## Philips PTS 6000 Software

In the PTS 6000 Bank Terminal System, both the hardware and the system applications are controlled by a software program present in the system memory. The concept of absolute software control, together with the modular design of the software, makes the system extremely flexible and so enables the user to devise an optimum solution for his particular application requirements. Moreover, the software has also been designed in such a way that the minimum of effort is needed in adapting the PTS 6000 Bank Terminal System to a bank's requirements. The software is thus employed to customise the PTS 6000 Bank Terminal System to make optimal use of the hardware configuration possibilities.

A number of different tasks (such as hardware control, transaction control and system control) must be performed by the software. To simplify matters, the software is divided into two major categories. One of these, the application software, is designed to control and perform transactions as specified by the user, and therefore varies from one bank to another. Application software is written as for a single operator, and the time-sharing and multi-tasking functions are handled by executive subsystems. The second category known as the system software is standardised and comprises an operating system, program development tools and support utilities.

The system software has interfaces with:

- the terminals
- the data communication network
- the application software
- the digital cassette recorder
- the terminal computer.
- other peripheral units of the terminal computer.

Subsystems have been developed for all of these interfaces, and each subsystem is given its own specific function.

## Philips PTS 6800 Software

### GENERAL

The PTS 6800 software has been specifically developed for the Philips PTS 6000 Terminal Systems. During development much emphasis was placed on the convenience of use and suitability for the different configurations. Moreover, the software has also been designed in such a way that a minimum of effort is needed to enable the user to derive an optimum solution for his particular application requirements. The software is thus designed to customize the PTS 6000 Terminal System to optimize the hardware configuration capabilities.

Each Philips PTS 6800 computer can control several terminal stations, and since the software is modular in concept, each terminal can have its own tailor-made software module. However if several terminals share idential modules, the application software is written as for a single terminal.

Supporting software such as a CREDIT-translator, linkage editor, debugging programs and utilities are available to assist in the speedy implementation of the application software.

Memory economy and programming economy is ensured by using an interpreter to carry out the individual operations of the application programs. The competition between terminals to use the central processing unit is controlled on a priority and time allocation basis by running the application packages and interpreter under the control of a Terminal Operating System (TOSS) which shares out processor time, handles I/O requests, etc. according to a defined interrupt system.

The interrupt system is used for all peripheral operations and for handling internally generated interrupts. The system handles a number of hardware interrupt levels according to their priority, which is established by pre-wiring on channel units. The priority interrupt request is accepted and compared with the priority level of the running program. If the priority level of the interrupt is higher than that of the running program, the program is interrupted and the significant register contents are stored in a memory stack. A new program is then started by the interrupt and this program runs until stopped by a higher priority or until it is completed, whereafter the interrupted program is continued.

The application software is written in CREDIT. This is an application transaction language specifically designed for use with the PTS 6800 Terminal System. It is a symbolic language containing all the functions required for terminal applications. In this way it is very easy to prepare tailor-made software modules. Some samples are given below.

Each transaction routine is controlled by the software precisely as desired and, if the terminal operator makes an error during the transaction procedure, or inadvertent tries to use the wrong procedure, an immediate warning is given to the operator and the routine is stopped until the correct procedure is carried out.

Also the layout of printed information is defined by the program. To decrease the chance of errors being made at the keyboard, those keys not required for a particular operation are disabled.

Weighting techniques are used to supplement data stored in registers and allow check digit verification to further reduce the risk of errors. Before information is forwarded, in on-line systems, it is automatically edited and arranged into suitable format for subsequent processing.

# Philips PTS 6800 Software — TOSS

## INTRODUCTION

TOSS (Terminal Operating System Software) is a real time operating system designed specifically for use with the PTS 6000 Terminal System, where a terminal computer out of the PTS 6800 range is used.

The concept of absolute software control, together with the modular design of the software, makes the system extremely flexible, and so enables the user to derive an optimum solution for his particular application requirements. Moreover, a transaction language called CREDIT has been designed so that the minimum of effort is needed in adapting the PTS 6000 Terminals System to the needs of any application.

