GOOSY
Id.: GOOSY Display
Version: 1.0
Date: 14–Jun–1988
Revised: June, 28 1988

GSI **O**nline **O**ffline **S**ystem

GOOSY Display

W. Spreng

June, 28 1988

GSI, Gesellschaft für Schwerionenforschung mbH
Postfach 11 05 52, Planckstraße 1, D-64220 Darmstadt
Tel. (0 6159) 71–0

List of Figures

Chapter 1

Preface

GOOSY Copy Right

Conventions used in this Document

$\boxed{\texttt{Fn}}$, $\boxed{\texttt{PFn}}$, $\boxed{\texttt{1}}$, $\boxed{\texttt{Do}}$, **or** $\boxed{\texttt{Return}}$ **key** — All key in frame boxes refer to the special keypads on VTx20 compatible terminals like VT220, VT320, VT330, VT340, VT420, VT520, PECAD, PERICOM terminals or DECterm windows under DECwindows/Motif on top or right to the main keyboard, to control characters, or to the delete and return keys of the main keyboard.

<Fn>, <PFn>, <KPn>, <Do>, or <Ctrl>— This is the alternative way of writing the keypad or control keys.

$\boxed{\texttt{GOLD}}$, **<GOLD>**— The $\boxed{\texttt{PF1}}$ key is called $\boxed{\texttt{GOLD}}$ in most utility programs using the keypad.

PERICOM— On the PERICOM terminal keyboard the function keys are marked opposite to all other terminals, i.e. the 4 $\boxed{\texttt{PFn}}$ of the rightmost VTx20 compatible keypad are named $\boxed{\texttt{Fn}}$ and the 20 $\boxed{\texttt{Fn}}$ keys on the top of each VTx20 compatible keyboard are named $\boxed{\texttt{PFn}}$ on a PERICOM.

$\boxed{\texttt{Return}}$— The $\boxed{\texttt{Return}}$ is not shown in formats and examples. Assume that you must press $\boxed{\texttt{Return}}$ after typing a command or other input to the system unless instructed otherwise.

$\boxed{\texttt{Enter}}$— If your terminal is connected to IBM, the $\boxed{\texttt{Enter}}$ key terminates all command lines.

$\boxed{\texttt{Ctrl}}$ **key** — The $\boxed{\texttt{Ctrl}}$ box followed by a letter means that you must type the letter while holding down the $\boxed{\texttt{Ctrl}}$ key (like the $\boxed{\texttt{Shift}}$ key for capital letters). Here is an example:

- $\boxed{\texttt{Ctrl}}$ `Z` means hold down the $\boxed{\texttt{Ctrl}}$ key and type the letter `Z`.

$\boxed{\texttt{PFn}}$ **key** — The $\boxed{\texttt{PFn}}$ followed by a number means that you must press the $\boxed{\texttt{PFn}}$ key and *then* type the number. Here is an example:

- $\boxed{\texttt{PF1}}$ `6` press the $\boxed{\texttt{PF1}}$ key and then type the number 6 on the main keyboard.

$\boxed{\texttt{PFn}}$ **or** $\boxed{\texttt{Fn}}$ **keys** — Any $\boxed{\texttt{PFn}}$ or $\boxed{\texttt{Fn}}$ key means that you just press this key. Here is an example:

- $\boxed{\texttt{PF2}}$ means press the $\boxed{\texttt{PF2}}$ key.

Examples— Examples in this manual show both system output (prompts, messages, and displays) and user input, which are all written in `typewriter` style. The user input is normally written in capital letters. Generally there is no case sensitive input in GOOSY, except in cases noted explicitly. In UNIX all input and with it user and file names are case sensitive, that means for TCP/IP services like Telnet, FTP, or SMTP mail one has to define node names, user names, and file names in double quotes "name" to keep the case valid for Open-VMS input. Keywords are printed with uppercase characters, parameters to be replaced by actual values with lowercase characters. The computer output might differ depending on the Alpha AXP or VAX system you are connected to, on the program version described, and on other circumstances. So do not expect identical computer output in all cases.

Registered Trademarks are not explicitly noted.

1.1 GOOSY Authors and Advisory Service

The authors of GOOSY and their main fields for advisory services are:

M. Richter GOOSY Data Management, VAX/VMS System Manager (Tel. 2394)

R. Barth GOOSY and PAW software (since 1995) (Tel. 2546)

H.G. Essel (GOOSY 1983-1993) Data Acquisition (Tel. 2491)

N. Kurz Data Acquisition (since 1992) (Tel. 2979)

W. Ott Data Acquisition (since 1994) (Tel. 2979)

People who have been involved in the development of GOOSY.

B. Dechant GOOSY software (1993-1095) (Tel. 2546)

R. S. Mayer Data Acquisition (1992-1995) (Tel. 2491)

R. Fritzsche Miscellanea (1989-1995) (Tel. 2419)

H. Grein Miscellanea (1984-1989)

T. Kroll Miscellanea, Printers (1984-1988)

R. Thomitzek Miscellanea, Printers, Terminals (1988-1989)

W. Kynast GIPSY preprocessor (1988)

W.F.J. Müller GOONET networking, Command interface (1984-1985)

H. Sohlbach J11, VME (1986-1989)

W. Spreng Display, Graphics (1984-1989)

K. Winkelmann GOOSY Data Elements, IBM (1984-1986)

1.2 Further GOOSY Manuals

The GOOSY system is described in the following manuals:

- GOOSY Introduction and Command Summary
- GOOSY Data Acquisition and Analysis
- GOOSY Data Management
- GOOSY Data Management Commands

- GOOSY Display
- GOOSY Hardware
- GOOSY DCL Procedures. GOOSY Error Recovery
- GOOSY Manual
- GOOSY Commands

Further manuals are available:

- GOOSY Buffer structures
- GOOSY PAW Server
- GOOSY LMD List Mode Data Generator
- SBS Single Branch System
- TCP-Package
- TRIGGER Bus
- VME Introduction
- OpenVMS Introduction

1.3 Intended Audience

This introduction is written for GOOSY users gone through the long way of creating dataelements, especially spectra, and filling them with any data and which are curious to have a look at the things they have produced.

The 'GOOSY Display' manual assumes that you are familiar with the basic data management and dynamic list concepts. If not have a look at the **GOOSY Introduction** manual.

This manual provides all information necessary to have a graphical control about your experimental data and to produce nice GOOSY-pictures. The author would be grateful for any critical comment or any suggestion about this manual.

1.4 Overview

- Section 2:
  Introduction of the basic key words.
- Section 3:
  Usage of display devices.
- Section 4:
  Displaying spectra and scatter plots.
- Section 5:
  Picture concept.
- Section 6:
  Diplay parameter default concept. Window concept.
- Section 7:
  Usage of spectrum calibrations.
- Section 8:
  Description of some often used display operations as EXPAND, INTEGRATE, REFRESH, OVERLAYS, UPDATE, PLOT, PROJECT, and FIT. Setting window conditions and polygons.
- Section 9:
  Usage of the fast display version.
- Section 9:
  Glossary of display.

Chapter 2

Introduction

This manual presents to you an overview about the main facilities offered by the GOOSY Display. The description of the whole functionality would overcharge the extent of this manual. Therefore the concept and the main functions of the GOOSY display are shortly explained. A short description of all display commands is given in Appendix A on page 65. For detail information about the GOOSY display commands you are referred to the **GOOSY Display Command** manual.

The GOOSY Display is activated by :

```
  $ MDISP                    ! in stand alone mode
 or as a component in an GOOSY environment by
  $ GOOSY CREATE PROCESS DISP $DSP
```

The process is started by default with priority 3. Specify another priority by `CREATE PROCESS ... PRIO=p`. To summarize the possibilities of the GOOSY Display, a short description of the fundamental ideas is given:

Devices Are graphical terminals which are used for all graphical outputs. In the GOOSY Display up to 10 different graphical output devices can be operated in parallel.

Spectra and Scatterplots Single spectra or scatterplots can be displayed in different modes. The definition of global display representations is possible.

Pictures Several spectra and scatterplots can be displayed together, if they are organized in pictures. Pictures are elements in your Data Base (see the **GOOSY Data Management** manual), therefore the picture definitions are available even if the display process terminates. Each spectrum or scatterplot is imbedded in frames, which can be modified separately.

Frames The basic entities of a picture. It is possible to manipulate single frames without distoring the other components on the screen.

Calibrations Each calibration can be connected to one or several spectra. Displaying the corresponding spectra, the calibration is applied in the appropriate modes, if it is requested.

Overlays Beside the overlay of a single spectrum, it is possible to define any number of overlays for each frame in a picture. These overlays are displayed by a single command.

Projection The projection of two dimensional spectra on any axis, with arbitrary projection windows and projection polygons is implemented.

Updates The continuous update of any frame on the screen in a specified delta time is possible.

Display Versions Two different display versions are available. The *standard* implementation with the complete functionality, and the *fast* version, which consums only 60% of the CPU–time of the standard version, but with a reduced functionality (see section 9.

The Data Elements referred in the examples of this manual are GOOSY Data Element name specifications (see **GOOSY Data Management** manual). Directory and Data Base names are not explicitly indicated, except they have to be specified. In all commands missing Directory or Data Base names these are substituted by common default values; the standard GOOSY defaults or the user defined defaults. The standard GOOSY defaults are:

DB	for the Data Base name
$CALIB	for the calibration Directory
$CONDITION	for the condition Directory
$DYNAMIC	for the Dynamic List Directory
$PICTURE	for the picture Directory
$POLYGON	for the polygon Directory
$SPECTRUM	for the spectrum Directory
DATA	for the event Directory

All commands creating a Data Element need a Pool specification. The default Pools provided by the GOOSY-Display are:

$CAL_POOL	for the calibration Pool
$PIC_POOL	for the picture and polygon Pool
$SPEC_POOL	for the spectrum Pool

Chapter 3

Display Devices

The GOOSY Display offers the possibility to send the graphical output to up to 10 different devices simultaneously. At any time it is possible to connect or disconnect any device to or from the GOOSY Display. The graphical input occurs at one user definable master device.

3.1 Allocation of Devices

To generate graphical outputs the GOOSY Display has to be connected to a graphical output device by:

```
    ALLOCATE DEVICE name type
```

You have to specify the physical address "name" of the device and a GOOSY alias name "type" for the device type.

The GOOSY Display supports two different device classes. The first class contains the interactive devices (e.g. graphical terminals), at which each graphical output can be seen directly. The available device types are:

GOOSY Alias name	Device type
MG600	Monterey MG600 or MG620 This is the default device type!
MV	Micro–VAX II graphic terminal
GPX	Micro–VAX II color graphic terminal
PECAD	Pecad terminal
TEK4014	Tektronix 4014
TEK4107,TEK4109,TEK4111,TEK4115	Tektronix 4107, 4109,4111,4115
VT240	VT 240, VT 241, VT 330, VT 340, REGIS terminals

Allocating such an interactive device the physical terminal address has to be known:

```
    ALLOCATE DEVICE txa9   pecad   ! terminal directly connected to the host
    ALLOCATE DEVICE lta999 tek4014 ! terminal connected to a DEC-server
```

To get the physical address of a terminal connected to a terminal server, login to the `Local>` terminal server mode (e.g. with the break key ⟨F5⟩), then type the server commands:

```
  Local> SHOW SERVER
  Local> SHOW PORT
  Local> LOGOUT
```

The server information shows you the device number range, e.g `LTA241-LTA248`. The port information gives you the port number, e.g `Port:  3`. The physical address of the graphical terminal yields to `LTA243`. Before creating any output at this terminal you have to logout from the server! Creating outputs at your current terminal specify as the physical address "SYS$COMMAND".

The second class contains the non–interactive devices (e.g. plotter). Allocating these devices, the specified physical address is interpreted as a file name, which is completed by a default file extension, if necessary. Available non–interactive devices are:

GOOSY Alias name	Device type	Default extension
LN03	LN03–Plus laser printer	.LN3
HP7550A4	HP7550A pen plotter with DINA4 sheets	.HP4
HP7550A3	HP7550A pen plotter with DINA3 sheets	.HP3
METAOUT	Output to metafile	.MET

To allocate a LN03 laser printer you have two possibilities:

```
  1.)  ALLOCATE DEVICE test ln03
  2.)  ALLOCATE DEVICE test.out ln03
```

The first case yields to the creation of the file "TEST.LN3" with device dependent sequences for the LN03–PLUS laser printer. The second example creates the file "TEST.OUT". Both can be plotted at any LN03–PLUS printer.

Beside the device dependent plotfiles you have the possibility to create a device independent metafile (e.g. "TEST.MET"):

```
    ALLOCATE DEVICE test.met metaout
```

It is possible to redisplay this metafile on any interactive or non–interactive device. Furthermore you have the possibility to send it to the IBM and to display it there at any available output device (see chapter 8.7 on page 47)!

If at least one device is allocated the GOOSY Display is ready for any graphical output.

3.2 The GOOSY Main Device

The GOOSY Display provides the possibility to allocate up to 10 graphical output devices; with the restriction that the device types are different. The graphical outputs are sent to all allocated devices simultaneously. But the requested graphical inputs have to be performed at one master device. This device is the GOOSY main device. By default the first allocated device is used as the GOOSY main device. If graphical inputs should occur at another device, allocate it with the "/MAIN" switch:

```
ALLOCATE DEVICE lta999 tek4014 /MAIN
```

In principle any device can be allocated as the GOOSY main device. But don't be surprised when no graphical input is possible using a plotter as the master device.

The main device is also used to adjust the graphical output. Normally the facilities of the allocated devices are different, e.g different character heights are supported. The GOOSY Display adjusts the graphical outputs using the character heights supported by the main device. Therefore in any case one main device has to be defined. The other device facilities, like the color, are supported in a device dependent manner.

3.3 Deallocation of Devices

If a plotfile or a metafile should be printed, it is necessary to close and to deallocate the file. In that case the corresponding device has to be disconnected from the GOOSY Display by:

```
DEALLOCATE DEVICE name
```

"name" is the device name as stored internally in the GOOSY Display. In the case of interactive devices it is the physical address, for non–interactive devices it is the file name; like "TEST.LN3". A list of all allocated devices and their internal GOOSY Display names can be get by:

```
SHOW DEVICES
```

The device names and types as well as the current main device type are listed. You also see which device is able to perform graphical inputs, e.g:

```
INFORMATION ON ALLOCATED DEVICES:
Device number:             1
  Logical device name : LTA999
  Physical device name: LTA999
  Device type         : Monterey MG600
  Device category     : Output and input

MAIN DEVICE TYPE        : Monterey MG600
```

Chapter 4

Spectra and Scatterplots

To display single spectra and scatterplots different commands have to be applied. Spectra can be displayed in many different modes, some are discussed in the following. The definition of global display modes for spectra is possible, they can be activated, deactivated, and modified at any time.
The display of scatterplots after a condition check against any parameter is possible. Scatterplots can run in different modes: asynchronously, or synchronously to the analysis.

4.1 Displaying Spectra

A single spectrum "name" could be displayed by:

```
DISPLAY SPECTRUM name
```

If nothing is specified the spectrum is displayed in the full spectrum range. Because each output device has a well defined resolution (e.g. 1024 pixels), it is not necessary to display more bins as pixels provided by the output device. Therefore the GOOSY Display optimizes the spectrum data by displaying them with an internal display binsize which is adjusted to the resolution of the GOOSY main device. Two binning modes are implemented:

- The average about several spectrum bins is calculated, the integral count rate remains unchanged. Therefore the displayed count rates are no longer integer values. Because this mode yields in a smooth spectrum contour it is activated by:

  ```
  DISPLAY SPECTRUM name /SMOOTH
  ```

- The maximum and minimum count rate of the gathered spectrum bins are displayed. With that mode spikes in the spectra are not lost. This mode is the default and can be activated by:

```
        DISPLAY SPECTRUM name /NOSMOOTH
```

The internal display binsize is activated if more than 1024 bins should be displayed.

Sometimes the automatic display binning is not desired. In that case you have the possibility to display a spectrum without binning by:

```
    DISPLAY SPECTRUM name BINFACTOR=1
```

If the binfactor is larger than 1 it specifies the number of spectrum bins to be summed up. This is identical to a temporary modification of the spectrum binsize. A binfactor of 0 yields to an automatic display binning. The actual binning is displayed in the spectrum information field on the screen.

A definite range of a spectrum is displayed specifying:

```
    DISPLAY SPECTRUM name 100,500        ! for one dimensional spectra
    DISPLAY SPECTRUM name 100,500,0,400  ! for two dimensional spectra
```

The scaling axis can be fixed by:

```
    DISPLAY SPECTRUM name SCALIM=100,700
```

Three different display modes (linear, logarithmic and squareroot) for each axis are available:

```
    DISPLAY SPECTRUM name /LOG   ! mode of scaling axis:
                                 ! y-axis for one dim. spectra
                                 ! z-axis for two dim spectra
    DISPLAY SPECTRUM name /XSQRT ! mode of x-axis
    DISPLAY SPECTRUM name /YLIN  ! mode of y-axis
```

All limits and qualifiers specified in one display command, the *temporary display modes*, are available until the next display and can be re–called by:

```
    DISPLAY SPECTRUM name /TEMP
```

The limits and qualifier specified in the display command have the highest priority and modify the corresponding values in the temporary display:

```
    DISPLAY SPECTRUM name /TEMP/LIN ! use temporary modes, but scaling axis
                                    ! is displayed linear.
```

For one dimensional spectra the following representations are implemented:

/HISTO The spectrum data are displayed as histograms.

/VECTOR The spectrum contents are connected by a polyline.

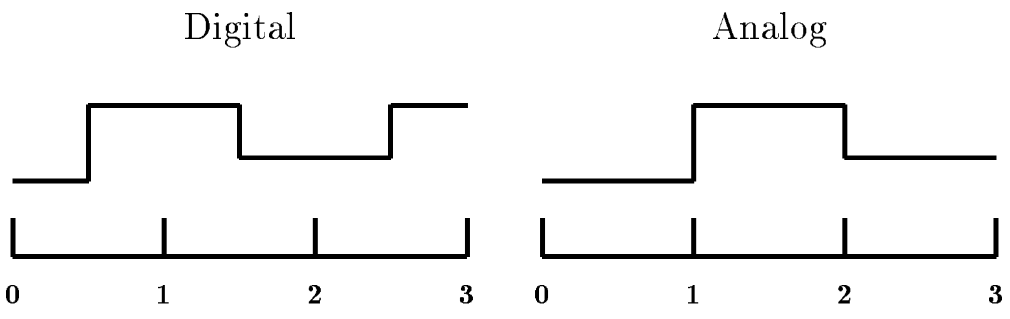

Figure 4.1: Display of Digital and Analogous Spectra

/MARKER The spectrum contents are indicated by markers.

For two dimensional spectra the following spectra representations are implemented:

/CLUSTER The spectrum count rates are indicated by clusters of different size and color.

/CONTOUR Contour lines are displayed.

/ISO A 3D plot is generated.

To display two dimensional spectra some additional parameters are necessary. For contour and cluster plots cuts or intervals have to be defined. They are specified in relative units of the spectrum maximum, e.g.

```
DISPLAY SPECTRUM name CUTS=0.25,0.5,0.75 /CONTOUR
```

draws cuts at $1/4, 1/2, 3/4$ of the spectrum maximum. The number of specified cuts is unlimited. If the cuts are omitted the last ones are used. As default they are drawn in 10% steps. The isometric representation of two dimensional spectra can be rotated in any direction. The rotation is performed in two steps: a clockwise rotation around the x–axis by an angle "THETA" (Default is 25.0^0) and a clockwise rotation around the new y–axis by an angle "PHI" (Default is -15.0^0):

```
DISPLAY SPECTRUM name THETA=245.0 PHI=45.0 /ISO
```

4.1.1 Digital and Analogous Spectra

GOOSY provides two types of spectrum binnings:

DIGITAL spectra are used to accumulate integer or bit data. Each spectrum bin corresponds to a number of integer values accumulated into that bin, where the number of integer values is given by the binsize. E.g. a digital spectrum in the range of 0 to 3 and a binsize of 1 contains the values $0, 1, 2, 3$. The GOOSY Display shows the center of gravity of a spectrum bin at the mean value of the data accumulated into that bin: The displayed bin range is the mean value of the data accumulated into that bin $\pm$ the half of the spectrum binsize. As indicated in figure 4.1 only the half of the first and last bin is displayed. For digital spectra the displayed bin does not indicate the range of values accumulated into that bin!

ANALOG spectra are used for the accumulation of float variables. Each spectrum bin corresponds to a range of real values accumulated into that bin. The lower limit of the bin is inclusive, the upper limits is exclusive. E.g. an analogous spectrum in the range of 0 to 3 and a binsize of 1 contains the ranges $[0, 1)$, $[1, 2)$, $[2, 3)$. The analogous spectrum has one bin less than the digital spectrum and the upper limits is exclusive. The displayed bin range indicates the range of values accumulated into that bin, as shown in figure 4.1. One bin less is shown than in the digital spectrum.

4.2 Global Display Parameters for Spectra

Sometimes the default display parameters, used by the GOOSY Display are not sufficient for your demand. In that case it is possible to predefine the parameters of the `DISPLAY SPECTRUM` command. E.g. a lower limit threshold for the display and the display modes of the axis are set by:

```
DEFINE DISPLAY SPECTRUM (5,) /LOG/XSQRT
```

Except the "BINFACTOR", "CUTS", "THETA", and "PHI" all other parameters of the `DISPLAY SPECTRUM` command can be set. Successive definitions of global display parameters are cumulative and do not distroy the earlier parameter set. Normally the global display modes are activated and used after their definition. At any time a deactivation and activation is possible:

```
DEFINE DISPLAY SPECTRUM /NOACTIVE ! Deactivate global parameters
DEFINE DISPLAY SPECTRUM /ACTIVE   ! Activate global parameters
```

The global parameters are released or activated for one display command by:

```
DISPLAY SPECTRUM /GLOBAL          ! Activate global parameters
DISPLAY SPECTRUM /NOGLOBAL        ! Suspend global parameters
```

All global parameters are listed by:

```
SHOW DISPLAY GLOBALS /SPECTRUM
```

4.3 Displaying Scatterplots

A scatterplot is used if correlations between two event parameters should be displayed:

```
    DISPLAY SCATTER event.x_axis event.y_axis (0,2048,0,500)
                    XLETTER=Energy YLETTER="Delta E"
```

In that example the coincidences of the two members ”x_axis” and ”y_axis” in the Data Element ”event” are shown. The first parameter is displayed in x–direction (horizontal axis) the second one in y–direction (vertical axis). The range, in which the parameter values vary has to be defined both in x and y. The default range is $(0, 1023, 0, 1023)$. At the x-axis the lettering ”Energy” appears, whereas at the y-axis ”Delta E” is displayed. If the parameter names and the displayed range are omitted the previously given values are used.

The display gets the scatter data from the analysis. Therefore the scatter plot information has to be passed to the analysis. This is done internally by creating a dynamic list entry in the default dynamic list $SCATTER (for the explanation of dynamic lists see the **GOOSY Data Acquisition and Analysis** manual). If any other list should be used specify the dynamic list name in the display scatter command:

```
    DISPLAY SCATTER event.x_axis event.y_axis (0,2048,0,500) DYN_SCAT=list
```

The analysis recognizes the new dynamic list entry, opens a communication path to the GOOSY Display process and starts the transfer of scatter data to it. The dynamic list has to be specified only in the first display scatter command, it is used in all subsequent commands requiring a dynamic list name.

The dynamic list must be attatched and the analysis must run to see scatter plot points on the screen. The dynamic list should be attached as the last list, to make sure that all referred conditions are executed propperly.

Sometimes the scatter data should be filtered by a condition:

```
  DISPLAY SCATTER event.x_axis event.y_axis (0,2048,0,500) condition object
```

The condition can be specified with or without a condition object. If no object has been defined it is assumed that the condition has been executed by the analysis or by any dynamic list. In that case only the result bit is checked. If an additional condition object has been specified, an entry in the dynamic list is created and the condition check against the specified object (any event parameter) is performed. In any case if the result bit is true the scatter data are sent to the display. If a condition object has been specified, take care that no entry in the dynamic list for the same condition exists, except the condition objects are the same. If this is not the case, the result from the last condition check is used for all accumulations and scatter plots, which could yield to unexpected results. In the analysis the event loops perform the following steps (for details see the **GOOSY Data Acquisition and Analysis** manual):

1. Call unpack routine (event input).
2. Call user analysis routine.

3. Execution of Dynamic Lists.
4. Fill output event into output buffer (event output).

Normally the scatterplot is activated by the `DISPLAY SCATTER` command, but this is suspended by:

```
DISPLAY SCATTER /NOACTIVE
```

The scatterplot can be activated at any time in two modes:

1. in an **asynchronous** mode. The analysis continues when a scatter buffer has been sent to the display. It may happen that more data will be analysed than displayed in the scatterplot. That mode is selected by:

   ```
   START SCATTER /ASYNCHRONOUS
   ```

   This is the default mode for all scatterplots.

2. in the **synchronous** mode. The analysis waits for an acknowledge from the display and continues if the scatter data are accepted by the display. This mode slows down the analysis! All data analysed is displayed in the scatterplot. This mode is activated by:

   ```
   START SCATTER /SYNCHRONOUS
   ```

An active scatter plot can be stopped any time by:

```
STOP SCATTER
```

ATTENTION: **Never stay in menu after displaying scatterplots. Display cannot scatter during menu prompt!**

NOTE after starting a **new** enviroment one should clear the dynamic list previously used for scatterplots from all old entries. This is done by:

```
DELETE DYNAMIC ENTRY SCATTER list * *
```

This command is executed in the Data Base Manager.

4.4 Plotting Scatterplots

If one wants to get a hardcopy of a scatter plot, the easiest way is to use a local copy of the screen. If a hardcopy is not available at the display terminal, one must allocate a plotfile, display the scatterplot, deallocate the plotfile, and send the plotfile to the printer, i.e.:

```
GOOSY> ALLOCATE DEVICE s1 LN03  ! File s1.LN3
GOOSY> DISPLAY SCATTER ....
GOOSY> DEALL DEV s1.ln3
GOOSY> <CTRL>Z
$ PATEX s1.ln3                  ! Print file
```

Note that the default file type is .LN3 and that you must specify the file type in the `DEALLOCATE` command. The plotfile in the above example is sent to the laser printer LN03_A. The LN03 device may be the second or first allocated. If it is the only one, scatterplot files can be produced in batch jobs.

