
1270      WAIT
1280 Addp:OFF KEY #
1290      ON KEY #1:"ACCEPT DATA" GOTO Compl
1300      DISP Clr$;
1310      GET FORM "QAF1"
1320      CLEAR FORM
1330 In1:INPUT
1340      GOTO In1
1350 Compl:   !
1360      ENTER Part_no$,Part_ds$
1370      IF NOT LEN(Part_no$) THEN Addp
1380      DELETE FORM
1390      DISP Clr$;
1400      DBPUT (Base$,"PARTS",1,S(*),Lst$,Buff$)
1410      IF S(0)=43 THEN Dup
1420      IF S(0) THEN Dberr
1430      DISP Clr$;
1440      DISP "Part: ";Part_no$;"  ";Part_ds$;"  being added"
1450      WAIT 3000
1460      GOTO Kys
1470 Dup:DISP
1480      DISP "Duplicate Part - Try Again!"
1490      WAIT 3000
1500      GOTO Kys
1510 Delp:OFF KEY #
1520      DISP Clr$;
1530      GET FORM "QAF2"
1540      CLEAR FORM
1550      ON KEY #1:"ACCEPT DATA" GOTO A1
1560 In2:INPUT
1570      GOTO In2
1580 A1: ENTER Part_no$
1590      DBGET (Base$,"PARTS",7,S(*),Lst$,Buff$,Part_no$)
```

```
1610      IF S(0) THEN Dberr
1620      DISP Part_ds$
1630      LDISP "Is this the part to be deleted?"
1640      LDISP "  Press key for YES or NO"
1650      OFF KEY #
1660      ON KEY #1:"YES" GOTO Dlt
1670      ON KEY #3:"NO" GOTO Delp
1680      ON KEY #8:"EXIT" GOTO Endd
1690      WAIT
1700 Dlt:DBDELETE (Base$,"PARTS",1,S(*))
1710      DELETE FORM
1720      IF S(0) THEN Dberr
1730      GOTO Kys
1740 Dberr:DISP Clr$
1750      DELETE FORM
1760      CURSOR (1,12)
1770      DISP "DATA BASE ERROR - ABORTED",S(0),S(6)
1780 Endd:DBCLOSE (Base$," ",1,S(*))
1790      END
1800 Nomatch:DISP "       NO  MATCH - RE-ENTER"
1810      WAIT 2000
1820      GOTO Delp
```

```
1000 !
1010 !           HPBB LAB          SORT QUALTY DB
1020 !                                                          M.B.
1030 !  RE-STORE "QASORT"
1040      DIM Lst$[2],Buff$[100]
1050      INTEGER S(9)
1060      Base$="  QUALTY"
1070      Pass$="ME"                  ! THIS LINE SHOULD BE REPLACED BY THE NEXT ONES
1080      ! LOAD SUB "SUBPR"
1090      ! CALL GETPASSWD(PASS$,VALID)
1100      ! IF NOT VALID THEN
1110      !     DISP "NO VALID PASSWORD FOR QUALTY DB"
1120      !     BEEP
1130      !     END
11        ! END IF
1150      Lst$="@;"
1160      DISP "  "
1170      CURSOR (5,18)
1180      DISP "One moment while data being sorted"
1190      DBOPEN (Base$,Pass$,3,S(*))
1200      IF S(0) THEN Dberr
1210      DBASE IS Base$
1220      IN DATA SET "VENDOR" USE ALL
1230      IN DATA SET "SHIPMT" USE ALL
1240      ASSIGN "WORKF" TO #1
1250      WORKFILE IS #1;THREAD IS "VENDOR","SHIPMT"
1260      FIND (Vend_nm$[1,1]>"A") AND (Vend_nm$[1,1]<"E")
1270      SORT BY Vend_no,Part_no$,Ardate
1280      DISP "  "
1290      PRINT USING Fm2
1300      FOR I=1 TO WFLEN(1)
1310         READ #1;Rec_no1,Rec_no2
1320         DBGET (Base$,"VENDOR",4,S(*),Lst$,Buff$,Rec_no1)
1330         IF S(0) THEN Dberr
1340         DBGET (Base$,"SHIPMT",4,S(*),Lst$,Buff$,Rec_no2)
1350         IF S(0) THEN Dberr
1360         PRINT USING Fm3;Vend_no,Vend_nm$,Part_no$,Amt,Ardate
1370      NEXT I
1380 Cls:DBCLOSE (Base$," ",1,S(*))
1390      END
1400 Dberr:DISP "Data base error - Aborted",S(0),S(6)
1410      GOTO Cls
1420 Fm2:IMAGE  //"VENDOR #        VENDOR NAME                    PART #          AMOUN
1430 Fm3:IMAGE  2X,4D,5X,30A,15A,DDCDDD,5X,6D
```

```
1000      SUB Getpasswd(Passwd$,Accepted)
1010          !   HPBB LAB
1020          !   SUBPROGRAM TO GET THE PASSWORD FOR THE QUALTY DB
1030          !   ACCEPTED: PASSWORD IS VALID
1040          !   PASSWD$:  NAME OF THE PASSWORD
1050          !