The TOSS monitor aided by the intelligence of a PTS 6800 Terminal Computer enables the PTS 6000 Terminal System to handle several terminals simultaneously in use, without restricting the performance of the terminal operators. This is achieved by effectively controlling all transfers of data and operating modes of the terminal devices and by appropriately allocating processing time according to the load on each terminal.

Four main executive areas are used in the operating system to satisfy these requirements:

- dispatcher
- monitor processors
- monitor tables
- input/output drivers and interrupt routines.

An advanced data management package is available to enable transfer of data to and from disk, which supports random, sequential and indexed random files. The integrated display management facilities perform an optimal use of the display without almost any programming effort.

Furthermore a memory management technique is implemented for the application package to improve the core economy. So modules which are not frequently used can share the memory. This memory management is fully controlled by the system software.

## GENERAL ORGANIZATION

### The task principle

TOSS monitor is a real time monitor controlling several independent tasks that can run on different priority levels.

A task is associated to an area in the monitor describing the status and device configuration. The number of tasks is a parameter at system generation.

In a system a physical terminal is usually corresponding to one task. In on-line systems some processing is related to the trunk line, thus requiring another task. Additionally tasks can be defined for general peripherals such as disk, cassette recorder(s) and system operators panel, e.g. with a special purpose to process off-line logged transactions in recovery situations.

Each task runs its own sequence of instructions. When requested monitor function has been carried out, the task is put on queue for continued execution. This queuing is done using the principle of first in first out per priority level. Thus, a task with a high priority will interrupt a task with a lower priority. The task priority feature is used when different applications are run in the same computer.

The terminal program then is defined as a task executing the terminal functions and the terminal state is mainly described by the program counter and register contents.

A single terminal interface is implemented to perform easy writing of application, if several tasks have to be supported by the same terminal program.

### Monitor configuration

The monitor is table-oriented which means that system generation consists of defining the proper table contents and, after that, linking the tables to the monitor library. The following tables are important for the configuration:

- Task control table
  containing an indication for each task in the operating system.
- Task table — one per task
  which fully describes the task configuration and is also used by the monitor to save registers when the task is in a wait state.
- Common device table
  which lists the common devices. It has the same organization as the configuration part of the task table. Some devices such as the cassette recorders can be common to all tasks. This table is referenced when a device is not found in the configuration list of the current task table.
- Device work table — one per physical device
  containing all variable information associated with a physical device.

Each device has a file code which is used to select a device at I/O requests.

**Note:** There is no restriction to share one device among several tasks. One device can also be referenced by several file codes.

A monitor generation program is available to define the

## Philips PTS 6800 Software — TOSS

hardware configuration possibilities, this comprises definition of items such as :

- number of terminal classes
- individual terminal configurations per terminal class
- tasks definition
- type of on-line connections
- different types of terminal connections
- recording devices.

Program loading is performed normally from flexible disk, magnetic tape cassette or a disk depending on the configuration.

In order to minimize the number of program cassettes e.g. needed for different configurations in a project, a configuration program is available, which generates configuration dependent tables at program loading time. This feature enables an easy software distribution in a project where several terminal computers are included with different configurations.

### TOSS SUBSYSTEMS

#### TOSS monitor

The TOSS monitor is built up around main functional blocks:

- the dispatcher
  which allocates control processor unit resources to the different tasks and monitor modules.
- Monitor processors
  which perform different functions such as input/output operation, activate another task, get/release of buffer, wait for a certain period of time, intertask communication etc. as described later in this chapter.
- Monitor tables
  which describe the configuration for the system and all its terminals. Monitor tables also contain work areas for devices, stack for the interrupt system, and queues for different jobs.
- I/O drivers and interrupt handlers
  which take care of all communication with the devices. The drivers generate or delete control characters that are specific for each device. The I/O drivers are written reentrant which means that just one module (in memory) supports several identical devices simultaneously. A standard I/O interface is introduced to get the same I/O handling seen from the application program, independent of device type.

