3 ADMINISTRATION IN A RUNNING PLANT ................................................................. 3-2

3.1 PROGRAM COMPONENTS OF THE OM 650 SOFTWARE ........................................ 3-2
3.2 FILE SYSTEM STRUCTURE ON OM-650 PCS .......................................................... 3-4
3.3 DISTRIBUTED OM 650 SYSTEM .............................................................................. 3-6
3.4 IKZ - SIGNIFICATION ......................................................................................... 3-7
3.5 ENGINEERING DATA IN THE OM SYSTEM .......................................................... 3-9

3.5.1 INF ................................................................................................................. 3-9
3.5.2 MMI ............................................................................................................. 3-13
3.5.3 ASR ............................................................................................................. 3-17
3.5.4 MAC ........................................................................................................... 3-21
3.5.5 ARC ........................................................................................................... 3-22
3.5.6 LZA .......................................................................................................... 3-24
3.5.7 PRT ........................................................................................................... 3-25
3.5.8 BDM ........................................................................................................... 3-26

3.6 DIAGNOSTIC FILES ............................................................................................. 3-27
3.7 DATA TRANSFER ES -> OM ................................................................................. 3-28
3.8 COUPLING AS 620 <-> OM 650 ........................................................................ 3-29
3.9 PRINTER ADMINISTRATION ................................................................................. 3-32
3.10 HARDCOPY ....................................................................................................... 3-36

3.11 OM650 COMPONENTS - COMMISSIONING TOOLS ........................................ 3-38

3.11.1 Arc.Read - reading values from the archive .................................................. 3-38
3.11.2 Fb.Statist - List of the most frequent events of a function complex ............. 3-40
3.11.3 Arc.Moni - number of events in the short-term archive ............................. 3-41
3.11.4 Lza.Moni - number of events in the long-term archive ............................. 3-42
3.11.5 Fb.Read - reading function complex designations ...................................... 3-43
3.11.6 Ikz.Read - reading IKZ data for a KKZ ....................................................... 3-44
3.11.7 PL - indicating the status of the OM component and the OM manager .... 3-45
3.11.8 SAMPLER - collecting data for later analysis ............................................. 3-46
3.11.9 Mv.Reset - resetting alarm processing for each function complex ............ 3-47
3.11.10 Spool.Kill - deleting the data in a spool ...................................................... 3-48
3.11.11 Carrying out poll commands in the OM network .................................... 3-49
3.11.12 Compare - comparing command results .................................................. 3-50
3.11.13 rdb - reading out the AS620B component status .................................... 3-52

3.12 OM650 SUBCOMPONENT MOD HANDLING .................................................. 3-53

3.12.1 MOD handling .............................................................................................. 3-53
3.13 READING THE STATUS FROM THE AS620B (CPU'948R) ................................ 3-55
3.14 SETTING THE MAC ADDRESS ON THE CP1430 (AS) .................................. 3-57
3.15 UPS UNINTERRUPTIBLE POWER SUPPLY ..................................................... 3-58
3.16 TTD (TIME-TAGGED DATA) CHECK ............................................................... 3-59
3.17 DATA UPDATE .................................................................................................. 3-60
3 Administration in a running plant

3.1 Program components of the OM 650 software

The OM 650 contains the following software packages (called object manager):

- MMI - Man-Machine-Interface
- ASR - AS-Representing module or ASE by OMME
- ARC - Archive (short-term)
- MAC - Module Administration and Control (processing functions and ASD)
- LZA - Long-term archive
- NTB - Notepad
- PRT - Log manager (reports)
- BDM - Description data manager (KKS database)

A PC where only the MMI is running is called OT (Operating Terminal).
PCs having the managers ASR, ARC and MAC are called PU (Processing Units).
On a SU (Server Unit) the managers LZA, PRT, NTB and BDM are present.
Several plants also have PCs called PUSU. This is a combination of the functions of a PU and an SU.
A CU (Compact Unit) is a PC where all object managers are running.

Moreover the program package INF (infrastructure) is available. This package runs on every OM PC independently of its function. The infrastructure ensures OM-internal communication.

An internal number is assigned to every object manager. These numbers are important for trouble shooting (see the chapter on diagnostic files).
Object manager - number assignment

OM_TYP_LTK  -10  /* Reference for LT-component type
OM_TYP_ASR  -20  /* Reference for the object manager type ASR-TXP
  */
OM_TYP_ASE  -21  /* Reference for the object manager type ASR-TME
  */
OM_TYP_ARC  -30  /* Reference for the object manager type ARC
  */
OM_TYP_BDM  -40  /* Reference for the object manager type BDM
  */
OM_TYP_MAC  -50  /* Reference for the object manager type MAC
  */
OM_TYP_PRT  -60  /* Reference for the object manager type PRT
  */
OM_TYP_LZA  -70  /* Reference for the object manager type LZA
  */
OM_TYP_MMI  -80  /* Reference for the object manager type MMI
  */
OM_TYP_NTB  -90  /* Reference for the object manager type NTB
  */
3.2 File system structure on OM-650 PCs

The file system of an OM PC is principally the same on every computer type (OT, PU, SU or CU). There are only differences in size.
In OM release 3 (SCO 3) the structure is as follows:

```
/root
  /txpsys
  /txpproj
  /txptest
  /txpproz
```

The operating system is installed on the root file system. The OM software is stored on the other file systems.
The executable programs of the OM software are located in the file system txsys.
In the txpproj file system the engineering data are stored (as for example the pictures for the MMI or the user-specific coupling data from ASR).
In the directory txptest the diagnostic files are located (hints and errors), which are created by every OM program.
The process data (hard copies, notes,...) are stored in the directory txpproz.

Each file system has a component-related organisation (depending on the object managers):

```
/txpsys     /txpproj/proj_std     /txpproz     /txpproz
  /inf       /inf                 /inf        /inf
  /mmi       /mmi                 /mmi        /mmi
  /asr       /asr                 /asr        /asr
  /arc       /arc                 /arc        /arc
  /mac       /mac                 /mac        /mac
  /lza       /lza                 /lza        /lza
  /prt       /prt                 /prt        /prt
  /ntb       /ntb                 /ntb        /ntb
  /bdm       /bdm                 /bdm        /bdm
  /swi
```
This means that the structure for a PU will be as follows:

```
/root
  /txpsys
  /inf
  /asr
  /arc
  /mac
  /swi  (directory for usual OM tools)
  /txpproj/proj_std
  /inf
  /asr
  /arc
  /mac
  /txpproz
  /inf
  /asr
  /arc
  /mac
```

On an SU the directory `/txpproz` has a special structure. Lza is not a directory but a link to the file system `/txparc.00.00`. On this file system all archived data are stored. There are also special solutions with 9 GB long-term archives.

For specific directories, environment variables are set:

```
For example:  /txpsys                 $OmSys
              /txpproj/proj_std       $OmProjData
              /txpproz                $OmProzData
              /txpsys/txpconf         $OmConfData
              /txptest                $OmDiagData
```
3.3 Distributed OM 650 system

One OM system can be composed of the various OM computers described above.

Our example of a TXP installation consists of 5 OTs, 2 redundant PUs, 1 redundant SU and 4 ASs. The object manager infrastructure (INF) needs to be configured in order for these components to be able to communicate. Also see chapter 3.5 /Engineering data/.
3.4 IKZ - signification

In a power plant ID codes such as KKS or AKS are used. These numbers are unambiguous. Since it is easier for a computer to handle numbers than letters and strings, TXP-internal coding is used. These codes are called IKZ (Interne Kennzeichen). An IKZ is an unambiguous translation of a KKS or AKS.

An IKZ has the following structure:

- Event type (warning, alarm, device fault, ....)
- Function complex, here we speak about FC-id
- General component, corresponding to the AS number; a negative number means an OM-internal signal.
- Part component, currently not used by TXP; always 0
- Instance number, consecutive number
- IO number, reference for input/ output from signal
- Element type, description of the element (analog transmitter, controller, .... ); the element types for all AS signals are stored in the file R_BstTypen in the directory $OmProjData/asr.
- Signal type, analog, binary or counting value

The following event types exist:

<table>
<thead>
<tr>
<th>Hex value</th>
<th>Type</th>
<th>Signification</th>
</tr>
</thead>
<tbody>
<tr>
<td>0001</td>
<td>S</td>
<td>Signal change</td>
</tr>
<tr>
<td>0002</td>
<td>A</td>
<td>Alarm</td>
</tr>
<tr>
<td>0004</td>
<td>W</td>
<td>Warning</td>
</tr>
<tr>
<td>0008</td>
<td>T</td>
<td>Tolerance cross</td>
</tr>
<tr>
<td>0010</td>
<td>B</td>
<td>Operating request</td>
</tr>
<tr>
<td>0020</td>
<td>L</td>
<td>Local fault</td>
</tr>
<tr>
<td>0040</td>
<td>F</td>
<td>Function fault</td>
</tr>
<tr>
<td>0080</td>
<td>M</td>
<td>Maintenance and service</td>
</tr>
<tr>
<td>0100</td>
<td>I</td>
<td>Indirect device fault</td>
</tr>
<tr>
<td>0200</td>
<td>P</td>
<td>Operating event</td>
</tr>
<tr>
<td>0400</td>
<td>SM</td>
<td>Centralised alarm</td>
</tr>
<tr>
<td>0800</td>
<td>Z</td>
<td>Free configurable event</td>
</tr>
<tr>
<td>1000</td>
<td>ST</td>
<td>IRD event</td>
</tr>
<tr>
<td>2000</td>
<td>-</td>
<td>Compressed value (set by the archive)</td>
</tr>
<tr>
<td>4000</td>
<td>Ü</td>
<td>I&amp;C fault</td>
</tr>
<tr>
<td>8000</td>
<td>G</td>
<td>Device fault</td>
</tr>
</tbody>
</table>
Signal types:

<table>
<thead>
<tr>
<th>Value</th>
<th>Signification</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>NIL-value</td>
</tr>
<tr>
<td>-1</td>
<td>Analog value</td>
</tr>
<tr>
<td>-2</td>
<td>Counting value</td>
</tr>
<tr>
<td>-3</td>
<td>Binary value</td>
</tr>
<tr>
<td>-4</td>
<td>Operated NIL-value</td>
</tr>
<tr>
<td>-5</td>
<td>Operated analog value</td>
</tr>
<tr>
<td>-6</td>
<td>Operated counting value</td>
</tr>
<tr>
<td>-7</td>
<td>Operated binary value</td>
</tr>
<tr>
<td>-8</td>
<td>Command</td>
</tr>
<tr>
<td>-9</td>
<td>Analog value with time stamp</td>
</tr>
<tr>
<td>-10</td>
<td>Counting value with time stamp</td>
</tr>
<tr>
<td>-11</td>
<td>Binary value with time stamp</td>
</tr>
<tr>
<td>-12</td>
<td>Analog value with time interval</td>
</tr>
<tr>
<td>-13</td>
<td>Counting value with time interval</td>
</tr>
<tr>
<td>-14</td>
<td>Binary value with time interval</td>
</tr>
</tbody>
</table>

A quality code exists for each event. This code defines the validity of the signal. The following quality codes are possible:

<table>
<thead>
<tr>
<th>Hex value</th>
<th>Signification</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>all ok.</td>
</tr>
<tr>
<td>0001</td>
<td>out of range</td>
</tr>
<tr>
<td>0002</td>
<td>default value</td>
</tr>
<tr>
<td>0004</td>
<td>uncalibrated</td>
</tr>
<tr>
<td>0008</td>
<td>manually entered</td>
</tr>
<tr>
<td>0010</td>
<td>test mode</td>
</tr>
<tr>
<td>0020</td>
<td>no data</td>
</tr>
<tr>
<td>0040</td>
<td>hardware error</td>
</tr>
<tr>
<td>0080</td>
<td>undefined error</td>
</tr>
<tr>
<td>0100</td>
<td>unknown address</td>
</tr>
</tbody>
</table>

If several faults occur simultaneously the quality codes are added in hexadecimal mode. 0060 means "hardware error" and "no data".

The line PICIKZ shows the complete IKZ of this picture. It is assigned to function complex 2, the general component is -2, meaning OM-internal, the instance number is 23006. This picture shows the subloop controller "0 0TAM00 EE003 XA92". This subloop controller is also assigned to FC 2 and is implemented in AS number 4.
3.5 Engineering data in the OM system

All engineering data are located in the path $OmProjData (/txpproj/proj_std).

3.5.1 INF

For the engineering of the distributed OM system, that is the infrastructure (INF), the following files have to be adapted:

- InfOmk.proj
- InfFb.proj
- InfDevInst.proj
- InfObm.inst

All these files are stored in the directory $OmProjData/inf

**InfOmk.proj**

In the file InfOmk.proj all OM components are listed and an internal unique component number is assigned to each component. The LTK-No. must always be negative, the LTK-Inst is the same number but positive, and the OMK-Inst is equal to LTK-Inst plus 1.

In release 3 the component numbers need to be a multiple of thousand. In release 4 we can also use multiples of hundred.

For our TXP installation mentioned in chapter 3.3, the file InfOmk.proj could be structured as follows:

```plaintext
# File InfOmk.proj

# Line building:
# Hostname in etc/hosts   LTK-Nr   RedPrio   OMK-Inst  LTK-Inst
# LTK-Nr = Negative number in multiples of hundred (release 4)
# RedPrio = Redundancy priority. Has no influence on OM but must be set to 1 and 2 for redundant components
# OMK-Inst = Instance number from OM Component
# LTK-Inst = Instance number from the superordinate I&C component

# Hostname in etc/hosts   LTK-Nr   RedPrio   OMK-Inst  LTK-Inst
ot1.khe.scn.de    -1000   1    1001      1000
ot2.khe.scn.de    -2000   1    2001      2000
ot3.khe.scn.de    -3000   1    3001      3000
ot4.khe.scn.de    -4000   1    4001      4000
ot5.khe.scn.de    -5000   1    5001      5000
pl1.khe.scn.de    -10000  1    10001     10000
pl2.khe.scn.de    -10000  2    10002     10000
p2a.khe.scn.de    -11000  1    11001     11000
p2b.khe.scn.de    -11000  2    11002     11000
su1.khe.scn.de    -20000  1    20001     20000
su2.khe.scn.de    -20000  2    20001     20000
```

**Caution:** this file must be identical on each OM computer.
InfFb.proj

In the file InfFb.proj every function complex from a plant is assigned to exactly one redundant PU couple. For example, lets say the following function complexes are in our AS code: 1,2,3,4,5,6. Moreover, there is also exactly one special function complex for the OM monitoring where no AS signals are assigned. For example we may call it 50. When distributing the FCs on the PUs, we must take care of the number of PU - AS couplings. The maximum number of ASs which can be coupled to one PU is 10.

The assignment to a component is done via the LTK-Instanz in the InfOmk.proj.

One possibility for the InfFb.proj in our TXP installation from chapter 3.3 is the following:

50 ARC=10000 MAC=10000 BDM=20000 LZA=20000 PRT=20000 NTB=20000
1 ASR=10000 ARC=10000 MAC=10000 LZA=20000 BDM=20000 PRT=20000 NTB=20000
2 ASR=10000 ARC=10000 MAC=10000 LZA=20000 BDM=20000 PRT=20000 NTB=20000
3 ASR=10000 ARC=10000 MAC=10000 LZA=20000 BDM=20000 PRT=20000 NTB=20000
4 ASR=11000 ARC=11000 MAC=10000 LZA=20000 BDM=20000 PRT=20000 NTB=20000
5 ASR=11000 ARC=11000 MAC=10000 LZA=20000 BDM=20000 PRT=20000 NTB=20000
6 ASR=11000 ARC=11000 MAC=10000 LZA=20000 BDM=20000 PRT=20000 NTB=20000

In this file the FC for OM monitoring must be written into the first line. Moreover, in this FC no signals are coming from the ASs. Therefore the component ASR is not started for this FC.

Caution: this file must be identical on each OM computer.
InfDevInst.proj

Example:
One printer is installed on the parallel port of OT1 and another printer is connected to the serial port
of OT3. One MOD drive is installed on each of the two SUs.
The file InfDevInst.proj is needed for device monitoring.
File structure: one line for each device
1. Column: OMK-Instance number from the corresponding OM computer
2. Column: The instance number of the device (this number is freely selectable,
   but it needs to be unique within the OM FC)
3. Column: Reference for the device (20550 = printer, 20500 = MOD drive)
4. Column: Device name (like the name present in /dev)

As a result our example would be as follows:

1001 800 20550 par01
2001 801 20550 ser01
20001 802 20500 MOD00
20002 803 20500 MOD00

Caution: this file must be identical on each OM computer.
InfObm.inst

In the file InfObm.inst you can set up the OM computer type.

Some examples:

OT:

```
#ObjMgr DirName
MMI mmi
```

PU:

```
#ObjMgr DirName
ASR asr
MAC mac
ARC arc
```

PU:

```
#ObjMgr DirName
LZA lza
BDM bdm
NTB ntb
PRT prt
```

⚠️ Caution: this file is specific to each OM computer
3.5.2 MMI

The following relevant engineering files are available for the object manager MMI:

- Ot.pwd
- Ot.hrn
- Ot.amu
- Ot.frappl (only release 4)

In addition the directories cnt, bin and pic are filled in with picture information from the ES.

**Ot.pwd**

The file `Ot.pwd` has to be created in the directory `$/HOME/om/mmi` on the ES. It is copied to the corresponding OT via MMI transfer.

The access rights of every user and for every function complex are described in this file.

```
# Login name: Password: Acknowledge group: UserId:
# FB: Functional_complex_id access_rights
# OM-OP: WS4JRSTANUK.6:1:1

FB: 13
FB: 23
FB: 33
FB: 43
FB: 53
FB: 63

# BEOB: WS4JRSTANUK.6:1:2

FB: 11
FB: 21
FB: 31
FB: 41
FB: 51
FB: 61

# OM-LT: GNvVH/cmZ1810:8:3

FB: 13
FB: 23
FB: 33
FB: 43
FB: 53
FB: 63
```
Signification:

Login name: This name appears after successful login at the operator terminal

Password: Coded password string. To get this code the tool `getPasswd` must be used. For example: `getPasswd OMLT`. The authorised symbols for passwords are only capital letters. Figures and special characters are not allowed.

Acknowledge group: 1-7 are reserved for the different operators

E.g.
1 = boiler
2 = turbine
8 is reserved for the I&C engineer.

If the groups 2-7 are used every function complex in the ASD is set to automatic acknowledgement at the first login. If a group is not used any longer the automatic acknowledgement must be set up again before the changes are made in the file `Ot.pwd`.

UserId: The user-id is the reference of the corresponding operator. This reference is used for example in operating logs in order to assign one operation to one operator.

Function complex: At this point the FC identifiers for which the respective operator has the corresponding rights, must be inserted.

Access rights: The access rights are set up for each FC independently.

Signification:
0 ⇒ No access rights for pictures of this FC
1 ⇒ No operating rights for pictures of this FC
2 ⇒ Normal operating access for pictures of this FC
3 ⇒ Complete operating rights for pictures of this FC
(also stepping mode for sequences, lock / unlock manual operating, couple / uncouple automatic commands)

Od.hrn

Data for the horn control on the OTs (on each OT it is possible to plug 2 horns).

File structure:
```
# HRN: x
# FB: y zzzz
```

Explanation:
x : is the horn number. In one OT we can plug up to 2 horns.
y : is the function complex identifier
zzzz: is the internal hexadecimal code of event types. If several types of events are needed the respective hex codes must be added.
Example:
# HRN: 1 # Horn 1
# FB: 1 0002 # Horn 1 is active on alarms from FC1
FB: 2 0006 # Horn 1 is active on alarms and warnings from FC2
FB: 3 000D # Horn 1 is active on alarms, warnings and tolerances from FC3
# HRN: 2 # Horn 2
FB: 4 0002 # Horn 2 is active on alarms from FC1
FB: 5 0006 # Horn 2 is active on alarms and warnings from FC2
FB: 6 000D # Horn 2 is active on alarms, warnings and tolerances from FC3

This file should be created in the ES because it is transferred at each MMI transfer. It can be empty when no horn is connected.