Chapter 5

Pictures

To control a complex detector array the display of single spectra or scatterplots is not sufficient. Therefore the definition of several spectra and/or scatterplot displays can be gathered in pictures, which are Data Elements in your Data Base. Pictures may be composed of up to 64 spectra or scatterplots. Any single picture element, called frame can be modified at any time. Furthermore global display parameters valid for all frames may be defined.

5.1 Creating Pictures

Creating a picture you have to know the total number of involved spectra and scatterplots. Each spectrum or scatterplot is organized in one independent frame. The total number of frames must be known to create a picture:

```
    CREATE PICTURE name 10
```

creates the picture "name" with 10 frames. The number of frames is fixed after the picture creation and could not be changed later. The picture frames are numbered starting with 1 for the upper left frame (see figure on page 27). If you intend to produce scatterplots in the picture you have the possibility to define a main condition for all scatter frames:

```
    CREATE PICTURE name 10 condition object
```

Similar to the scatterplots the condition can be specified with or without a condition object. If no object is declared, it is assumed that the condition has been executed earlier and only the result bit will be checked. If additionally a condition object is given the condition limits will be checked against the object and an earlier check of the same condition is overwritten. If the result bit of the condition is true the scatter data are sent to the GOOSY Display process. An existing picture is deleted by:

```
    DELETE PICTURE name
```

The CREATE PICTURE and DELETE PICTURE commands are executed in the Data Base Manager!

After the creation of a picture all frames are undefined. Before displaying a picture **all** frames have to be pre–set with valid spectra names or scatterplot parameters.

5.2 Modifying Picture Frames

In a single picture frame all informations necessary for the display of that frame are stored. One has to distinguish between spectra and scatter frames:

Spectrum frame For a spectrum frame all parameters of the "DISPLAY SPECTRUM" command, exept "BINFACTOR", "CUTS", "THETA", and "PHI", can be set.

```
MODIFY FRAME SPECTRUM picture frame ....
```

The modify command changes any specified parameter in the "frame" of "picture". The modification of several frames within one modify command is supported if the picture should be filled with members of a one dimensional spectrum array. In that case a range of members in the spectrum array and a range of frames can be specified. Subsequent spectra are put into subsequent frames:

```
MODIFY FRAME SPECTRUM picture 1:4 spectrum(3:6)
```

In this example the spectra "spectrum(3)" to "spectrum(6) are put into the frames 1 to 4!

Additionally it is possible to define a dynamic scaling axis. This is useful if you like to increase or decrease the default GOOSY Display scaling axis by a certain amount. But normally the spectrum contents change when the analysis is active, so no fixed values should be specified in that case. The dynamic scaling factor is used as a factor to the maximum spectrum content. E.g. to increase the scaling axis to twice the spectrum maximum, specify:

```
MODIFY FRAME SPECTRUM picture frame SCALEFACTOR=2.0
```

Scatter frame For scatter frames all parameters of the "DISPLAY SCATTER" command, exept the "DYN_LIST", can be set:

```
MODIFY FRAME SCATTER picture frame ....
```

The modify command changes any specified parameter in the scatter "frame" of "picture".

Beside the main picture condition, the definition of a frame condition is possible with or without a condition object. Scatter data for this frame are shown if both the picture and the frame conditions are true.

The MODIFY FRAME commands are executed in the Data Base Manager!

If frame conditions with a condition object are specified, take care that the same condition never refers to different condition objects!

If a condition object is specified in the picture the condition is inserted in the dynamic list when the picture is displayed. The result of this condition check is used for the scatter display.

Successive frame modifications are cumulative, which means that only the specified parameters are changed, the other values in the frame remains unchanged. Normally it is possible to change a scatter frame into a spectrum frame and vice versa, if no overlays are defined in the picture (see chapter 8.5 on page 45).

Additionally to the frame definition via the modify commands, it is possible to invoke interactive prompting menus. In that mode for each frame a menu is created, which allows to specify all necessary input parameters. This can be used in interactive sessions, to be sure that all frames are properly specified. The interactive prompting menus are created by:

```
CREATE PICTURE name 10 /PROMPT
CREATE PICTURE name 10
```

To create pictures in a command procedure specify:

```
CREATE PICTURE name 10 /NOPROMPT
```

to prevent the interactive inquiry. The definition of the single frames occurs by successive `MODIFY` commands. The parameters set in the picture are listed by:

```
SHOW PICTURE picture /FULL/DATA
```

In the picture and Directory name wildcards are allowed. The switch "/FULL" effects the listing of the spectra and scatter parameters for each frame. With the "/DATA" switch all other specified values are listed.

5.3 Displaying Pictures

If all frames are defined the picture is displayed by a single command:

```
DISPLAY PICTURE picture
```

The parameters stored in the picture definition are used, creating the display of the single frames. But they can be changed for the current display command, if other parameters are specified, e.g:

```
DISPLAY PICTURE picture /LIN
```

displays the picture frames with a linear scaling axis. Similar to the "DISPLAY SPECTRUM" command a binning factor can be specified to prevent the automatic display binning (see page 16):

```
DISPLAY PICTURE picture BINFACTOR=bins
```

If scatter frames are involved in the picture the declaration of a dynamic list is possible if an other list than the GOOSY default $SCATTER should be used. In that dynamic list a dynamic entry is created to activate the preparation of the scatter data in the analysis (see page 19).

```
DISPLAY PICTURE picture DYN_SCAT=list
```

ATTENTION: **Never stay in menu after displaying pictures containing scatterplots. Display cannot scatter during menu prompt!**.

NOTE after starting a **new** enviroment one should clear the dynamic list previously used for scatterplots from all old entries. This is done by:

```
DELETE DYNAMIC ENTRY SCATTER list * *
```

This command is executed in the Data Base Mananger.

5.4 Global Display Parameters for Pictures

The global display parameters for pictures are used in the same way as the global parameters for spectra. They can be set and activated by:

```
DEFINE DISPLAY PICTURE ...    ! specify any parameter
```

Except the "CONDITION", "DYN_SCAT", and "BINFACTOR" all other parameters of the display picture command can be set. Successive definitions of global display parameters are cumulative and do not destroy the earlier parameter set. Normally the global display modes are activated and used after their definition. At any time a deactivation and activation is possible:

```
DEFINE DISPLAY PICTURE /NOACTIVE ! Deactivate global parameters
DEFINE DISPLAY PICTURE /ACTIVE   ! Activate global parameters
```

Furthermore the global parameters are released or activated for one display command by:

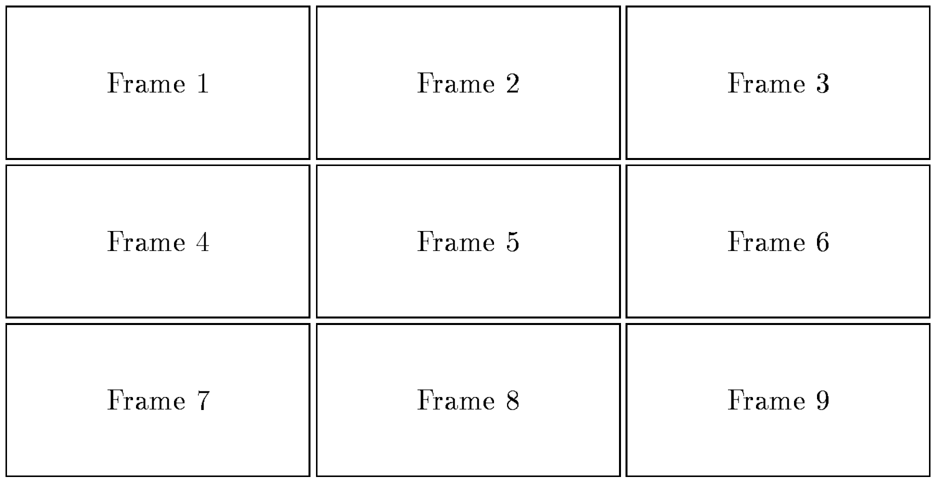

Figure 5.1: Standard Set–Up for Picture with 9 Frames

```
    DISPLAY PICTURE /GLOBAL          ! Activate global parameters
    DISPLAY PICTURE /NOGLOBAL        ! Suspend global parameters
```

All global parameters are listed by:

```
    SHOW DISPLAY GLOBALS /PICTURE
```

5.5 Picture Set–Up

Creating a picture the frames are arranged in rows and columns on the screen and the size of the frames is equal (see figure 5.1). In principle the size of the frames and their position on the screen is arbitrary, with the restriction that an overlap of two frames could yield in unexpected results if cursor inputs are provided. There is one command which allows a redefinition of the picture frames at any time after the creation of the picture. You can specify the number of rows in which the frames should be arranged, that means the vertical extension of all frames is the same. Furthermore for each row the number of frames in that row have to be specified (see figure 5.2):

```
    DEFINE PICTURE SETUP picture 4 3,3,2,1 ! 4 horizontal rows
                                           ! 3 frames in row 1 and 2
                                           ! 2 frames in row 3
                                           ! 1 frame in row 4
                                           ! total of 9 frames
```

Normally the frames in each row are spread over the horizontal screen range. If the size of the frames should be identical specify (see figure 5.3):

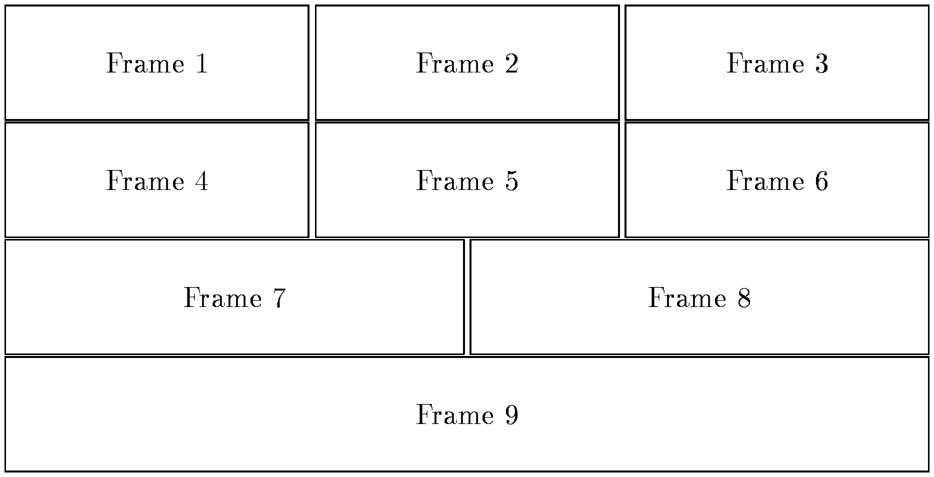

Figure 5.2: User Defined Set–Up for Picture with 9 Frames

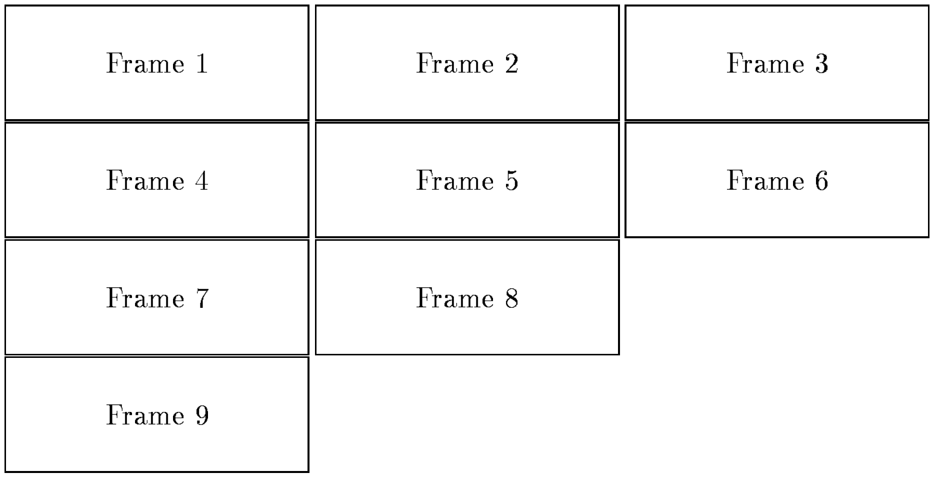

Figure 5.3: Set–up with /EQUAL Switch for Picture with 9 Frames

```
DEFINE PICTURE SETUP picture 4 3,3,2,1 /EQUAL
```

5.6 Clear Pictures

Normally pictures are used to summarize several spectra or scatterplots which are in a physical relation. And sometimes it might happen that you wish to clear all the spectra arranged in one picture, e.g if analysis parameters are changed. This is done by:

```
CLEAR PICTURE picture 3      ! Clear spectrum in frame 3
CLEAR PICTURE picture 1:8    ! Clear spectra in frame 1 to 8
CLEAR PICTURE picture *      ! Clear spectra in all frames
CLEAR PICTURE * * /LOG       ! Clear spectra in all frames of all
                             ! pictures
```

Chapter 6

Special GOOSY Display Concepts

6.1 The GOOSY Display Mode Concept

Discussing the display commands for spectra and pictures, temporary and global display modes have been introduced. But two other parameter sets exist: the defaults provided by the GOOSY Display and the modes specified directly in the commands. Now the priority and correlations of these parameter sets will be described. They are listed in the following with increasing priority:

GOOSY defaults are the values provided by the GOOSY Display to guarantee that all parameters are specified, which are absolutely necessary for the display. *The GOOSY defaults are placed at the lowest priority level.*

Global modes are user defined default values. By default they are active, but a deactivation is possible at any time. A permanent [de]activation is obtained by:

```
DEFINE DISPLAY SPECTRUM /[NO]ACTIVE
DEFINE DISPLAY PICTURE /[NO]ACTIVE
```

A [de]activation for a single command is possible, too:

```
DISPLAY SPECTRUM /[NO]GLOBAL
DISPLAY PICTURE /[NO]GLOBAL
```

For pictures and spectra separate parameter sets exist.

Temporary modes are the values specified in the previous display command. They are identical for spectra and pictures and can be activated in the display commands:

```
DISPLAY SPECTRUM /TEMP
DISPLAY PICTURE /TEMP
```

Command modes are specified in the actual display commands and are available in the next display command as temporary parameters. *The command parameter have the highest priority.*

Parameter values in sets of lower priority are superseded by values of higher priority. If e.g. the global parameter "/LOG" is specified for the scaling axis, the GOOSY Display default "/LIN" is no longer valid. The global value is superseded, if e.g. "/SQRT" is specified in the command. The hierarchy of the GOOSY Display parameter sets is shown in figure 6.1.

6.2 The GOOSY Display Window Concept

For displaying spectra or pictures the following windows are available in the GOOSY Display :

1. The **full window**, which displays the spectrum data in the full spectrum range. It is selected by:

   ```
   DISPLAY SPECTRUM /FULL
   DISPLAY PICTURE  /FULL
   ```

   This full window is the default in the DISPLAY SPECTRUM command.

2. The **specified window**, which displays the spectrum data in the specified range. The specified range are the limits defined in the modify frame commands or directly in the display spectrum command. It is activated by:

   ```
   DISPLAY SPECTRUM LIMITS=range
   DISPLAY PICTURE /SPECIFIED
   ```

   The specified window is the default in the DISPLAY PICTURE command if in the MODIFY FRAME SPECTRUM command limits has been defined.

3. The **last window** is the window of the last display command. Normally, these windows are different for each frame, depending on the actions peformed with the single frames (e.g which region has been expanded). These windows are kept separately for each picture and for one and two dimensional spectra! These windows are indicated in each frame by a # and can be selected by:

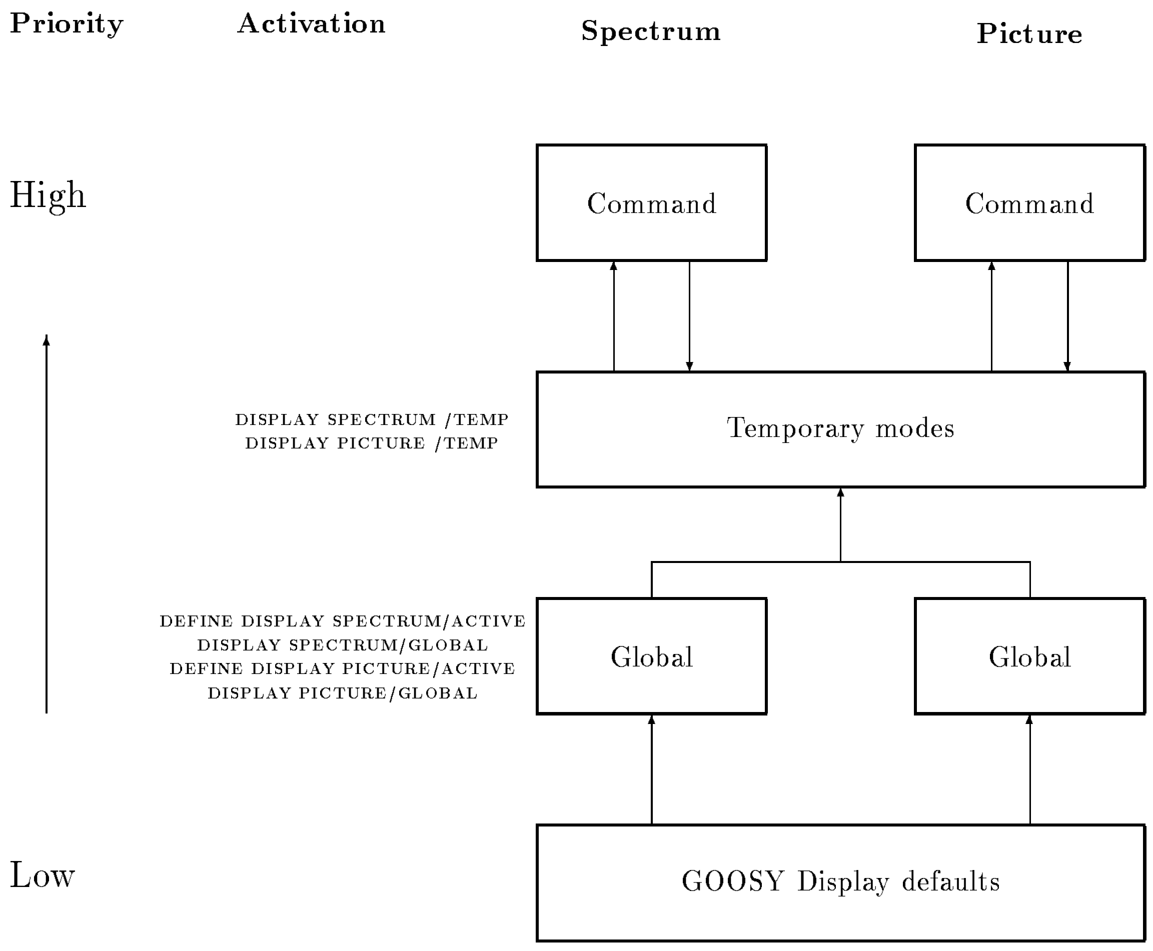

Figure 6.1: Priorities of the Different GOOSY Display Modes

```
          DISPLAY SPECTRUM /LAST
          DISPLAY PICTURE  /LAST
```

The last window is the default in the DISPLAY PICTURE command if in the MODIFY FRAME SPECTRUM command no limits are specified. If the display range of a frame is changed, this range is used in the next display picture command!

4. The **actual window** is the currently displayed range, it can be used for the next display, specifying:

```
          DISPLAY SPECTRUM /ACTUAL
          DISPLAY PICTURE  /ACTUAL
```

In several commands it is posible to specify the window limits directly; e.g.:

```
    DISPLAY SPECTRUM LIMITS=window
```

The GOOSY Display provides special window specification possibilities. The window limits are enclosed in brackets and the values are separated by commas. The first two values are reserved for the minimum and maximum values for the x–axis (horizontal direction), the next two values are the limits on the y–axis (vertical direction) for two dimensional spectra:

```
     DISPLAY SPECTRUM LIMITS=(xmin,xmax)           ! one dimensional window
     DISPLAY SPECTRUM LIMITS=(xmin,xmax,ymin,ymax) ! two dimensional window
```

In all window specifications it is possible to declare only one limit, the upper or lower values can be skipped by an empty input, they are replaced by the highest, lowest possible input:

```
     DISPLAY SPECTRUM LIMITS=(5,)     ! Lower display limit is 5
                                      ! Upper limit is upper spectrum range.
     DISPLAY SPECTRUM LIMITS=(,400)   ! Display from lower spectrum limit
                                      ! up to 400
     DISPLAY SPECTRUM LIMITS=(,,200)  ! Display in full x-range and
                                      ! from 200 till upper spectrum limit
                                      ! in the y-range
```

For other commands, like EXPAND or INTEGRATE, the lowest and highest possible input values are determined by the actual display window! With these window concept is possible to define lower or upper display thresholds which are independent on the limits of the actually displayed spectra.

Chapter 7

Displaying Calibrated Spectra

Displaying spectra in arbitrary units is sometimes not very informative. Normally one is interested in physical units, especially if different spectra should be compared. Therefore the GOOSY Display provides user defined calibration functions calibrate spectra. The calibrations are kept in tables in the Data Base and can be connected to an arbitrary number of spectra. Afterwards a display of the spectrum in calibrated units is possible.

7.1 Creating Calibrations

To display spectra in calibrated units you need a well defined unambigious function. On the other hand, if you prefer to think in calibrated units, the inputs of all spectrum limits should occur in calibrated units. In that case an unambigious inverse calibration function is needed to convert the inputs back into spectrum units. If that is requested, only one method is possible: the calibrations have to be kept in tables, which allow to calibrate and recalibrate each value. GOOSY provides three different types of calibration tables, to fit all requirements:

1. The **linear calibrations** are linear polynoms. For these functions the calibration and recalibration is defined and can be determined exactly. Therefore the polynom parameters are kept in the calibration Data element. They are created by:

   ```
   CREATE CALIBRATION LINEAR name
   ```

2. The **fixed calibrations** are created with a fixed step width for the table entries of the calibrated values. Therefore only a list of calibrated values is kept in the calibration table. The table range is defined by the value of the first table entry, the step width between two entries and the table length. These calibrations should be used for functions which vary slowly. They are created by:

```
CREATE CALIBRATION FIXED name entries
```

Besides the calibration "name" the number of calibrated values ("entries"), stored in the table, must be specified.

3. The **float calibrations** are created with a variable step width for the table entries of the uncalibrated values. Therefore a list of calibrated as well as the corresponding uncalibrated values are kept in the Data Element. These calibrations should be used for functions which vary drastically. They are created by:

   ```
   CREATE CALIBRATION FLOAT name entries
   ```

   Besides the calibration "name" the number of calibrated values ("entries"), stored in the table, must be specified.

7.2 Set Calibration Tables

The calibration tables can be set in different modes:

1. The table entries are directly specified.
2. A list of calibrated and uncalibrated values is given and a polynom fit should be performed. This polynom is used to determine the table entries.
3. A list of calibrated values is given. The corresponding uncalibrated values are specified via cursor inputs. Then a polynom fit is performed.
4. A user module is called, which is used to fill the calibration table.

Sometimes the calibrated values are calculated by a user program and stored in a disk file. If the file is written in a specific format, it is possible to read the required inputs from that file.

The input possibilities for the specification of the calibration tables are different for each type of calibration:

Linear calibration In the linear calibration tables the parameters of a linear polynom are kept. There are different possibilities to determine the polynom parameters:

1. They can be specified directly:

   ```
   SET CALIBRATION LINEAR name "units" PAR=(100.0,0.5) /PARAMETER
   ```

The linear calibration "name" is specified with an offset 100.0 and a polynom factor of 0.5. The calibration description "units" is displayed at the calibration axis, if this calibration has been applied to a spectrum.

2. Another possibility is to determine the polynom parameter by fitting a list of calibrated and uncalibrated values:

   ```
   SET CALIBRATION LINEAR name "units" CALIB=list UNCALIB=list/FIT
   ```

   It is possible to read these two lists of values from a file:

   ```
   SET CALIBRATION LINEAR name "units" file/FILE/FIT
   ```

   The format of the input file is described below on page 39.

3. Sometimes the list of calibrated values is fixed (e.g for calibration sources) in that case this list can be read from file or specified directly and the corresponding uncalibrated values can be prompted by cursor:

   ```
   SET CALIBRATION LINEAR name "units" file/FILE/PROMPT
   SET CALIBRATION LINEAR name "units" CALIB=list/PROMPT
   ```

   After that a polynom fit is performed and the parameters are stored in the calibration Data Element.

Fixed calibration The step width between two successive uncalibrated values is fixed. Therefore the range of the calibration table is defined by the first uncalibrated value and the step width.

1. The corresponding calibrated values can be specified directly or read from a disk file by:

   ```
   SET CALIBRATION FIXED name "units" 100.0 1.5 CALIB=list /TABLE
   SET CALIBRATION FIXED name "units" 100.0 1.5 file /FILE/TABLE
   ```

   The first uncalibrated value for which a table entry exists is 100.0 and the step width is 1.5; these are spectrum units. "/TABLE" means that all table entries are specified and "/FILE" directs the input to "file". The format of the input file is described on page 39.

2. Similar to the setting of the linear calibrations it is possible to specify the parameter of an arbitrary polynom, which is used to calculate the table of calibrated values:

   ```
   SET CALIBRATION FIXED name "units" 100.0 1.5 PARA=(10,2,0.5) /PARAM
   ```

The following polynom is used to determine the calibrated values:

```
cal = 10.0 + 2.0*uncal + 0.5*(uncal**2)
```

3. The polynom parameter can be determined by fitting calibrated and uncalibrated data. The same features as for the linear calibrations are realized:

```
SET CALIBRATION FIXED name "units" 100 1.5 CAL=list UNCAL=list/FIT
SET CALIBRATION FIXED name "units" 100 1.5 file /FILE/FIT
SET CALIBRATION FIXED name "units" 100 1.5 CAL=list/FILE/PROMPT
SET CALIBRATION FIXED name "units" 100 1.5 file /FILE/PROMPT
```

4. Additionally the calibration table can be set by calling a user module, which is linked into a sharable image:

```
SET CALIBRATION FIXED name unit 100.0 1.5 MOD=module IMAG=image/MOD
```

The user module is called with the following parameter list:

```
L_sts = module(R_start,R_step,L_entries,RA_table)
                ! R_start - start value of table
                ! R_step  - Step width between two entries
                ! L_entries - Number of table entries
                ! RA_table -  Array which contains the final
                            calibration values.
```

Float calibrations The step width between two successive uncalibrated values varies. In that case all uncalibrated and calibrated table entries have to be stored.