ASCII is standard code set at input and output. EBCDIC is available for DC and flexible disk. All references between user program and monitor are done using specific instructions. I/O drivers are available for devices such as:

- Teller Terminal Printer
- General Terminal Printer
- Numeric and Alphanumeric keyboard
- Character display
- Badge card reader
- Magnetic stripe unit
- System Operator's panel
- Digital Cassette Recorder
- Cartridge disk
- Flexible disk
- Console typewriter
- Magnetic tape unit
- Line printer
- Card reader
- Data communication

The modularity of the TOSS monitor enables a choice of any of the drivers, the corresponding monitor tables, and the monitor library. To generate a TOSS monitor for a specific configuration a sophisticated system generation program (SYSGEN) is available.

### TOSS DATA COMMUNICATION

The data communication is implemented as a standard driver to enable TOSS to communicate with other systems. In that respect master as well as slave drivers are available. The interface to the application is such that each communicating terminal regards the line as a device upon which the application program can execute read and write requests. This interface is independent of the lineprocedure used.

The DC-driver has the following functions:

- line procedure
- buffer management
- request time out supervision
- status control

There are different drivers for different line procedures, however, only the line procedure function is different. The other parts are options and parameter controlled.

Several slave drivers are available which are serving

## Philips PTS 6800 Software – TOSS

synchronous as well as asynchronous procedures such as :

- BSC multipoint (IBM 3270)
- BSC contention (IBM 2780/3780)
- SDLC (IBM 3600)
- Uniscope 100
- VIP 700
- HDLC
- MSV 1/2

Master drivers, which can be used in concentrator environment are serving protocols such as :

- BSC multipoint
- HDLC multipoint

### TOSS DATA MANAGEMENT

TOSS data management contains a set of routines which enable the user to handle data recorded on a disk in an optimal way.

It has the following features :

- Multi volume organization
- All accesses to a data file are on record level
- All records within one data file are of fixed length
- Data files can be used concurrently by a variable number of tasks, or related to one task only
- System software takes care that concurrent updating of the same record by a number of tasks is not possible (exclusive access mechanism)
- Access methods :
  - Sequential access
  - Random access
  - Indexed random access
- Functions to process a record :
  - Read record
  - Write record (including rewriting)
  - Delete record
- Creation/deletion of data files can be done off-line via utility programs
- File relations are defined by the user .

Each volume of a disk contains a number of cylinders; each cylinder being divided into tracks, and a track into sectors of fixed length. User programs are not aware of this division, they only address records within a file.

A volume can be :

- A cartridge, containing a removable disk
- A disk, as a permanent feature, mounted in a disk drive
- A flexible disk
- A fixed disk

TOSS data management supports mixed disk configurations. Furthermore a data file can be divided into several extents, on several volumes. Note that the flexible disks supported by the data management are labelled according to TOSS labelling system. However, TOSS supports fully the IBM labelling for flexible disks too, so compatibility to IBM 3740 formats is included.

### TOSS FILE MANAGEMENT

TOSS File Management is a tool for elementary file handling on direct access (disk) memories. File management is structured in two main parts :

1. File administration functions
   - Open an existing file
   - Create a new file
   - Close an opened file
   - Delete a file
   - Extend an existing file
   - Read file parameters
2. Input/Output functions
   - Attach file
   - Detach file
   - Read consecutive sectors
   - Write consecutive sectors

All file management functions are available in runtime.
TOSS file management is available as a layer between TOSS data management and the disk device drivers. TOSS file management can be used by the application programs even without data management.

### TOSS UTILITY PACKAGE

TOSS utility package contains a number of utilities which can be called, either by the application program or via stand-alone Control Command Interpreter (CCI).

Following utilities are included :

- Create volume
- Create file
- Delete file
- Print file
- Copy volume
- Copy file to file ( on disk, as well as on 1/2 inch magnetic tape or magnetic tape cassette)
- Copy cards to file
- Sort data / index file
- Create / reorganization of index file

In addition, IBM labelled flexible disks are supported with respect to :

- Create volume
- Copy files ( to and from TOSS labelled flexible disks).