The hexadecimal codes for the event types are:

/* Event type (Bitmaske) */
#define OM_ETYP_S 0x0001 /* Signal change */
#define OM_ETYP_A 0x0002 /* Alarm */
#define OM_ETYP_W 0x0004 /* Warning */
#define OM_ETYP_T 0x0008 /* Tolerance */
#define OM_ETYP_B 0x0010 /* Operating request */
#define OM_ETYP_L 0x0020 /* Local fault */
#define OM_ETYP_F 0x0040 /* Function fault */
#define OM_ETYP_WS 0x0080 /* Maintenance & Service */
#define OM_ETYP_I 0x1000 /* Ind. device fault */
#define OM_ETYP_P 0x2000 /* Operating event */
#define OM_ETYP_SM 0x4000 /* Group event */
#define OM_ETYP_Z 0x8000 /* Status change event */
#define OM_ETYP_STAD 0x1000 /* IRD event */
#define OM_ETYP_V 0x2000 /* Compressed value */
#define OM_ETYP_UE 0x4000 /* I&C fault */
#define OM_ETYP_G 0x8000 /* Device fault */

Ot.amu

The file Ot.amu is used for automatic hiding of events in the ASD. Event types can be hidden for a specified time interval.

Example
#
#
#
MU: 11 115 0008 # In 11 sec 115 events => hide tolerances
MU: 11 135 0004 # In 11 sec 135 events => hide warnings
MU: 11 185 0002 # In 11 sec 185 events => hide alarms

The hiding of event types is signalled by the grey colour of the flags in the header of the ASD.

⚠️ Hiding events does not mean automatic acknowledgement.

This file can be created or changed on every OT.
Ot.frappl

With this file it is possible to lock or unlock functionalities on the local OM-OT.

Example:

```
#######
#     #   #####          ######  #####     ##    #####   #####   #
#     #     #            #       #    #   #  #   #    #  #    #  #
#     #     #            #####   #    #  #    #  #    #  #    #  #
#     #     #     ###    #       #####   ######  #####   #####   #
#     #     #     ###    #       #   #   #    #  #       #       #
#######     #     ###    #       #    #  #    #  #       #       ######
#
# @(#) - FRAPPL Konfigurationsdatei
# History:
# 08.03.96,13:59 <wlu> Stapellauf.
# 02.04.96 <kel> video geaendert
# 29.11.96 <fl> Leda neu
# 20.01.97 <fl> online neu
#(#$KORS 14.05.97 <bs> mit OmPicLock-Vorlage
# --------------------------------------------------------------------------
# KME-Archiv     @(#)$Source: /home/kme/mmi/confd/mmi/Ot.frappl,v $>
# Ausgabestand/   @(#)$Revision: 5.9 $>
# Version        @(#)$Date: 1997/05/14 09:55:26 $>
# Letzte Aenderung @(#)$Author: weinbrec $>
# von             @(#)$Source: /home/kme/mmi/confd/mmi/Ot.frappl,v $>
#Ausgabestand/   @(#)$Revision: 5.9 $>
# --------------------------------------------------------------------------
# ich file sollte da liegen, wo &APPL_CONF_FILE im MMI_CONFILE hinzieht
FRPROZ clock.bm xclock -bg "#880088" -hd "yellow" -fg "green" -update 1 &
APPLSYS prot.bm prot
&APPLSYS protN.bm protN
&APPLSYS lza.bm lza
&APPLSYS video.bm video
APPLSYS Leda protL
APPLSYS EP protE
FRPROZ any.bm xshowcmap &
FRPROZ DIGEST $OmProjData/mmi_start_digest &
FRPROZ SystemMenuName  ${OmSys}/OmPicLock.sh -d ${OmProjData}/Pic.lock &
# --------------------------------------------------------------------------
# Ende Konfigurationsdatei: Ot.frappl
#--------------------------------------------------------------------------
```

By changing the & character before line &APPLSYS lza.bm lza into a blank, the LZA interface on the respective OT will be deactivated.

This file can be changed on every OT separately.
### 3.5.3 ASR

For the ASR module only one engineering file exists: the `Asr.proj`. This file is generated on the ES by executing the script `AsrGen.sh`. It is stored in the path `$HOME/om/asr`. Since this file is generated for all ASs it is necessary to make a manual adaptation for every PU.

Example:

The PU1 is responsible for the function complexes 1, 2 and 3, PU2 for 4, 5 and 6. In the ASs, the following function complexes are running:

<table>
<thead>
<tr>
<th>AS</th>
<th>Function complexes</th>
</tr>
</thead>
<tbody>
<tr>
<td>AS 1</td>
<td>1,2</td>
</tr>
<tr>
<td>AS 2</td>
<td>2,3,4</td>
</tr>
<tr>
<td>AS 3</td>
<td>1,4,5,6</td>
</tr>
<tr>
<td>AS 4</td>
<td>6</td>
</tr>
</tbody>
</table>

This yields the following matrix:

<table>
<thead>
<tr>
<th></th>
<th>AS 1</th>
<th>AS 2</th>
<th>AS 3</th>
<th>AS 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>PU1</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>PU2</td>
<td>x</td>
<td>x</td>
<td></td>
<td>x</td>
</tr>
</tbody>
</table>

i.e. PU1 must be coupled to AS 1, 2 and 3. PU2 to AS 2, 3, and 4

From this configuration the script `AsrGen.sh` has generated the following file:

```bash
# ASR - mode is normal operation
IBS 0

# Information for AS 1
IKZ_GK 1
IKZ_FB 2
IKZ_IN 3
AS_ueberwacht 0
AS_FB 1
AS_FB 2
#LT_FB 1
#LT_FB 2
end

# Information for AS 2
IKZ_GK 2
IKZ_FB 4
IKZ_IN 1
AS_ueberwacht 0
AS_FB 2
AS_FB 3
AS_FB 4
#LT_FB 2
#LT_FB 3
#LT_FB 4
end

# Information for AS 3
IKZ_GK 3
IKZ_FB 6
IKZ_IN 3
AS_ueberwacht 0
AS_FB 1
AS_FB 4
```
Explanations for the different lines:

Every line beginning with # are comments

- **IBS 0**: If a PU loses contact to all configured ASs, the system expects a PU internal failure or a LAN failure. Therefore this PU is shut down in order to let the partner PU (redundant PU) take over process operation with all AS couplings. If this mode of operation is not needed the parameter IBS should be set to 1. Recommendation: IBS 0.

- **IKZ_GK 1**: General component number = AS number

- **IKZ_FB 2**: Function complex of the AP

- **IKZ_IN 3**: Instance number of the AP

- **AS_ueberwacht 0**: If a general failure of an AS occurs, this failure must be recognised by the OM system and an error message should be sent to the ASD. If several PUs are coupled to the same AS it doesn't make sense that every PU sends a message to the ASD. If the parameter AS_ueberwacht is set to 1 the PU sends a message. If AS_ueberwacht is set to 0 the PU does not send a message. Recommendation: use AS_ueberwacht 1 on the PU processing the function complex of the AP and AS_ueberwacht 0 on every other PU.

- **AS_FB 1**: Function complex id. For this FC the PU should get data from the AS and process them.

- **LT_FB 1**: Function complex id. For this FC the PU should get data from the AS but doesn't have to process them.

- **end**: End mark for the definition of one AS.

It is important that every entry with LT_FB is written after all AS_FB entries from the corresponding AS.

Our goal now is to adapt this Asr.proj for every PU in order to get a valid file for every PU computer. Asr.proj.pu1:

```plaintext
# ASR - mode is normal operation
IBS 0

# Information for AS 1
IKZ_GK 1
IKZ_FB 2
```
IKZ_IN 3
AS_uueberwacht 1
AS_FB 1
AS_FB 2
#LT_FB 1
#LT_FB 2
end

# Information for AS 2
IKZ_GK 2
IKZ_FB 4
IKZ_IN 1
AS_uueberwacht 0
AS_FB 2
AS_FB 3
#AS_FB 4
#LT_FB 2
#LT_FB 3
LT_FB 4
end

# Information for AS 3
IKZ_GK 3
IKZ_FB 6
IKZ_IN 3
AS_uueberwacht 0
AS_FB 1
#AS_FB 4
#AS_FB 5
#AS_FB 6
#LT_FB 1
LT_FB 4
LT_FB 5
LT_FB 6
end

Asr.proj.pu2:

# ASR - mode is normal operation
IBS 0

# Information for AS 2
IKZ_GK 2
IKZ_FB 4
IKZ_IN 1
AS_uueberwacht 1
#AS_FB 2
#AS_FB 3
AS_FB 4
LT_FB 2
LT_FB 3
#LT_FB 4
end

# Information for AS 3
IKZ_GK 3
IKZ_FB 6
IKZ_IN 3
AS_uueberwacht 1
#AS_FB 1
AS_FB 4
AS_FB 5
AS_FB 6
LT_FB 1
#LT_FB 4
#LT_FB 5
#LT_FB 6
end

# Information for AS 4
IKZ_GK 4
IKZ_FB 6
IKZ_IN 1
AS_ueberwacht 1
AS_FB 6
#LT_FB 6
end
3.5.4 MAC

From OM release 03.01.3X on it is possible to have a new behaviour of the signalling processing by message buffer overflow.

The signalling function block is able to store 2000 messages in the ASD and 600 messages in the IC-ASD. Up to now it could happen that "back-to-normal" messages were not processed due to a buffer overflow. Consequently the corresponding "arriving" messages after acknowledgement were stored on the old side of the ASD despite the fault was no longer present.

The new signalling function block has the following properties:

When the message buffer for one FC is X% full automatic acknowledgement is used until a rest of Y% messages is left.

The X and Y factors can be adapted relatively to the plant in the file Meld.proj.

The file is located in the path $OmProjData/mac and is set up by default as follows:

By reaching 98% (=X) all messages are acknowledged (Y=0%). Y=0% should be changed only in exceptional cases!! It has the same function as an overall acknowledgement in the system menu and creates the least CPU load.

| WARN_ATW:     | warning threshold MFA  |
| WARN_LTW:     | warning threshold LT-MFA |
| QUITT_ATW:    | acknowledge threshold MFA |
| QUITT_LTW:    | acknowledge threshold LT-MFA |
| SOLL_ATW:     | acknowledge rest MFA |
| SOLL_LTW:     | acknowledge rest LT-MFA |

Example:

```
#WARN_ATW WARN_LTW QUITT_ATW QUITT_LTW SOLL_ATW SOLL_LTW
80 90 95 98 0 0
```

I.e. a message appears in the ASD when the message buffer is 80% full. At 95%, all messages will be acknowledged automatically (acknowledgement to 0%).