1060          ! RE-STORE "SUBPR"                                    M.B.
1070         Accepted=0
1080         Counter=0
1090 Main:   DISP "  "
1100         Counter=Counter+1
1110         DISP "HPBB LAB        SUBPROGRAM TO GET PASSWORD FOR QUALITY DB"
1120         CURSOR (10,10)
1130         DISP "ENTER PASSWORD: ";
1140         ACCEPT Passwd$
1150         Passwd$=TRIM$(Passwd$)
1160         IF NOT LEN(Passwd$) THEN Wrongpasswd
1170         IF Passwd$<>"ME" THEN Wrongpasswd
1180 Exit:   Accepted=1
1190         DISP "  "
1200         SUBEXIT
1210 Wrongpasswd:BEEP
1220         DISP "WRONG PASSWORD; PLEASE CHECK LOWER AND UPPER CASE"
1230         IF Counter>3 THEN
1240            DISP "SORRY, BUT THE PASSWORD IS SET TO 'ME'"
1250            Passwd$="ME"
1260            WAIT 2500
1270            GOTO Exit
1280         END IF
1290         DISP "TO ENTER 'ME' WOULD BE A GOOD SUGGESTION"
1300         WAIT 2500
1310         GOTO Main
1320      SUBEND
```

```
updpar
  10           GLOBAL OPTION BASE 0 REAL!** ADDED LINE
  20           EXTERNAL Getpasswd(PS,R)
  30 !         HPBB  - LAB
  40 !         UPDATE PARTS FILE       -  PARTUP                    M.B.
  50 !
  60 ! RE-STORE "UPDPAR"
  70           DIM Lst$[2],Buff$[100],Clr$[4],Ans$[30]
  80           DIM Part_no$[10],Part_ds$[30]
  90           SHORT INTEGER S(9)!** SYNTAX CHANGE
 100           Base$="  QUALTY"
 110 ! if interpreted then getsub "subpr"
 120 ! call getpasswd(pass$,valid)
 130 ! if not valid then
 140 !    disp "NO VALID PASSWORD FOR QUALTY DB"