1. The table can be set by a user module with the following command:

```
SET CALIBRATION FLOAT name "units" MODULE=module IMAGE=image
```

Your module has to be linked into the sharable image "image" and is called with the following parameter list:

```
status=program(L_entries,RA_uncal,RA_cal);
   ! L_entries    Number of table entries
   !              BIN FIXED(31) Input
   ! RA_uncal     Array with uncalibrated values
   !              (L_entries) BIN FLOAT(24) Output
   ! RA_cal       Array with calibrated values
   !              (L_entries) BIN FLOAT(24) Output
```

2. Or it is possible to read the total calibration table (the lists of calibrated and uncalibrated values) directly from the file "file":

   ```
   SET CALIBRATION FLOAT name "units" file /FILE
   ```

Take care that in the calibration table for each uncalibrated value only one calibrated value exists, and vice versa. If this is not the case the result obtained displaying a calibrated spectrum is unpredictable.

After the calibration tables has been set, the unambigiuty of the inverse calibration is proofed and a warning message is created if the calibration is not correct. The calibration function can be displayed by:

```
DISPLAY CALIBRATION name
```

This is a convient way to control the correctness of the calibration.

Calibration Input File Format

Calibration files are ASCII text files. The format of the files is very simple: An exclamation point (!) in the first column indicates a comment line. In each record one or two input values are expected depending on the input mode:

/FIT Two input values, separated by comma or blanks, are expected in each line. The first one is the uncalibrated value, the second one the corresponding calibrated value.

/PROMPT For each line one calibrated input value is expected.

/TABLE For a fixed calibration one calibrated value is expected in each line. For the float calibrations a list of uncalibrated and calibrated values are read.

E.g. for the "/FIT" input mode the file looks like:

```
!
!    uncalibrated          calibrated  value
!
      500.0                  800.0
      700                    1020.0
!  Another comment line
      1020,                  2000.0
      1040,2100
!
!   end of file
!
```

Linking Modules into a Sharable Image

Dynamically loaded user modules have to be linked into a sharable image. This can be done by the DCL command `LSHARIM`:

```
$ LSHAR module image
```

One or several modules can be linked together into the specified sharable image. If the modules uses internally global variables (COMMON blocks in FORTRAN or EXTERNAL variables in PL/1), they must be defined explicitly and must be placed into an unshared section in the image. This is done by `LSHARIM` if global parameters are specified:

```
$ LSHAR module image /GLOBAL=variables
```

For the dynamic load of the sharable image a logical name has to be assigned to it:

```
$ LSHAR module image /SHARELOG=name
```

7.3 Calibrating Spectra

Up to now the created calibrations are independent Data Elements in your Data Base. But they should be connected to one or several spectra:

```
CALIBRATE SPECTRUM spectrum calibration
```

The connection between spectra and calibrations is established creating links between the Data Elements (see the **GOOSY Data Management** manual).

Up to now only one dimensional spectra can be calibrated.

It is possible to connect one calibration to several spectra, but it is not possible to connect one spectrum to several calibrations. Furthermore the calibration of a spectrum can be changed any time by disconnecting it from its current calibration and then connecting it to another one. The decalibration of a spectrum is performed by

```
DECALIBRATE SPECTRUM spectrum
```

the "spectrum" is disconnected from its current calibration. If you want to delete the spectrum, it must be decalibrated first. A list of all spectra connected to a calibration can be obtained by:

```
SHOW CALIBRATION calibration /LINKS /TABLE
```

The switch "/LINKS" shows all established links and "/TABLE" lists the total contents of the calibration table.

7.4 Displaying Calibrated Spectra

When the calibrations are made up calibrated spectra can be displayed. In the GOOSY Display two calibration modes are implemented:

1. A calibrated axis is drawn below a spectrum shown in the original coordinate system. Using non–linear calibrations the distance between two neighbouring tics varies. This mode is activated by:

```
DISPLAY SPECTRUM /CALAX
DISPLAY PICTURE /CALAX
```

2. The spectrum is drawn in calibrated units. Then the size of the displayed spectrum bins vary. This mode is activated by:

```
DISPLAY SPECTRUM /CALSP
DISPLAY PICTURE /CALSP
```

In any case the uncalibrated axis is displayed, too. This can be suppressed by:

```
DISPLAY SPECTRUM /CALAX/NOCHAN
DISPLAY PICTURE /CALAX/NOCHAN
DISPLAY SPECTRUM /CALSP/NOCHAN
DISPLAY PICTURE /CALSP/NOCHAN
```

Chapter 8

Often Used Display Commands

In this chapter some useful commands are presented. The most common actions are the integration of a spectrum range or the expansion of spectra and scatterplots.
Comparing spectra in comfortable manner can be done by overlays. Definable, linear transformations are applied to the overlayed spectra in x and y–direction.
The standard GOOSY Display keeps all graphical output in memory in a device independent form. Therefore it is possible to plot the currently active picture on any supported plotter.

8.1 Expand

Sometimes the displayed range in one or several frames is not sufficient and you like to change the display limits. This is done by the "EXPAND" command. You have the possibility to specify the new range graphically by:

```
EXPAND                  ! without any parameter
```

The cursor appears at the GOOSY main device (see chapter 3.1 on page 12) and the new display range can be specified in any frame. The corresponding frame is deleted from the display screen and redrawn within the specified limits. For all terminals, which support the erasure of local display parts, the other frames are not affected by that operation. For dump terminals (like TEKTRONIX 4014) the whole screen is erased and redrawn. If the expansion limits are known, they can be specified directly in the command:

```
EXPAND (100,567)            ! for one dimensional spectra
EXPAND (100,567,234,1056)   ! for two dimensional spectra or
                            ! scatter plots.
```

In that examples, the cursor appears, if more than one frame is displayed, to select the frame which should be used. Furthermore it is possible to expand more than one frame by:

```
EXPAND (100,500) 2:4       ! expand frame 2 to 4
EXPAND (100,500) *         ! expand all frames
EXPAND FRAME=*             ! the cursor appears to specify the limits
```

If calibrated spectra are in the selected frames the new display limits can be specified in calibrated or uncalibrated units:

```
EXPAND (100,567) /CHAN      ! limits are uncalibrated units
EXPAND (100,567) /CALIB     ! limits are calibrated units
```

8.2 Integrate

The syntax of the "INTEGRATE" command is similiar to the "EXPAND" command. Limits can be specified directly or graphically and in calibrated or uncalibrated units. Furthermore the integration of spectra in one or in several frames is possible. Some examples:

```
INTEGRATE                         ! cursor input
INTEGRATE FRAME=frame             ! cursor appears in frame
INTEGRATE (100,200) *             ! integrate spectra in all frames.
```

8.3 Sum Up Spectrum

The `INTEGRATE` command provides graphical inputs. If the limits of the window which should be integrated are known you can use `SUMUP` instead of `INTEGRATE`:

```
SUMUP SPECTRUM spectrum (100,200)
```

It is possible to define more than one window by:

```
SUMUP SPECTRUM spectrum (100,200,400,500)
```

Similiar to the `INTEGRATE` command the window limits can be specified in original spectrum units or in calibrated units. *The* `SUMUP SPECTRUM` *command is executed in the Data Base manager!.* No display of the spectrum is needed.

8.4 Refresh

Sometimes the current picture should be re–displayed to get rid of superfluous information. E.g if scatterplots are in the picture you can erase the scatterpoints, without another `DISPLAY PICTURE` or `DISPLAY SCATTER` command and without creating a dynamic entry, which takes some time. This is done by:

```
REFRESH 5             ! Erase frame 5 and redisplay it
REFRESH *             ! Redraw all frames
```

If spectrum frames are refreshed, the old spectrum data are displayed. To get the new spectrum contents, specify:

```
    REFRESH 5 /UPDATE
    REFRESH * /UPDATE
```

8.5 Overlays

If spectra or pictures are displayed, overlays of one- and two dimensional spectra are possible:

```
    OVERLAY spectrum FRAME=frame
```

The frame has to be specified if more than one frame is displayed. The overlayed spectrum can be shifted and scaled in x and y–direction:

```
    OVERLAY spectrum FRAME=frame TRANS=(xfac,xoff,yfac,yoff)
    OVERLAY spectrum FRAME=frame
            TRANS=(1.0,0.0,2.0,100.0) ! Shift in y-direction by 100
                                      ! Factor in y-direction: 2.0
    OVERLAY spectrum FRAME=frame
            TRANS=(,,2.0,100.0)       ! Shift in y-direction by 100
                                      ! Factor in y-direction: 2.0
    OVERLAY spectrum FRAME=frame
            TRANS=(0.5,-30)           ! Shift in x-direction by -30.0
                                      ! Factor in x-direction: 0.5
```

Sometimes the transformation factors are unknown. To spare you the calculation of the transformation an automatic adjustment is implemented, which is called by:

```
    OVERLAY spectrum FRAME=frame /ADJUST
```

In that mode the cursor appears twice to select two points in the original spectrum. Then the spectrum, which should be overlayed is drawn and the cursor appears again to inquire two points. These cursor inputs define a linear transformation applied to the overlayed spectrum.

Besides the overlay of spectra the GOOSY Display provides overlays of several scatter parameters:

```
    OVERLAY XPARA=event.x_para YPARA=event.y_para FRAME=frame /SAVE
```

The defined parameters are scattered additionally into the specified frame. The `/SAVE` switch is necessary to save the scatter parameter in the data element of the actually displayed picture data element and to initiate the update of the SCATTER entry in the dynamic list, which enables the analysis to create scatter buffers containing the additional parameter set. The scatter points are displayed in another color, if that is possible. If `/SAVE` is not specified no update of the SCATTER entry in the dynamic list is performed and therefore the additional parameters are not displayed.

If you know that overlays of different spectra are necessary, you can define a set of overlays for each existing picture. These overlay definitions are kept in the picture definition in your database and can be displayed by a single command:

```
OVERLAY            ! without any parameter
```

Define your overlays by:

```
CREATE OVERLAY picture frame spectrum        ! for spectrum frames
CREATE OVERLAY picture frame XPARA=x YPARA=y ! for scatter frames
```

All parameters of the "OVERLAY" command can be specified in the "CREATE OVERLAY" command, the syntax of both commands is nearly identical. If overlayed spectra should be shifted into y–direction you run into problems, because the scaling axis of the original spectra changes if an analysis is active. Therefore a dynamic shift can be specified, which is used as a factor to the actual maximum spectrum contents. An overlay of "spectrum" in frame "2" of "picture" shifted by half of the maximum of the original frame spectrum is defined by:

```
CREATE OVERLAY picture 2 spectrum DYNSHIFT=0.5
```

If the count rate in the overlayed spectrum exceeds the maximum of the original frame spectrum it is possible to generate a dynamic scaling axis (see page 24):

```
MODIFY FRAME SPECTRUM SCALEFACTOR=2.0
```

The scaling factor and the dynamic shift allow an optimal definition of shifted overlayed spectra, independent of the actual spectra maxima.

If the overlays created for a single frame or for the total picture are no longer needed, they can be deleted by:

```
DELETE OVERLAY picture 3   ! overlays of frame 3 are deleted
DELETE OVERLAY picture 2:5 ! overlays of frame 2 to 5 are deleted
DELETE OVERLAY picture  *  ! all overlays are deleted
```

8.6 Updating Frames

The spectrum accumulation can be controlled if in a specified time the new spectrum contents are drawn across the original spectrum. To update all frames on the screen, declare an update time in the display commands:

```
DISPLAY SPECTRUM UPDATE=10    ! Update of spectrum every 10 sec.
DISPLAY PICTURE UPDATE=2      ! Update of all frames every 2 sec.
```

The update of definite frames is performed by:

```
    UPDATE FRAMES 5,4,1 10
```

The spectra in frames $1,4,5$ are drawn every 10 seconds. An arbitray number of valid frames can be specified.

The update of frames is canceled by any "DISPLAY SPECTRUM" or "DISPLAY SCATTER" or "DISPLAY PICTURE" command.

8.7 Plotting Pictures

The standard GOOSY Display (see page 53) keeps the most graphical output in a device independent format in a workstation independent segment storage (see the **GKS Primer** manual). This storage is kept in the memory or in a disk file if the required memory gets too large. But there are two exceptions: scatter data and the data obtained by the picture update (see chapter 8.5 on page 46) are not stored, because it is unpredictable how many data are produced with scatterplots and frame updates. To spare disk space these outputs are not kept in the device independent format.

The segment storage allows the output of all stored graphical data on any other device, especially a printout on plotter; e.g:

```
    PLOT PICTURE sys$ln03_c ln03 2
```

The actual picture is printed on plotter queue "SYS$LN03_C". The plotter is a LN03–PLUS laser printer and 2 copies are printed.

If there are some scatter frames on the screen they remain empty on the plot, if it is produced with the "PLOT PICTURE" command. To plot the scatter data you have to allocate a plotter or a metafile to save all graphical outputs (see chapter 3 on page 11) on file. After the deallocation of the plotter the produced files can be printed by:

```
    PLOT PLOTFILE file queue copies
```

The plotfile "file" (wildcards are supported) is plotted on "queue" in "copies" versions. Metafiles are printed by:

```
    PLOT METAFILE file queue type
```

The "file" is interpreted and converted in device dependent sequences of a device of the specified "type" and printed out on the specified "queue". The metafile can be sent to the IBM for plotting:

```
    PLOT METAFILE file IBM::   RP01  ! output on RP01: Large Benson
    PLOT METAFILE file IBM::   RP02  ! output on RP02: Small Benson
    PLOT METAFILE file IBM::   VP01  ! output on VP01: Vector plotter
```

8.8 Setting and Displaying Window Conditions

Condition windows are set with the REPLACE CONDITION WINDOW and SET CONDITION WINDOW commands. The difference between REPLACE and SET is that the REPLACE command provides cursor inputs and is therefore available in the GOOSY Display . The SET command requires the window limits and is executed in the GOOSY Database Manager.

A window condition can be multidimensional, therefore it is necessary to specify the dimension in which the condition limits should be set:

```
REPLACE CONDITION WINDOW condition limits 3   ! set limits in dimension 3
REPLACE CONDITION WINDOW condition limits 5:7 ! set dimension 5 to 7 with
                                              ! the specified limits
REPLACE CONDITION WINDOW condition limits *   ! set all dimensions
```

These examples are identical with the SET CONDITON WINDOW command! The specification of the limits is optional in the REPLACE command. If they are omitted the cursor appears and a graphical input is possible. Furthermore it is possible that the limits in one or several members of a condition array should be set:

```
REPLACE CONDITION WINDOW cond(1)   DIM=3   ! set COND(1) in dimension 3
REPLACE CONDITION WINDOW cond(3:5) DIM=5:7 ! set COND(3) to COND(5)
                                           ! in dimension 5 to 7
REPLACE CONDITION WINDOW cond(*)   DIM=*   ! set all members of COND
                                           ! in all dimensions
```

In the examples above the cursor appears and the window limits have to be specified graphically. These inputs are possible in the SET command, too, with the restriction that the limits are then required. In the REPLACE command the input of the window limits in calibrated units is supported by:

```
REPLACE CONDITION WINDOW cond(1) DIM=3/CALIB
```

If the condition windows should be set in two dimensional spectra, it is possible to specify the axis at which the graphical input should occur:

```
REPLACE CONDITION WINDOW cond(1:5) DIM=*/YAXIS
REPLACE CONDITION WINDOW cond(1:5) DIM=*/XAXIS
```

The verification of the condition windows is possible with the DISPLAY CONDITION command. Dimension "5" of the condition window "condition" in frame "2" is displayed by:

```
DISPLAY CONDITION condition FRAME=2 DIMENSION=5
```

Further input possibilities are:

```
    DISPLAY CONDITION cond(1:5) FRAME=1 DIM=3:5
    DISPLAY CONDITION cond(*) FRAME=1 DIM=3:5
    DISPLAY CONDITION cond(*) FRAME=1 DIM=*
    DISPLAY CONDITION cond(*) FRAME=2 DIM=* /DISTRIBUTE
```

The "/DISTRIBUTE" switch distributes the single members of a condition array to subsequent frames, starting at the specified frame. In the above example COND(1) is shown in frame 2, COND(2) in frame 3 etc. Furthermore the outputs can be directed to the y–axis by:

```
    DISPLAY CONDITION cond(1:5) FRAME=1 DIM=3:5 /YAXIS
    DISPLAY CONDITION cond(*) FRAME=2 DIM=* /DISTRIBUTE/YAXIS
```

8.9 Setting and Displaying Polygons

Polygons are set with the `REPLACE POLYGON` command. The polygon data can be specified in a list of x- and y-values:

```
  REPLACE POLYGON polygon XPOINTS=list YPOINTS=list
```

The number of x and y values have to be the same, if specified. If no points are specified a graphical polygon editor is envoked, which allows to delete, insert, or append new points in the polygon. The polygon must be displayed in that case. The following help appears if the polygon editor is activated:

```
    GOOSY Polyline editor is active
    To append or overstrike point press  : SPACE
    To stop editor input press           : RETURN
    To skip one point backward press     : B or b
    To skip one point forward press      : F or f
    To delete current point press        : D or d
    To delete backspaced point press     : <BS>
    To toggle insert mode press          : I or i
```

The insert mode is indicated at the right bottom on the screen. These inputs occur at the graphical input device. A polygon, which has been replaced earlier, can be modified with this editor. In that case the polygon points have to be displayed in an existing frame, which must be specified:

```
  DISPLAY POLYGON polygon 3   ! show existing polygon points in frame 3
  REPLACE POLYGON polygon 3   ! Edit polygon in frame 3
```

Points in an existing polygon are deleted and can be set again by:

```
  REPLACE POLYGON polygon /DELETE
```

If a polygon is specified which does not exist, it will be created. The number of polygon points can be increased at any time, the size of the polygon data element is increased if necessary.
The correctness of the polygon data can be controlled by:

```
DISPLAY POLYGON polygon frame       ! the contour of the polygon is shown
DISPLAY POLYGON polygon frame /FILL ! the interior of the polygon is filled
```

8.10 Projection of Two Dimensional Spectra

The GOOSY Display provides the possibility to project two dimensional source spectra to any dimension. Furthermore it is possible to specify a projection window to restrict the range of the source spectrum, which should be projected to the specified dimension. The result is kept in a specified target spectrum. If the target does not exist, it will be created with the same attributes as found in the projection dimension of the source spectrum. The syntax of the command is:

```
PROJECT source target window dimension
PROJECT source target (100,400) 2 ! Project window at x-axis 100,400
                                  ! Project to the y-axis.
PROJECT source target   *  1      ! Full y-range of source is projected
                                  ! to x-axis.
```

If the window limits are omitted the cursor appears for a graphical input.

If the target spectrum exists arithmetic operations with the target and the result of the projection are supported:

```
PROJECT source target DIM=2 /ADD   ! Add projection to target
PROJECT source target DIM=2 /SUB   ! Subtract projection from target
PROJECT source target DIM=2 /CLEAR ! Clear target spectrum first
```

Beside the specification of a simple projection window it is possible to specify a polygon, which should be used to define a projection region:

```
PROJECT source target WINDOW=polygon /POLYGON
```

The switch "/POLYGON" suggests that the specified window is not interpreted as a pair of limits, but as the name of a polygon! All spectrum bins for which the middle of the bin is inside the polygon are taken into account during the projection.

8.11 Fitting Spectra

In the GOOSY Display it is possible to fit one dimensional spectra by polynomial and gaussian peaks. As default the data displayed in the specified frames are used for the fit. A polynomial of an arbitrary order is fitted to your data by:

```
FIT SPECTRUM 2 5 /BACKGROUND
```

The data of the spectrum in frame 2 are used and a polynomial of a power of 5 is fitted to the data. The cursor appears to select up to 10 data ranges taken into account for the fit. Gaussian peaks are fitted by:

```
FIT SPECTRUM 1 /GAUSS
```

For each peak start values for the peak–position as well as the peak–width have to be specified by cursor input. In many cases the peaks have similiar peak–widhts; then it is possible to specify the width just once

```
FIT SPECTRUM 1 /GAUSS/SAMEWIDTH
```

In both examples the data of the spectrum in frame 1 are used!

Furthermore it is possible to combine gaussian peaks with a polynomial background:

```
FIT SPECTRUM 2 5 /BACKGROUND/GAUSS/SAMEWIDTH/SHOW
```

The "/SHOW" switch effects that the results of all iterations are shown. As default the fit uses the whole data as displayed actually on the screen, but the data region can be expanded or enclosed by specifying a data window:

```
FIT SPECTRUM 2 5 100,600/BACKGROUND/GAUSS
FIT SPECTRUM 2 5 ?/BACKGROUND/GAUSS
```

In the first example the data between the channels 100 and 600 are used for the fit. In the second example the data region to be taken into account during the fit is specified via cursor input.

The data are normally wheightened by their statistical errors. Fitting gaussian peaks this yields in an adjustment to the peak flanks. For data with bad statistics or none gaussian peaks the result may be unsatisfying. Better results are obtained without error weightening:

```
FIT SPECTRUM 2 5 /BACKGROUND/GAUSS/NOERROR
```

8.12 Defining the Frame Setup

The GOOSY Display provides the possibility to set some parameters, which manipulate the layout of the frames. The number of tics at the axis can be increased or decreased, the text fonts can be changed, the size of the spectra in the frames can be increased, the number of channels, which should be displayed for two dimensional spectra is variable, and you can select weather isometric plots will display a grid or not:

```
DEFINE FRAME SETUP tic_number text_font linewidth xyratio channels
                   /[NO]GRID /[NO]INFO
```

The switch "/NOINFO" yields to a larger representation of the spectra, but in that mode no space is available to display detailed spectrum informations. Nice pictures, for any kind of representation, can be produced by increasing the linewidth of all polylines, or by changing the format of the whole picture by the "xyratio".

Chapter 9

Different Display Versions

The GOOSY Display can be activated as a normal image or as a GOOSY process:

```
$ MDISP                           ! standalone display
$ GOOSY CREATE PROCESS DISP $DSP ! creates a GOOSY process
```

Scatterplots and frame updates are not possible if the display is activated as a normal image. The scatterplot and the update are based on the communication mechanism provided by the normal GOOSY environment, which is available only for a GOOSY process.

For scatterplots and frame updates activate the GOOSY Display as a GOOSY process.

As mentioned above the GOOSY Display saves most of the graphical data in a separate workstation independent storage (see page 47). The management of this storage consumes CPU–time. Therefore a display version with a deactivated segment storage is available. This version needs about 60% of the CPU–time of the standard display version. Using the fast display version the functionality of the GOOSY Display is reduced:

1. The "PLOT PICTURE" is not available.
2. The "SAVE DISPLAY" command does not work.
3. The "REFRESH" command works only as

   ```
   REFRESH */UPDATE
   ```

4. The "EXPAND" command works properly only if devices with a local deletion facility are allocated. If e.g. a plotter is active the screen is cleared and the whole picture will be redrawn.

The two display versions are activated by:

```
    SET DISPLAY MODE/STANDARD  ! standard version
    SET DISPLAY MODE/FAST      ! fast display version
```

At any time it is possible to switch from one version to the other. If you work with the fast version and you decide to produce plots, activate the standard version, display a new picture, and then the `PLOT PICTURE` command will work!

The fast version should be used if you run GOOSY on a MICRO–VAX or a VAX 750. On the VAX 8600 an effect is seen if the CPU is occupied by more than 50%.

Display Glossary

Calibration Calibrations are tables describing the dependence between the original spectrum units and a calibrated units. Each spectrum can be connected to a calibration and a display of the spectrum in calibrated units is possible.

CONDITION In contrast to SATAN, GOOSY conditions are independent of spectra. Besides the multi window conditions which are similar to SATAN analyzer conditions, GOOSY provides window-, pattern-, composed-, polygon- and userfunction-conditions. Each condition has counters associated for true/false statistics. Conditions can be executed in a Dynamic List or by the $COND macro in an analysis routine. Each condition can be used as filter for spectrum accumulation or scatter plots.

CONNECT A calibration can be connected to any number of spectra with the GOOSY command `CALIBRATE SPECTRUM`.

Device All graphical outputs occur at ouput devices; interactive terminals or plotters. In GOOSY up to 10 different devices can be handled at once.

Data Base A Data Base is located in a file and has a Data Base name. It is recommended to use the same name for the file and the Data Base. The file type should be .SEC. A logical name may be defined for the Data Base name. To activate a Data Base it must be mounted. It is dismounted during a system shutdown or by command. If a Data Base runs out of space, it can presently NOT be expanded.

Data Base Directory Similar to a VMS disk, GOOSY Data Bases are organized in Directories. They must be created.

Data Base Pool The storage region of a Data Base is splitted in Pools. All Data Elements are stored in Pools. A Pool can be accessed by a program with READ ONLY protection or with READ/WRITE protection. Pools must be created. They are automatically expanded if necessary, up to the space available in a Data Base.

Data Element A Data Element is allocated in a Data Base Pool. Its name is kept in a Directory. Data Elements can be of atomic Types (scalars or arrays), or of the structure Type (PL/1 structures). Besides the data structure a Data Element can be indexed (one or two dimensional). Such Data Elements are called name arrays. Each name array member has its own data and Directory entry.

Data Element Type GOOSY Data Elements can be PL/1 structures. The structure declarations must be in a file or text library module. They are used to create a Data Element Type in the Data Base and can be included in a program to access the Data Element.

Dynamic List A Dynamic List has several Entries, each specifying an action like condition check or spectrum accumulation. It is executed for each event in the analysis program. The Entries are added or removed by commands even without stopping the analysis.

Dynamic List Entry An Entry in a Dynamic List keeps all information to execute an action. For example, an accumulation Entry contains the spectrum name, an object and optional a condition and an increment parameter.

Global modes are user defined display defaults. For spectra and pictures different parameter sets exist.

Metafile The metafile is a device independent plotfile. This metafile can be redisplayed at any interactive ot non–interactive output device.

Picture A Picture is a complex display. A picture is a set of up to 64 frames with spectra and/or scatterplots. Once created and specified they remain in a Data Base independent of programs. They are displayed by `DISPLAY PICTURE` command. Pictures are composed of frames.