In the IC-ASD a message appears when the message buffer is 90% full. At 98%, all messages will be acknowledged automatically (acknowledgement to 0%).
### 3.5.5 ARC

In the file `Arc.proj` the only thing to do is to insert the function complex ids which are processed by the corresponding PU. In our example we will have the following files:

**Arc.proj for pu1**

```plaintext
#       Arc.proj
#       Projektierungsdaten eines Kurzzeit-Archivs
#       Anlage          Transmittal
#       PU                      xx
#       Proj-Stand       xx
#       Datum           18.04.95
#       Aufzählung aller Funktionsbereiche
#       zu denen Ereignisse zu archivieren sind
1 2 3 50
#       maximale Anzahl projektierter Signale
AnzAikz  20000
#       maximale Anzahl typspezifischer Signale
AnzLikz  50000
#       Groesse des Ringspeichers (in Ereignissen)
AnzEr   500000
#       zu archivierende Ereignistypen (hex)
EreigTypen 0xffff
```

**Arc.proj for pu2**

```plaintext
#       Arc.proj
#       Projektierungsdaten eines Kurzzeit-Archivs
#       Anlage          Transmittal
#       PU                      xx
#       Proj-Stand       xx
#       Datum           18.04.95
#       Aufzählung aller Funktionsbereiche
#       zu denen Ereignisse zu archivieren sind
4 5 6
#       maximale Anzahl projektierter Signale
AnzAikz  20000
#       maximale Anzahl typspezifischer Signale
AnzLikz  50000
#       Groesse des Ringspeichers (in Ereignissen)
AnzEr   500000
#       zu archivierende Ereignistypen (hex)
EreigTypen 0xffff
```

Another important parameter is `AnzEr`. This parameter determines the size of the FIFO stack from the short-term archive. Here it is set to 500,000 events and needs approximately 32 MB RAM mem-
ory. In order to save RAM memory, this parameter can be set to a smaller value. The parameters AnzAikz, AnzLikz and EreigTypen can only be changed after asking the hotline. They determine the internal buffer size of ARC.
LZA

In the engineering file `Lza.proj` only the function complex ids need to be listed

```plaintext
# Lza.proj
# globale Projektierungsdaten des TXP-OM-Langzeit-Archivs
# Anlage       KME-Stand
# PU           xx
# Proj-Stand   xx
# Datum        26.07.94

Aufzaehlung aller Funktionsbereiche
1 2 3 4 5 6 50
```

# AnzAikz         Maximalzahl projektierter IKZ
# AnzLikz         Maximalzahl typspezifischer IKZ
# PDA hat nur Platz fuer (AnzAikz+AnzLikz) IKZs
# AnzEr           Groesse eines Wechselarchives

AnzEr sollte signifikant grosser als (AnzAikz+AnzLikz) sein (Faktor 5-20)
maximaler Platzbedarf eines Teilarchivs :
100 + (AnzAikz+AnzLikz)*24 + AnzEr*44 Bytes
bei der Standardprojektierung (AnzAikz=30000, AnzLikz=70000, AnzEr=1000000)
ergibt sich damit ein Platzbedarf von ca. 46 MB je Teilarchiv
Plattenspeicherbedarf des Prozessdatenarchivs: (AnzTa aus Lza.conf)
(AnzTa + 1) * Teilarchivgroesse + Buchfuehrung
       der Platzbedarf fuer die Buchfuehrung wachst mit der Laufzeit
       und der Auslagerungsrate (Prozessdaten und Protokolle/Files)
Er duerfte unter realistischen Bedingungen kaum ueber 50 MB liegen

BITTE beachten !
Eine Vergroesserung der untenstehenden Parameter kann dazu fuhren, dass
der Speicherplatz des PDA-Filesystems nicht mehr ausreicht.
In diesem Fall muss evtl. der Parameter AnzTa aus Lza.conf verringert
werden. Danach koennen die ueberzaehligen Teilarchive geloescht werden.
Dies ist evtl. mit Ereignisverlust verbunden !
Erhohung sich (AnzAikz+AnzLikz), muss gegebenenfalls der Parameter
ShmAnzEr in LzaSp.conf erhoeht werden.
Damit erhohet sich der Shared-Memory-Bedarf des PDA !

# AnzAikz         30000         # Anzahl anlagenspez. Ereignisse
# AnzLikz         70000         # Anzahl typspez. Ereignisse
# AnzEr           1000000       # Groesse eines Wechselarchives
# EreigTypen      0xffff         # Zu archivierende Ereignistypen

Ende der Projektierungsdatei
3.5.6 PRT

In the file PrtProj the allocation between the available printers, the OM-PCs and the symbolic printer name is written.

Example:

```
**
**      KME-Archiv                      @(#)$Source: /RCSTREE/rcstree/prt/proj/PrtProj,v $>
**
**      Ausgabestand/                @(#)$Revision: 1.8 $>
**
**      Version
**
**      Letzte Aenderung              @(#)$Date: 1994/12/09 13:58:45 $>
**      von                        @(#)$Author: fluck $>
**
**
**      KME                           @(#)$Header: /RCSTREE/rcstree/prt/proj/PrtProj,v 1.8
1994/12/09 13:58:45 fluck stab $>
**
**
**      KMP                                    @(#)$version_id:$>
**
**      IDENT                                     @(#)$TXP/OM/SEG/PRT/PrtProj>
**
**
**
**      Projektierung der Drucker im System
**
**---------------------------------------------
**
**      symbolischerDruckername:Rechnername:realerDruckername[:par1[:par2[:...]]]
**
**
**      Struktur der Datensaezte:
**
**
**      Hinweise:  1. Tabs in den Datensaezten sind verboten, Blanks nur in par1 erlaubt.
**      2. In den Datensaezten sind keine Kommentare zulaessig
**      3. Zeilen, die mit "**","#" oder " " beginnen, werden als Kommentarzeilen
**      ueberlesen
**      4. Zeilen, die kein zulaessiges Format besitzen, werden ueberlesen
**
**      Die symbolischen Druckernamen "DruckerX" (X=1,2,...,9,10,11,...) sind fuer die
**      Druckfunktion von Prot reserviert:
**      par1 : Name des Druckers, der in den Auswahlmasken erscheint (maximal 10 Zeichen).
**      Hinweis: Die Namen aller Drucker muessen disjunkt sein.
**      Der symbolische Druckername "onlineprinter" ist fuer das Online-Protokoll reserviert.
**      Die symbolischen Druckernamen "Plotter_X" (X=1,2,...,9) sind fuer die Plot-Funktion
**      reserviert:
**      par1 : Name des Plotters, der in den Auswahlmasken erscheint (maximal 60 Zeichen)
**      par2 : Format (A3 oder A4).
**      Hinweis: Die Namen aller Plotter muessen disjunkt sein.
**
**      Drucker_1:ot1:par01:Name D1
Drucker_2:ot2:ser01:Name D2
Drucker_3:bt02:par01:Name D3
Drucker_4:ot5:ser01:Name D4
onlineprinter:ot2:par01
ntb_printer:ot1:par01
Plotter_1:ot3:plot01:Description text plotter1:A4
Plotter_2:ot3:plot01:Description text plotter2:A3

These inputs are enough if the printer is also configured with sysadmsh (SCO 3) or scoadmin SCO 5) (see the chapter on printer administration)
3.5.7 NTB

The engineering file Ntb.proj from the module NTB is a link to the file PrtProj. As a result, both files are identical.

3.5.8 BDM

For the BDM component no engineering files have to be adapted.
3.6 Diagnostic files

Each object manager writes so-called diagnostic files. In these files error and hint messages are stored as a report. All diagnostic files are located in the path /txptest. The infrastructure module writes directly into directory /txptest. Every other object manager writes into its respective subdirectory. For example the ASR writes into /txptest/asr.

The diagnostic files are realised via a cyclic buffer. First all diagnostic messages are written into file DiagMld.0. If this file reaches a size greater than 1 MB, then the file DiagMld.1 is created. If this file also becomes greater than 1 MB, then DiagMld.0 is deleted and created again. Thanks to this mechanism, the hard disk will never be full and the latest diagnostic data are always available.

Let's now take a look at a part of a diagnostic file from the infrastructure:

```
02.12.97 15:30:27 OMroot 8630 ILTus.c 0.00 529 InfLtukUSInit
  -> 20 28 OMK Ltk -11000 Prio 1 gestartet
02.12.97 15:30:37 OMroot 8630 ILTuc.c 0.00 498 InfLtukUCInit
  -> 20 64 Verbindungsaufbau zu Server Ltk -11000 Prio 2 erg=0
02.12.97 15:30:38 OMroot 8630 ILTus.c 0.00 2705 InfLtukUSLtkInit
  -> 20 29 OMK Ltk -11000 Prio 1 synchronisiert
11.12.97 19:11:57 OMroot 715 ILTus.c 0.00 529 InfLtukUSInit
  -> 20 28 OMK Ltk -11000 Prio 1 gestartet
11.12.97 19:12:07 OMroot 715 ILTuc.c 0.00 498 InfLtukUCInit
  -> 20 64 Verbindungsaufbau zu Server Ltk -11000 Prio 2 erg=0
11.12.97 19:12:08 OMroot 715 ILTus.c 0.00 2705 InfLtukUSLtkInit
  -> 20 29 OMK Ltk -11000 Prio 1 synchronisiert
11.12.97 19:13:03 OMroot 715 IOM.c 1.21 1253 InfOMgrObjmTerm
  -> 20 9 Objektmanager LtkNr -11050 ist abgebrochen
11.12.97 19:13:08 OMroot 715 IAB.c 1.28 318 InfAblAbort
  -> 20 42 OMK beendet Status=0
```

All diagnostic files are always built with a two line set. The date and time of the message are written in the first line, as well as the program name, process id, module name, module version, line number and function name. In the second line we can find the error message number and the error message description.

We can see in this part that the computer was started on 2.12.97. Another start message appears on the 11.12.97. But between these two messages no message is displayed for shutting down the computer. This means that the computer was not shut down properly. Maybe only the power supply was switched off.

On 11.12.97 19:13 the object manager with number -11050 broke down. Using the component number from chapter 1 we can see that the MAC has shut down. Since the component 11000 is a redundant PU (see the example above), the complete OM software of the computer was shut down in order to enable a redundancy switchover.

For more detail it is possible from this point on to analyse the diagnostic file of the MAC to find the reason for its shut down. Unfortunately these diagnostic files have been written by developers and for other developers. Therefore they are often very difficult to understand.
3.7 Data transfer ES -> OM

The following data must be transferred from the Es to the OM:

- The picture data for MMI on every OT: Transfer - OM - MMI Picture
  Transfer - OM - MMI FUP
- For ASR on every PU: Transfer - OM - LAN (only release 4)
  Transfer - OM - ASR
- For BDM on every SU: Transfer - OM - BDM
- For PRT on every SU: Transfer - OM - Protocols

For transfer, the ES uses the user txpes in release 3, and the user txpproj in release 5 on the OM-PCs.

Data are copied with the following commands:

rcp <file> txpproj@<OM-component>:<goal directory>

Programs are called up as follows:

remsh <OM-component> -l txpproj <Job>

Both commands need free access from the ES to the OM computers. This means that the ES must be able to login on the OM computers without password check.

For example:
The ES is called hpa4004 and has the address 142.17.5.1. The engineering login is voerde. A MMI transfer on OT vo_ot1 with address 142.17.1.1 is needed.

In the file /etc/hosts on the ES following lines must be present:

# /etc/hosts
hpa4004 142.17.5.1
vo_ot1 142.17.1.1

The /etc/hosts of the OT must be principally the same.
The file .rhosts from user txpproj on the OT needs to be as follows:

# .rhosts
hpa4004 voerde

The file .rhosts needs the access rights 644.
3.8 Coupling AS 620 <-> OM 650

For the coupling between AS and OM, the LAN data and the Asr.proj file must be present on the PU.

The LAN data are constituted with the following items:

```
/usr/scp/H1tools/h1.ldb
/txpproj/proj_std/asr/RES.red_conf.dat
/txpproj/proj_std/asr/ROS.red_conf.dat
/txpproj/proj_std/asr/RSS.red_conf.dat
/txpproj/proj_std/asr/Asr.stf_conf.dat
```

The LAN data are generated on the ES. For every component of the topology diagram which contains a CP1413 a file with subsequent name convention is built in the directory `$HOME/listen/lan`:

```
cxxn1413.dat
```

xx is the component index, n is the subcomponent index.
Both indexes can be read in the module parameter mask of the CP1413:

In this example we can see that the ASCII database name for pu1a is c0821413.asc and for pu1b it is c0831413.asc. During LAN transfer, the corresponding file is copied to the PU into the path /usr/scp/H1tools. The file is then converted to a loadable database called h1.ldb via the tool cvh1tf.

Command: cvh1tf c0821413.asc

Now the database is loaded with the following commands:

scp_mon -r  # Reset CP1413
scp_mon -c  # Load operating system into CP 1413
scp_mon -l  # Load user database (h1.ldb)

The computer should be rebooted with the command init 6 for safety reasons.

During LAN transfer some configuration files for the coupling of ASR to CP1413 are also copied into the path /txpproj/proj_std/asr. The names of these files have the following form:

xx_res_conf.dat
xx_ros_conf.dat
xx_rss_conf.dat

where xx is the component index

These files are then moved onto the PU with the following names:

RES.red_conf.dat
ROS.red_conf.dat
RSS.red_conf.dat

In our example:

08_res_conf.dat  ➔ RES.red_conf.dat
08_ros_conf.dat  ➔ ROS.red_conf.dat
08_rss_conf.dat  ➔ RSS.red_conf.dat

Finally, in release 4 the file Asr.stf_conf is also transferred.

All steps described here are completely automatic in release 4. In release 3 these steps must be done manually.

Moreover, it is useful to check the parameter "database" in the file /usr/scp/startup/scp0_conf.dat. It must be set to /usr/scp/H1tools/h1.ldb.
 SIEMENS
UserAdmin Workshop - TELEPERM XP Release 4
Chapter 3

/*------------------------------------------------------------------------*
* Copyright (C) Siemens AG 1993 All Rights Reserved. Confidential. *
*------------------------------------------------------------------------*/

boardtyp = 1413  /* board type */
config_reg = 3E0  /* Configuration Register */
dpram_addr = 10F00000  /* DPRAM Address */
int_vector = 10  /* Interrupt Vector */
eternet_addr = 08.15.00.00.47.11  /* default Ethernet address */
DSP_comp = sca  /* Driver Software Package */
Database = /usr/scp/H1tools/h1.ldb  /* local Database filename */
num_channel = 2  /* number of channels */
channel_name = ADM,STF  /* external name of channel */
channel_type = SCP,SCP  /* type of channel */
umhostbuffer = 40,80  /* number of host buffers */
umboardbuffer = 40,80  /* number of board buffers */
umproc = 20,50  /* total number of processes */
Download : kern.lad  /* SCP kernel software */

/*------------------------------------------------------------------------*
* Copyright (C) Siemens AG 1993 All Rights Reserved. Confidential. *
*------------------------------------------------------------------------*/
3.9 Printer administration

Check the printer entry in the operating system

![Printer administration window 1](image1)

![Printer administration window 2](image2)
The image shows a printer configuration interface. The configuration includes two printers:

- **ser01**: IR241 an tty1a
- **tty1a**: cn.de

The interface provides options for managing the printers, such as add remote, set to default, examine, and delete.
The following printer models are adapted for TXP applications:

- **PrtIfDR241** for Matrix lineprinter DR241
- **PrtIfDR241_Jl_p1** for Matrix lineprinter DR241 on JetLan port 1
- **PrtIfDR241_Jl_p2** for Matrix lineprinter DR241 on JetLan port 2
- **PrtIfHpLj4** for HP LaserJet 4 (old version)
- **PrtIfHpLj4_Jl_p1** for HP LaserJet 4 (old version) on JetLan port 1
- **PrtIfHpLj4_Jl_p2** for HP LaserJet 4 (old version) on JetLan port 2
- **PrtIfPclL** for all PCL printers such as HP LaserJet
- **PrtIfPclL_Jl_p1** for all PCL printers on JetLan port 1
- **PrtIfPclL_Jl_p2** for all PCL printers on JetLan port 2
- **PrtIfPlt** plotter driver

The example above shows how to set up a new HP-Laserjet printer plugged into the parallel interface.
on an OT and with SCO 5.
3.10 Hardcopy

For the hardcopy function, two solutions are available.

- Software solution
- External hardcopy-PC with OS/2 or Windows 95

In both cases the hardcopy files are created locally on each OT and stored in the directory /txpproz/mmi/hc_spool.

Software solution

For the software solution hardcopy software is installed on one OM computer (normally an OT) in the path /usr/hcx. This software cyclically checks whether there are data present in the spool directory of each OT. If there are data present, then these data are copied to the directory /usr/hcx/spool and deleted on the source computer. Thereafter the files are output directly on the parallel interface, by passing the Unix spool system.

The mechanism described above is parameterised in file /usr/hcx/etc/hcsp1.cn.

```
*-----------------------------------------------------------------------*
*                                                                       *
*             Spoolingparameter DYNAVIS-X Hardcopy -Server               *
*                                                                       *
*                                                                       *
*-----------------------------------------------------------------------*

*   - Kommentarzeilen beginnen mit '*'                                   *
*   - Kommandos werden ueber Schlüsselwörter eingegeben.                *
*   POLL : {1-3600}                                 Default: 10          *
*                   Pollingrate in Sekunden.                              *
* LOCSPPOOL : <cat> <fil>                              Default: keine     *
*                   Katalogs- und Dateinamenskonventionen lokal.           *
* LOCNAME  : <fil>                                    Default: locname.hc  *
*                   Name der lokalen Temporaerdatei fuer die              *
*                   extern akquirierten CGM-Files. Dieser Name              *
*                   muß der lokalen Dateinamensmaske entsprechen          *
*                   (da die Datei sonst nicht gedruckt wird).             *
*                   Falls weitere, lokale CGM-Datei Quellen               *
*                   existieren, darf dieser Name nicht von diesen          *
*                   Quellen erzeugt werden.                               *
* EXTSPOOL : <internet> <user> <password> <cat> <fil> Default: keine    *
*                   Internetadresse, User- und Passwort, Katalog-          *
*                   und Dateinamenskonventionen extern                     *
*                                                                       *
*                                                                       *
*                                                                       *
*                                                                       *
*                                                                       *
*                                                                       *
*                                                                       *
*locspool                 /usr/hcx/spool  *.hc
*extspool internet      user  pass     catalog                    a*.hc
*locname  locname.hc
*poll     10

On this TXP installation (example above) a cyclic check is carried out every 10 seconds on the computers with addresses 142.14.30.25 and 142.14.30.207 in order to check the presence of hardcopy files (*.hc).
```
Hardcopy PC

For an external hardcopy PC the mechanism is principally the same. Here we can find the parameterisation file in the path `c:\hcx\etc\hcspl.cn`.

```plaintext
*-----------------------------------------------------------------------*
*             Spoolingparameter DYNAVIS-X Hardcopy-Server               *
*              geändert am 15.03.1995 von J. Hanken ANL (I+T)           *
*-----------------------------------------------------------------------*

- Kommentarzeilen beginnen mit '*'
- Kommandos werden über Schlüsselwörter eingegeben.

** POLL : (1-3600) Default: 10 **
Pollingrate in Sekunden.

** LOCSPOOL : <cat> <fil> Default: keine **
Katalogs- und Dateinamenskonventionen lokal.

** LOCNAME : <fil> Default: locname.hc **
Name der lokalen Temporärdatei für die
extern akquirierten CGM-Files. Dieser Name
müß der lokalen Dateinamensmaske entsprechen
(da die Datei sonst nicht gedruckt wird).
Falls weitere, lokale CGM-Datei Quellen
existieren, darf dieser Name nicht von diesen
Quellen erzeugt werden.