 150 !    beep
 160 !    end
 170 ! end if
 180           Pass$="ME"! this line should be replaced by the previous
 190           Clr$=" 27"H" 27"J"!** SYNTAX CHANGE
 200           Lst$="@;"
 210           DBOPEN Base$,PASSWORD=Pass$,MODE=3,STATUS=S(*)!** SYNTAX CHANGE
 220           IF S(0) THEN Dberr
 230           DBASE IS Base$
 240 Set1:     IN DATASET "PARTS" USE Part_no$,Part_ds$  !** UNTRANSLATABLE
 250 Kys:  !
 260           CLOSE FORM!** SYNTAX CHANGE
 270           DISP Clr$;
 280           OFF KEY !** SYNTAX CHANGE
 290           BEEP
 300           CURSOR (10,10)
 310           DISP "Please select a function"
 320           ON KEY 1 GOTO Addp;LABEL="ADD PART"!** SYNTAX CHANGE
 330           ON KEY 3 GOTO Delp;LABEL="DELETE PART"!** SYNTAX CHANGE
 340           ON KEY 8 GOTO Endd;LABEL="EXIT"!** SYNTAX CHANGE
 350           LOOP                !** ADDED LINE
 360              ACCEPT !** SYNTAX CHANGE
 370              EXIT IF RESPONSE=1!** ADDED LINE
 380           ENDLOOP          !** ADDED LINE
 390 Addp:     OFF KEY !** SYNTAX CHANGE
 400           ON KEY 1 GOTO Compl;LABEL="ACCEPT DATA"!** SYNTAX CHANGE
 410           DISP Clr$;
 420           OPEN FORM "QAF1"+":"+"QAF1"!** SYNTAX CHANGE
 430           CLEAR FORM !** SYNTAX CHANGE
 440 In1:      LOOP                !** ADDED LINE
 450              INPUT
 460              EXIT IF RESPONSE>0!** ADDED LINE
 470           ENDLOOP          !** ADDED LINE
 480           GOTO In1
 490 Compl:    !
 500           ENTER Part_no$,Part_ds$
 510           IF NOT LEN(Part_no$) THEN Addp
 520           CLOSE FORM!** SYNTAX CHANGE
 530           DISP Clr$;
 540           DBPUT Base$ USING Set1,DATASET="PARTS",STATUS=S(*),ITEMS=Lst$!** SYNTAX CHANGE
 550           IF S(0)=43 THEN Dup
 560           IF S(0) THEN Dberr
 570           DISP Clr$;
 580           DISP "Part: ";Part_no$;"  ";Part_ds$;"  being added"
 590           WAIT (3000)/1000!** SYNTAX CHANGE
 600           GOTO Kys
 610 Dup:      DISP
 620           DISP "Duplicate Part - Try Again!"
 630           WAIT (3000)/1000!** SYNTAX CHANGE
 640           GOTO Kys
 650 Delp:     OFF KEY !** SYNTAX CHANGE
 660           DISP Clr$;
 670           OPEN FORM "QAF2"+":"+"QAF2"!** SYNTAX CHANGE
 680           CLEAR FORM !** SYNTAX CHANGE
 690           ON KEY 1 GOTO A1;LABEL="ACCEPT DATA"!** SYNTAX CHANGE
 700 In2:      LOOP                !** ADDED LINE
 710              INPUT
 720              EXIT IF RESPONSE>0!** ADDED LINE
 730           ENDLOOP          !** ADDED LINE
 740           GOTO In2
 750 A1:       ENTER Part_no$
 760           DBGET Base$ USING Set1,DATASET="PARTS",MODE=7,STATUS=S(*),ITEMS=Lst$,KEY=Part_no$  !**
 770           IF S(0)=17 THEN Nomatch
 780           IF S(0) THEN Dberr
 790           DISP Part_ds$
 800           LDISP "Is this the part to be deleted?"
 810           LDISP "  Press key for YES or NO"
 820           OFF KEY !** SYNTAX CHANGE
 830           ON KEY 1 GOTO Dlt,LABEL="YES"!** SYNTAX CHANGE
 840           ON KEY 3 GOTO Kys;LABEL="NO"!** SYNTAX CHANGE
 850           ON KEY 8 GOTO Kys;LABEL="EXIT"!** SYNTAX CHANGE
 860           LOOP                !** ADDED LINE
 870              ACCEPT !** SYNTAX CHANGE
 880              EXIT IF RESPONSE=1!** ADDED LINE
```

```
 890             ENDLOOP          !** ADDED LINE
 900 Dlt:        DBDELETE Base$,DATASET="PARTS",STATUS=S(*)!** SYNTAX CHANGE
 910             CLOSE FORM!** SYNTAX CHANGE
 920             IF S(0) THEN Dberr
 930             OFF KEY
 940             DISP "Part :";Part_no$,Part_ds$,"has been deleted"
 950             WAIT 2
 960             GOTO Kys
 970 Dberr:      DISP Clrs
 980             CLOSE FORM!** SYNTAX CHANGE
 990             CURSOR (12,1)!** SYNTAX CHANGE
1000             DISP "DATA BASE ERROR - ABORTED",S(0),S(6)
1010 Endd:       DBCLOSE Base$,DATASET=" ",MODE=1,STATUS=S(*)  !** SYNTAX CHANGE
1020             END
1030 Nomatch:    DISP "       NO  MATCH - RE-ENTER"
1040             WAIT (2000)/1000!** SYNTAX CHANGE
1050             GOTO Delp
```

```
!  qasort
    10          GLOBAL OPTION BASE 0,REAL!** ADDED LINE
    20          EXTERNAL Getpasswd(PS,R)
    30 !