Picture Frame Each frame is a coordinate system for a spectrum or scatter plot. Up to 64 different frames may inserted to a picture.

Scatter Plot The GOOSY display component can display any pairs of Data Element members event by event in scatter plot mode (live mode). Several scatter plots can be displayed on one screen (pictures). Scatter plots are executed in Dynamic Lists and may be filtered by conditions.

Spectrum A GOOSY spectrum differs from a SATAN analyzer in that there are no windows or conditions associated. A spectrum can be filled in a Dynamic List Entry or in an analysis routine by macro $ACCU. GOOSY distinguishes between `DIGITAL` spectra for the accumulation of integer data and `ANALOG` spectra to accumulate float variables.

Temporary modes are the display modes specified in the last `DISPLAY SPECTRUM` or `DISPLAY PICTURE` command.

Appendix A

Summary of GOOSY–Display Commands

ALLOCATE DEVICE name type xsize ysize
/[NO]MAIN
Allocate a graphical device

ATTACH BASE base node
/READ
Attach data base.

CLEAR DEVICE –
Clear all active workstations

DEALLOCATE DEVICE device

Deallocate a graphical device

DEFINE DISPLAY HEADER string
Define display header line.

DEFINE DISPLAY PICTURE update refresh
/LIN/LOG /SQRT [SCALE]
/XLIN/XLOG/XSQRT [=X_SCALE]
/YLIN/YLOG/YSQRT [=Y_SCALE]
/ZLIN/ZLOG/ZSQRT [=Z_SCALE]
/[NO]ACTIVE
/[NO]ERROR
/[NO]WINDOW
/[NO]LIFE
/[NO]ROTATE

/[NO]SMOOTH
/[NO]LETTER
/[NO]NUMBER
/NOCAL/CALAX/CALSPEC [=CALIB]
/[NO]CHANNELS
/FULL/LAST/ACTUAL [=RANGE]
/AUTOSCALE/SCALE [=SCALE_RANGE]
/HISTO/VECTOR/MARKER [=STYLE]
/CONTOUR/ISO/CLUSTER/SCATTER
Set Spectrum parameter

DEFINE DISPLAY SPECTRUM limits scalim update refresh
/LIN/LOG /SQRT [SCALE]
/XLIN/XLOG/XSQRT [=X_SCALE]
/YLIN/YLOG/YSQRT [=Y_SCALE]
/ZLIN/ZLOG/ZSQRT [=Z_SCALE]
/[NO]ACTIVE
/[NO]ERROR
/[NO]WINDOW
/[NO]LIFE
/[NO]ROTATE
/[NO]SMOOTH
/[NO]LETTER
/[NO]NUMBER
/NOCAL/CALAX/CALSPEC [=CALIB]
/[NO]CHANNELS
/FULL/LAST/ACTUAL [=RANGE]
/AUTOSCALE/SCALE [=SCALE_RANGE]
/HISTO/VECTOR/MARKER [=STYLE]
/CONTOUR/ISO/CLUSTER/SCATTER
Set Spectrum parameter

DEFINE FRAME SETUP tic_number text_font linewidth xyratio channels
/[NO]GRID
/[NO]TICOUTSIDE
/[NO]INFO [=TYPE]
Define the design of picture frames

DETACH BASE base node
Detach data base.

DETACH DISPLAY base

Detach display data base.

DISPLAY CALIBRATION name cal_dir base node
Display calibration table.

DISPLAY CONDITION condition frame dimension
cond_dir base node
/CHAN/CALIB [=CALIBR]
/DISTRIBUTE
/XAXIS/YAXIS [=AXIS]
Display window condition limits

DISPLAY GRAPH frame file module image
Display user graphics.

DISPLAY METAFILE file directory
Read screen image from file and display it

DISPLAY PICTURE picture condition object dyn_scat
binfactor update refresh
binfactor update refresh
node base pic_dir dyn_dir cond_dir
buffer_size
/LIN /LOG /SQRT [=SCALE]
/XLIN /XLOG /XSQRT [=X_SCALE]
/YLIN /YLOG /YSQRT [=Y_SCALE]
/ZLIN /ZLOG /ZSQRT [=Z_SCALE]
/[NO]WINDOW
/[NO]LIFE
/[NO]ROTATE
/[NO]SMOOTH
/[NO]LETTER
/[NO]NUMBER
/NOCAL/CALAX/CALSPEC [=CALIB]
/[NO]CHANNELS
/SPECIFIED/FULL/LAST/ACTUAL [=RANGE]
/AUTOSCALE/SCALE [=SCALE_RANGE]
/HISTO/VECTOR/MARKER/CONTOUR-
/POINT/ISO/CLUSTER/SCATTER [=STYLE]
/TEMP /GLOBAL [=MODE]
/NOSCATTER
/[NO]ERROR
Display screen image as described by picture

DISPLAY POINT point frame
/LOOP
[CALIBR=]/CHAN/CALIB
Mark point on screen and get channel contents.

DISPLAY POLYGON polygon frame poly_dir base node
/FILL
Display polygon points.

DISPLAY SCATTER xparam yparam limits condition
object dyn_scat xletter yletter
node base par_dir dyn_dir
cond_dir buffer_size
/LAST /ACTUAL [=RANGE]
/TEMP /GLOBAL [=MODE]
/NOSCATTER
Display single scatter frame for two parameters.

DISPLAY SPECTRUM spectrum limits scalim
binfactor update refresh cuts theta phi
node base spec_dir
/LIN /LOG /SQRT [=SCALE]
/XLIN /XLOG /XSQRT [=XSCALE]
/YLIN /YLOG /YSQRT [=YSCALE]
/ZLIN /ZLOG /ZSQRT [=ZSCALE]
/[NO]WINDOW
/[NO]LIFE
/[NO]ROTATE
/[NO]SMOOTH
/[NO]LETTER
/[NO]NUMBER
/NOCAL/CALAX/CALSPEC [=CALIB]
/[NO]CHANNEL
/FULL/LAST/ACTUAL [=RANGE]
/AUTOSCALE/SCALE [=SCALE_RANGE]
/HISTO/VECTOR/MARKER/CONTOUR-
/POINT/ISO/CLUSTER/SCATTER [=STYLE]
/TEMP /[NO]GLOBAL [=MODE]
/[NO]ERROR
Display spectrum using default picture

DISPLAY TEXT text frame xposition yposition font size
/CENTER /LEFT /RIGHT [=LOCATE]

```
                              /ABSOLUTE /RELATIVE [=UNIT]
                 Display text into box specified by cursor

EXPAND           limits frame
                        /CHAN/CALIB [=CALIBR]
                        /LOG/LIN/SQRT [=SCALE]
                        /XLOG/XLIN/XSQRT [=X_SCALE]
                        /YLOG/YLIN/YSQRT [=Y_SCALE]
                 Expand spectrum or scatterplot within specified window.

FIT SPECTRUM      frame poly window iter file
                   /[NO]OUTPUT
                   /[NO]APPEND
                   /BACKGROUND
                   /GAUSS
                   /SAMEWIDTH
                   /SHOW
                   /[NO]ZERO
                   /[NO]MARK
                   /NOERROR /STATISTICAL [=ERROR]
                 Fit spectrum with polynom and/or with gaussian peaks

INTEGRATE        window frame file
                     /[NO]OUTPUT
                     /[NO]APPEND
                     /CHAN/CALIB [=CALIBR]
                     /LOOP
                 Integrate specified window

OVERLAY          spectrum xpara ypara binfactor
                         trans=(xfactor,xoffset,yfactor,yoffset)
                         frame node base spec_dir par_dir
                       /ADJUST
                       /SAVE
                       /[NO]ERROR
                       /HISTO/VECTOR/MARKER - =[STYLE]
                       /ISO/CLUSTER/CONTOUR/SCATTER
                 Add spectrum to frame

PLOT METAFILE      file type command queue copies font
                 Plot a metafile.

PLOT PICTURE      type command queue copies font file
                              /[NO]FLAG
```

/[NO]PRINT
Send the current active picture to a plotter

PLOT PLOTFILE file command queue copies
/[NO]DELETE
/[NO]FLAG
Plot device specific plotfile

PRINT command printer form file
/DELETE
Plot device specific plotfile

PROJECT spectrum target window dimension node base spec_dir
/ADD/SUB/CLEAR [=ACTION]
/POLYGON
Project window in 2-dim spectrum

REFRESH frame
/[NO]UPDATE
/[NO]WINDOW
/[NO]OVER
Refresh picture as displayed

REPLACE CONDITION WINDOW condition limits dimension cond_dir base node
/CHANN/CALIBR [=CALIBR]
/XAXIS/YAXIS [=AXIS]
Set or replace window condition by cursor

REPLACE POLYGON polygon frame xpoints ypoints
poly_dir base node poly_pool
/DELETE/MODIFY [=MODE]
Set points in polygon

SAVE DISPLAY file directory
Save displayed picture in a metafile.

SET CALIBRATION FIXED name unit start step input calib uncalib parameters
polynom module image cal_dir base node
/[NO]FILE
/FIT/MODULE/PARAMETER/PROMPT/TABLE [=ACTION]
Set table for a fixed-type calibration.

SET CALIBRATION FLOAT name unit input module image
cal_dir base node
/[NO]FILE
Set table for a float-type calibrations.

SET CALIBRATION LINEAR name unit input parameters
calib uncalib cal_dir base node
/FILE
/FIT/PROMPT/PARAMETER [=/ACTION]
Set parameters for a linear calibration.

SET DEVICE COLOR name index r g b
/UPDATE
Allocate a graphical device

SET DISPLAY MODE –
/STANDARD/FAST [=VERSION]
Select display version.

SHOW DEVICES SHOW DEVICE
Show all allocated user devices

SHOW DISPLAY GLOBALS –
/SPECTRUM
/PICTURE
Show global display parameter.

SHOW MAPPING base area
Show mapping of a Data Base

START SCATTER –
/SYNCHRONOUS /ASYNCHRONOUS [=/MODE]
Start scatter plots for actual picture

STOP SCATTER –
Stop scatter plots for actual picture

UPDATE FRAMES frames seconds
/REFRESH
Update frames in the actual picture on screen

ZOOM FRAME frame picture dyn_scat
pic_dir dyn_dir buffer_size base node
Zoom one frame of a picture.

Appendix B

GOOSY–Display Commands

ALLOCATE DEVICE

```
ALLOCATE DEVICE name type xsize ysize
    /[NO]MAIN
```

PURPOSE Allocate a graphical device

PARAMETERS

name Logical device name
use TT for current terminal.
default is DECW$DISPLAY

type Device type. Allowed devices:

MG600	Monterey MG600 and MG620
PECAD	Pecad terminal
TEK4014	Tektronix 4014
TEK41xx	Tektronix 4107, 4109, 4111, 4115
VWS,MV,GPX	MICRO-VAXII graphic Terminal with VWS
DECWINDOW,MOTIF	DEC window / motif
VT240	graphic VT240 and REGIS terminals
VT340	graphic VT340 and REGIS terminals
LN03	LN03-PLUS laser printer file (.LN3)
SIXEL	sixel output file (.SIX)
POST	Postscript file (.PS)
COLOR	Color Postscript file (.PS)
LJ250	Inkjet file (.LJ250)
HP7550A4	HP7550 pen plotter, DINA4 sheets (.HP4)
HP7550A3	HP7550 pen plotter, DINA3 sheets (.HP3)
METAOUT	Metafile output (.MET) MOTIF

xsize Window x-size in Meter (VWS)

	DECwindow	Size in units of screen. i.e. 0.9
ysize	Window y-size in Meter (VWS)	
	DECwindow	Size in units of screen. i.e. 0.9
	For DECwindows/motif the size can be changed using the cursor!	
/[NO] MAIN	Allocate the specified device as the GOOSY-main device.	
Caller	MDISP,MGOODISP,D$DSPCM	
Author	W. Spreng	

Example

1.) ALLOCATE DEVICE txnn
txnn is allocated as MONT600
2.) ALLOCATE DEVICE txnn TEK4014/MAIN
txnn is allocated as Tektronix 4014 and as a GOOSY-main device.
3.) ALLOCATE DEVICE plotter ln03
A plotfile PLOTTER.LN3 will be generated which could be printed on any LN03-Laser printer by the GOOSY command PLOT PICTURE.
4.) ALLOCATE DEVICE scatter.met metaout
generates the metafile SCATTER.MET
5.) ALLOCATE DEVICE goosy VWS
On a MICRO-VAX II graphic Terminal under VWS a Window with the name "goosy" is created and the graphical output is sent to that window
6.) DECwindow:

```
Create a virtual display by (/NODE is optional):
 $ SET DISPLAY/CREATE/NODE=node::0.0
 $ SHO DISPLAY
The SHOW command outputs the name of the virtual
display, i.e. WSA8:
GOOSY> ALLOCATE DEVICE WSA8: DECW .9 .9
```

Remarks

File name	D$ALLOC.PPL
Created by	GOO$DISP:D$DSPCM.PPL

Description

CALLING	STS=D$ALLOC(CV_name,CV_type,r_xsize,R_ysize,CV_main)
COMMAND	ALLOCATE DEVICE name type xsize ysize /[NO]MAIN

NAME

Routine par. Input CHAR(*) VAR

Command arg. String required, def=DECW$DISPLAY

Logical or physical device name. For a graphical Terminal it is the VMS-Terminal address; e.g.:

TXA6	for a terminal connected directly to one VAX.
LTA999	for a terminal connected to a LAT server.
TT	for the terminal of the session.

For spooled devices (all supported plotters) you can specify a file name with or without a file type.

For Metafiles a file name or a logical name which refers to a file is required.

For MICRO-VAX II and GPX devices under VWS the device name is arbitrary, it is displayed at the generated graphic window.

For DECwindow a virtual display must be created by DCL command SET DISP/CREATE/NODE=node::0.0 where /NODE= is optional (remote display). The DCL command SHO DISP then outputs the virtual display device name WSAnn:

Normally one can use DECW$DISPLAY witch is the default.

TYPE

Routine par. Input CHAR(*) VAR

Command arg. String required default=MOTIF

The type of the device to be allocated has to be specified with this parameter. The following device types are supported:

MG600	Monterey MG600 and MG620
PECAD	Pecad terminal

TEK4014	Tektronix 4014
TEK41xx	Tektronix 4107, 4109, 4111, 4115
VWS,MV,GPX	MICRO-VAXII graphic Terminal with VWS
DECWINDOW,MOTIF	DEC window
LN03	LN03-PLUS laser printer
SIXEL	sixel output
POST	Postscript output format
COLOR	Postscript output format
HP7550A4	HP7550 pen plotter, DINA4 sheets
HP7550A3	HP7550 pen plotter, DINA3 sheets
LJ250	Inkjet output format
VT240	graphic VT240 and REGIS terminals
VT340	graphic VT340 and REGIS terminals
METAOUT	Metafile output

For all supported plotters the plot-file 'name' is generated which could be plotted out later. If no file-type is specified the default plotfile extensions are:

LN03 generates a plotfile name.LN3
SIXEL generates a plotfile name.SIX
POST generates a plotfile name.PS
COLOR generates a plotfile name.PS
LJ250 generates a plotfile name.LJ250
HP7550A3 generates a plotfile name.HP3
HP7550A4 generates a plotfile name.HP4

You can plot them with the PLOT PLOTFILE command.

Metafiles could be sent to all supported plotters with the PLOT METAFILE command or they could be displayed later with DISPLAY METAFILE.

But in any case do not forget to give the DEALLOCATE DEVICE command first to close the metafile or plotfile properly!

XSIZE

Routine par.	Input BIN FLOAT(24)
Command arg.	FLOAT, default = 0.0

For window orietated devices, e.g. VAXstations, this parameter specifies the window size in X-direction. The size has to be specified in METERS, except for DECwindow, where it must be specified in units of the screen, i.e. 0.9 is 90% of screen. For DECwindows/motif the size can be changed using the cursor. The window origin is always at the lower left screen edge.

YSIZE

Routine par. Input BIN FLOAT(24)

Command arg. FLOAT, default = 0.0

For window orietated devices, e.g. VAXstations, this parameter specifies the window size in Y-direction. The size has to be specified in METERS, except for DECwindow, where it must be specified in units of the screen, i.e. 0.9 is 90% of screen. For DECwindows/motif the size can be changed using the cursor. The window origin is always at the lower left screen edge.

MAIN

Routine par. Input CHAR(*) VAR

Command arg. Set, default = /NOMAIN

Allocate the specified device as the GOOSY-main device. All displays are then adjusted to make optimal use of the hardware facilities of the specific graphical device. Futhermore the GOOSY-main device is used for all graphical inputs. If not specified the first device which is allocated is used as the GOOSY-main device.

For a detail description of the GOOSY-main device see the GOOSY Display manual.

Function

The specified device is allocated as a workstation in a GKS-session.

An entry in the device description table is created and all hardware facilities of the device are stored in this table. If /MAIN is specified the GOOSY-main device type is defined by the specified device. All pictures produced in the display process are then adjusted to make optimal use of the facilities of the main device. Futhermore this device is additionally used for all graphical inputs.

In principle each device can be allocated as a GOOSY main device, but do not wonder if you try this with a plotter and no graphical inputs are possible!

If /MAIN is not specified, the procedure checks if a main device is defined. If not the type of the first allocated output device is used as the default main device.

If a METAFILE should be allocated the device name has to be a file name or a logical name which references to a file.

ATTACH BASE

```
ATTACH BASE base node
   /READ
```

PURPOSE	Attach data base.
PARAMETERS	
base	Data Base name required common default
node	optional node optional
/READ	Map readonly
Caller	M$DMCMD
Author	H.G.Essel
File name	M$AATDB.PPL
EXAMPLE	ATT BASE mybase

Remarks

REMARKS	-

Description

CALLING	STS=M$AATDB(CV_BASE,CV_NODE,I_READ)
ARGUMENTS	
CV_BASE	Data Base name CHAR(*) VAR
CV_NODE	optional node CHAR(*) VAR

I_READ	Map readonly BIN FIXED(15)
FUNCTION	Map total bata base.
REMARKS	Module is an action routine.
EXAMPLE	-

CLEAR DEVICE

CLEAR DEVICE

PURPOSE	Clear all active workstations
Caller	MDISP,MGOODISP
Author	W. Spreng

Description

CALLING	STS=D$CLRWK;
COMMAND	CLEAR DEVICE

Function

This procedure clears all active workstations defined during a GKS-session.

All the following actions are executed:

- the display surface is cleared
- all segments are deleted from the workstation state list
- all actions pending on the workstation are updated, e.g. changes of the workstation window, workstation viewport etc.

DEALLOCATE DEVICE

DEALLOCATE DEVICE device

PURPOSE	Deallocate a graphical device
PARAMETERS	
device	Logical device name or * for all devices. TT for login terminal
Caller	MDISP,MGOODISP,D$DSPCM
Action rout.	D$DEALL
Author	W. SPRENG

Examples

1. DEALLOCATE DEVICE txnn
   Device txnn will be freed
2. DEALLOCATE DEVICE *
   All allocated devices are freed.

Remarks

File name	D$DEALL.PPL
created by	GOO$DISP:D$DSPCM.PPL

Description

CALLING	STS=D$DEALL(CV_device)
COMMAND	DEALLOCATE DEVICE device

DEVICE

Routine arg.	CHAR(*) VAR
Command par.	String required Logical or physical device name or * for all devices. For plotter use the file name of the generated plotfile!

Function

Deallocate graphical devices. The following actions are performed:
- The device is disconnected from the GKS-session
- The entry in the main device table is deleted
- If there are no devices of the same type allocated, the device description table is deleted
- If the specified device is the GOOSY-main device all actions will be performed, but additionally it will be checked if another device of the same type exists which could be used as GOOSY-main device. If this is not the case a device of the category INPUT/OUTPUT will be searched and will be allocated as the GOOSY-main device for all following actions.

If it is not possible to find an other main main device the request to deallocate the device will be canceled. To deallocate this device you have to deallocate all devices or you must allocate a new device which could be used as a GOOSY-main device! This is necessary to be sure that always a main device exists.

DEFINE DISPLAY HEADER

DEFINE DISPLAY HEADER string

PURPOSE	Define display header line.
PARAMETERS	
string	String to be displayed on top of picture. The string is defined as user mode logical name in the process table. It remains valid until program exit or redefinition.
Caller	MDISP,MGOODISP,D$DSPCM
Action rout.	D$SDHD
Author	H.G.Essel

Examples

DEF SP HEAD "This is picture xx"

Remarks

File name	D$SDHD.PPL
created by	GOO$DISP:D$DSPCM.PPL

Description

CALLING	STS=D$SDHD(CV_string)
COMMAND	DEFINE DISPLAY HEADER string

STRING

Routine arg.	CHAR(*) VAR
Command par.	String required String to be displayed on top of picture.

Function

Define string to be displayed on top of picture. The string definition is valid as long as the program is up.

DEFINE DISPLAY PICTURE

```
DEFINE DISPLAY PICTURE update refresh
          /LIN/LOG /SQRT [SCALE]
          /XLIN/XLOG/XSQRT [=X_SCALE]
          /YLIN/YLOG/YSQRT [=Y_SCALE]
          /ZLIN/ZLOG/ZSQRT [=Z_SCALE]
          /[NO]ACTIVE
          /[NO]ERROR
          /[NO]WINDOW
          /[NO]LIFE
          /[NO]ROTATE
          /[NO]SMOOTH
          /[NO]LETTER
          /[NO]NUMBER
          /NOCAL/CALAX/CALSPEC [=CALIB]
          /[NO]CHANNELS
          /FULL/LAST/ACTUAL [=RANGE]
          /AUTOSCALE/SCALE [=SCALE_RANGE]
          /HISTO/VECTOR/MARKER [=STYLE]
          /CONTOUR/ISO/CLUSTER/SCATTER
```

PURPOSE Set Spectrum parameter

PARAMETERS

update Update time interval.

refresh Refresh time interval.
***************** not yet implemented *******

SCALE Scaling mode for Y- or Z-axis for one or two dimensional spectra.

/LIN	Display axis in a linear scale
/LOG	Display axis in a logarithmic scale
/SQRT	Display axis in a square root scale

X_SCALE	Scaling mode for X-axis	
	/XLIN	Display axis in a linear scale
	/XLOG	Display axis in a logarithmic scale
	/XSQRT	Display axis in a square root scale
Y_SCALE	Scaling mode for Y-axis	
	/YLIN	Display axis in a linear scale
	/YLOG	Display axis in a logarithmic scale
	/YSQRT	Display axis in a square root scale
Z_SCALE	Scaling mode for Z-axis	
	/ZLIN	Display axis in a linear scale
	/ZLOG	Display axis in a logarithmic scale
	/ZSQRT	Display axis in a square root scale
/[NO] WINDOW	Window switch ***************** not yet implemented *******	
/[NO] LIFE	Activate life display of spectra. ***************** not yet implemented *******	
/[NO] SMOOTH	Display the spectrum with smooth binning.	
/[NO] LETTER	Display lettering on axis	
/[NO] NUMBER	Display numbering on axis	
CALIB	Perform calibration	
	/CALAX	Calibrate axis.
	/CALSPEC	Calibrate spectrum data.
	/NOCAL	no calibration is performed.
/[NO] CHANNELS	Display channel numbers	
RANGE	Select display window	
	/FULL	Use full spectrum range.
	/LAST	Use last displayed range.
	/ACTUAL	Use actual displayed range.
SCALE_RANGE	Scaling method	

/AUTOSCALE	Autoscaling is performed
/SCALE	Scaling as in MODIFY FRAME command

STYLE Display mode of spectra. 1. One dimensional spectra

/HISTO	histograms are generated
/VECTOR	spectrum contents are connected with olylines
/MARKER	spectrum contents are signed with markers

2. Two dimensional spectra

/CONTOUR	Contour lines are displayed.
/CLUSTER	spectrum contents in cluster mode
/ISO	pseudo 3D isometric plot is generated.
/SCATTER	Simulate scatter plot data.

Caller	MDISP,MGOODISP,D$DSPCM
Author	W. Spreng

Example

DEFINE DISPLAY PICTURE /LOG/SMOOTH/VECTOR

The spectra are displayed with a logarithmic scaling axis (Y for one and Z for two dimensional spectra).The display bins are smoothed and the one dimensional spectra are displayed in VECTOR mode. If the two dimensional spectra should be displayed in CLUSTER mode specify additionally:

DEFINE DISPLAY PICTURE/CLUSTER

Remarks

File name	D$SPPAR.PPL
Created by	GOO$DISP:D$DSPCM.PPL

Description

CALLING STS=D$SPPAR(L_update,L_refresh,
CV_scale,CV_Xscale,CV_Yscale,CV_Zscale,I_active,
I_error,I_window,I_life,I_rotate,I_smooth,
CV_letter,CV_number,CV_calib,CV_channels,CV_range,
CV_scale_range,CV_style,B_mask)

COMMAND

```
DEFINE DISPLAY PICTURE update refresh
          /LIN/LOG /SQRT [SCALE]
          /XLIN/XLOG/XSQRT [=X_SCALE]
          /YLIN/YLOG/YSQRT [=Y_SCALE]
          /ZLIN/ZLOG/ZSQRT [=Z_SCALE]
          /[NO]ACTIVE
          /[NO]ERROR
          /[NO]WINDOW
          /[NO]LIFE
          /[NO]ROTATE
          /[NO]SMOOTH
          /[NO]LETTER
          /[NO]NUMBER
          /NOCAL/CALAX/CALSPEC [=CALIB]
          /[NO]CHANNELS
          /FULL/LAST/ACTUAL [=RANGE]
          /AUTOSCALE/SCALE [=SCALE_RANGE]
          /HISTO/VECTOR/MARKER [=STYLE]
          /CONTOUR/ISO/CLUSTER/SCATTER
```

UPDATE

Routine arg. Input BIN FIXED(31)

Command par. Integer replaceable

Update interval. If <= 0 , no update.

If > 0 every 'update' seconds the whole picture will be updated with the new channel contents of each spectrum.

REFRESH

Routine arg. Input BIN FIXED(31)

Command par. Integer replaceable

Refresh intervall. If <=0 , no refresh.

If > 0 all spectra in the picture will be deleted and redrawn every 'refresh' seconds. The scatter plots will be stored on the display and not deleted if a device with selective erase operation is used (e.g. Tektronix 4115) if not the whole spectrum is deleted and the collected data in scatter plots are lost. Therefore be careful using this parameter.