# Philips PTS 6800 Software — CREDIT

## INTRODUCTION

CREDIT is an application orientated transaction language designed specifically for use with the PTS 6800 Terminal System. It is a symbolic language containing all the functions required for terminal applications and reduces programming effort significant compared with e.g. assembler programming.

A further advantage is brought by CREDIT being an interpreter based language. Total memory requirements based on the interpreter and application program requirements are less than the needs of an application program written in a direct executable language.

The source information is first compiled into an object code by a translator and then linked to interpretative code. This code and the interpreter are then loaded into core memory where the code is interpreted and executed by the interpreter during execution time.

The main objectives of using this procedure are to provide storage economy, and to obtain an easy coding language. Furthermore, CREDIT as an interpretive language enables tracing and debugging straight-forward.

## GENERAL ORGANIZATION

### Source program structure

A source program consists of a number of statements. It is composed of two main parts; the Data division and the Procedure division.

The Data division contains all statements defining the working storage structure, namely, all data definition statements. It preceeds the Procedure division.

The Procedure division contains all procedural statements, namely, statements defining operations and literal constants. It can contain different main routines and subroutines, as needed by the classes of terminals involved.

Assembly language written subroutines may be linked to a CREDIT-program. Any kind of data item or literal constant may be passed as an argument to an assembly written subroutine.

### Data definition statements

Data definitions have to be entered in a hierarchous structure, due to the various levels of data defining pseudo-operations. These levels are: terminal storage access defining, block defining, and variable defining.

Working storage blocks exist of the following categories:

- Common work blocks (CWB)
- Terminal work blocks (TWB)
- User work blocks (UWB)
- Swappable workblocks (SWB)

Variables exist of the following categories:

- Single variables
- Indexed variables with 1 index
- Indexed variables with 2 indexes.

These variables can be of the following types:

- Binary numericals
- BCD numericals
- String variables
- Boolean variables

### Procedural statements

Procedural statements describe the operations to take place. All procedural statements are contained in the procedure division. There are several types of procedural statements:

- Arithmetic statements
- Logic statements
- String handling statements
- Branch statements
- Subroutine control statements
- Input/output statements
- Scheduling and storage control statements.

Special attention is given to the I/O statements to perform input (e.g. from keyboard) and output programming in an easy way. Therefore keytables are used in the input handling and output are defined by write lists and pictures.

### Memory management

In order to make it possible to run CREDIT programs larger than physical core memory, a dynamic memory management technique can be used. The system software supervises the dynamic allocation of memory. The segments are defined by the user. Procedure division as well as data division are handled by the memory management.

### Display management

To enable an advanced programming of display applications e.g. in data entry environment, inquiry etc. display handling facilities are included. With these facilities the user can build up his own display application. However, a general display management can be used to handle forms on a display in conjunction with an input device e.g. keyboard. Basic tests and

## Philips PTS 6800 Software — CREDIT

controls are included and own defined tests can be supplemented.

The following functions are available:

- extended format directives, which enable an easy definition of the output part as well as the input part of the total picture.
- several standard input controls such as
  - numeric/alphanumeric
  - min./max. length
  - compulsory field
- standard functions to handle the different input and output fields and to control the cursor according to the relevant format list.

## Philips PTS 6800 Software — Assembler language

The assembler language can be used for programming the system software as well as the application software, although it is mainly used for system software. Assembler operations fall into the following types:

- memory reference instructions
- register to register instructions
- constant instructions
- shift instructions
- input/output instructions
- miscellaneous
- assembly process instructions

To improve the use of the assembly language in application programming an element addressing system is available together with a set of routines which performs functions as decimal addition and subtraction, multiplication and division and a powerful editing.
Though it is still possible to use the assembly language for application programming, it is strongly advised to use CREDIT.