    40 !          HPBB LAB          SORT QUALTY DB
    50 !
    60 !  RE-STORE "QASORT"                                          M.B.
    70          DIM Lst$[2],Buff$[100]
    80          DIM Part_no$[10],Part_ds$[30],Projct$[10],Vend_ad$(1)[30],Vend_nm$[30]
    90          REAL Amt,Ardate
   100          SHORT INTEGER Vend_no,Yield
   110          INTEGER Rec_no1,Rec_no2
   120          SHORT INTEGER S(9)!** SYNTAX CHANGE
   130          Base$="  QUALTY"
   140          Pass$="ME"! this line should be replaced by the following
   150 ! if interpreted then getsub "subpr"
   160 ! call getpasswd(pass$,valid)
   170 ! if not valid then
   180 !    disp "NO VALID PASSWORD FOR QUALTY DB"
   190 !    beep
   200 !    end
   210 ! end if
   220          Lst$="@;"
   230          DISP '27"H"'27"J"!** SYNTAX CHANGE
   240          CURSOR (18,5)!** SYNTAX CHANGE
   250          DISP "One moment while data being sorted"
   260          DBOPEN Base$,PASSWORD=Pass$,MODE=3,STATUS=S(*)!** SYNTAX CHANGE
   270          IF S(0) THEN Dberr
   280          DBASE IS Base$
   290 Set1:    IN DATASET "PARTS" USE Part_no$,Part_ds$
   300 Set2:    IN DATASET "VENDOR" USE Vend_no,Vend_nm$,Vend_ad$(*)
   310 Set3:    IN DATASET "SHIPMT" USE Part_no$,Vend_no,Amt,Projct$,Ardate,Yield
   320          ON ERROR GOTO Purgeerr
   330          PURGE "wfile"
   340 Purgeerr: ON ERROR GOTO Createerr
   350          CREATE BINARY "wfile",FILESIZE=128
   360 Createerr: ON ERROR GOTO Assignerr
   370          ASSIGN "wfile" TO #1
   380          WORKFILE IS #1!** SYNTAX CHANGE
   390          POSITION #1;RESET!** ADDED LINE
   400          OFF ERROR
   410          GOTO Contin
   420 Assignerr: BEEP
   430          DISP "Unable to access workfile 'wfile'"
   440          END
   450 Contin: !
   460 Thread1: THREAD IS Set2,Set3
   470          SEARCH USING Thread1;(Vend_nm$[1,1]>"A") LAND (Vend_nm$[1,1]<"E")!** UNTRANSLATABLE
   480          SORT USING Thread1;Vend_no,Part_no$,Ardate
   490          DISP '27"H"'27"J"!** SYNTAX CHANGE
   500          PRINT USING Fm2
   510          FOR I=1 TO NUMREC(@ABS(1))!** SYNTAX CHANGE
   520             READ #1;Rec_no1,Rec_no2
   530             DBGET Base$ USING Set2,DATASET="VENDOR",MODE=4,STATUS=S(*),ITEMS=Lst$,KEY=Rec_no
   540             IF S(0) THEN Dberr
   550             DBGET Base$ USING Set3,DATASET="SHIPMT",MODE=4,STATUS=S(*),ITEMS=Lst$,KEY=Rec_no
   560             IF S(0) THEN Dberr
   570             PRINT USING Fm3;Vend_no,Vend_nm$,Part_no$,Amt,Ardate
   580          NEXT I
   590 Cls:     DBCLOSE Base$,DATASET=" ",MODE=1,STATUS=S(*) !** SYNTAX CHANGE
   600          END
   610 Dberr:   DISP "Data base error - Aborted",S(0),S(6)
   620          GOTO Cls
   630 Fm2:     IMAGE //"VENDOR #       VENDOR NAME                PART #          AMOUNT       DATE"/
   640 Fm3:     IMAGE 2X,4D,5X,30A,15A,DDCDDD,5X,6D
```

```
! subpr
  10          GLOBAL OPTION BASE 0,REAL!** ADDED LINE
  20          SUB Getpasswd(Passwd$,Accepted)
  30        !   HPBB LAB
  40        !   SUBPROGRAM TO GET THE PASSWORD FOR THE QUALTY DB
  50        !   ACCEPTED: PASSWORD IS VALID
  60        !   PASSWD$:  NAME OF THE PASSWORD
  70        !