***************** not yet implemented *******

SCALE

Routine arg. Input CHAR(*) VAR

Command par. Set replaceable

Set the display mode on the scaling axis. You can select between three modes:

/LIN	Display the linear count rate.
/LOG	Display the spectrum contents in logarithmic mode.
/SQRT	Display the square root of the spectrum contents.

X_SCALE

Routine arg. Input CHAR(*) VAR

Command par. Set replaceable

Display mode of X-axis. You can select between three modes:

/XLIN	Display the axis linear.
/XLOG	Display the axis in logarithmic mode.
/XSQRT	Display the square root of axis.

Y_SCALE

Routine arg. Input CHAR(*) VAR

Command par. Set replaceable

Display mode of Y-axis. You can select between three modes:

/YLIN	Display the axis linear.
/YLOG	Display the axis in logarithmic mode.
/YSQRT	Display the square root of axis.

Z_SCALE

Routine arg. Input CHAR(*) VAR

Command par. Set replaceable

Display mode of Z-axis. You can select between three modes:

/ZLIN	Display the axis linear.
/ZLOG	Display the axis in logarithmic mode.
/ZSQRT	Display the square root of axis.

ACTIVE

Routine arg. Input BIN FIXED(15) valid values 0 or 1.

Command par. Switch replaceable negatable default=/ACTIVE

Activate or deactivate the global display parameter for spectra. Possible inputs are:

/ACTIVATE	activate display parameter. This is the default. The parameter can be deactivated any time by DEFINE DISPLAY SPECTRUM/NOACTIVE or for a single display by DISPLAY SPECTRUM/NOGLOBAL
/NOACTIVE	deactivate the display parameter. They can be activated later by DEFINE DISPLAY SPECTRUM /ACTIVE or for a single display by DISPLAY SPECTRUM/GLOBAL

ERROR

Routine arg. Input BIN FIXED(15) valid values 0 or 1.

Command par. Switch replaceable negatable

Display statistical errors at each channel of a one dimensional spectrum.

/ERROR	Statistical errors (the square root of the count rate) are displayed.
/NOERROR	No errors are displayed

WINDOW

Routine arg. Input BIN FIXED(15) valid values 0 or 1.

Command par. Switch replaceable negatable

*************** Not yet implemented **********

LIFE

Routine arg. Input BIN FIXED(15) valid values 0 or 1.

Command par. Switch replaceable negatable

*************** Not yet implemented **********

ROTATE

Routine arg. Input BIN FIXED(15) valid values 0 or 1.

Command par. Switch replaceable negatable

*************** Not yet implemented **********

SMOOTH

Routine arg. Input BIN FIXED(15) valid values 0 or 1.

Command par. Switch replaceable negatable

Display the spectrum with smooth binning, the mean values of the channel contents about the display binsize will be shown. The effect is that the spectrum looks smoother, but the displayed spectrum contents could be fractional numbers.

/SMOOTH	Display the spectra with smooth binning
/NOSMOOTH	The minimum and maximum contents of the display bins are shown.

LETTER

Routine arg.	Input CHAR(*) VAR
Command par.	Set replaceable

Activate or deactivate the lettering on the axis.

/LETTER	The lettering is displayed.
/NOLETTER	The lettering is not displayed.

NUMBER

Routine arg.	Input CHAR(*) VAR
Command par.	Set replaceable

Activate or deactivate the numbering on the axis.

/NUMBER	The numbering is displayed.
/NONUMBER	The numbering is not displayed.

CALIB

Routine arg.	Input CHAR(*) VAR
Command par.	Set replaceable

Display the spectrum in calibrated units. To do this the displayed spectra has to be connected to an existing calibration. Different calibration modes are available:

/CALAX	The axis is drawn in calibrated units and the spectrum in uncalibrated units. Therefore the distance between two subsequent tics varies.
/CALSPEC	The spectrum is drawn in calibrated units. Then the width of the displayed spectrum bins varies.
/NOCAL	No calibration is performed to the displayed spectra.

In any case the axis with uncalibrated units is displayed, too. To prevent this specify the /NOCHANNELS switch!

CHANNELS

Routine arg.	Input CHAR(*) VAR
Command par.	Set replaceable

Activate or deactivate the display of an axis with the original channel units.

/CHANNELS	Display an axis with original units.
/NOCHANNEL	The axis with original units is not displayed.

RANGE

Routine arg.	Input CHAR(*) VAR
Command par.	Set replaceable

Specify one of the available display windows:

/FULL	The full spectrum range is displayed.
/LAST	The last displayed range is used. This window is indicated be a "#".
/ACTUAL	Use the actual displayed range.

The specified window is used for the display of the spectra.

SCALE_RANGE

Routine arg.	Input CHAR(*) VAR
Command par.	Set default=/SCALE

Defines the range of the scaling axis.

/AUTOSCALE	Perform autoscaling
/SCALE	Use the scaling range defined in the MODIFY FRAME command.

STYLE

Routine arg.	Input CHAR(*) VAR
Command par.	Set replaceable

Defines the display style of the spectrum data. For one and two dimensional spectra different styles are implemented.

One dimensional spectra:

/HISTO	Histograms are generated
/VECTOR	The spectrum contents are connected by poly-lines.
/MARKER	The spectrum contents are indicated by markers.

For two dimensional spectra:

/CONTOUR	Contour lines are displayed.
/CLUSTER	The spectrum count rates are indicated by clusters of a variable size and colour.
/ISO	A 3D isometric plot is generated.
/SCATTER	Scatter plot data are simulated. In each the countrate is indicated by a number of points, randomly distributed in the bin.

Function

Define the global display parameters for the display of pictures. Subsequent definitions of global parameter are cumulative and do not distroy the earlier settings. Therfore at any time additional parameters can be set or existing ones can be modified.

By default they are activated after their definition. But the parameter can be deactivated at any time by DEFINE DISPLAY PICTURE/NOACTIVE

DEFINE DISPLAY SPECTRUM

```
DEFINE DISPLAY SPECTRUM limits scalim update refresh
          /LIN/LOG /SQRT [SCALE]
          /XLIN/XLOG/XSQRT [=X_SCALE]
          /YLIN/YLOG/YSQRT [=Y_SCALE]
          /ZLIN/ZLOG/ZSQRT [=Z_SCALE]
          /[NO]ACTIVE
          /[NO]ERROR
          /[NO]WINDOW
          /[NO]LIFE
          /[NO]ROTATE
          /[NO]SMOOTH
          /[NO]LETTER
          /[NO]NUMBER
          /NOCAL/CALAX/CALSPEC [=CALIB]
          /[NO]CHANNELS
          /FULL/LAST/ACTUAL [=RANGE]
          /AUTOSCALE/SCALE [=SCALE_RANGE]
          /HISTO/VECTOR/MARKER [=STYLE]
          /CONTOUR/ISO/CLUSTER/SCATTER
```

PURPOSE Set Spectrum parameter

PARAMETERS

limits	Display limits for spectrum
scalim	Limits for scaling axis
update	Update time interval.
refresh	Refresh time interval. **************** not yet implemented *******
SCALE	Scaling mode for Y- or Z-axis for one or two dimensional spectra.

/LIN	Display axis in a linear scale

	/LOG	Display axis in a logarithmic scale
	/SQRT	Display axis in a square root scale
X_SCALE	Scaling mode for X-axis	
	/XLIN	Display axis in a linear scale
	/XLOG	Display axis in a logarithmic scale
	/XSQRT	Display axis in a square root scale
Y_SCALE	Scaling mode for Y-axis	
	/YLIN	Display axis in a linear scale
	/YLOG	Display axis in a logarithmic scale
	/YSQRT	Display axis in a square root scale
Z_SCALE	Scaling mode for Z-axis	
	/ZLIN	Display axis in a linear scale
	/ZLOG	Display axis in a logarithmic scale
	/ZSQRT	Display axis in a square root scale
/[NO] WINDOW	Window switch ***************** not yet implemented *******	
/[NO] LIFE	Activate life display of spectra. ***************** not yet implemented *******	
/[NO] SMOOTH	Display the spectrum with smooth binning.	
/[NO] LETTER	Display lettering on axis	
/[NO] NUMBER	Display numbering on axis	
CALIB	Perform calibration	
	/CALAX	Calibrate axis.
	/CALSPEC	Calibrate spectrum data.
	/NOCAL	no calibration is performed.
/[NO] CHANNELS	Display channel numbers	
RANGE	Select display window	
	/FULL	Use full spectrum range.
	/LAST	use last displayed range.

	/ACTUAL	Use actual displayed range.
SCALE_RANGE	Scaling method	
	/AUTOSCALE	Autoscaling is performed
	/SCALE	Scaling as in MODIFY FRAME command
STYLE	Display mode of spectra. 1. One dimensional spectra	
	/HISTO	histograms are generated
	/VECTOR	spectrum contents are connected with olylines
	/MARKER	spectrum contents are signed with markers
	2. Two dimensional spectra	
	/CONTOUR	Contour lines are displayed.
	/CLUSTER	spectrum contents in cluster mode
	/ISO	pseudo 3D isometric plot is generated.
	/SCATTER	Scatter plot data are simulated.
Caller	MDISP,MGOODISP,D$DSPCM	
Author	W. Spreng	

Example

DEFINE DISPLAY SPECTRUM 5, /LOG/SMOOTH/VECTOR

The spectra are displayed with a lower window limit of 5. The scaling axis (Y for one and Z for two dimensional spectra) is displayed in a logaritmic scale. The display bins are smoothed on the one dimensional spectra are displayed in VECTOR mode. If the two dimensional spectra should be displayed in CLUSTER mode specify additionally:

DEFINE DISPLAY SPECTRUM /CLUSTER

Remarks

File name	D$SDPAR.PPL
Created by	GOO$DISP:D$DSPCM.PPL

Description

CALLING

```
STS=D$SDPAR(CV_limits,CV_scallim,
    L_update,L_refresh,
    CV_scale,CV_Xscale,CV_Yscale,CV_Zscale,I_active,
    I_error,I_window,I_life,I_rot,I_smooth,CV_letter,
    CV_number,CV_calib,CV_channels,CV_range,
    CV_scale_range,CV_style,B_mask)
```

COMMAND

```
DEFINE DISPLAY SPECTRUM limits scalim update refresh
             /LIN/LOG /SQRT [SCALE]
             /XLIN/XLOG/XSQRT [=X_SCALE]
             /YLIN/YLOG/YSQRT [=Y_SCALE]
             /ZLIN/ZLOG/ZSQRT [=Z_SCALE]
             /[NO]ACTIVE
             /[NO]ERROR
             /[NO]WINDOW
             /[NO]LIFE
             /[NO]ROTATE
             /[NO]SMOOTH
             /[NO]LETTER
             /[NO]NUMBER
             /NOCAL/CALAX/CALSPEC [=CALIB]
             /[NO]CHANNELS
             /FULL/LAST/ACTUAL [=RANGE]
             /AUTOSCALE/SCALE [=SCALE_RANGE]
             /HISTO/VECTOR/MARKER [=STYLE]
             /CONTOUR/ISO/CLUSTER/SCATTER
```

LIMITS

Routine arg. Input CHAR(*) VAR

Command par. String replaceable

Display limits for spectrum. The input of a one or two dimensional window is possible, e.g.:

(100,560) - for a one dimensional window

(100,500,200,400) - for a two dimensional window

Any lower and upper limits in the window specification can be skipt. The missings values are replaced by the limits of the displayed spectrum, e.g: (100,). The lower limit is 100. The upper limits are given by the upper limit of the spectrum, displayed with these window.

SCALIM

Routine arg.	Input CHAR(*) VAR
Command par.	String replaceable

Specifies limits on the scaling axis; for one dimensional spectra this is the y-axis, for two dimensional spectra the z-axis. The lower and upper limits has to be specified, e.g:

(-100,1000)

UPDATE

Routine arg.	Input BIN FIXED(31)
Command par.	Integer replaceable

Update interval. If <= 0 , no update. If > 0 every 'update' seconds the whole spectrum will be updated with the new channel contens of each spectrum.

REFRESH

Routine arg.	Input BIN FIXED(31)
Command par.	Integer replacable

Refresh intervall. If <=0 , no refresh.

If > 0 all spectra in the picture will be deleted and redrawn every 'refresh' seconds. The scatter plots will be stored on the display and not deleted if a device with selective erase operation is used (e.g. Tektronix 4115) if not the whole spectrum is deleted and the collected data in scatter plots are lost. Therefore be careful using this parameter.

***************** not yet implemented *******

SCALE

Routine arg.	Input CHAR(*) VAR
Command par.	Set replaceable

Set the display mode on the scaling axis. You can select between three modes:

/LIN	Display the linear count rate.

/LOG	Display the spectrum contents in logarithmic mode.
/SQRT	Display the square root of the spectrum contents.

X_SCALE

Routine arg. Input CHAR(*) VAR

Command par. Set replaceable

Display mode of X-axis. You can select between three modes:

/XLIN	Display the axis linear.
/XLOG	Display the axis in logarithmic mode.
/XSQRT	Display the square root of axis.

Y_SCALE

Routine arg. Input CHAR(*) VAR

Command par. Set replaceable

Display mode of Y-axis. You can select between three modes:

/YLIN	Display the axis linear.
/YLOG	Display the axis in logarithmic mode.
/YSQRT	Display the square root of axis.

Z_SCALE

Routine arg. Input CHAR(*) VAR

Command par. Set replaceable

Display mode of Z-axis. You can select between three modes:

/ZLIN	Display the axis linear.
/ZLOG	Display the axis in logarithmic mode.
/ZSQRT	Display the square root of axis.

ACTIVE

Routine arg. Input BIN FIXED(15) valid values 0 or 1.

Command par. Switch replaceable negatable default=/ACTIVE

Activate or deactivate the global display parameter for spectra. Possible inputs are:

/ACTIVATE	activate display parameter. This is the default. The parameter can be deactivated any time by DEFINE DISPLAY SPECTRUM/NOACTIVE or for a single display by DISPLAY SPECTRUM/NOGLOBAL
/NOACTIVE	deactivate the display paramter. They can be activated later by DEFINE DISPLAY SPECTRUM /ACTIVE or for a single display by DISPLAY SPECTRUM/GLOBAL

ERROR

Routine arg. Input BIN FIXED(15) valid values 0 or 1.

Command par. Switch replaceable negatable

Display statistical errors at each channel of a one dimensional spectrum.

/ERROR	Statistical errors (the square root of the count rate) are displayed.
/NOERROR	No errors are displayed

WINDOW

Routine arg. Input BIN FIXED(15) valid values 0 or 1.

Command par. Switch replaceable negatable

*************** Not yet implemented **********

LIFE

Routine arg.	Input BIN FIXED(15) valid values 0 or 1.
Command par.	Switch replaceable negatable

*************** Not yet implemented **********

ROTATE

Routine arg.	Input BIN FIXED(15) valid values 0 or 1.
Command par.	Switch replaceable negatable

*************** Not yet implemented **********

SMOOTH

Routine arg.	Input BIN FIXED(15) valid values 0 or 1.
Command par.	Switch replaceable negatable

Display the spectrum with smooth binning, that mean mean values of the channel contents about the display binsize willl be shown. The effect is that the spectrum looks smoother, but the displayed spectrum contents could be fractional numbers.

/SMOOTH	Display the spectra with smooth binning
/NOSMOOTH	The minimum and maximum contents of the display bins are shown.

LETTER

Routine arg.	Input CHAR(*) VAR
Command par.	Switch replaceable negatable

Activate or deactivate the lettering on the axis.

/LETTER	The lettering is displayed.
/NOLETTER	The lettering is not displayed.

NUMBER

Routine arg.	Input CHAR(*) VAR
Command par.	Switch replaceable negatable

Activate or deactivate the numbering on the axis.

/NUMBER	The numbering is displayed.
/NONUMBER	The numbering is not displayed.

CALIB

Routine arg.	Input CHAR(*) VAR
Command par.	Switch replaceable negatable

Display the spectrum in calibrated units. To do this the displayed spectra has to be connected to an existing calibration. Different calibration modes are available:

/CALAX	The axis is drawn in calibrated units and the spectrum in uncalibrated units. Therefore the distance between two subsequent tics varies.
/CALSPEC	The spectrum is drawn in calibrated units. Then the width of the displayed spectrum bins varies.
/NOCAL	No calibration is performed to the displayed spectra.

In any case the axis with uncalibrated units is displayed, too. To prevent this specify the /NOCHANNELS switch!

CHANNELS

Routine arg.	Input CHAR(*) VAR
Command par.	Set replaceable

Activate or deactivate the display of an axis with the original channel units.

/CHANNELS	Display an axis with original units.
/NOCHANNEL	The axis with original units is not not displayed.

RANGE

Routine arg. Input CHAR(*) VAR

Command par. Set replaceable

Specify on of the available display windows:

/FULL	The full spectrum range is displayed.
/LAST	The last displayed range is used. This window is indicated be a "#".
/ACTUAL	Use the actual displayed range.

The specified window is used for the display of the spectra.

SCALE_RANGE

Routine arg. Input CHAR(*) VAR

Command par. Set default=/SCALE

Defines the range of the scaling axis.

/AUTOSCALE	Perform autoscaling
/SCALE	Use the scaling range defined in the MODIFY FRAME command.

STYLE

Routine arg. Input CHAR(*) VAR

Command par. Set replacable

Defines the display style of the spectrum data. For one and two dimensional spectra different styles are implemented.

One dimensional spectra:

/HISTO	Histograms are generated
/VECTOR	The spectrum contents are connected by polylines.
/MARKER	The spectrum contents are indicated by markers.

For two dimensional spectra:

/CONTOUR	Contour lines are displayed.
/CLUSTER	The spectrum count rates are indicated by clusters of a variable size and colour.
/ISO	A 3D isometric plot is generated.
/SCATTER	Scatter plot data are simulated. In each the countrate is indicated by a number of points, randomly distributed in the bin.

Function

Define the global display parameters for the display of spectra. Subsequent definitions of global display parameter are cumulative and do not distroy the earlier settings. Therfore at any time additional parameters can be set or existing ones can be modified.

By default they are activated after their definition. But the parameter can be deactivated at any time by DEFINE DISPLAY SPECTRUM/NOACTIVE

DEFINE FRAME SETUP

```
DEFINE FRAME SETUP tic_number text_font linewidth xyratio channels
  /[NO]GRID
  /[NO]TICOUTSIDE
  /[NO]INFO [=TYPE]
```

PURPOSE	Define the design of picture frames
PARAMETERS	
tic_number	Number of tics on axis
text_font	Text font for all descriptions
linewidth	Factor to increase the width of all polylines.
xyratio	X/Y-ratio of the maximum used display surface.
channels	Number of channels which should be displayed for two dimensional spectra.
/[NO] GRID	If set the lines of isometric plots are draw parallel to the x- and y-axis. If not set only parallel to the x-axis.
/TICOUTSIDE	Draw tics at spectrum axis outside the frame.
/[NO] INFO	Frame type

/INFO	Frame with information
/NOINFO	Frame without any information

Caller	MDISP
Author	W. Spreng

Remarks

Created by	GOO$DISP:D$DSPCM.PPL
File name	GOO$DISP:

Examples

DEFINE FRAME SETUP 20
Draw 20 tic at the axis, if that is possible.
DEFINE FRAME SETUP FONT=-3
Change the font for all descriptions.
DEFINE FRAME SETUP CHANNELS=200 /NOGRID
Two dimensional spectra are shown with 200 channels and for isometric plots no grids are produced.
DEFINE FRAME SETUP /NOINFO
Draw frames with a large spectrum area, but without detail spectrum informations.

Description

CALLING

```
STS=D$DEFRA(L_tic_number,L_font,R_linewidth,
            R_xyratio, L_CHAN, I_grid, I_ticoutside,
            cv_type)
```

COMMAND

```
DEFINE FRAME SETUP tic_number text_font linewidth xyratio
channels
  /[NO]GRID
  /[NO]TICOUTSIDE
  /[NO]INFO [=TYPE]
```

TIC_NUMBER

Routine arg. Input BIN FIXED(31)

Command par. Integer replaceable default=10

Number of tics which should be drawn at the axis. The real number of tics can be different from this number, because it depends on the range of the axis and the size of the frames. But with this parameter you can increase or decrease the number of tics.

FONT

Routine arg. Input BIN FIXED(31)

Command par. Integer replaceable default=0

Font number to change text fonts defined in the GKS-bundle tables which are used in case of hardware character requirement.

NOTE That means, hardware font must be enabled, i.e.:
DEFINE FRAME SETUP 10 0 (is default after start)

GTS-GRAL GKS The following GKS-font numbers are supported:

0	Hardware text font
-1...-11	proportional fonts.
-51	thick proportional font
-101..-111	proportional italics
-151	thick proportional italics
-201..-211	mono spaced fonts
-251	thick mono spaced font
-301..-311	mono spaced italics
-351	thick mono spaced italics

Recommended is font -51.

DEC GKS The standard DEC software fonts are:

0	Hardware text font
-1,1	Standard ISO font (Thin, ugly).
-15	Thick Roman (But underscore is arrow)

Recommended is font -15.
With Postscript format you may use one of the following fonts:

-101..-104	Times (Roman, italic, Bold, Bold italic)
-105..-108	Helvet. (Roman,italic,bold,bold italic)
-109..-112	Courier (Roman,italic,bold,bold italic)
-114..-117	Lubalin (Roman,italic,bold,bold italic)
-118..-121	School (Roman,italic,bold,bold italic)
-122..-125	Av.Garde (Roman,italic,bold,bold italic)
-126..-129	Souvenir (Roman,italic,bold,bold italic)

LINEWIDTH

Routine arg. Input BIN FLOAT(24)

Command par. REAL replaceable default=0 min=0 max=10
Factor to increase the width of all polylines!

XYRATION

Routine arg. Input BIN FLOAT(24)

Command par. REAL replaceable default=0 min=0 max=10

XY-ratio of the maximum display surface used for the display of pictures. If set to 0 the default GOOSY display ratio is used (the whole screen). If you want to draw all pictures to be higher than wideeeee specify a ratio lower than 1.0, then the total height of the screen is used, but the width is reduced.

CHANNELS

Routine arg. Input BIN FIXED(31)

Command par. Integer replaceable default=100 min=50 max=500

Number of channels which should be displayed for two dimensional spectra.

GRID

Routine arg. Input BIN FIXED(15)

Command par. Switch replaceable default=/GRID

If set the lines of isometric plots are draw parallel to the x- and y-axis. If not set only parallel to the x-axis.

TICOUTSIDE

Routine arg. Input BIN FIXED(15)

Command par. Switch replaceable default=/NOTICOUTSIDE

The tics at the spectrum axis can be drawn inside or outside the spectrum frame. Defauld is inside.

TYPE

Routine arg. Input CHAR(*) VAR

Command par. Set replaceable default=/INFO

Specifies if all spectrum and scatterplot information should be displayed or if the data area should be increased and a minimum set of information should be displayed.

/INFO	Smaller spectra and large information field to display the spectrum header, integration and fit results etc.
/NOINFO	The spectra appear larger, but only the spectrum name is displayed, no place available for futher information.

Function

FUNCTION In the GOOSY-Display the user has the possibility to modify the default frame set-up. The number of tic-marks on the axis and the text font could be changed.

The maximum number of channels which should be displayed for two dimensional spectra could be defined. This is sometimes useful if the default display set-up of 100 channels is to rough to make details visible. But be careful to increase this parameter too much, the memory requirements and the CPU-time will be increased with the 2th power of this parameter.

If the two-dimensional isometric plots should be displayed as a grid specify /GRID!

Futhermore two different spectrum sizes are implemented. Small spectra with an information area large enough to display the spectrum header and all intgration or fit results(/INFO). Or large spectra with no information area (/NOINFO) are availible.

DETACH BASE

DETACH BASE base node

PURPOSE	Detach data base.
PARAMETERS	
base	Data Base name required common default
node	optional node optional
Caller	M$DMCMD
Author	H.G.Essel
File name	M$ADADB.PPL
EXAMPLE	ATT BASE mybase

Remarks

REMARKS	-

Description

CALLING	STS=M$ADADB(CV_BASE,CV_NODE)
ARGUMENTS	
CV_BASE	Data Base name CHAR(*) VAR
CV_NODE	optional node CHAR(*) VAR
FUNCTION	Map total bata base.

REMARKS Module is an action routine.

EXAMPLE -

DETACH DISPLAY

DETACH DISPLAY base

PURPOSE	Detach display data base.
PARAMETERS	
base	Data base name or * for all bases.
Caller	MDISP,MGOODISP,D$DSPCM
Action rout.	D$DEALL
Author	W. SPRENG

Examples

DETACH DISPLAY db
 Detach data base DB form display process.
 DETACH DISPLAY *
 Detach all data bases located by the display process

Remarks

File name	D$DETDI.PPL
created by	GOO$DISP:D$DSPCM.PPL

Description

CALLING	STS=D$DEALL(CV_base)
COMMAND	DETACH DISPLAY base

DEVICE

Routine arg. CHAR(*) VAR

Command par. String required

Name of database which should be detached or * if the display process should detach all attached bases.

Function

The specified data base is detached from the display process.

DISPLAY CALIBRATION

DISPLAY CALIBRATION name cal_dir base node

PURPOSE	Display calibration table.
PARAMETERS	
name	Name of calibration
cal_dir	Directory for calibration Data Elements.
base	Data Base name
node	Node name for Data Base file
Caller	MDISP,MGOODISP
Author	W. Spreng

Remarks

File name	GOO$DISP:D$DCAL.PPL
Created by	GOO$DISP:D$DSPCM.PPL

Description

CALLING	STS=D$DCAL(CV_NAME,CV_CAL_DIR,CV_BASE,CV_NODE)
COMMAND	DISPLAY CALIBRATION name cal_dir base node

NAME

Routine arg.	Input CHAR(*) VAR
Command par.	String required Name of the calibration, which should be displayed.

CAL_DIR

Routine arg.	Input CHAR(*) VAR
Command par.	String global replaceable default=$CALIBRATION Default Directory for calibrations.

BASE

Routine arg.	Input CHAR(*) VAR
Command par.	String global replaceable default=DB Default Data Base name.

NODE

Routine arg.	Input CHAR(*) VAR
Command par.	String global replaceable default=* Default node name.

Function

The correlation of the calibrated and uncalibrated units is shown graphically. Futhermore it will be checked if the functional dependence is unique. If this is not the case a warning message is produced and the critical points are listed.