** EXTSPOOL : <internet> <user> <password> <cat> <fil> Default: keine **
Internetadresse, User- und Passwort, Katalog-
und Dateinamenskonventionen extern

Internet User Password Katalog Filenames
locspool catalog * .hc
extspool internet user pass catalog a* .hc

poll 10
locname locname.hc

locspool c:\hcx\spool * .hc
extspool 142.17.10.2 txpom /txpom /txpproz/mmi/hc_spool a* .hc
extspool 142.17.10.1 txpom /txpom /txpproz/mmi/hc_spool a* .hc```

If the printer has been out of service for a longer period of time a lot of print requests (hardcopy files) are stored on the hard disk. To avoid the disk from getting full, these requests can be deleted simply by removing the files from the spool directory: `rm * .hc`. 
3.11 OM650 components - commissioning tools

3.11.1 Arc.Read - reading values from the archive

SYNTAX
Arc.Read [-d Date] [-u time] [-i time duration] [-a] [-l] FID [SID]

DESCRIPTION
Archived values are read for the signals defined by KKS (tag no.) and SID.

The input of an SID is not an obligation. If there is a space character in the KKS or SID code, then the
KKS and/or SID name needs to preceded and ended by " " or ". It is possible to use wildcards (" or ?). At the end of a KKS or SID input no wildcard is needed because Arc.Read sets it automatically.

If no options are specified, the default values are:

- 5 minutes time interval
- without initial values
- from the short-term archive

OPTIONS

- **-d Date**  The beginning date of the desired time interval is indicated as follows:
  *Date*: Day[.Month],
  e.g. 17.11, i.e. the 17th of November
  or 14 , i.e. the 14th of the current month
  This means that it is possible to output values for the past 365 days.
  The default value of the date is the actual day

- **-u time**  The beginning time for the desired time interval is indicated as follows:
  *Time*: hour[.minute]
  The default value is the actual time.

- **-i Duration**  The duration of the time interval is indicated as follows:
  *Duration*: [Hour.]Minute
  The default value is 5 minutes.

- **-a**  The initial values are also indicated.

- **-l**  The events are read directly from the long-term archive.
  If the option -l is not set, the Arc.read access is first to the short-term arch-
  ive and then automatically to the long-term archive if the desired values
  are already overwritten in the short-term archive.

EXAMPLE
Arc.Read ,0,0GAA"
The archived values for the signals with KKS="0 0GAA" are read:
- for the last 5 minutes
- without initial values
- from the short-term archive

Arc.Read -u 9.30 -i 30 -l ,0 0GAA" XH01
The archived values for the signals with KKS="0 0GAA" and SID="XH01" are read:
- from 9:30 on the actual day
- for a time interval of 30 minutes (i.e. 10 o'clock)
• without initial values
• from the long-term archive

REMARKS
All signals with the same KKS number need to be processed by the same archive manager.

The archive access is done via the command OciSignalRecallPast and the result is output thanks to the TOC (Test Object Client). After the execution, the used input file is stored in /temp/TOC.
3.11.2 Fb.Statist - List of the most frequent events of a function complex

SYNTAX


DESCRIPTION

This tool allows you to get a list of the most frequent events with the number of occurrences within a

time interval. As a result it is possible to locate signals sending to many TTDs due to an engineering
error, for example.

The function complex FC has to be specified as FC name and not FC id. This name consists of two

letters or numbers, which appear in the ASD.

If no options are specified the 20 most frequent signals for the last 55 minutes are read from the

short-term archive.

OPTIONS

-d Date
The beginning date of the desired time interval is indicated as follows:

Date: Day[.Month],
e.g. 17.11, i.e. the 17th of November
or 14 , i.e. the 14th of the current month
This means that it is possible to output values for the past 365 days.
The default value of the date is the actual day

-u Time
The beginning time for the desired time interval is indicated as follows:

Time: hour[.minute]
The default value is the actual time.

-i Duration
The duration of the time interval is indicated as follows:

Duration: [Hour.]Minute
The default value is 5 minutes.

-n number
The number of the most frequent events to read is indicated.
The default value is 20.

-a
All signals within the given time interval are listed.

-l
The events are read directly from the long-term archive.
If the option -l is not set, the Arc.read access is first to the short-term ar-

chive and then automatically to the long-term archive if the desired va-

lues are already overwritten in the short-term archive.

EXAMPLE

Fb.Statist NA
The 20 most frequent signals from function complex NA are read for the last 5 minutes and
from the short-term archive.

Fb.Statist -u 9.30 -i 1 -n 40 -l 03 > myfile
The 40 most frequent signals of FC 03 are read between 9:30 and 9:31 on the actual day and
directly from the long term archive. The result is output in the file called "myfile". This file can
be read later using the Unix system program "more", for example.

REMARKS

The archive access is done via the command OciSignalRecallPast and the result is output thanks to
the TOC (Test Object Client). After the execution, the used input file is stored in /temp/TOC.
3.11.3 Arc.Moni - number of events in the short-term archive

SYNTAX

Arc.Moni [CycleTime]

DESCRIPTION

The following values are output for the short-term archive of an OM component (PU or CU). The values printed in bold print are refreshed cyclically.

- **ME-Len:** Length of an event in bytes.
- **RSP-Len:** Number of events storable in the FIFO stack.
- **Zyklus:** Time delay since the last cyclic refresh. If the computer has a high load this time can be greater than the cycle time specified.
- **RSP-Pos:** Position of the last stored event in the FIFO stack.
- **Count:** Number of stored events since the last start-up of the OM software.
- **SZ-Time-Anf:** Time tag of the oldest event present in the FIFO stack. In the second line this tag is converted into a Unix format.
- **SZ-Time-End:** Time tag of the newest event present in the FIFO stack. In the second line this tag is converted into a Unix format.
- **Me/Sec:** Average number of events in the last cycle. In the second line we can see the maximum number of events in one cycle since Arc.Moni has been started.

Arc.Moni delivers values only if the short-term archive is running on the OM component where it was started.

OPTIONS

- **Cycle:** The refreshing cycle can be specified in seconds. The default value is 3 seconds.

EXAMPLE

- Arc.Moni
- Arc.Moni 5

REMARKS

To end the program use the "Del" or "Entf" key
After shutting down the OM software Arc.Moni has to be stopped and, if needed, started again after starting up the OM software
3.11.4 Lza.Moni - number of events in the long-term archive

SYNTAX
Lza.Moni [Cycle time]

DESCRIPTION
The following values are output for the long-term archive of an OM component (SU or CU). The values printed in bold type are refreshed cyclically:

- **Anzahl ME:** Number of events copied from the short-term archive into the long-term archive since the last start-up of the OM software. The events read from the LZA of the redundant machine during the checkpoint (just after start-up) are not counted here.
- **Zyklus:** Time delay since the last cyclic refresh. If the computer has a high load this time can be greater than the cycle time specified.
- **Durchsatz:** Number of events that occurred during the last cycle. If during this period of time no events have arrived the last value is displayed.
- **Max. Durchsatz:** Highest number of events that occurred during one cycle since the start of Lza.Moni.
- **Beginn der Speicherung:** Point in time when normal operation started after the last start of the long-term archive; i.e. the time when checkpoint processing was completed.
- **Letzte Daten erhalten:** Point in time when the last events were transferred from the short-term archive to the long-term archive.
- **Zeitstempel des jüngsten Ereignisses:** The time tag of the youngest event normally is one minute older than the time specified in "Letzte Daten erhalten" (last data received at), since the long-term archive reads the events from the short-term archive with a delay.

Lza.Moni only provides values if the long-term archive is running on the OM component.

OPTIONS
- **Zyklus:** The update cycle can be specified in steps of seconds if the default value of 3 seconds needs to be changed. Since the archived values in the short-term archive are read by the long-term archive in a 20-seconds cycle (as a standard) the display does not change during the procedure.

EXAMPLES
- Arc.Moni  Display with a 3-seconds update cycle
- Arc.Moni 5 Display with a 5-seconds update cycle

REMARKS
To deselect the cyclically updated display press the Del key (on English keyboards) or the Entf key (on German keyboards). After shutdown of the OM system, Arc.Moni must be terminated and started again after a restart, if required.
3.11.5 Fb.Read - reading function complex designations

SYNTAX
Fb.Read [FC]

DESCRIPTION
If no function complex no. FC is specified the following data is listed for all function complexes known to be present in the OM:
  • The function complex no. FC
  • The function complex identification FC ID
  • The function complex name FC name.
If a function complex number FC is specified this data is output for the corresponding function complex.

EXAMPLES
Fb.Read
  FC,FC ID and FC name are listed for all function complexes.
Fb.Read NA
  FC,FC ID and FC name are output for the function complex NA.

REMARKS
The ES designations are used:

The "function complex number" FC is a concise designation consisting of two characters, which is allocated by the configuring engineer on the ES. It appears in the individual alarms of the alarm sequence display, for example. The tool PI with the option -L indicates the FCs for which the individual servers are responsible.

The "function complex identification" FC ID is the number allocated by the ES. It is used internally in the OM software as for example in the IKZ. The tool PL with the option -l indicates the FC IDs for which the individual servers are responsible.

The function complex name consists of up to 16 characters and is allocated by the configuring engineer.
3.11.6 Ikz.Read - reading IKZ data for a KKZ

SYNTAX

DESCRIPTION
Output of IKZ data for the signals or function block instances to which the KKZ (tag no.) specified by FID and SID is applicable.

If no options are specified the IKZ data is listed for all associated signals in the form of a table.

The SID can be omitted from the KKZ. If blanks are used in an FID or SID they must be preceded and ended by "..." or '...'. Within an FID and SID the wildcards * and ? can be used; the wildcard * is unnecessary at the end of an FID or SID since it is added by Ikz.Read anyway.

OPTIONS
-\( -i \)
Lists the IKZ data for the function block instances to which the FID specified applies.
-\( -p \)
Lists the IKZ data for the configured signals - i.e. the signals for which an SID is specified.
-\( -s \)
The IKZ data has a form that is needed for the TOC (Test-Object-Client). If an SID is specified, the IKZ for the corresponding function block instance is displayed or otherwise the IKZ for the signals.
-\( -a \)
Same as for option -s, except for signals only.

EXAMPLES
Ikz.Read "0 0GAA"
The IKZ data for all signals to which the FID ="0 0GAA" applies is displayed in the form of a table.

Ikz.Read "0 0GAA" XH04
The IKZ data for all signals to which the FID ="0 0GAA" and SID ="XH04" apply is displayed in the form of a table.