  80        ! RE-STORE "SUBPR"                                          M.B.
  90            Accepted=0
 100            Counter=0
 110 Main:      DISP '27"H"'27"J"!** SYNTAX CHANGE
 120            Counter=Counter+1
 130            DISP '27"&dDHPBB"'128" "'132"LAB"'128"       "'132"SUBPROGRAM"'128" "'132"TO"'128
    ORD"'128" "'132"FOR"'128" "'132"QUALITY"'128" "'132"DB"'128!** SYNTAX CHANGE
 140            CURSOR (10,10)!** SYNTAX CHANGE
 150            DISP "ENTER PASSWORD: ";
 160            LOOP              !** ADDED LINE
 170               ACCEPT Passwd$
 180               EXIT IF RESPONSE>0!** ADDED LINE
 190            ENDLOOP     !** ADDED LINE
 200            Passwd$=TRIM$(Passwd$)
 210            IF NOT LEN(Passwd$) THEN Wrongpasswd
 220            IF Passwd$<>"ME" THEN Wrongpasswd
 230 Exit:      Accepted=1
 240            DISP '27"H"'27"J"!** SYNTAX CHANGE
 250            SUBEXIT
 260 Wrongpasswd: BEEP
 270            DISP
 280            DISP "WRONG PASSWORD; PLEASE CHECK LOWER AND UPPER CASE"
 290            IF Counter>3 THEN
 300               DISP "SORRY, BUT THE PASSWORD IS SET TO 'ME'"
 310               Passwd$="ME"
 320               WAIT (2500)/1000!** SYNTAX CHANGE
 330               GOTO Exit
 340            ENDIF
 350            DISP "TO ENTER 'ME' WOULD BE A GOOD SUGGESTION"
 360            WAIT (2500)/1000!** SYNTAX CHANGE
 370               GOTO Main
 380         SUBEND
```

Lab4: HP260 to HPBB Migration Training
****

This lab should be used to get familar with the joinform editor.
In your handouts you will find the manual for the joinform editor.

1) Create two forms with the joinform editor. (:run joinedit.pub.sys)

2) Put the before created forms in one forms- file.

[3)] Write a program to load and use the forms.

Lab 5: HP260 to HPBB Migration Training

In this lab the HPBB compiler will be used.

1) Modify the programs "UPDPAR" and "QASORT" in this way that the DB password "ME" will be read in the routine "getpasswd" which is part of the subprogram "subpr".

2) Compile the three programs "UPDPAR", "QASORT" and "SUBPR" and put the generated code into one USL.

3) Use the SEGMENTER to have a look into this USL and to modify the file.

4) Generate a running version of the two programs. (:PREP usl-file)

5) SAVE the temporary program file and run it.