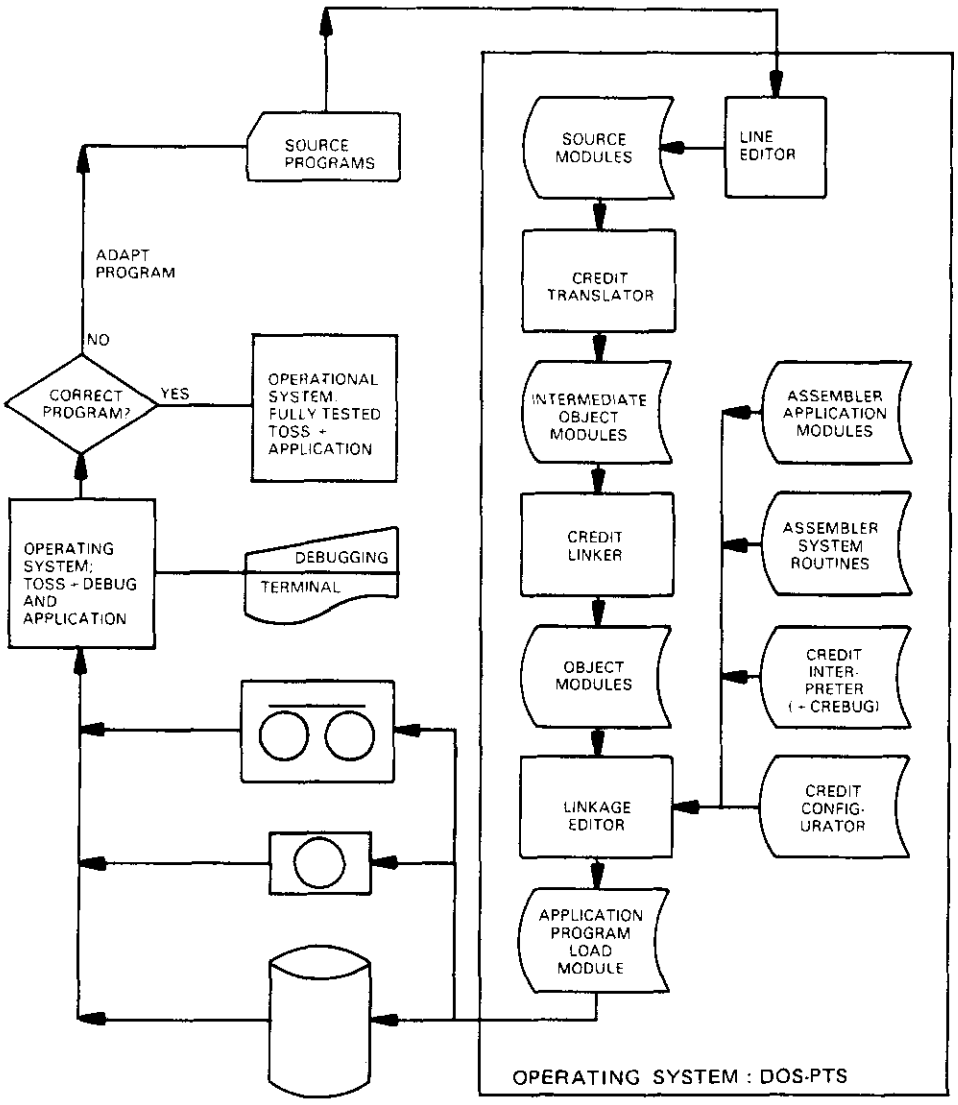
## Philips PTS 6800 Software — Program Development

Program development for the PTS 6800 system is handled by the disk operating system tool DOS-PTS.
DOS-PTS provides a disk-oriented system with all the facilities for program development and giving the necessary degree of control for the secure allocation and utilisation of both user and system programs within the system.

As can be seen on the illustration, source program modules in mnemonic form are fed onto disk. This source information is then translated by a translator program into object code which is also stored on disk. Modules are then linked via the Link Editor program to form an application package. This package is dumped onto cassette, flexible disk, or disk for subsequent running together with a debugging program on the PTS 6800 Terminal Computer. The program is recycled using the line editor when necessary until the application package is finalized.