Ikz.Read -i "0 0GAA"
The IKZ data for the function block instance to which the FID ="0 0GAA" applies is displayed in the form of a table.

REMARKS
The description data of the BDM is accessed in the form of a database access using SQL, since no Oci instance, which provides the data for all signals and not only the configured ones, has been implemented.

If more IKZ data is provided than can be displayed on one screen the data should be displayed over several pages using the UNIX command more, e.g.

Ikz.Read "0 0GAA"|more
3.11.7 PL - indicating the status of the OM component and the OM manager

SYNTAX

PL [-l] [-L] [-1] : if only one ring is present
PL [-l] [-L] [-1] [-a] [Om component] : if several rings are present

DESCRIPTION

The current statuses present in the infrastructure are displayed for the OM components and the object managers. The display refers to the local OM component, i.e. the OM component on which PL has been selected.

Unless option -1 has been specified the display is updated every 6 seconds. Use the key Del (English keyboard) or Entf (German keyboard) to deselect cyclic output.

In multi-unit plants the OM components may be distributed over several infrastructural rings. An OM component knows the OM components of another ring if it can establish connections to these components according to the configuration of the infrastructure.

OPTIONS

- **-l**  
  
  ("l" for "long") the IKs of those function complexes are listed for each object manager in addition for which the IK works as a server (according to the configuration of the infrastructure). In addition, information about the OM software version is output.

- **-L**  
  
  Same as option "l" except that the "function complex numbers" are listed.

- **-1**  
  
  (1 for "one") This indicator appears once; without this option it is updated every 6 seconds (until you terminate it by pressing the Del or Entf. key).

- **-a**  
  
  This option only makes sense if several rings exist in the OM.

  **OM component**  
  
  The statuses for the OM components and managers of the ring to which the OM component specified belongs to are displayed.

EXAMPLES

**PL**

The statuses of the OM components and object managers are indicated and cyclically updated. If the OM infrastructure has several rings, the ring to which the OM component belongs to is considered.

**PL -l1**

The statuses of the OM components and object managers are indicated once. If the OM infrastructure has several rings the ring to which the OM component belongs to is considered. In addition, the IDs of those function complexes are listed for each object manager for which it works as a server.

**PL pu01a**

The statuses of the OM components and object managers which belong to the same ring as the OM component pu01a are indicated and cyclically updated.

REMARKS

The object manager ASE is displayed with the name ASR.

PL internally calls up the program *pl* which was created for fault analysis of the OM infrastructure and conditions its outputs.
3.11.8 SAMPLER - collecting data for later analysis

SYNTAX
SAMPLER [-h] [-u] [-c] [-s] [-i] [-v] [file...]

DESCRIPTION
The script SAMPLER collects data required for later fault analysis and stores it in a compressed file. The file name is automatically generated and contains the name of the OM component and the date and time. The file path can be found in the help text which can be called up with SAMPLER -h.

The fault analysis files created as a standard contain:

- Data for OM software identification
- The system installation logs
- The diagnostic messages
- The existing core dumps
- Shared memory images (for MMI only)

The additional fault analysis files contain, among other things:

- Information about the Kernel parameters of UNIX
- Information about the status of the computer network.

if no option is specified both the fault analysis files created as a standard and the additional ones are taken over into the output file.

OPTIONS

- **-h** Calls up the help text on the SAMPLER functions; this text also contains information about the path for the compressed file stored by SAMPLER
- **-u** Calls up the help text on the parameters available
- **-c** Individual files must be acknowledged by the user before they are taken over into the output file
- **-s** Do not take over the fault analysis files created as a standard
- **-i** Do not take over the fault analysis files created in addition
- **-v** Log the names of the files taken over

file... Also take over the specified files into the compressed file

EXAMPLES

SAMPLER
The fault analysis files created as a standard and the ones created in addition are taken over into the output file.

SAMPLER -iv ownfile
The faulty analysis file created as a standard and the file ownfile are taken over into the output file. The names of the files taken over are logged.

REMARKS
The output file must be submitted to the OM hotline (phone (+49) 721/595/6485) for evaluation.
3.11.9 Mv.Reset - resetting alarm processing for each function complex

SYNTAX
Mv.Reset FB

DESCRIPTION
Alarm processing is carried out for each function complex of an OM system by the responsible processing function object manager MAC.

The lists created by MAC for a function complex can be reset using the script Mv.Reset. The function complex FC must be identified by the "function complex number"; the "function complex number" is a concise designation consisting of two characters which is allocated by the configuring engineer on the ES. It appears in the individual alarms of the alarm sequence display, for example.

Using skript fb.read Funktionsbereich (FB).

OPTIONS
None

EXAMPLES
Mv.Reset <FC No.>
The alarm processing for the function complex FC no. is reset.

REMARKS
None
3.11.10 Spool.Kill - deleting the data in a spool

SYNTAX
Spool.Kill Spoolname

DESCRIPTION
On the OTs connected to printers or plotters spools are provided in the UNIX operating system which are used to administer these devices and to buffer the data to be output.

The script Spool.Kill is used to delete the data buffered in a spool.

The names of the spools generated on the OT can be determined using the command lpstat -a. The names of the spools used for the OM logs are specified by configuration and can be found in the file $OmProjData/prt/PrtProj on the associated SU.

lpstat -t is used to output the print jobs.

OPTIONS
None

EXAMPLES
Spool.Kill par01
    Deletes the data buffered in spool par01.

REMARKS
None
3.11.11 Carrying out poll commands in the OM network

Syntax
Poll [-a] [-h computer...] command

DESCRIPTION
The command specified is carried out in sequence on several computers and the command result is output. If no options are specified the command is carried out on all OM components in the local network.

Poll can be used provided that the home path, the files of the commands specified and the files activated in the computers involved are stored on the same path. This requirement is met as a standard in the OM system.

REMARKS ON THE HANDLING OF SPECIAL CHARACTERS
Wildcards such as * or ? in the command are converted on the local computer; the file names in the command to be executed are thus the same on all computers.

Environment variables must be preceded and ended by the character ' in order to ensure that they are converted on each individual computer.

If redirection using the special characters | or > is to be used on each individual computer they must be preceded and ended by the character ' before interpretation by the Shell of the local computer.

OPTIONS

- The command is not only carried out on the components of the corresponding ring but on all OM components known to the local OM component corresponding to the configuration of the infrastructure.

-h computer
The names of the computers on which the command is to be carried out they are separated by a comma (without blanks).

EXAMPLES
Poll date
The current value of the date and time is scanned on all OM components of the local ring.

Poll -a cat /txpsys/Version /txpsys/Date
The OM version and the date of creation are scanned on all OM components of the local ring and all other OM components known to the local OM component.

Poll -h su01a, su01b PL
PL is called up once on the computers su01a and su01b (see the remark below)

Poll echo '$LANG'
The value of the environment variable $LANG is scanned on all OM components of the local ring.

Poll 'ps -ef | grep Init' > initfile
The UNIX command ps is called up on all OM components of the local ring; the result is transferred to the filter grep on each individual OM component. The filtered result of all OM components is stored in the file initfile on the local OM component.

REMARKS
If the tool PL is called up via poll, cyclic activation of PL is suppressed (as for the call-up of PL -1).
3.11.12 Compare - comparing command results

SYNTAX
Compare [-a] [-h computer...] command

DESCRIPTION
The command specified is first executed on the local computer, the command result is output. There-
after the same command is sequentially executed on other computers and the command result is
compared to the result of the local computer.

If no options have been specified the command results of all OM components in the local network are
compared to the result of the local OM component.

Compare can be used provided that the home path, the files of the command specified and the files
activated on the computers involved are stored in the same path. This requirement is met as a stan-
dard in the OM system.

REMARKS ON THE HANDLING OF SPECIAL CHARACTERS
Wildcards such as * or ? in the command are converted on the local computer; the file names in the
command to be executed are thus the same on all computers.

Environment variables must be preceded and ended with the character ' in order to ensure that they
are converted on each individual computer.

If redirection using the special characters | or > is to be used on each individual computer they must
be preceded and ended by the character ' before interpretation by the Shell of the local computer.

OPTIONS
-a The command is not only carried out on the components of the own
ring but on all OM components known to the local OM component corre-
sponding to the configuration of the infrastructure. The result is com-
pared to the result of the local OM component.

-h computer The names of the computers on which the command is to be carried
out are separated by a comma (without blanks).

EXAMPLES
Compare date
The current value of the date and time is scanned on all OM components of the local ring.
The result is compared to the result of the local computer.

Compare -a cat /txpsys/Version /txpsys/Date
The OM version and the date of creation are scanned on all OM component of the local ring
and all other OM components known to the local OM component. The result is compared to
the result of the local computer.

Compare -h su01a, su01b PL
PL is called up once on the computers su01a and su01b (see the second remark). The result
is compared to the result of the local computer.

Compare echo ‘$LANG’
The value of the environment variable $LANG is scanned on all OM components of the local
ring. The result is compared to the result of the local computer.
Compare `ps -ef | grep Init' > initfile

The UNIX command `ps' is called up on all OM components of the local ring; the result is transferred to the filter `grep' on each individual OM component. The filtered results of all OM components is compared to the result of the local computer. The associated output data (command result of the local OM component and the results of the comparison) are stored in the file initfile on the local OM component.

REMARKS

Compare compares the results of commands and not files. So do not enter:

```
Compare /txpsys/Version  \textit{wrong!}
```

but:

```
Compare cat / txpsys/Version
```

If the tool PL was called up via Compare, cyclic activation of PL is suppressed (same as for the call-up PL -1).
3.11.13 rdb- reading out the AS620B component status

SYNTAX

rdb

DESCRIPTION

The application of this powerful tool has been limited to two functions for the standard user. The other functions are not disabled but should only be used by the Hotline personnel.

FUNCTION 3: The tool can be used to indicate the status via the communication connection to the AS620B.

FUNCTION 16: Reads an event from the ASR image. The IKZ (FC ID and instance) of the desired event must be specified.