```
  27252 THREAD IS !!"GL-TRANS"  !** ADDED LINE  !** UNTRANSLATABLE
                ^
Syntax error at character 19
Statement needs  number,  identifier
  27300      SEARCH USING !(Account_no>=Store_no*1000000) LAND (Account_no<=Stor
e_no*1000000+999999)  !** UNTRANSLATABLE
                        ^
Syntax error at character 27
Statement needs  number,  identifier
  28352 THREAD IS !!"GL-BALANCE"  !** ADDED LINE  !** UNTRANSLATABLE
                ^
Syntax error at character 19
Statement needs  number,  identifier
  28400      SEARCH USING !(Account_no>=Store_no*1000000) LAND (Account_no<=Stor
e_no*1000000+999999)  !** UNTRANSLATABLE
                        ^
Syntax error at character 27
Statement needs  number,  identifier
  30752 THREAD IS !!"REQ"  !** ADDED LINE  !** UNTRANSLATABLE
                ^
Syntax error at character 19
Statement needs  number,  identifier
  30800   SEARCH USING !(Req_no>=VAL(VAL$(Store_no)+"0000")) LAND (Req_no<=VAL(V
AL$(Store_no)+"9999"))  !** UNTRANSLATABLE
                      ^
Syntax error at character 24
Statement needs  number,  identifier
  31002 THREAD IS !!"INVENTORY"  !** ADDED LINE  !** UNTRANSLATABLE
                ^
Syntax error at character 19
Statement needs  number,  identifier
  31050   SEARCH USING !I_store_no=Store_no  !** UNTRANSLATABLE
                      ^
Syntax error at character 24
Statement needs  number,  identifier
  31250      Mesg$="STORE NUMBER '"+VAL$(Store_no)+"' CANNOT BE DELETED BECAUSE
"+Mesg$  !** SYNTAX CHANGE
Error 110
Program unit is too large.  No space available to process this line.
>LIST
 !  CV6
      1 GLOBAL OPTION BASE 0,REAL  !** ADDED LINE
  10000 ! COPYRIGHT MANAGEMENT BUSINESS SYSTEMS,1982.  ALL RIGHTS ARE RESERVED.*
***
  10050 ! CMSTOR  - ADD,MODIFY,DELETE,DISPLAY STORE INFO PROGRAM  12/24/82
  10100 ! REVISION: 1.13.00 LAST EDIT: 01/27/87 MODIFIED BY : MBS-RIU,JGA
  10150 ! FORMS:    CMfm02.00.13
  10200 ! FILES:    NONE
  10250 ! DATABASES/DATASETS: CMDB/BRANCH,TYPE,STORE,WAREHOUSE,LOCATION
  10300 Dell: OPTION BASE 1
  10350   COM Date$[8],Title$[40], SHORT INTEGER Printer,Width,Menu,S(0:9),Title
$(7)[40] !** SYNTAX CHANGE
  10400   COM Base$[16],Pass$[8],Buff$[1000],Lst$[2],Rpt_title$[30],Wfvol$[8]
  10450   COM Mesg$[80],Clearkey$,Shiftclear$, SHORT INTEGER Curkey !** SYNTAX C
HANGE
  10500   COM SHORT INTEGER Day,Backup,Dailyend,Monthend,Yearend  !** SYNTAX CHA
NGE
  10550   COM Base1$[16], SHORT INTEGER Period !** SYNTAX CHANGE
  10600 !          ***************  INITIALIZE LOCAL VARIABLES  **************
***
  10650 Entry: DIM Entry$[40],Entry1$(12)[30],Store_name$[30],Dataset$[15],Whse_
name$[20],Base2$[16]
```

```
  10700   DIM Branch_name$[30],Type_title$[20],L_name$[20],S_whs_name$(3)[20],R$
[3]
  10750   SHORT INTEGER I,S_branch_no,Select,S_location_no,Whse_no(3),Type,S_his