All program development tools are disk resident. The following tools can be called by control commands:

- TRANSLATOR
- LINKER
- ASSEMBLER
- LINKAGE EDITOR
- LINE EDITOR



## Philips PTS 6800 Software — CREDIT translator

The CREDIT translator is a processor which converts CREDIT-source statements into intermediate object code. Source modules are translated separately, resulting in producing individual object modules.

During translation the translator generates a listing in three parts. Part one contains the CREDIT source statements, intermediate object code and error messages. The error messages are printed immediately after the source-line containing the error. An error counter is maintained by the translator and is printed at the end of the listing.

Other parts contain the data item name table and the procedure label table.

## Philips PTS 6800 Software — CREDIT Linker

The CREDIT — Linker is a two pass processor which converts intermediate object code produced by the CREDIT — Translator into object code which can be processed by the Linkage Editor.

Intermediate object modules may contain references to:

- labels in same module
- literals in same module
- labels in other CREDIT — modules
- assembler application modules

The first three types of reference are satisfied by the CREDIT— Linker. The remaining types of reference must be satisfied by the Linkage Editor.

The CREDIT Linker also converts the byte oriented addressing system used in the intermediate object code to a word oriented addressing system. A set of object modules is produced which bears a one-to-one relationship to the intermediate object modules.

During the linker process relevant listings are produced with information of tables, pools, cross reference statistics etc.

## Philips PTS 6800 Software — ASSEMBLER

ASSEMBLER is a programming tool which is used with the PTS 6800 software program which can be written in a low-level symbolic language. Each instruction in this language corresponds to a single instruction of one or two words in the machine language. The process of converting the program from the low-level language to machine code is called assembly and the program which performs the conversion is ASSEMBLER. An alternative description is to say ASSEMBLER is used to convert source modules written in assembly language into object modules suitable for linking to other object modules.

Additional facilities available with ASSEMBLER include error reporting and recovery, assembler listing, and the selection of peripheral devices to be used during the assembly run. A program written in assembler, cannot be used in a system with memory management.

## Philips PTS 6800 Software — Linkage Editor

The Linkage Editor running under DOS-PTS provides the system with the facility to link separate object modules either for direct loading and execution or for output, to be loaded later or used within a further linkage process. By the use of linking, all the advantages of modular programming are easily available. Modules which are to be linked contain specified external references and entry points to be used during linkage. The control of the linking process, by the operator, allows for the selection of the devices and mode to be used during processing. In addition, the linkage editor includes the facility to provide a list of the concerned modules and error reporting.

## Philips PTS 6800 Software — Line Editor

The Line Editor, running under DOS-PTS, provides all the facilities for updating of a source module and an additional facility to enable the alteration of a specified character string wherever such a string appears in a module.

The Line Editor operates at character, string, or line level. Functions include alteration, insertion, deletion, and listing of text. The system is extremely simple to operate as it has been designed for conversational interaction via the operator's typewriter. Editing is performed by a comprehensive set of operator messages.

An auxiliary input file can be used, to aid the manipulation of large portions of text.

## Philips PTS 6800 Software — Test and Debugging

### INTRODUCTION

Debugging programs are available to enable rapid error detection within programs modules, and to provide the programmer with the means to stop a program at specific points so that the contents of memory and/or registers may be checked or altered if necessary.

These debuggers offer conversational debugging facilities for CREDIT language programs or for Assembly language programs. Thus, it provides a convenient means with which the user can trace the execution of his program and perform extensive diagnostic procedures.

User programs can be processed step by step until completion, or up to a specific statement number.

Facilities include accessing the variables, commands to set or dump data, and breakpoints with conditional debugging.

### FUNCTIONAL DESCRIPTION

PTS 6800 has a debugger program called CREBUG intended for debugging application programs written in CREDIT and a debugger program called DEBUG to assist the debugging of assembler written software.

The debugger operates on an interrupt basis which means that programs can run simultaneously.