DISPLAY CONDITION

```
DISPLAY CONDITION condition frame dimension
      cond_dir base node
    /CHAN/CALIB [=CALIBR]
    /DISTRIBUTE
    /XAXIS/YAXIS [=AXIS]
```

PURPOSE	Display window condition limits
PARAMETERS	
condition	Name of condition.
frame	Number of frame in which the condition should be displayed
dimension	Dimension of condition which should be displayed.
cond_dir	Default Directory name for condition
base	Default Data Base
node	default node name
CALIBR	Units in which the condition should be displayed.

/CHAN	Original spectrum units.
/CALIB	Calibrated spectrum units.

/DISTRIBUTE	Distribute the specified mebers of the condition name array to different frames started with the specified "frame".
AXIS	Specifies the axis at which the limits should be marked:

/XAXIS	The limits are marked at the x-axis.
/YAXIS	The limits are marked at the y-axis.

Caller	MDISP,MGOODISP,D$DSPCM

Action rout.	D$DCWIN
Author	W. Spreng

Example

1.) DISPLAY CONDITION WINDOW C(1) 2 3:5
Dimension 3 to 5 of condition array "C(1)" is displayed in frame 2
2.) DISPLAY CONDITION WINDOW c2
Limits of condition C2 is displayed in frame 1 Dimension 1 is used.
3.) DISPLAY CONDITION WINDOW c(2:4) 3 * /DISTRIBUTE
All dimensions of the condition array members c(2), c(3) and c(4) are displayed in different frames:

C(2) in frame 3
C(3) in frame 4
C(4) in frame 5

Remarks

File name	D$DCWIN.PPL
Created by	GOO$DISP:D$DSPCM.PPL
REMARKS	Only one dimensional conditions are supported.

Description

CALLING

```
STS=D$DCWIN(CV_condition,CV_frame,CV_dimension,
            CV_cond__dir,CV_base,CV_node,CV_calibr,
            I_distribute,CV_axis)
```

COMMAND

```
DISPLAY CONDITION condition frame dimension
     cond_dir base node
    /CHAN/CALIB [=CALIBR]
    /DISTRIBUTE
    /XAXIS/YAXIS [=AXIS]
```

CONDITION

Routine arg. CHAR(*) VAR

Command par. String replaceable

Name of window condition Supported condition name specifications are:

1.) COND_1
2.) COND_ARRAY(1)
3.) COND_ARRAY(2:5)
4.) COND_ARRAY(*)

Supported Condition types are: WINDOW and MULTIWINDOW conditions.

FRAME

Routine arg. CHAR(*) VAR

Command par. String replaceable; default=1

Number of frame in which this condition should be displayed.

DIMENSION

Routine arg. CHAR(*) VAR

Command par. String replaceable; default=*

Dimension of condition limits which should be shown. Possible inputs are:

1. n - single number
2. n:m - range of dimensions
3. * - all condition limits will be shown

COND_DIR

Routine arg. CHAR(*) VAR

Command par. String replaceable global default=$CONDITION

Default Directory for conditions

BASE

Routine arg. CHAR(*) VAR

Command par. String replaceable global default=DB

Default data Base

NODE

Routine arg. CHAR(*) VAR

Command par. String replaceable global default=*

Default node name

CALIBR

Routine arg. CHAR(*) VAR

Command par. Set default=/CHAN

Specified the units in which the condition limits should be displayed

/CHAN	Original spectrum units.
/CALIB	Calibrated spectrum units.

DISTRIBUTE

Routine arg. BIN FIXED(15); valid values 0 and 1

Command par. Switch

Distribute the single members of a name array to subsequent frames. The first frame is "frame".

If the condition array members COND(2:5) and frame 1 is specified the single array members are displayed in subsequent frames in the following manner:

COND(2) into frame 1
COND(3) into frame 2
COND(4) into frame 4
COND(5) into frame 5

AXIS

Routine arg. CHAR(*) VAR

Command par.	Switch default="/XAXIS"	
	Specifies the axis at which the condition limits should be marked.	
	/XAXIS	The condition limits are marked at the x-axis.
	/YAXIS	The condition limits are marked at the y-axis. This is done in any case, even if the spectrum in the frame is one dimensional.

Function

The limits of a specified condition are displayed in the specified frame. For multiwindow conditions the "dimension(s)" which should be shown have to be specified.

The members of condition arrays can be distributed to subsequent frames, if /DISTRIBUTE is specified.

Futhermore the condition limits can be marked at the X- or at the Y-axis.

DISPLAY GRAPH

DISPLAY GRAPH frame file module image

PURPOSE	Display user graphics.
PARAMETERS	
frame	Number of frame into which the graph should be drawn.
file	Specifies a file which can be used to read the graphical x and y vectors.
module	User module which should be dynamicly linked out of a sharable image.
image	Sharable image which contains the specifeid user module.
Caller	MDISP,MGOODISP
Author	W. Spreng

Example

1.) DISPLAY GRAPH 2 test.dat
The graphical data are read from file test.dat and are displayed in frame 2.
2.) DISPLAY GRAPH 5 *::DB:[SPECTRUM]test fit user
The module FIT is called, it has to be linked into the sharable image USER. The specified spectrum name is passed the the user module.

Remarks

Created by	D$DSPCM.PPL
File name	GOO$DISP:D$DGRAP.PPL

Description

CALLING	STS=D$DGRAP(L_frame,CV_File,L_index,l_size, CV_MODULE,CV_IMAGE,CV_type)
COMMAND	DISPALY GRAPH frame file index size module image /MARKER/LINE [=type]

Frame

Routine arg. Input BIN FIXED(31)

Command par. Integer

Number of frame into which the user graphics should be displayed.

File

Routine arg. Input CHAR(*) VAR

Command par. String

Specifies a file from which the X and Y-Vectors of the user graph should be read. If file input is required the module and image name have to be unspecified.

In each record of the file two values are expected. The first is the x-coordinate the second the y-coordinate.

If a module and image is specified this parameter is passed to the user module as a parameter!

INDEX

Routine arg. Input BIN FIXED(31)

Command par. Integer valid values MIN=1, MAX=100

Specifies the linetype index or markerindex which should be used. You can chose a combination of color and type, the index is calculated like:

10*colour+type

The colour representation depends on the output device. The linetypes are the following:

1	solid
2	dashed
3	dotted

4	dashed-dotted
5	long-dashed

It is possible that for serveral devices more linetypes are available. The marker types are:

1	dot
2	plus
3	asterix
4	circle
5	cross
6	square
7	triangle
8	rhomb
9	star
10	large cross

INDEX

Routine arg. Input BIN FIXED(31)

Command par. Integer valid values MIN=1

Specifies trhe linethickness or marker size.

MODULE

Routine arg. Input CHAR(*) VAR

Command par. String

User module which should be linked dynamicly from a user defined sharable image. The user routine must set the calibration table. The user-module is called with the following parameters:

```
  L_status=user(CV_file,L_length,RA_x,RA_y)
CV_file : Input. The string specified in the
             File parameter.
L_length: Used length in the arrays RA_x and RA_y.
             Output.
```

RA_x : Output. X-vector containing the
x-coordinates to be displayed.
RA_y : Output. Y-vector containing the
x-coordinates to be displayed.

A module "user" can be linked into the sharable image "SHARE.EXE" by:

```
LSHARE user.obj share.exe /share=usershr
```

"USERSHR" is the logical image name, which can be inserted into the "IMAGE" parameter of this command.

IMAGE

Routine arg. Input CHAR(*) VAR

Command par. String

Sharable image in which the user module can be found.

Function

A set of user created x/y coordinates can be displayed in the specified frame.

DISPLAY METAFILE

DISPLAY METAFILE file directory

PURPOSE	Read screen image from file and display it
PARAMETERS	
file	Metafile name produced with GKS
directory	VMS-directory for file
Caller	MDISP,MGOODISP,D$DSPCM
Author	W. Spreng

Example

1.) DISPLAY METAFILE file.met [user.metafile]
Metafile "[user.metafile]file.met" is interpreted and displayed on all active devices
2.) DISPLAY METAFILE file.met []
Metafile "[]file.met" is interpreted and displayed on all active devices

Remarks

File name	D$DSAVE.PPL
Created by	D$DSPCM.PPL
REMARKS	The format of the metafile is not fixed in the GKS standard, therefore it is possible that files produced with other GKS-implementation could not be interpreted.

Description

CALLING	STS=D$DSAVE(CV_file,CV_dir)
COMMAND	DISPLAY METAFILE file directory

FILE

Routine arg. Input CHAR(*) VAR

Command par. String required

Metafile name produced with GOOSY, on IBM or with any user software using the GKS-implementation available at GSI!

DIRECTORY

Routine arg. Input CHAR(*) VAR

Command par. String replacable default= SYS$LOGIN:[METAFILE]

VMS-directory where the metafile can be found. Set it "[]" if your file is on the current directory.

Function

Every Metafile produced in GOOSY on IBM or with any user software could be displayed.

To be shure that all information is displayed the workstation viewport has to be set in the metafile. Default is the x/y ration of your main-device to be shure that the whole screen is used.

Metafiles produced with GOOSY are only documented graphical informations! The metafile does not keep any informations about the spectra which are included in your picture. Therefore you can only display the metafile, but it is not possible to manipulate these pictures, e.g. to EXPAND frames, to FIT SPECTRA, to INTEGRATE, etc...

DISPLAY PICTURE

```
DISPLAY PICTURE picture condition object dyn_scat
          binfactor update refresh
          binfactor update refresh
          node base pic_dir dyn_dir cond_dir
          buffer_size
          /LIN /LOG /SQRT [=SCALE]
          /XLIN /XLOG /XSQRT [=X_SCALE]
          /YLIN /YLOG /YSQRT [=Y_SCALE]
          /ZLIN /ZLOG /ZSQRT [=Z_SCALE]
          /[NO]WINDOW
          /[NO]LIFE
          /[NO]ROTATE
          /[NO]SMOOTH
          /[NO]LETTER
          /[NO]NUMBER
          /NOCAL/CALAX/CALSPEC [=CALIB]
          /[NO]CHANNELS
          /SPECIFIED/FULL/LAST/ACTUAL [=RANGE]
          /AUTOSCALE/SCALE [=SCALE_RANGE]
          /HISTO/VECTOR/MARKER/CONTOUR-
          /POINT/ISO/CLUSTER/SCATTER [=STYLE]
          /TEMP /GLOBAL [=MODE]
          /NOSCATTER
          /[NO]ERROR
```

PURPOSE Display screen image as described by picture

PARAMETERS

picture	Name of picture
condition	Name of condition to be used for all scatter plot frames.
object	Object for specified condition

dyn_scat	Name of dynamic listfor the scatter-plot entry	
binfactor	Binning Factor	
update	Update interval. If <= 0 , no update	
refresh	Refresh intervall. If <= 0 , no refresh ******** not yet implemented ********	
node	Node name	
base	Data Base name	
pic_dir	Picture Directory name	
dyn_dir	Directory name for dynamic list	
cond_dir	Directory name for conditions	
buffer_size	Number of scatter points in scatter buffer	
SCALE	Scaling mode for Y- or Z-axis	
	/LIN	Linear scaling axis
	/LOG	Logarithmic scaling axis
	/SQRT	Squareroot scaling axis
X_SCALE	Scaling mode for X-axis	
	/XLIN	Linear X-axis
	/XLOG	Logarithmic X-axis
	/XSQRT	Squareroot X-axis
Y_SCALE	Scaling mode for Y-axis	
	/YLIN	Linear Y-axis
	/YLOG	Logarithmic Y-axis
	/YSQRT	Squareroot Y-axis
Z_SCALE	Scaling mode for Z-axis	
	/ZLIN	Linear Z-axis
	/ZLOG	Logarithmic Z-axis
	/ZSQRT	Squareroot Z-axis
/[NO] WINDOW	Window switch	

	/NOWINDOW	Display no windows
	/WINDOW	Display all associated windows of spectrum

***************** not yet implemented *******

/[NO] LIFE Life mode switch

/NOLIFE	No life mode
/LIFE	life mode (update event by event)

***************** not yet implemented *******

/[NO] ROTATE Rotate displayed spectra

/NOROTATE	No rotate
/ROTATE	Rotate

***************** not yet implemented *******

/[NO] SMOOTH Binnig mode

/SMOOTH	Smooth binning
/NOSMOOTH	min/max binning

/[NO] LETTER Display lettering

/LETTER	Display lettering on axis
/NOLETTER	Display no lettering

/[NO] NUMBER Display numbering

/NUMBER	Display numbers on axis
/NONUMBER	Display no numbers on axis

CALIB Perform calibration

/NOCAL	no calibration performed
/CALAX	Calibrate axis
/CALSPEC	Calibrate spectrum

/[NO] CHANNELS Display channel numbers

/CHANNEL	Display spectrum channels.
/NOCHANNEL	Display no spectrum channels.

RANGE Display predefined window

	/FULL	Display full spectrum range
	/LAST	Display last (#) range
	/ACTUAL	Display actual range
	/SPECIFIED	as defined in MODIFY FRAME command.
SCALE_RANGE	Scaling method	
	/AUTOSCALE	Autoscaling is performed
	/SCALE	Scaling as in MODIFY FRAME command
STYLE	Define style of displayed spectrum. For one dimensional spectra:	
	/HISTO	Draw histograms
	/VECTOR	Connect spectrum bins with lines
	/MARKER	Signe spectrum contents with markers
	For two dimensional spectra:	
	/CONTOUR	Contour lines are displayed
	/CLUSTER	Clusters indicating the count rate
	/ISO	Show spectrum as an isometric plot.
	/SCATTER	Simulate scatter plot data.
MODE	Select temporary or global display modes	
	/TEMP	Use last display parameters
	/GLOBAL	Use global picture parameters
	/NOGLOBAL	do not use global parameters.
/NOSCATTER	Deactivtivate switch for scatter plots	
	/NOSCATTER	Do not start scattering.
/[NO] ERROR	Draw statistical error bars for each bin.	
	/NOERROR	No error bars drawn.
	/ERROR	Error bars are drawn at each bin.
Caller	MDISP,MGOODISP,D$DSPCM	
Author	W. Spreng	

Examples

1.) DISPLAY PICTURE a
Picture "a" is diplayed as defined in the Data Element.
2.) DISPLAY PICTURE a /global
Picture "a" is diplayed with the switches defined in the SET PICTURE PARAMETER command.
3.) DISPLAY PICTURE a /GLOBAL/LIN
Picture "a" is diplayed with ths switches defined in the SET PICTURE PARAMETER command. but with linear scaling axis.
4.) DISPLAY PICTURE a /YLOG
All y-axis are displayed in "LOG" mode, this is valid for scatterframes too!

Remarks

File name D$DPICT.PPL

Created by GOO$DISP:D$DSPCM.PPL

REMARKS All parameters, qualifiers and switches specified in this command acts on all frames! If overlayed spectra or scatter-plot parameters have been defined with the OVERLAY command and are not saved in the picture Data Element they are lost when this command is given! Only overlays specified in the Data Element will be considered and could be displayed with the OVERLAY command.

Description

CALLING

```
STS=D$DPICT(CV_picture,CV_condition,CV_object,
     CV_dyn_scat,L_binfactor,L_update,L_refresh,
     CV_node,CV_db,CV_pic_dir,CV_dyn_dir,CV_cond_dir
    ,CV_buffer_size,CV_scale,CV_Xscale,CV_Yscale
    ,CV_Zscale,I_window,I_life,I_rotate
    ,I_smooth,CV_letter,CV_number,
     CV_calib,CV_channels,CV_range,
    ,CV_scale_range,CV_style,CV_mode,I_noscatter,
     I_error,B_mask)
```

COMMAND

```
DISPLAY PICTURE picture condition object dyn_scat
        binfactor update refresh
        node base pic_dir dyn_dir cond_dir
        buffer_sizeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeeee
```

```
/LIN /LOG /SQRT [=SCALE]
/XLIN /XLOG /XSQRT [=X_SCALE]
/YLIN /YLOG /YSQRT [=Y_SCALE]
/ZLIN /ZLOG /ZSQRT [=Z_SCALE]
/[NO]WINDOW
/[NO]LIFE
/[NO]ROTATE
/[NO]SMOOTH
/[NO]LETTER
/[NO]NUMBER
/NOCAL/CALAX/CALSPEC [=CALIB]
/[NO]CHANNELS
/SPECIFIED/FULL/LAST/ACTUAL [=RANGE]
/AUTOSCALE/SCALE [=SCALE_RANGE]
/HISTO/VECTOR/MARKER/CONTOUR-
/POINT/ISO/CLUSTER/SCATTER [=STYLE]
/TEMP /GLOBAL [=MODE]
/NOSCATTER
/[NO]ERROR
```

PICTURE

Routine arg.	Input CHAR(*) VAR
Command par.	String required replaceable
	Name specification of Picture Data Element which should be displayed. All frames of the picture has to be defined, before a display is possible!

CONDITION

Routine arg.	Input CHAR(*) VAR
Command par.	String
	General picture condition for all scatterplots in this picture. The scatter data are show if the result flag of the condition is true. If this parameter is specified the global picture condition defined in the CREATE PICTURE command will be overwritten.

OBJECT

Routine arg.	Input CHAR(*) VAR

Command par. String

Condition object for the general picture condition. If this parameter is specified the condition is checked against the object. This will destroy the result of an earlier check of the same condition!

DYN_SCAT

Routine arg. Input CHAR(*) VAR

Command par. String global default=$SCATTER

Name of dynamic list for the scatter-plot entry

BINFACTOR

Routine arg. Input BIN FIXED(31)

Command par. Integer, default=0

Binning Factor. If larger than "0" it specifies the number of bins to be summed up in the display. This corresponds to a temporarily modification of the spectrum binsize. E.g. if the binning factor is 2 the count rate in one displayed spectrum bin is given by the sum of two neighbouring bins in the original spectrum!

If =0 an internally calculated display binsize is used to optimize the displayed data.

UPDATE

Routine arg. Input BIN FIXED(31)

Command par. Integer replaceable

Update interval. If <= 0 , no update. If > 0 every 'update' seconds the whole spectrum will be updated with the new channel contents of each spectrum.

REFRESH

Routine arg. Input BIN FIXED(31)

Command par. Integer replaceable

Refresh intervall. If <=0 , no refresh. If > 0 all spectra in the spectrum will be deleted and redrawn every 'refresh' seconds. The scatter plots will be stored on the display and not deleted if a device with selctive erase operation is used (e.g. Tektronix 4115) if not the whole spectrum is deleted and the collected data in scatter plots are lost. Therefore be careful using this parameter.

*************** Not yet implemented **********

NODE

Routine arg. Input CHAR(*) VAR

Command par. String global default=*

Default node name.

BASE

Routine arg. Input CHAR(*) VAR

Command par. String global default=DB

Default Data Base where the picture is assumed.

PIC_DIR

Routine arg. Input CHAR(*) VAR

Command par. String global default=$PICTURE

Default Directory name.

DYN_DIR

Routine arg. Input CHAR(*) VAR

Command par. String global default=$DYNAMIC

Default Directory name for dynamic list

COND_DIR

Routine arg. Input CHAR(*) VAR

Command par. String global default=$CONDITION

Default Directory name for conditions

BUFFER_SIZE

Routine arg. Input CHAR(*) VAR

Command par. String global replace default=1000

Number of scatter points in scatter buffer.

SCALE

Routine arg. Input CHAR(*) VAR

Command par. Set

Set the display mode on the scaling axis. You can select between three modes:

/LIN	Display the linear count rate.
/LOG	Display the spectrum contents in logarithmic mode.
/SQRT	Display the square root of the spectrum contents.

X_SCALE

Routine arg. Input CHAR(*) VAR

Command par. Set

Display mode of X-axis. You can select between three modes:

/XLIN	Display the axis linear.
/XLOG	Display the axis in logarithmic mode.
/XSQRT	Display the square root of axis.

Y_SCALE

Routine arg. Input CHAR(*) VAR

Command par. Set

Display mode of Y-axis. You can select between three modes:

/YLIN	Display the axis linear.
/YLOG	Display the axis in logarithmic mode.
/YSQRT	Display the square root of axis.

Z_SCALE

Routine arg. Input CHAR(*) VAR

Command par. Set

Display mode of Z-axis. You can select between three modes:

/ZLIN	Display the axis linear.
/ZLOG	Display the axis in logarithmic mode.
/ZSQRT	Display the square root of axis.

WINDOW

Routine arg. Input BIN FIXED(15) valid values 0 or 1.

Command par. Switch negatable

*************** Not yet implemented **********

LIFE

Routine arg. Input BIN FIXED(15) valid values 0 or 1.

Command par. Switch negatable

*************** Not yet implemented **********

ROTATE

Routine arg. Input BIN FIXED(15) valid values 0 or 1.

Command par. Switch negatable

*************** Not yet implemented **********

SMOOTH

Routine arg.	Input BIN FIXED(15) valid values 0 or 1.
Command par.	Switch negatable

To optimize the displayed spectrum data, the display reduces the number of displayed bins by an internal display binsize. The spectrum bins can be gathered in to ways:

/SMOOTH	Display the spectra with smooth binning. In that mode the mean values of the channel contents over the display binsize will be shown. The effect is that the spectrum looks smoother, but the displayed spectrum contents could be fractional numbers.
/NOSMOOTH	The minimum and maximum contents of the display bins are shown. In that modes spikes in the spectra are not smoothed out.

LETTER

Routine arg.	Input CHAR(*) VAR
Command par.	Set

Activate or deactiveate the lettering on the axis.

/LETTER	The lettering is displayed.
/NOLETTER	The lettering is not displayed.

NUMBER

Routine arg.	Input CHAR(*) VAR
Command par.	Set

Activate or deactiveate the numbering on the axis.

/NUMBER	The numbering is displayed.
/NONUMBER	The numbering is not displayed.

CALIB

Routine arg. Input CHAR(*) VAR

Command par. Set

Display the spectrum in calibrated units. To do this the displayed spectra has to be connected to an existing calibration. Different calibration modes are available:

/CALAX	The axis is drawn in calibrated units and the spectrum in uncalibrated units. Therefore the distance between two subsequent tics varies.
/CALSPEC	The spectrum is drawn in calibrated units. Then the width of the displayed spectrum bins varies.
/NOCAL	No calibration is performed to the displayed spectra.

In any case the axis with uncalibrated units is displayed too. To prevent this specify the /NOCHANNELS switch!

CHANNELS

Routine arg. Input CHAR(*) VAR

Command par. Set

Activate or deactivate the display of an axis with the original channel units.

/CHANNELS	Display an axis with original units.
/NOCHANNEL	The axis with original units is not not displayed.

RANGE

Routine arg. Input CHAR(*) VAR

Command par. Set default=/LAST

Specify on of the available display windows:

/FULL	The full spectrum range is displayed.

/LAST	The last displayed range is used. This window is indicated be a "#".
/ACTUAL	Use the actual displayed range.
/SPECIFIED	Use the range as defined in the MODIFY FRAME command.

The specified window is used for the display of the spectra.

SCALE_RANGE

Routine arg. Input CHAR(*) VAR

Command par. Set default=/SCALE

Defines the range of the scaling axis.

/AUTOSCALE	Perform autoscaling
/SCALE	Use the scaling range defined in the MODIFY FRAME command.

STYLE

Routine arg. Input CHAR(*) VAR

Command par. Set

Defines the display style of the spectrum data. For one and two dimensional spectra different styles are implemented.

One dimensional spectra:

/HISTO	Histograms are generated
/VECTOR	The spectrum contents are connected by polylines.
/MARKER	The spectrum contents are indicated by markers.

For two dimensional spectra:

/CONTOUR	Contour lines are displayed.
/CLUSTER	The spectrum count rates are indicated by clusters of a variable size and colour.
/ISO	A 3D isometric plot is generated.

/SCATTER	Simulate scatter plot data.

For one dimensional spectra /HISTO is default for two dimensional /CONTOUR is default.

MODE

Routine arg. CHAR(*) VAR

Command par. SET

Use the global or temporary display parameters. The temporary parameters, are the modes specified in the last DISPLAY command. The global parameters are specified with DEFINE DISPLAY PICTURE.

/TEMP	Use temporary parameters.
/GLOBAL	Use the global parameters.

NOSCATTER

Routine arg. Input BIN FIXED(15) valid values 0 or 1.

Command par. Switch negatable

Switch to deactivate scatter-plot

/NOSCATTER	Scatter plot have to be activated via command

If nothing specified the scatterplot is activated

ERROR

Routine arg. Input BIN FIXED(15) valid values 0 or 1.

Command par. Switch negatable

Display statistical errors at each channel of a one dimensional spectrum.

/ERROR	Statistical errors (the square root of the count rate) are displayed.
/NOERROR	No errors are displayed

Function

Display the picture in the modes and limits defined with the MODIFY FRAME commands. All frames have to be defined before the picture can be displayed.

The specifications kept in the frames are changed by the parameter of the DISPLAY PICTURE command, but only for the current display. Futhermore the parameter act on all frames!

The global picture condition is changed permanently if a "condition" with or without an "object" has been specified.

DISPLAY POINT

```
DISPLAY POINT point frame
              /LOOP
   [CALIBR=]/CHAN/CALIB
```

PURPOSE	Mark point on screen and get channel contents.
PARAMETERS	
point	Name of point for input or value
frame	Number of frame or spectrum name
/LOOP	Enter cursor loop to mark several points
CALIBR	Specifies the unit in which the coordinates are given.
	/CHAN Original spectrum units.
	/CALIB Calibrated units.
Caller	MDISP,MGOODISP,D$DSPCM
Author	W. Spreng

Example

1. DISPLAY POINT (1023) 1 or DISPLAY POINT 1023 1
returns contents of channel 1023 of 1dim. spectrum in frame 1
DISPLAY POINT (100,600) 2
the same for 2 dimensional spectrum in frame 2.
2. DISPLAY POINT 1023
3. DISPLAY POINT frame=1
4. DISPLAY POINT
5. DISPLAY POINT /LOOP
6. DISPLAY POINT frame=1/LOOP

Remarks

File name D$DPOIN.PPL

Created by GOO$DISP:D$DSPCM.PPL

Description

CALLING STS=D$DPOIN(CV_point,CV_frame,CV_calibr)

COMMAND

```
DISPLAY POINT point frame
              /LOOP
    [CALIBR=]/CHAN/CALIB
```

POINT

Routine arg. Input CHAR(*) VAR

Command par. String

Coordinates of the point to be marked. The coordinates should be specified like:
(number) for 1.-dim spectra
(n1,n2) for 2.-dim spectra
If no coordinates have been specified the cursor appears to select one or several points.