tory,S  !** SYNTAX CHANGE
  10800   SHORT INTEGER S_type,S_whs_no(3),S_req_no,Type_no,Branch_no,S_print,S_
seq_no  !** SYNTAX CHANGE
  10850   SHORT INTEGER G_whse_no1,G_whse_no2,G_whse_no3,G_whs_no,G_type_no,J,Re
c_no  !** SYNTAX CHANGE
  10900   SHORT INTEGER G_branch_no,G_location_code,G_location_no,Entry_count,Sa
ve_type,Save_seq_no,Save_print  !** SYNTAX CHANGE
  10950   SHORT REAL S_cost_factor,Store_no,I_store_no,S_rate,Cash_acct_no  !**
SYNTAX CHANGE
  11000   REAL Req_no,Account_no,Cp_gl_trans(28),Np_gl_trans(7),Cy_gl_balances(1
3),Py_gl_balances(13)
  11050   IF INTERPRETED THEN GET SUB "CMUTIL"   !** SYNTAX CHANGE
  11100   ON HALT CALL Halt
  11150   ON ERROR CALL Error
  11200   Entry$,Base$="  CMDB"
  11250   DBOPEN Base$,PASSWORD=Pass$,MODE=1,STATUS=S(*)     !** SYNTAX CHANGE
  11300   IF S(0)=-1 THEN
  11350 Busy_db: CALL Message("ANOTHER USER HAS EXCLUSIVE ACCESS TO THE "+Entry$
+" DATA BASE",3,1,1)  !** SYNTAX CHANGE
  11400       WAIT (4000)/1000  !** SYNTAX CHANGE
  11450       GOTO Exit
  11500   ENDIF
  11550   IF S(0) THEN Dberror
  11600   Entry$,Base1$="  HGLDB"
  11650   DBOPEN Base1$,PASSWORD=Pass$,MODE=1,STATUS=S(*)     !** SYNTAX CHANGE
  11700   IF S(0)=-1 THEN Busy_db
  11750   IF S(0) THEN Dberror
  11800   DBASE IS Base1$
  11850   IN DATASET "GL-TRANS" USE All  !** UNTRANSLATABLE
  11900   IN DATASET "GL-BALANCE" USE All  !** UNTRANSLATABLE
  11950   Entry$,Base2$="  CMIN"
  12000   DBOPEN Base2$,PASSWORD=Pass$,MODE=1,STATUS=S(*)     !** SYNTAX CHANGE
  12050   IF S(0)=-1 THEN Busy_db
  12100   IF S(0) THEN Dberror
  12150   DBASE IS Base2$
  12200   IN DATASET "INVENTORY" USE All  !** UNTRANSLATABLE
  12250   DBASE IS Base$
  12300   IN DATASET "BRANCH" USE Branch_no,Branch_name$,SKIP 1,Cash_acct_no  !*
* SYNTAX CHANGE
  12350   IN DATASET "STORE" USE All  !** UNTRANSLATABLE
  12400   IN DATASET "TYPE" USE All  !** UNTRANSLATABLE
  12450   IN DATASET "WAREHOUSE" USE All  !** UNTRANSLATABLE
  12500   IN DATASET "LOCATION" USE All  !** UNTRANSLATABLE
  12550   IN DATASET "REQ" USE Req_no
  12600 !          ***********  INPUT CODES AND DESCRIPTIONS  ****************
***
  12650 Strt: CALL Getform("CMfm02","STORE INFORMATION DATA ENTRY SCREEN",Date$,
Rpt_title$)
  12700 Strt_ret: CURSOR OFLD(4)  !** SYNTAX CHANGE
  12750   Title$="STORE INFORMATION DATA ENTRY SCREEN"
  12800   DISP Shiftclear$
  12850   CURSOR OFLD(3)  !** SYNTAX CHANGE
  12900   DISP RPT$(" ",(40.5-LEN(Title$))/2)+Title$  !** SYNTAX CHANGE
  12950   OFF KEY  !** SYNTAX CHANGE
  13000   ON KEY 1 GOTO Selection;LABEL="ADD STORE"  !** SYNTAX CHANGE
  13050   ON KEY 2 GOTO Selection;LABEL="MODIFY STORE"  !** SYNTAX CHANGE
  13100   ON KEY 3 GOTO Selection;LABEL="DELETE STORE"  !** SYNTAX CHANGE
  13150   ON KEY 4 GOTO Selection;LABEL="DISPLAY STORE"  !** SYNTAX CHANGE