The following functions are available in CREBUG:

- Relocation registers which allow the setting of module base addresses and use the relative address in the program list.
- Examine and modify data elements and memory locations
- Test and verify on data elements
- Trace parts of the program
- Program control for program start, set breakpoint, remove breakpoint, proceed, loop, etc.
- Dump memory
- Hexadecimal calculation

Data elements are related to the current tasks, however, it is easy possible to get data elements of other tasks.

The following functions are available in DEBUG:

- Relocation registers: which allow the setting of module base addresses and use the relative address in the program list.
- Register dump/modification: which allow CPU registers to be display and modified
- Memory dump/modification
- Program control: for program start, set breakpoint, remove breakpoint, proceed, loop, etc.

The debugging tools CREBUG and DEBUG can respectevely be included in the interpreter or the TOSS monitor in a test environment.

The DEBUG program operates a system console typewriter only, while CREBUG operates a system console typewriter or any general printer or display terminal (defined at system generation time).

## Philips PTS 6800 Software – Utilities

Utility programs have been specially developed for use with the PTS 6000 Terminal System. The following utility programs are used with PTS 6800 system software to assist in loading, dumping, converting and copying data, and to provide necessary marking and labelling facilities during normal system operation or during program development.

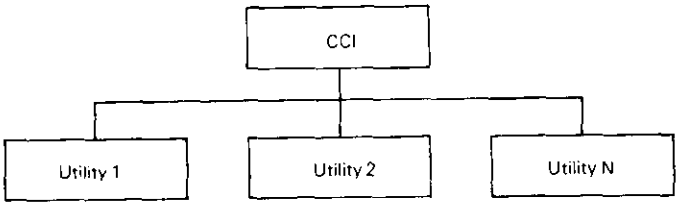
### DOS–PTS–UTILITIES

The PTS 6800 utility programs are delivered with the DOS-PTS software package. The following utilites are available, running under DOS-PTS.

- **CPLGEN :**
  Used to write IPL on cassette for PTS 6800 program cassettes
- **JESPER :**
  Used to convert load modules on disk and write them on cassette in a loadable form for the PTS 6800
- **PMPTS :**
  A premark program for formatting a DOS-PTS disk
- **DMPGEN :**
  Used to generate a cassette with the program DUMPER. DUMPER is used to dump core memory onto a cassette
- **PRDUMP :**
  Used to list a core dump on a line printer produced by the program DUMPER or DUMPFD
- **DUMPA :**
  Used to dump the contents of a cassette or magnetic tape onto a line printer
- **SUM :**
  Used to save the whole contents or a file of a complete disk onto magnetic tape
- **RUM :**
  Used to restore the whole contents or a file of a complete disk from a magnetic tape
- **XRF :**
  Cross reference program for source module
- **OBX :**
  Cross reference program for object modules
- **DMPGEF :**
  Used to generate a flexible with the program DUMPFD. DUMPFD is used to dump memory onto a flexible .
- **CASCOP :**
  Copies a cassette to another cassette.
- **FLXCOP :**
  Copies a flexible to another flexible.
- **PCAS :**
  Copies programs from disk load modules to a program loading cassette
- **PDISC :**
  Copies a configuration data file or a load file from a DOS-PTS format disk to a TOSS format disk
- **SYSGEN :**
  Used to generate a TOSS monitor
- **TOSSUT -**
  Controls the running of TOSS utilities CRF, CRV, DLF and PVC under DOS–PTS.

### TOSS UTILITY PACKAGE

A package which includes a TOSS monitor, Control Command Interpreter (CCI) and a number of utility functions. The Control Command Interpreter (CCI) reads input parameters necessary for the utility functions and stores them in a parameter block.
The CCI program also calls the required utility functions, thus forming the following structure for the utility package.




Each utility function is an independent subroutine which may also be incorporated in an application program environment.
Utility package comprises e.g.