FRAME

Routine arg. Input CHAR(*) VAR

Command par. String

Number of frame in which the coordinates should be be marked or selected. The spectrum in this frame is used to determine the channel contents. If no frame is specified and if more than one frame is on the screen, the cursor appears to pick the frame.

LOOP

Routine arg. Input BIN FIXED(15); valid input are 0 and 1

Command par. Switch

Enter cursor loop until break facility is given. If specified an arbitrary number of points in any frame could be marked.

CALIBR

Routine arg.	Input CHAR(*) VAR
Command par.	Set default="/CHAN"

If the spectrum in the selected frame is calibrated the input of the coordinates can occur in in calibrated or uncalibrated units:

/CHAN	Coordinate of point is in channels
/CALIB	Coordinates of point are in calibrated units

If the coordinates are specified via cursor input the units are known and this switch has no effect!

FUNCTION

The selected points will be marked in the selected frames and the contents of the spectrum in that frame is displayed.

Different inputs are possible (look at the examples):

1. 'point', 'frame' are specified:

The specified point is marked on the display and information about the spectrum contents is returned.

2. 'point' specified:

The cursor appears to select the frame.

3. 'frame' is specified:

Then only points in that frame could be marked.

4. Nothing specified:

Cursor appears to select the point and the frame

5. /LOOP is specified :

The cursor appears to select an arbitrary number of points in different frames. The input will be finished if the break facility is given. The parameter CV_point is ignored.

6. /LOOP and 'frame' is specified:

An arbitrary number of points could be selected in the specified frame.

The point to be displayed could be specified in channels or in calibrated units, depending on the switches "/CHAN or /CALIB". If calibrated units are given, but the spectrum in the specified frame is not connected to a calibration an error is signaled.

In any case the points are marked on the screen and the spectrum contents is displayed.

DISPLAY POLYGON

DISPLAY POLYGON polygon frame poly_dir base node
/FILL

PURPOSE	Display polygon points.
PARAMETERS	
polygon	Name of polygon.
frame	Frame in which the polygon should be displayed.
poly_dir	Directory for polygons.
base	Data Base name
node	Node name for Data Base file
/FILL	Fill interior of polygon.
Caller	MDISP,MGOODISP
Author	W. Spreng

Remarks

File name	D$DPOLY.PPL
Created by	D$DSPCM.PPL

Description

CALLING

```
STS=D$DPOLY(CV_polygon,L_frame,CV_poly_DIR,
            CV_BASE,CV_NODE,I_fill)
```

COMMAND

```
DISPLAY POLYGON polygon frame poly_dir base node
      /FILL
```

POLYGON

Routine arg.	Input CHAR(*) VAR
Command par.	String required
	Name of the polygon , which should be displayed.

FRAME

Routine arg.	Input BIN FIXED(15)
Command par.	Integer
	Number of the frame in which the polygon should be displayed. If pictures with more than one frame are active, the frame number has to be specified.

POLY_DIR

Routine arg.	Input CHAR(*) VAR
Command par.	String global replaceable default=$POLYGON
	Default Directory for polygons.

BASE

Routine arg.	Input CHAR(*) VAR
Command par.	String global replaceable default=DB
	Default Data Base name.

NODE

Routine arg.	Input CHAR(*) VAR
Command par.	String global replaceable default=*
	Default node name.

FILL

Routine arg.	Input BIN FIXED(15) valid inputs are 0 and 1
Command par.	Integer
	Fill the interior of the polygon to indicate the points inside. If not set only the contour of the is drawn.

Function

The specified polygon is displayed in the specified frame.

DISPLAY SCATTER

```
DISPLAY SCATTER xparam yparam limits condition
                object dyn_scat xletter yletter
                node base par_dir dyn_dir
                cond_dir buffer_size
                /LAST /ACTUAL [=RANGE]
                /TEMP /GLOBAL [=MODE]
                /NOSCATTER
```

PURPOSE Display single scatter frame for two parameters.

PARAMETERS

xparam	Name of parameter for x- value
yparam	Name of parameter for y- value
limits	Window limits for X and Y.
condition	Name of condition
object	Parameter object on which the condition acts.
dyn_scat	Name of dynamic list for scatter plot
xletter	X-lettering on scatterplot axis.
yletter	Y-lettering on scatterplot axis.
node	Default node name for all Data element name specifications.
base	Default Data Base name for all Data Elements
par_dir	Default Directory for parameters (objects)
dyn_dir	Default Directory name for dynamic list
cond_dir	Default Directory name for condition
buffer_size	Number of scatter points in scatter buffer

RANGE	Display window to be used.
	/LAST Window of last DISPLAY SCATTER command
	/ACTUAL Window of the current active scatterplot
MODE	Display modes to be used.
	/TEMP Modes of the last DISPLAY command
	/GLOBAL Global display modes should be used
/NOSCATTER	Wait for START SCATTER comand.
Caller	MDISP,MGOODISP,D$DSPCM
Author	W. Spreng

Remarks

File name D$DSCAT.PPL

Created by GOO$DISP:D$DSPCM.PPL

Description

CALLING

```
STS=D$DSCAT(CV_xparam,CV_yparam,
            CV_window,CV_condition,CV_obj,CV_dyn
            CV_xletter,cv_yletter,
            CV_node,CV_db,CV_par_dir,CV_dyn_dir,
            CV_cond_dir,CV_buffer_size
            CV_range,CV_mode,I_noscatter,B_mask)
```

COMMAND

```
DISPLAY SCATTER xparam yparam limits condition object dyn_scat
xletter yletter node base par_dir dyn_dir cond_dir buffer_size
    /LAST /ACTUAL [=RANGE]
    /TEMP /GLOBAL [=MODE]
    /NOSCATTER
```

XPARAM

Routine arg. Input CHAR(*) VAR

Command par. String required replaceable

Name of the parameter displayed in x-direction. If no directory the default parameter Directory is used. Valid input, e.g for J11 standard event:

EVENT.IA$EVENT(1)

YPARAM

Routine arg. Input CHAR(*) VAR

Command par. String required replaceable

Name of parameter for y-value. If no Directory is specified the Directory of "xparam" is used as default, e.g a valid input for a standard J11 event is:

EVENT.IA$EVENT(2)

LIMITS

Routine arg. Input CHAR(*) VAR

Command par. String replaceable default=(0,1023,0,1023)

Window limits to specify the range of the displayed parameter correlations. Valid inputs are any GOOSY display window specification (see the GOOSY Display Manual), e.g.

(0,100,300,400)

(,,300,400) xmin=0, xmax=1023, ymin=300, ymax=400

CONDITION

Routine arg. Input CHAR(*) VAR

Command par. String

Name of condition which is used to filter the scatter data. If no condition object is specified only the result bit of the condition is checked. If it is true the scatter data are displayed.

OBJECT

Routine arg. Input CHAR(*) VAR

Command par. String

Condition object for the general picture condition. If this parameter is specified the condition is checked against the object. This will destroy the result of an earlier check of the same condition!

DYN_SCAT

Routine arg. Input CHAR(*) VAR

Command par. String global replaceable default=$SCATTER

Name of dynamic list used to create the entries necessary for the scatter plot. The SCATTER entry in this dynamic list is necessary to tell the analysis which scatterplots are requested by your display.

ATTENTION The dynamic list has to be attatched before a scatterplot could be created.

XLETTER,YLETTER

Routine arg. Input CHAR(*) VAR

Command par. String replaceable

X- and Y-lettering for scatter plot axis. If nothing is specified the parameter names are used

NODE

Routine arg. Input CHAR(*) VAR

Command par. String global replaceable default=*

Default node name for all Data Element name specifications.

BASE

Routine arg. Input CHAR(*) VAR

Command par. String global replaceable default=DB

Default Data Base name for all Data Elements

PAR_DIR

Routine arg. Input CHAR(*) VAR

Command par. String global replaceable default=DATA

Default Directory for parameter.

DYN_DIR

Routine arg.	Input CHAR(*) VAR
Command par.	String global raplacable default=$DYNAMIC
	Default Directory name for dynamic list

COND_DIR

Routine arg.	Input CHAR(*) VAR
Command par.	String global replaceable default=$CONDITION
	Default Directory name for conditions

BUFFER_SIZE

Routine arg.	Input CHAR(*) VAR
Command par.	String global replace default=1000
	Number of scatter points in scatter buffer.

RANGE

Routine arg.	Input CHAR(*) VAR
Command par.	SET
	Display window to be used.

/LAST	Window of last DISPLAY SCATTER command
/ACTUAL	Window of the current active scatterplot

MODE

Routine arg.	Input CHAR(*) VAR
Command par.	SET
	Display modes to be used.

/TEMP	Modes of the last DISPLAY command
/GLOBAL	Global display modes should be used

NOSCATTER

Routine arg.	Input BIN FIXED(15) valid inputs are 0 and 1
Command par.	SWITCH Do not start with the scatter plot, wait for START SCATTER command.

Function

With this command correlations between two event parameters 'xparam' and 'yparam'could be displayed in live mode. To pass the data for the required scatterplot to the analysis an entry in the dynamic list 'dyn_scat' is created.

The condition is used as a trigger for the scattered correlations. If the condition is specified without any object only the result bit of this condition is checked, therefore take care that the condition has been executed. If an object has been specified the condition will be applied to it, even if the condition has been executed earlier. Therefore take care that the specified condition is not refered twice in the specified dynamic list, the last condition check wins. This is the reason why it is recommended to use a separate dynamic list for all scatter plots.

As default mode the scatter-plot works asynchronously to the analysis so that the analysis does not slow down. If the whole data should be displayed on screen, the scatter-plot have to run synchronously to the analysis. In that case you have to specify /NOSCATTER to prevent the scatter mode running on. After that you have to give the command:

```
START SCATTER /SYNCHRONOUS
```

In any case the scatter-plot data are not stored on disk or in the local storage of a display device. So the scatter plot provides a high resolution 2-dimensional display without occupying any computer storage. Therefore it is not possible to send scatter data to a plotter. If you try this the only effect is that you see on your plot a frame without any scatter data in it. This has been done for two reasons:

1. to reduce the used storage and
2. to ensure a scatter rate which is only limited by the physical transmission rate of the terminal line.

To get a nice plot of your scatter data you have to produce a plotfile by allocating a plotter device or by creating a metafile:

```
ALLOCATE DEVICE scatter ln03
ALLOCATE DEVICE scatter metaout
```

which can be sent to a plotter by the corresponding PLOT PLOTFILE or PLOT METAFILE commands!

DISPLAY SPECTRUM

```
DISPLAY SPECTRUM spectrum limits scalim
            binfactor update refresh cuts theta phi
            node base spec_dir
            /LIN /LOG /SQRT [=SCALE]
            /XLIN /XLOG /XSQRT [=XSCALE]
            /YLIN /YLOG /YSQRT [=YSCALE]
            /ZLIN /ZLOG /ZSQRT [=ZSCALE]
            /[NO]WINDOW
            /[NO]LIFE
            /[NO]ROTATE
            /[NO]SMOOTH
            /[NO]LETTER
            /[NO]NUMBER
            /NOCAL/CALAX/CALSPEC [=CALIB]
            /[NO]CHANNEL
            /FULL/LAST/ACTUAL [=RANGE]
            /AUTOSCALE/SCALE [=SCALE_RANGE]
            /HISTO/VECTOR/MARKER/CONTOUR-
            /POINT/ISO/CLUSTER/SCATTER [=STYLE]
            /TEMP /[NO]GLOBAL [=MODE]
            /[NO]ERROR
```

PURPOSE Display spectrum using default picture

PARAMETERS

spectrum	Name of spectrum
limits	Display limits for spectrum
scalim	Limits for scaling axis
binfactor	Binning factor.
update	Update interval.

refresh	Refresh interval. ******** not yet implemented ********	
cuts	Cuts on z-axis for CONTOUR/CLUSTER plots.	
theta	Rotation angle about x-axis for ISOMETRIC plots.	
phi	Rotation angle about y-axis for ISOMETRIC plots.	
node	Default node name.	
base	Default Data Base where the picture is assumed.	
spec_dir	Default Directory name.	
SCALE	Scaling mode for Y- or Z-axis	
	/LIN	Linear scaling axis
	/LOG	Logarithmic scaling axis
	/SQRT	Squareroot scaling axis
X_SCALE	Scaling mode for X-axis	
	/XLIN	Linear X-axis
	/XLOG	Logarithmic X-axis
	/XSQRT	Squareroot X-axis
Y_SCALE	Scaling mode for Y-axis	
	/YLIN	Linear Y-axis
	/YLOG	Logarithmic Y-axis
	/YSQRT	Squareroot Y-axis
Z_SCALE	Scaling mode for Z-axis	
	/ZLIN	Linear Z-axis
	/ZLOG	Logarithmic Z-axis
	/ZSQRT	Squareroot Z-axis
/[NO] WINDOW	Window switch ***************** not yet implemented *******	
/[NO] LIFE	Life mode switch ***************** not yet implemented *******	

/[NO] ROTATE Rotate displayed spectra
**************** not yet implemented *******

/[NO] SMOOTH Binnig mode

/SMOOTH	Smooth binning
/NOSMOOTH	min/max binning

/[NO] LETTER Display lettering

/LETTER	Display lettering on axis
/NOLETTER	Display no lettering

/[NO] NUMBER Display numbering

/NUMBER	Display numbers on axis
/NONUMBER	Display no numbers on axis

CALIB Perform calibration

/NOCAL	no calibration performed
/CALAX	Calibrate axis
/CALSPEC	Calibrate spectrum

/[NO] CHANNELS Display channel numbers

/CHANNELS	Display spectrum channels.
/NOCHANNELS	Display no spectrum channels.

RANGE Display predefined window

/FULL	Display full spectrum range
/LAST	Display last (#) range
/ACTUAL	Display actual range

SCALE_RANGE Scaling method

/AUTOSCALE	Autoscaling is performed
/SCALE	Scaling as specified in last command

STYLE Define style of displayed spectrum. For one dimensional spectra:

/HISTO	Draw histograms
/VECTOR	Connect spectrum bins with lines

	/MARKER	Signe spectrum contents with markers
	For two dimensional spectra:	
	/CONTOUR	Contour lines are displayed
	/CLUSTER	Clusters indicating the count rate
	/ISO	Show spectrum as an isometric plot.
	/SCATTER	Scatter plot data are simulated.
MODE	Select temporary or global display modes	
	/TEMP	Use last display parameters
	/GLOBAL	Use global picture parameters
	/NOGLOBAL	do not use global parameters.
/[NO] ERROR	Draw statistical error bars for each bin.	
	/NOERROR	No error bars drawn.
	/ERROR	Error bars are drawn at each bin.
Caller	MDISP,MGOODISP,D$DSPCM	
Author	W. Spreng	

Examples

1.) DISPLAY SPECTRUM spec
"spec" is displayed in default mode
2.) DISPLAY SPECTRUM /TEMP
The last spectrum is displayed with the same modes and within the same range as before.
3.) DISPLAY SPECTRUM two CUTS=0.4,0.5,0.7,0.9
Spectrum "two" is displayed in contour mode with the contourlines at the specified cuts.
4.) DISPLAY SPECTRUM two THETA=0.0 PHI=0.0/iso
Spectrum "two" is displayed in isometric mode. The x-axis is identical to the horizontal screen axis and the y-axis points into the screen, the z-axis points along the vertical screen axis.
5.) DISPLAY SPECTRUM two THETA=90.0 phi=90.0/iso
Now the x-axis of the spectrum is along the vertical screen axis and the y-axis is along the horizontal screen axis (from right to left) the z-axis comes out of your screen. With this set-up you will see only crossing lines.

Remarks

File name	D$DSPEC.PPL
Created by	D$DSPCM.PPL

Description

CALLING

```
STS=D$DSPEC(CV_spectrum,CV_limits,CV_scalim,
        L_binfactor,L_update,L_refresh,L_NUM_CUTS,
        RA_cuts,CV_theta,CV_phi,
        CV_node,CV_base,CV_spec_dir,
      ,CV_scale,CV_Xscale,CV_Yscale
      ,CV_Zscale,I_window,I_life,I_rotate
      ,I_smooth,CV_letter,CV_number,
        CV_calib,CV_channels,CV_range,
      ,CV_scale_mode,CV_style,CV_mode,LA_array,B_mask)
```

COMMAND

```
DISPLAY SPECTRUM spectrum limits scalim
            binfactor update refresh numcuts cuts
            theta phi
            node base spec_dir
            /LIN /LOG /SQRT [=SCALE]
            /XLIN /XLOG /XSQRT [=XSCALE]
            /YLIN /YLOG /YSQRT [=YSCALE]
            /ZLIN /ZLOG /ZSQRT [=ZSCALE]
            /[NO]WINDOW
            /[NO]LIFE
            /[NO]ROTATE
            /[NO]SMOOTH
            /[NO]LETTER
            /[NO]NUMBER
            /NOCAL/CALAX/CALSPEC [=CALIB]
            /[NO]CHANNELS
            /FULL/LAST/ACTUAL [=RANGE]
            /AUTOSCALE/SCALE [=SCALE_RANGE]
            /HISTO/VECTOR/MARKER/CONTOUR-
            /POINT/ISO/CLUSTER/SCATTER [=STYLE]
            /TEMP /GLOBAL [=MODE]
            /[NO]ERROR
```

SPECTRUM

Routine arg. Input CHAR(*) VAR

Command par. String replaceable

Name of spectrum which should be displayed

LIMITS

Routine arg. Input CHAR(*) VAR

Command par. String

Display limits for spectrum. Valid inputs are any GOOSY display window specification, e.g:

(xmin,xmin) - Minimum and maximum value for x
(xmin,) - Maximum is upper spectrum limit.
(,xmax) - Minimum is lower spectrum limit.
(xmin,xmax,ymin,ymax) -
for a two dimensional window.

SCALIM

Routine arg. Input CHAR(*) VAR

Command par. String

Range of scaling axis. Valid inputs are:
(min,max) - Minimum and maximum limits of scaling axis.
(min,) - upper limits is maximum spectrum contents.
(,max) - lower limits is minimum spectrum contents.

BINFACTOR

Routine arg. Input BIN FIXED(31)

Command par. Integer, default=0

Binning Factor. If larger than "0" it specifies the number of bins to be summed up in the display. This corresponds to a temporarily modification of the spectrum binsize. E.g. if the binning factor is 2 the count rate in one displayed spectrum bin is given by the sum of two neighbouring bins in the original spectrum!

If =0 an internally calculated display binsize is used to optimize the displayed data.

UPDATE

Routine arg.	Input BIN FIXED(31)
Command par.	Integer replaceable

Update interval. If <= 0 , no update. If > 0 every 'update' seconds the whole spectrum will be updated with the new channel contents of each spectrum.

REFRESH

Routine arg.	Input BIN FIXED(31)
Command par.	Integer replaceable

Refresh interval. If <=0 , no refresh. If > 0 all spectra in the spectrum will be deleted and redrawn every 'refresh' seconds. The scatter plots will be stored on the display and not deleted if a device with selctive erase operation is used (e.g. Tektronix 4115) if not the whole spectrum is deleted and the collected data in scatter plots are lost. Therefore be careful using this parameter.

+++++++ not yet implemented +++++++++++++++++

NUMCUTS

Routine arg.	Input BIN FIXED(31)
Command par.	Integer, default=0

Defines the number of cuts used for the display of CONTOUR, CLUSTER and SCATTER spectra. If specified as 0 the cuts of the CUTS parameter are used, if larger than 0 the cuts are displayed in a fixed stepwidth and the cuts defined in CUTS are ignored.

CUTS

Routine arg.	Input CHAR(*) VAR
Command par.	Real arrray default=(0.1,0.2,0.3,0.4,0.5,0.6,0.7, 0.8,0.9,1.0)

Cuts for CLUSTER/CONTOUR plots. The cuts are interpreted in relative units of the difference between the spectrum maximum and the spectrum minimum (e.g if 0.5 is specified the contour line at half between the spectrum maximum and spectrum minimum will be drawn). The actual cuts are calculated like:

```
ACTUAL = min_spectrum +
         cuts*(max_spectrum-min_spectrum)
```

Therefore the specified values have to be in the range of 0.0<=R_cuts<=1.0.

THETA

Routine arg. Input CHAR(*) VAR

Command par. String replaceable default=25.0

Rotation angle about X-Axis. This is the first rotation which is performed clockwise looking in direction of the X-axis.

PHI

Routine arg. Input CHAR(*) VAR

Command par. String replaceable default=25.0

Rotation angle about Z-Axis. This is the second rotation which is performed clockwise looking in direction of the Z-axis.

NODE

Routine arg. Input CHAR(*) VAR

Command par. String global replaceable default=*

Default node name.

BASE

Routine arg. Input CHAR(*) VAR

Command par. String global replaceable default=DB

Default Data Base name.

SPEC_DIR

Routine arg. Input CHAR(*) VAR

Command par. String global replaceable default=$SPECTRUM

Default Directory name for spectra

SCALE

Routine arg. Input CHAR(*) VAR

Command par. Set default=/LIN

Set the display mode on the scaling axis. You can select between three modes:

/LIN	Display the linear count rate.
/LOG	Display the spectrum contents in logarithmic mode.
/SQRT	Display the square root of the spectrum contents.

X_SCALE

Routine arg. Input CHAR(*) VAR

Command par. Set default=/XLIN

Display mode of X-axis. You can select between three modes:

/XLIN	Display the axis linear.
/XLOG	Display the axis in logaritmic mode.
/XSQRT	Display the square root of axis.

Y_SCALE

Routine arg. Input CHAR(*) VAR

Command par. Set default=/YLIN

Display mode of Y-axis. You can select between three modes:

/YLIN	Display the axis linear.
/YLOG	Display the axis in logaritmic mode.
/YSQRT	Display the square root of axis.

Z_SCALE

Routine arg.	Input CHAR(*) VAR
Command par.	Set deafult=/ZLIN

Display mode of Z-axis. You can select between three modes:

/ZLIN	Display the axis linear.
/ZLOG	Display the axis in logaritmic mode.
/ZSQRT	Display the square root of axis.

WINDOW

Routine arg.	Input BIN FIXED(15) valid values 0 or 1.
Command par.	Switch negatable

*************** Not yet implemented **********

LIFE

Routine arg.	Input BIN FIXED(15) valid values 0 or 1.
Command par.	Switch negatable

*************** Not yet implemented **********

ROTATE

Routine arg.	Input BIN FIXED(15) valid values 0 or 1.
Command par.	Switch negatable

*************** Not yet implemented **********

SMOOTH

Routine arg.	Input BIN FIXED(15) valid values 0 or 1.

Command par.	Switch negatable

To optimize the displayed spectrum data, the display reduces the number of displayed bins by an internal display binsize. The spectrum bins can be gathered in to ways:

/SMOOTH	Display the spectra with smooth binning. In that mode the mean values of the channel contents over the display binsize will be shown. The effect is that the spectrum looks smoother, but the displayed spectrum contents could be fractional numbers.
/NOSMOOTH	The minimum and maximum contents of the display bins are shown. In that modes spikes in the spectra are not smoothed out.

LETTER

Routine arg.	Input CHAR(*) VAR
Command par.	Set

Activate or deactiveate the lettering on the axis.

/LETTER	The lettering is displayed.
/NOLETTER	The lettering is not displayed.

NUMBER

Routine arg.	Input CHAR(*) VAR
Command par.	Set

Activate or deactiveate the numbering on the axis.

/NUMBER	The numbering is displayed.
/NONUMBER	The numbering is not displayed.

CALIB

Routine arg. Input CHAR(*) VAR

Command par. Set

Display the spectrum in calibrated units. To do this the displayed spectra has to be connected to an existing calibration. Different calibration modes are available:

/CALAX	The axis is drawn in calibrated units and the spectrum in uncalibrated units. Therefore the distance between two subsequent tics varies.
/CALSPEC	The spectrum is drawn in calibrated units. Then the width of the displayed spectrum bins varies.
/NOCAL	No calibration is performed to the displayed spectra.

In any case the axis with uncalibrated units is displayed too. To prevent this specify the /NOCHANNELS switch!

CHANNELS

Routine arg. Input CHAR(*) VAR

Command par. Set

Activate or deactivate the display of an axis with the original channel units.

/CHANNELS	Display an axis with original units.
/NOCHANNELS	The axis with original units is not displayed.

RANGE

Routine arg. Input CHAR(*) VAR

Command par. Set

Specify one of the available display windows:

/FULL	The full spectrum range is displayed.
/LAST	The last displayed range is used. This window is indicated be a "#".

/ACTUAL	Use the actual displayed range.

The specified window is used for the display of the spectra. If no window is given the spectrum is displayed in its full range.

SCALE_RANGE

Routine arg. Input CHAR(*) VAR

Command par. Set

Defines the range of the scaling axis.

/AUTOSCALE	Perform autoscaling
/SCALE	Use the scaling range defined in the last DISPLAY SPECTRUM command.

STYLE

Routine arg. Input CHAR(*) VAR

Command par. Set

Defines the display style of the spectrum data. For one and two dimensional spectra different styles are implemented.

One dimensional spectra:

/HISTO	Histograms are generated
/VECTOR	The spectrum contents are connected by polylines.
/MARKER	The spectrum contents are indicated by markers.

For two dimensional spectra:

/CONTOUR	Contour lines are displayed.
/CLUSTER	The spectrum count rates are indicated by clusters of a variable size and colour.
/ISO	A 3D isometric plot is generated.
/SCATTER	Scatter plot data are simulated. In each the countrate is indicated by a number of points, randomly distributed in the bin.

For one dimensional spectra /HISTO is default, for two dimensional spectra /CONTOUR is default.

MODE

Routine arg.	CHAR(*) VAR
Command par.	SET

Use the global or temporary display parameters. The temporary parameters, are the modes specified in the last DISPLAY command. The global parameters are specified with DEFINE DISPLAY PICTURE.

/TEMP	Use temporary parameters.
/GLOBAL	Use the global parameters.
/NOGLOBAL	Do not use the global parameters.

ERROR

Routine arg.	Input BIN FIXED(15) valid values 0 or 1.
Command par.	Switch negatable

Display statistical errors at each channel of a one dimensional spectrum.