  13200   ON KEY 7 GOTO Report;LABEL="REPORT STORE"  !** SYNTAX CHANGE
  13250   ON KEY 8 GOTO Exit;LABEL="EXIT"  !** SYNTAX CHANGE
  13300   CALL Message("Please select a function key.",1,1,1)
  13350    LOOP                  !** ADDED LINE
  13351 ACCEPT  !** SYNTAX CHANGE
```

```
13352    EXIT IF RESPONSE=1   !** ADDED LINE
13353 ENDLOOP                 !** ADDED LINE
13400 Selection: Select=CURKEY MOD 8
13450 Selection1: OFF KEY  !** SYNTAX CHANGE
13500   IF Select=1 THEN Title$="ADD STORE INFORMATION"
13550   IF Select=2 THEN Title$="MODIFY STORE INFORMATION"
13600   IF Select=3 THEN Title$="DELETE STORE INFORMATION"
13650   IF Select=4 THEN Title$="DISPLAY STORE INFORMATION"
13700   CURSOR OFLD(3)  !** SYNTAX CHANGE
13750   DISP RPT$(" ",(40.5-LEN(Title$))/2)+Title$  !** SYNTAX CHANGE
13800   IF Select=1 THEN
13850 Dbcap: IF NOT FNDb_cap(Base$,"STORE",Buff$) THEN Stpp
13900   ENDIF
13950 Security: CURSOR OFLD(4)  !** SYNTAX CHANGE
14000   OFF KEY  !** SYNTAX CHANGE
14050   DISP Shiftclear$
14100   ON KEY 7 GOTO Strt_ret;LABEL="START OVER"  !** SYNTAX CHANGE
14150   ON KEY 8 GOTO Exit;LABEL="EXIT"  !** SYNTAX CHANGE
14200 Retry: CALL Message("Please input a STORE NUMBER from 100000 to 999999."
,1,1,1)
14250    LOOP                 !** ADDED LINE
14251 LINPUT "";Entry$
14252    EXIT IF RESPONSE>0   !** ADDED LINE
14253 ENDLOOP                 !** ADDED LINE
14300   IF FNNum_check((Entry$),12,Mesg$) THEN
14350      CALL Message(Mesg$,3,1,1)
14400      WAIT (3000)/1000  !** SYNTAX CHANGE
14450      GOTO Retry
14500   ENDIF
14550   IF (VAL(Entry$)>999999) LOR (VAL(Entry$)<100000) THEN Retry  !** SYNTA
X CHANGE
14600   Store_no=VAL(Entry$)
14650   G_branch_no=VAL(Entry$[1,3])
14700   Dataset$="STORE"
14750   DISABLE
14800   S,Save_type,Save_seq_no,Save_print=0
14850   DBGET Base$ INTO Buff$,DATASET="STORE",MODE=7,STATUS=S(*),ITEMS=Lst$,K
EY=Store_no       !** SYNTAX CHANGE
14900   IF S(0)=17 THEN
14950      IF Select=1 THEN
15000         GOSUB Branch_ck
15050         CURSOR OFLD(4)  !** SYNTAX CHANGE
15100         DISP Store_no,"",Branch_no,Branch_name$
15150         GOTO Input
15200      ELSE
15250         GOTO No_match
15300      ENDIF
15350   ENDIF
15400   IF S(0) THEN Dberror
15450   Save_type=S_type
15500   Save_seq_no=S_seq_no
15550   Save_print=S_print
15600   GOSUB Branch_ck
15650   GOSUB Get_recs
15700 Select_ck2: GOSUB Display
15750   IF Select=1 THEN Dupl_code
15800   IF Select=2 THEN Input
15850   IF Select=3 THEN Delete
15900   IF Select=4 THEN Display_wait1
15950    LOOP                 !** ADDED LINE
15951 ACCEPT  !** SYNTAX CHANGE
15952    EXIT IF RESPONSE=1   !** ADDED LINE
15953 ENDLOOP                 !** ADDED LINE
16000 Input: CALL Message("Please complete this form.",1,1,1)
16050   ENABLE
16100   CURSOR IFLD(1)  !** SYNTAX CHANGE
```