Example of function 3

The TXP system consists of four AS620B components: AS1 AS4 AS5
The AS4 component is not available.

rdb Start
<RETURN>
3 Function 3
3 3-seconds cycle
n No storage in the file
⇒ Display:

<table>
<thead>
<tr>
<th>Asin-</th>
<th>Status</th>
<th>TTD Tel No.</th>
<th>....</th>
</tr>
</thead>
<tbody>
<tr>
<td>dex</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>fc000710</td>
<td></td>
<td>corresponds to AS1</td>
</tr>
<tr>
<td>1</td>
<td>08000008</td>
<td></td>
<td>corresponds to AS4</td>
</tr>
<tr>
<td>2</td>
<td>fc000710</td>
<td></td>
<td>corresponds to AS5</td>
</tr>
</tbody>
</table>

Example of function 16

Follow the dialog

Press the key <del> or <entf> to terminate the tool.
3.12 OM650 subcomponent MOD handling

3.12.1 MOD handling
The dialog box for long-term archiving provides the following operations:
- Formatting a MOD
- Activating a MOD
- Deactivating a MOD
- Outputting an overview of the process data archived

- In redundant systems the MOD is only provides with data from the leading SU.
- No MOD may be inserted into the active SU.
- The current bookkeeping is located on the SU disk and on each MOD.

3.12.1.1 Formatting a MOD
Data can only be archived on a formatted MOD. Each MOD only needs to be formatted once.

⇒ Click on the drive in the drive selection list into which the MOD to be formatted is inserted.

- The keys Aktivieren Deaktivieren and Formatieren become operable

⇒ Click on the Formatieren key.

- The MOD is formatted (approx. 5 min.)

Errors that might occur during formatting are indicated in the output field for the LZA alarms. Possible causes are: the MOD has already been formatted and contains data, no MOD has been inserted or the MOD is defective.

3.12.1.2 Activating an MOD
If a formatted MOD which might contain data is inserted into one of the drives, the MOD first has to be activated. This means that the MOD content is notified to the OM 650.

Before externally storing data on a MOD for the first time a number is allocated to the MOD by the OM 650. The numbers are consecutively incremented by the OM 650. The numbers of the MODs can be indicated in a MOD list in an overview.

⇒ Click on the drive in the drive selection list into which the MOD to be formatted is inserted.

- The keys Aktivieren Deaktivieren and Formatieren become operable.

⇒ Click on the key Aktivieren.

- The MOD is activated. Its number and time range are indicated in the drive selection list.

Errors that might occur during formatting are indicated in the output field for the LZA alarms. Possible causes are: the MOD has not been formatted, no MOD has been inserted or the MOD is defective.
3.12.1.3 Deactivating a MOD

Before you remove an activated MOD from one of the drives the MOD must be deactivated. This means that the OM 650 is notified that the MOD is no longer available for data storage and that it can no longer be accessed.

→ Click on the key **Deaktivieren**.

- The MOD is deactivated. The note 'not activated' is output in the drive selection list. The MOD can then be removed from the drive.

Errors that might occur during deactivation are indicated in the output field for the LZA alarms. Possible causes are: a subarchive is being externally stored or another operator is accessing the data on the MOD.
3.13 Reading the status from the AS620B (CPU948R)

Information about pgmaster

- The program pgmaster can be used to read out various AG functions using the ES.
- The program is started using pgmaster <project>.
- Operation is dialog-based.
- To output the pgmaster directory enter the command:
  
  `find / -name pgmaster -print`

A number is specified before each command. You may activate the command by entering the number and pressing Return.

pgmaster provides 23 diagnostic functions. 18 functions are used by the standard user for diagnostic purposes.

1. Connect
   Establishes the connection to the AP. The AS number must be specified.

2. Disconnect
   Before you exit the program pgmaster the coupling must be logged off.

3. Remove block
   This command allows you to delete PB-DX function blocks, for example, in the AG function blocks. The function block type and the function block number of the function block concerned must be specified.

4. Read system parameter
   This command can be used to read out system information.

5. Compress CPU memory
   Compresses the memory.

6. Erase all
   General deletion of the CPU.

7. Read block directory
   This command is used to output all function blocks of one type, e.g. PB-DX function blocks.

8. Write block
   With this command a function block available in mc5 code can be transferred to the AG. Please note that not all function blocks can be loaded online.

9. Read block
   This command is used to reread a function block, e.g. a PB function block, from the AG. The function block is stored in the directory in which pgmaster was started.

10. Read interruption stack
    Outputs the U-stack if a CPU stops.

11. Read block info

12. Change state of PLC
    Starts and stops the CPU.

13. Write into address
    Writes a value into an address.
14. Read address
   Reads out the value of an address.

15. Unlock APF

16. Lock APF

17. Memory info
   Indicates the unused AG memory space available.

18. Image of Memory
   Writes the complete memory image of the AG into the ES. The files abzug.x -- abzug.y are written into the directory in which pgmaster was started. It therefore makes sense to create an empty directory first from which pgmaster is started. It is then easier to combine it using tar.

19. Steer variable
   Writes data words into data function blocks.

20. Status variable
   Indicates the contents of data words.

21. Diagnostic H-Error Block
   The H-error DB is indicated on the ES monitor. At the same time it is stored in the file hdberror.txt in the current directory. Note: the file is overwritten each time the function is started.

22. Diagnostic DX 4
   The diagnostic DX 4 is indicated on the ES monitor. At the same time it is stored in the file dx4.txt in the current directory. Note: the file is overwritten each time the function is started.

23. Scan memory
   This function is used to output PB-DB function blocks on the monitor.

**Example of the procedure after AG stop during which the memory contents are copied.**

A. Set up an empty directory called mkdir NAME and change to the directory.
   Then start pgmaster from this directory.
B. Enter 1 and press Return to establish the connection to the AG.
   You are then asked for the AS no. =. Enter the AS number (e.g. 4 and Return). Then enter A to establish a connection to the upper AG or B for the lower AG.
C. For the APF no.= just press Return.
D. If "Connection is up" is displayed the connection is established.
E. Then press Return again to return to the selection mask.
F. Enter 18 and press Return. The copying procedure is automatically started.
   "Write to abzug0.dat" to "Write to abzug.e.dat." is displayed on the screen.
   This takes about 10-20 minutes. Thereafter the 15 files abzugx.dat and the file PgFkt.inf are stored in the directory.
G. Press Return to return to the selection mask.
H. Enter 2 to disconnect the connection to the AG and then terminate by X pgmaster.
I. Thereafter these files should be combined using the command tar cvf name.tar *.*.
J. This tar-file should be copied to the path /tmp/mailbox so that it can be transferred using Tele service.
3.14 Setting the MAC address on the CP1430 (AS)

Requirement: S5 program SINEC.NCM under the directory COM1430

Parameterisation of the MAC address:

- Put the CP to Stop:
  - CP function Stop

- Enter the MAC address:
  - Edit CP init address from the ES680 under <project>/listen/lan/NET_ADR_??.ASC
    for AP A  232
    for AP B  236

- Restart CP
  - CP function Start
3.15 UPS Uninterruptible power supply

The UPS is a "stand alone" unit within TXP. The UPS writes the status of the battery into a diagnostic file (powerchute.log). The program "powerchute" provides further diagnostic options:

- Current loading status of the battery
- Current status of the mains voltage

Login: as root user
Path: /usr/lib/powerchute
File: powerchute.log
Program: powerchute;
Password: /1pwrchute

Call up the basic menu using <ctrl>G and terminate the program using Exit
### 3.16 TTD (Time-tagged Data) Check

The automation system sends TTD telegrams to the OM computers. If the connection to the process seems faulty on the OM computers, it makes sense to use a test tool to check the arrival of the TTD telegrams on the OM computers. This tool can also be used to check a signal, for example, which is not updated on a dynamic FUP.

<table>
<thead>
<tr>
<th>Input</th>
<th>Significance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sig.Attach &quot;&lt;KKS&gt;&quot; &lt;Signal&gt;</td>
<td>Checks once whether a TTD arrives on the OM computer for a &lt;signal&gt;, including KKS no. &lt;KKS&gt; or whether a TTD is present.</td>
</tr>
<tr>
<td>Sig.Attach -o&lt;Zyklus&gt; &quot;&lt;KKS&gt;&quot; &quot;&lt;Signal&gt;</td>
<td>Checks cyclically whether a TTD arrives on the OM computer for a &lt;signal&gt;, including KKS no. &lt;KKS&gt; or whether a TTD is present.</td>
</tr>
</tbody>
</table>

**Example 1:**

**Input:**

Sig.Attach "4 0LAC20 AP001" XB01

**Result:**

<table>
<thead>
<tr>
<th>No.</th>
<th>date</th>
<th>time</th>
<th>FC</th>
<th>INR</th>
<th>EA</th>
<th>ETyp</th>
<th>Eurs</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>12.03.98</td>
<td>08:48:02:900</td>
<td>10</td>
<td>1971</td>
<td>1</td>
<td>0001</td>
<td>8000 Binary: 0060 0000 0000</td>
</tr>
</tbody>
</table>

If no TTD telegram is present "Binary : 0060" is replaced by "Nil : 01200"

**Example 2:**

**Input:**

Sig.Attach -o5 "4 0LAC20 AP001" XB01

**Result:**

<table>
<thead>
<tr>
<th>No.</th>
<th>date</th>
<th>time</th>
<th>FC</th>
<th>INR</th>
<th>EA</th>
<th>ETyp</th>
<th>Eurs</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>12.03.98</td>
<td>08:48:02:900</td>
<td>10</td>
<td>1971</td>
<td>1</td>
<td>0001</td>
<td>8000 Binary: 0060 0000 0000</td>
</tr>
<tr>
<td>2</td>
<td>12.03.98</td>
<td>08:48:07:900</td>
<td>10</td>
<td>1971</td>
<td>1</td>
<td>0001</td>
<td>8000 Binary: 0060 0000 0000</td>
</tr>
<tr>
<td>3</td>
<td>12.03.98</td>
<td>08:48:12:900</td>
<td>10</td>
<td>1971</td>
<td>1</td>
<td>0001</td>
<td>8000 Binary: 0060 0000 0000</td>
</tr>
</tbody>
</table>

...
3.17 Data Update

The size of the following system files should be checked on a regular basis and their length reduced to zero, if required.

The following files should be checked:

```
/usr/adm/messages
/usr/adm/syslog
/usr/adm/sulog
/usr/spool/lp/logs/requests
/usr/spool/mail/*
/usr/informix/online.log (SU only)
```

The files should not be deleted. The command `> Dateiname` can be used to set the file length to zero. Thereafter the computer should be restarted using init 6.