- **CREATE VOLUME**
  Used to format a disk pack or flexible before it is actually used by the system. It writes a VOLUME LABEL (VL) and an empty VOLUME TABLE OF CONTENTS (VTOC) on disk. The utility program also writes cylinder identifiers in all sectors on the disk and tests for quality of each sector. If an unusable sector is found, this sector is withdrawn from the "user available area" and a dummy sector BADSPOT is created which occupies the sectors not to be used. The free space administration table in VTOC is also updated for the "badspot" area.
- **CREATE FILE**
  Used to create a file on a volume already formatted with the CREATE VOLUME utility program. The program searches for an empty space area large enough to occupy the file. If such an area is not available it searches for a maximum of 4 areas together being large enough to contain the file. The order in which the maximum of 4 volume names are given also determines the order in which a search is made for empty space.
- **DELETE FILE**
  Used to delete a data file or a file containing a loadable program from a volume. If the file to be deleted resides on more than one volume, the different volume names are entered as parameters. The free space administration table maintained in VTOC is updated for the released sectors and the file descriptor records in VTOC are set to spaces for the requested file name.

## Philips PTS 6800 Software — Utilities

- **PRINT VOLUME TABLE OF CONTENTS**
  Gives a listing of all relevant data in the VTOC. The number of free records in VTOC and free space administration table are also printed.
- **PRINT DISK SECTOR**
  Gives a listing of the contents of one or more sectors on a specified disk. Output format is hexadecimal together with ASCII representation.
- **UPDATE DISK SECTOR**
  Used to change one or more positions in a specified disk sector.
- **COPY VOLUME**
  This utility is used to copy a whole disk to another one which is formatted already. Checks are performed to protect against unproper handlings.
- **COPY FILES**
  The utility can handle input as well as output files on disk, flexible disk, 1/2 inch magnetic tape, magnetic tape cassette etc. Multivolume files are supported for disk devices as well as blocking facilities for the magnetic tape device. Input files from card reader are also supported.
- **COPY PROGRAM**
  The utility is used to move a program from DOS-PTS formatted disk and place it on a TOSS formatted volume (on cartridge disk or on flexible disk)
- **WRITE IBM-LABEL FLEXIBLE DISK**
  The write IBM-labels utility is used to write the necessary lables on a flexible disk to prepare it to be used for e.g. sequential logging. The initialization is according to GA 21—9182 document. (IBM manual)
- **COPY IBM-FLEXIBLE DISK**
  This copy utility transfers data sets from an IBM labelled flexible disk to a TOSS labelled disk and the other way round.
- **SORT FILE**
  The records in a disk file are sorted according to a symbolic key contained in the the records.
- **BUILD INDEX**
  An index file will be created from an existing data file which contains a symbolic key.
- **REORGANIZE INDEX FILE**
  The reorganize index file utility will create a correctly formatted index file from a sorted input file that contains all index records.
- **PRINT INDEX TRACK**
  The utility is used to print all the data relevant to an index track of an IBM formatted disk.
- **PRINT FILE SECTION**
  The utility lists records from a TOSS formatted disk file.
- **SCAN TAPE**
  The contents of a magnetic tape or cassette can be examined.
- **UNLOAD DEVICE**
  The utility is used to unlock the door of a flexible disk drive or to unload a cassette or magnetic tape.

### DATA COMMUNICATION UTILITIES

- **LINSIM / DLCSIM**
  This program is used at data communication tests to simulate the master on another PTS 6800 computer. In a number of buffers the user can specify his own test and control sequences to be transmitted. The received/transmitted data are outputted on a printer. Two different line codes are allowed, EBCDIC or ASCII. The program is procedure independent, although two versions are available for BSC, and HDLC-like procedures.
- **LINSPY / DLCSPY**
  The program is used for logging the line in a separate PTS 6800 Terminal Computer. The line information is stored in a circular buffer and printed out afterwards. To improve the printouts it is possible to define a pattern (e.g. idle poll) which is not printed but just counted. The transmitted and received data can be distinguished easily in the printout. EBCDIC as well as ASCII code is supported. The program is procedure independent, although two versions are available for BSC, and HDLC-like procedures.