/ERROR	Statistical errors (the square root of the count rate) are displayed.
/NOERROR	No errors are displayed

Function

The specified spectrum is displayed with the specified switches and modes. If 2-dim. spectra should be displayed in cluster or isometric mode the cuts and rotation angles can be modified:

a) in CLUSTER,CONTOUR mode an arbitrary number of cuts could be defined with "CUTS=...". The cuts are interpreted in relative units of the difference between the spectrum maximum and minimum.

b) in ISOMETRIC mode rotaion angles about the x-axis "THETA" and the y-axis "PHI" could be specified. Any values between -360.0 and + 360.0 are allowed.

DISPLAY TEXT

```
DISPLAY TEXT text frame xposition yposition font size
             /CENTER /LEFT /RIGHT [=LOCATE]
             /ABSOLUTE /RELATIVE [=UNIT]
```

PURPOSE	Display text into box specified by cursor
PARAMETERS	
text	String containing text.
frame	Frame into which the the text should be placed.
xposition	X-position where the text string should be placed.
yposition	Y-position where the text string should be placed.
font	GKS text font number for text string.
size	Factor to increase the text-size.
LOCATE	Specifies the location of the text.

/LEFT	Specified position is the lower left
/RIGHT	Specified position is the lower right
/CENTER	Specified coordinate is the center

UNIT	Specifies in which units the coordinates are given.

/ABSOLUTE	The coordinates are in absolute units.
/RELATIVE	The coordinates are in relative units.

Caller	MDISP,MGOODISP
Author	W. Spreng

Example

1.) DISPLAY TEXT "sdf sdf" 1

The cursor appears to specify the reference point in frame 1. The reference point is at the lower left corner of the text.

2.) DISPLAY TEXT "sdf sdf" xpos=100 ypos=200/CEN

The cursor appears to specify the frame. The text is centered.

3.) DISPLAY TEXT "sdf sdf" ypos=200/RIGHT

The cursor appears to specify the frame and the x-coordinate of the reference point. The specified point lies at the lower right edge of the text box.

4.) DISPLAY TEXT "sdf sdf" xpos=0.5 ypos=0.5/REL

The reference point lies in the middle of the physical screen.

Remarks

File name	D$DTEXT.PPL
Created by	D$DSPCM.PPL

Description

CALLING STS=D$DTEXT(CV_text,L_frame,R_xposition,R_yposition, L_font,R_size,CV_locate,CV_unit,B_mask)

COMMAND

```
DISPLAY TEXT text frame xposition yposition font size
        /CENTER /LEFT /RIGHT [=LOCATE]
        /ABSOLUTE /RELATIVE [=UNIT]
```

TEXT

Routine arg. Input CHAR(*) VAR

Command par. String required

String containing text. If text contains blanks, it has to be enclosed in apostrophes.

FRAME

Routine arg. Input BIN FIXED(31)

Command par. Integer replaceable

Frame into which the the text should be displayed. If the text should be placed in absolute units a valid frame number is required, because it is necessary to know in which coordinate system the text position should be interpreted and in which part of the picture it has to be drawn.

XPOSITION

Routine arg. Input BIN FLOAT(24)

Command par. FLOAT

X-position where the text string should be placed.

The X/Y-coordinates could be given in absolute coordinates according to the actual axis in the specified frame. Or in relative coordinates, given in units of the height and width of the total picture. If the coordinates are given in relative units the text could be placed anywhere one the screen, not only in a single frame!

YPOSITION

Routine arg. Input BIN FLOAT(24)

Command par. FLOAT

Y-position where the text string should be placed.

The X/Y-coordinates could be given in absolute coordinates according to the actual axis in the specified frame. Or in relative coordinates, given in units of the height and width of the total picture. If the coordinates are given in relative units the text could be placed anywhere one the screen, not only in a single frame!

FONT

Routine arg. Input BIN FIXED(31)

Command par. Integer replaceable default=0

GKS text font number for text string. Supported text-fonts are:

0	Hardwaretext
-1...-11	proportional
-101..-111	proportional italics
-201..-211	mono spaced
-301..-311	mono spaced italics

SIZE

Routine arg. Input BIN FLOAT(24)

Command par. FLOAT default=1.0

Factor to increase the text-size. This factor is relative to the minimum text height in the specified frame. E.g. a size of 2 increases the hight of the text to the double heigth of the lowest text height in the picture.

LOCATE

Routine arg. Input CHAR(*) VAR

Command par. Set default = /LEFT

Specifies the location of the text relativ to the X/Y-coordinates. Possible inputs are:

/LEFT	The specified position is the lower left corner of the text box.
/RIGHT	The specified position is the lower right corner of the text box.
/CENTER	The specified coordinate should be in the middle of the text box.

UNIT

Routine arg. Input CHAR(*) VAR

Command par. Set default = /ABSOLUTE

Specifies in which units the coordinates are given. Possible inputs are:

/ABSOLUTE	The coordinates are in spectrum units as indicated at the actual displayed axis in the specified frame. Therefore absolute units are defined only in one frame!
/RELATIVE	The coordinates are in relative units of the height and width of the total picture. (X,Y)=(0.0,0.0) is the lower left screen edge; (X,Y)=(1.0,1.0) is the upper right screen edge.

The ABSOLUTE units are only defined in one frame which has to be specified. The RELATIVE units describe a position on the total display-screen and are therefore picture independent!

FUNCTION

A text string is written into the picture displayed on the screen. The text string could be placed into a single frame, then the specified coordinates for the text reference point are interpreted in units of the actual axis in the specified frame. This mode is active if /ABSOLUTE has been set. The disadvantage of this mode is its correlation to the actually displayed picture!

But the text position can be specified in relative screen coordinates, if /RELATIVE is set. Then the reference point is interpreted in relative units of the screen height and width! In that mode the frame number is ignored. The relative position of e.g. (x,y) = (0.0,0.0) is the lower left screen edge.

If the frame number and/or the coordinates of the reference point are not specified they are promted via cursor input.

The text could be placed in different modes relative to the reference point:

/LEFT	The reference point is the lower left corner of the text box.
/RIGHT	The reference point is the lower right of the text box.
/CENTER	The reference point lies in the middle of the text box.

EXPAND

```
EXPAND limits frame
        /CHAN/CALIB [=CALIBR]
        /LOG/LIN/SQRT [=SCALE]
        /XLOG/XLIN/XSQRT [=X_SCALE]
        /YLOG/YLIN/YSQRT [=Y_SCALE]
```

PURPOSE Expand spectrum or scatterplot within specified window.

PARAMETERS

limits Limits for expansion range.

frame number of frame to be expanded.

CALIBR Specifies if the limits should be interpreted as spectrum cannels or as calibrated units.

/CHAN	Channels are specified.
/CALIB	Calibrated units are specified.

This two switches are valid inputs.

SCALE Modify the current display mode on the scaling axis.

/LIN	Linear scaling axis.
/LOG	Logarithmic Scaling axis.
/SQRT	Square root scaling axis.

X_SCALE Modify the current display mode on the X-axis.

/XLIN	Linear X-axis.
/XLOG	Logarithmic X-axis.
/XSQRT	Square root X-axis.

Y_SCALE	Modify the current display mode on the X-axis.	
	/YLIN	Linear Y-axis.
	/YLOG	Logarithmic Y-axis.
	/YSQRT	Square root Y/-axis.

Caller	MDISP,MGOODISP,D$DSPCM
Action rout.	D$DEXP
Author	W.Spreng

EXAMPLE

1.) EXPAND (100,300) 3
=> one dim spectrum in frame 3 is expanded within the specified limits. If in frame 3 is a two dimensional spectrum the y-range is unchanged.
2.) EXPAND (100,300,400,800) 3
=> expansion range of x-axis is 100 to 300. Expansion range on y-axis is 400 to 800 if a 2-dim spectrum or a scatterplot is in frame 3. For 1-dim spectra this y-range will be ignored.
3.) EXPAND (,300) 3
=> every value in the array specification could be set to the actual value if it is not specified. In this example the lower x-value is unchanged.
4.) EXPAND (100,,,700) 3
=> in this example x-max and y-min values are unchanged.
5.) EXPAND (100,300,400,800) *
=> ALL Frames on screen should be expanded. expansion range of x-axis is 100 to 300. Expansion range on y-axis is 400 to 800 if a 2-dim spectrum or a scatterplot is in frame 3. For 1-dim spectra this range will be ignored.
Cursor inputs:
6.) EXPAND
=> cursor appears to define expansion limits in one frame. Two limits have to be set.
7.) EXPAND frame=3
=> cursor appears to define expansion limits in frame 3. Two limits have to be set.
8.) EXPAND (0,100)
=> If more than one frame is on screen cursor appears to select the frame.
9.) EXPAND frame=3:6/LOG
=> Cursor appears to specify limits and the frames 3 to 6 are expanded and the scaling axis is displayed in logarithmic mode.
The limits could be specified as channels or as calibrated units. If calibrated units should be used but the spectrum is not connected to a calibration an error is signaled.

Remarks

File name D$DEXP.PPL

Created by GOO$DISP:D$DSPCM.PPL

Description

CALLING

```
STS=D$DEXP(CV_limits,CV_frame,CV_cal,
          cv_scale,cv_xscale,cv_yscale,b_mask)
```

COMMAND

```
EXPAND limits frame
        /CHAN/CALIB [=CALIBR]
        /LOG/LIN/SQRT [=SCALE]
        /XLOG/XLIN/XSQRT [=X_SCALE]
        /YLOG/YLIN/YSQRT [=Y_SCALE]
```

LIMITS

Routine arg. Input CHAR(*) VAR

Command par. String

Limits for expansion range. Possible inputs are is any GOOSY display window specification:
E.g. (100,200,400,700) for 2.-dim spectra
(100,300) for 1.- dim spectra
every position could be set to the actual display value by two subsequent commas. If no limits are given the cursor prompts for input. See the command examples.

FRAME

Routine arg. Input CHAR(*) VAR

Command par. String

Number of the frame in which the expansion should be performed.
Valid inputs are:
n : a single frame number as shown on screen
n:m : a range of valid frames
* : for all frames on screen
If devices with no local ereasure facility are active (e.g. TEKTRONIX 4014 or any plotter), and the standard GOOSY display is used do not specify a range of frames! Because in that case the whole picture is

completely redrawn for each specified frame. If all frames are specified the whole picture will be deleted and all frames are redrawn at once.

If no frame is specified the cursor appears to select a frame.

CALIBR

Routine arg. Input CHAR(*) VAR

Command par. Set default=/CHAN

Specifies if the limits should be interpreted as spectrum cannels or as calibrated units.

/CHAN	Channels are secified.
/CALIB	Calibrated units are specified

SCALE

Routine arg. Input CHAR(*) VAR

Command par. SET

Changes the display mode of the scaling axis in the expanded frames

/LIN	to LINEAR
/LOG	to LOGARITHMIC
/SQRT	to SQUARE ROOT

X_SCALE

Routine arg. Input CHAR(*) VAR

Command par. SET

Changes the display mode of the X-axis in the expanded frames

/XLIN	to LINEAR
/XLOG	to LOGARITHMIC
/XSQR	to SQUARE ROOT

Y_SCALE

Routine arg.	Input CHAR(*) VAR
Command par.	SET

Changes the display mode of the Y-axis in the expanded frames

/YLIN	to LINEAR
/YLOG	to LOGARITHMIC
/YSQRT	to SQUARE ROOT

FUNCTION

The spectrum or scatterplot in frame "frame" is expanded within the specified limits. Following inputs are valid:

1. frame = "
=> The cursor appears to specify the frame
2. limits = "
=> the cursor appears to specify the expansion range in frame "cv_frame"
3. limits = " and frame = "
=> the cursor appears to specify the expansion range in one arbitrary frame.
4. limits and frame specified
=> expansion is performed with no futher input

It is possible to expand several or all frames with the limits directly specified or selected by cursor input. A range of several frames is given by N:M all frames are expanded with 'frame'=*.

The limits could be specified as channels or as calibrated units. If calibrated units should be used but the spectrum is not connected to a calibration an error is signaled.

FIT SPECTRUM

```
FIT SPECTRUM frame poly window iter file
  /[NO]OUTPUT
  /[NO]APPEND
  /BACKGROUND
  /GAUSS
  /SAMEWIDTH
  /SHOW
  /[NO]ZERO
  /[NO]MARK
  /NOERROR /STATISTICAL [=ERROR]
```

PURPOSE	Fit spectrum with polynom and/or with gaussian peaks
PARAMETERS	
frame	Frame number to specify the spectrum used for the fit.
poly	Power of background polynom
window	Window to specify data region for gaussian fits.
iter	Maximum number of iterations.
file	File for the output of the results.
/[NO] OUTPUT	Output occus additionally onto the specified file.
/[NO] APPEND	Results are appended to an existing file.
/BACKGROUND	Fit background polynom
/GAUSS	Fit gaussian peaks
/SHOW	Show intermediate results of each iteration.
/SAMEWIDTH	All peaks are assumed to have same width.
/ZERO	Use points close to background too.

/[NO] MARK	Mark grphical input point by polylines.
	/MARK All points are indicated.
	/NOMARK The inputs are not indicated.
ERROR	Error wightening
	/STATISTICAL Statistical errors are used.
	/NOERROR No errors are taken into account.

Caller	MDISP
Author	W. Spreng

Remarks

File name	GOO$DISP:D$FISP.PPL
Created by	GOO$DISP:D$DSPCM.PPL

Description

CALLING

```
STS=D$FISP(L_frame,L_poly,CV_window,L_iter,
              CV_file,I_output,I_append,
              I_background,I_gauss,I_samewidth,I_show,
              I_mark,I_zero,CV_error,B_mask);
```

COMMAND

```
FIT SPECTRUM frame poly window iter file
  /[NO]OUTPUT
  /[NO]APPEND
  /BACKGROUND
  /GAUSS
  /SAMEWIDTH
  /SHOW
  /[NO]ZERO
  /[NO]MARK
  /NOERROR /STATISTICAL [=ERROR]
```

FRAME

Routine arg. Input BIN FIXED(31)

Command par. Integer; Default=1

The spectrum in the specified frame is used to perform the required fit.

POLY

Routine arg. Input BIN FIXED(31)

Command par. Integer; Default=1

Specifies the maximum power of the polygon, which is fitted to the spectrum data.

WINDOW

Routine arg. Input CHAR(*) VAR

Command par. String; Default=*

Window specification for the data region used to limit the data used to perform the gaussian fits. Following inputs are possible:

```
        * Use the whole displayed region
   ? or " Specify window with cursor input.
   (n,m) Use data between limit n and m.
```

ITER

Routine arg. Input BIN FIXED(31)

Command par. Integer; Default=10

Specifies the maximum number of iterations, which should be performed.

FILE

Routine arg. CHAR(*) VAR

Command arg. String replacable default=GOOSY_RESULT.LOG

Name of a file onto which the results should be written. To direct the outputs to that file the /OUTPUT switch is required!

OUTPUT

Routine arg. Input BIN FIXED(15); valid input are 0 and 1

Command par. Switch, default = /NOOUTPUT

Write results into the specified file.

/OUTPUT	The output is directed onto the file.
/NOOUTPUT	results are only written to terminal and into the session logfile.

APPEND

Routine arg. Input BIN FIXED(15); valid input are 0 and 1

Command par. Switch, default=/APPEND

Append the output to an existing file.

/APPEND	output is appended to an existing file.
/NOAPPEND	A new output file is created.

BACKGROUND

Routine arg. Input BIN FIXED(15); valid inputs are 0 and 1

Command par. Switch

A polygon should be fitted to the spectrum data. The power of the polygon is specified by the "poly" parameter! The cursor appears and up to 100 windows. The spectrum data in these windows are used for the fit.

If additionally gaussian peaks should be fitted, this polygon is used as the spectrum background.

GAUSS

Routine arg. Input BIN FIXED(15); valid inputs are 0 and 1

Command par. Switch

Gaussian peaks should be fitted to the spectrum data. The cursor appears to specify up to 100 data windows, which are used to fit a single gaussian peak. The peaks must be well separated.

SAMEWIDTH

Routine arg.	Input BIN FIXED(15); valid inputs are 0 and 1
Command par.	Switch

If specified the peak width is prompted only once. That width is used as a first guess for all peaks.

ITER

Routine arg.	Input BIN FIXED(15); valid inputs are 0 and 1
Command par.	Switch

If set the result of each iteration will be shown.

ZERO

Routine arg.	Input BIN FIXED(15); valid inputs are 0 and 1
Command par.	Switch

With this switch it is possible to ignore points which are not statistically significant about the background during the fit:

/ZERO	Use all data, even zeros!
/NOZERO	Ignore all points not clearly above the background.

All points in the range

ABS(Background-Data) < SQRT(Background)

are not used in the fit if zero suppression is required.

MARK

Routine arg.	Input BIN FIXED(15); valid inputs are 0 and 1
Command par.	Switch, default = /MARK

Mark graphical input points by polylines.

/MARK	mark the graphical input points.
/NOMARK	do not mark the inputs.

To control the acceptance of the graphical inputs it is recommended to use the /MARK switch. If nice pictures without any superfluous information should be produced, use the /NOMARK switch.

ERROR

Routine arg. Input CHAR(*) VAR

Command par. Set; Default=/STATISTICAL

Perform an error wightening during the fit. The spectrum data are wightened with their statistical errors.

/NOERROR	No errors are taken into account in the fit.
/STATISTICAL	The data are wightened with their statistical errors.

Function

Fit spectrum data with a polynom as background and an arbitrary number of well separated gaussian peaks. The spectrum displayed in the specified frame is used for the fit.

If /BACKGROUND is specified the cursor appears to specify background windows until the GKS-break facility is given at the graphical input device. Then a polynom with the power of 'poly' is fitted to the selected data regions, the result is stored in the resulting fit vector and the background is subtracted from the spectrum data.

If /GAUSS is specified the cursor appears again to specify the peak position and the peak width. The number of performed iterations is limited by the specified parameter 'ITER'. The results of each iteration can be controlled with the "/SHOW" switch.

The method applied fitting the background is described in Bevington page ???; for the gaussian peaks see page ???.

INTEGRATE

```
INTEGRATE window frame file
    /[NO]OUTPUT
    /[NO]APPEND
    /CHAN/CALIB [=CALIBR]
    /LOOP
```

PURPOSE	Integrate specified window
PARAMETERS	
window	Limits of integration window
frame	Frame which should be integrated.
file	File for the output of the results.
/[NO] OUTPUT	Output occus additionally onto the specified file.
/[NO] APPEND	Results are appended to an existing file.
CALIBR	Specifies the unit in which the coordinates are given.

/CHAN	Original spectrum units.
/CALIB	Calibrated units.

/LOOP	Enter cursor loop to mark several points
Caller	MDISP,MGOODISP,D$DSPCM
Author	W. Spreng

Example

1.) INTEGRATE
=> The cursor appears to define integration limits in any frame on the screen.
2.) INTEGRATE fr=5

=> the cursor appears to fix the limits. Inputs are only accepted in frame 5.

3.) INTEGRATE 100,600

=> cursor appears to select frame

4.) INTEGRATE /LOOP

=> Cursor appears to define limits and frames. Inputs are accepted from any frame on screen. The cursor loop interrupts with the GKS-Break facility.

5.) INTEGRATE fr=5/loop

=> Cursor appears to define limits. Inputs are accepted from frame 5. The cursor loop interrupts with the GKS-Break facility.

6.) INTEGRATE 100,600 *

=> All frames on screen are integrated.

Remarks

File name	D$INT.PPL
Created by	GOO$DISP:D$DSPCM.PPL

Description

CALLING STS=D$INT(CV_window,CV_frame,CV_file,I_output,
I_append,CV_calibr,I_LOOP)

COMMAND INTEGRATE window frame file
/[NO]OUTPUT
/[NO]APPEND
/CHAN/CALIB [=CALIBR]
/LOOP

WINDOW

Routine arg. Input CHAR(*) VAR

Command par. String

Window which should be used as integration range. Any valid display window specification is allowed (see the GOOSY Display Manual):
(xmin,xmax) for a one dimensional window
(xmin,xmax,ymin,ymax) for a two dimensional window.
If nothing has been specified the cursor appears to specify the window limits.

The upper window limits are exclusive! E.g. for a one dimensional analog spectrum of binsize 1 the window (1,3) yields to an integration of two bins! Although value 3 belongs to the third bin.

FRAME

Routine arg. Input CHAR(*) VAR

Command par. String

Number of frames in which the coordinates should be be marked or selected. The spectrum in this frame is used to determine the channel contents. If no frame is specified and if more than one frame is on the screen, the cursor appears to pick the frame.

Beside the specification of a single frame the following inputs are valid:

```
N:M - Integrate frame N to M with the specified
        or selected limits.
  * - Integrate all spectra on the screen within
        the specified or selected window.
```

FILE

Routine arg. CHAR(*) VAR

Command arg. String replacable default=GOOSY_RESULT.LOG

Name of a file onto which the results should be written. To direct the outputs to that file the /OUTPUT switch is required!

OUTPUT

Routine arg. Input BIN FIXED(15); valid input are 0 and 1

Command par. Switch, default = /NOOUTPUT

Write results into the specified file.

/OUTPUT	The output is directed onto the file.
/NOOUTPUT	results are only written to terminal and into the session logfile.

APPEND

Routine arg.	Input BIN FIXED(15); valid input are 0 and 1
Command par.	Switch, default=/APPEND

Append the output to an existing file.

/APPEND	output is appended to an existing file.
/NOAPPEND	A new output file is created.

CALIBR

Routine arg.	Input CHAR(*) VAR
Command par.	Set default="/CHAN"

If the spectrum in the selected frame is calibrated the input of the coordinates can occur in calibrated or uncalibrated units:

/CHANN	Coordinates of point are given in spectrum units.
/CALIB	Coordinates of point are in calibrated units

If the coordinates are specified via cursor input the units are known and this switch has no effect!

LOOP

Routine arg.	Input BIN FIXED(15); valid input are 0 and 1
Command par.	Switch

Enter cursor loop until break facility is given. If specified an arbitrary number of points in any frame could be marked.

FUNCTION

The spectrum in the specified frame is integrated within the specified limits. Possible inputs are:

1.) nothing specified
=> The cursor appears to define inegration limits in any frame on the screen.
2.) Frame specified
=> the cursor appears to fix the limits in that frame
3.) window specified
=> cursor appears to select frame

The window could be specified in calibrated units or in channels, depending on the given switch (/CHAN or /CALIB). If calibrated units are used but the spectrum in the specified frame is not connected to a calibration an error is signaled. If I_loop is set a cursor loop is entered which could be interrupted by GKS-Break facility.

OVERLAY

```
OVERLAY spectrum xpara ypara binfactor
          trans=(xfactor,xoffset,yfactor,yoffset)
          frame node base spec_dir par_dir
        /ADJUST
        /SAVE
        /[NO]ERROR
        /HISTO/VECTOR/MARKER - =[STYLE]
        /ISO/CLUSTER/CONTOUR/SCATTER
```

PURPOSE	Add spectrum to frame
PARAMETERS	
spectrum	Name of new spectrum, the spectrum could be one or two dimensional.
xpara	X-parameter for additional scatterplot parameters
ypara	Y-parameter for additional scatterplot parameters
binfactor	Binning factor.
xoffset	Offset X-direction for one dim. spectra
xfactor	Factor X-direction for one dim spectra
yoffset	Offset Y-direction for one dim spectra
yfactor	Factor Y-direction for one dim spectra
frame	Number of frame
node	Default node name
base	Default Data Base name
spec_dir	Default spectrum Directory
par_dir	Default parameter Directory

/ADJUST	New spectrum is adjusted by cursor
/SAVE	Save the defined overlays for the specified frame in the picture Data Element.

Example

1.) OVERLAY spec fr=4
Spectrum "spec" is overlayed in frame 4
2.) OVERLAY spec TRANS=1.5,100,2.0,-500 9 /save
Spectrum "spec" is added to frame 9 with the transformation factors:
x_fac = 1.5 x_shift = 100
y_fac = 2.0 y_shift = -500
and additionally the defined overlays are strored in the Data Base if a picture Data Element is on the screen.
3.) OVERLAY spec fr=2/adjust
The cursor appears to define a linear transformation before "spec" is added to frame 2. 4.) OVERLAY spec fr=2/NOERROR/MARKER
Spectrum is displayed in MARKER mode and without errorbars.
5.) OVERLAY xparam=[$event]x.geli(10) yparam=[$event]y.geli(9) frame=5
The specified scatterplot parameters are additionally displayed in the scatterframe 5.

Caller	MDISP,MGOODISP,D$DSPCM
Author	W. Spreng

Remarks

File name	D$DOVER.PPL
Created by	D$DSPCM.PPL
REMARKS	The dimensions of the original spectrum in the specifeid frame and of the overlayed spectrum have to be the identical.

Description

CALLING

```
STS=D$DOVER(CV_spectrum,CV_xpara,CV_ypara,
        RA_trans,L_frame,
        CV_node,CV_base,CV_spec_dir,CV_par_dir,
        I_adjust,I_save,I_error,CV_style,LA_dim,B_mask)
```

COMMAND

```
OVERLAY spectrum xpara ypara binfactor
  trans=(xfactor,xoffset,yfactor,yoffset)
  frame node base spec_dir par_dir
       /ADJUST
       /SAVE
       /[NO]ERROR
       /HISTO/VECTOR/MARKER - =[STYLE]
       /ISO/CLUSTER/CONTOUR/SCATTER
```

SPECTRUM

Routine arg. Input CHAR(*) VAR

Command par. String

Name of spectrum, which should be additionally displayed in a frame. The dimension of the original frame spectrum and of the specified spectrum have to be the same.

XPARA,YPARA

Routine arg. Input CHAR(*) VAR

Command par. String

The X and Y-parameter for additional scatterplot correlations, which should be displayed in the specified frame. Both parameter have to be specified.

BINFACTOR

Routine arg. Input BIN FIXED(31)

Command par. Integer, default=0

Binning Factor. If larger than "0" it specifies the number of bins to be summed up in the display. This corresponds to a temporarily modification of the spectrum binsize. E.g. if the binning factor is 2 the count rate in one displayed spectrum bin is given by the sum of two neighbouring bins in the original spectrum!

If =0 an internally calculated display binsize is used to optimize the displayed data.

TRANS

Routine arg. Input (*) BIN FLOAT(24)

Command par. Real array default=(1.0,0.0,1.0,0.0)

Defines a linear transformation for overlayed spectra in the horizontal (x) and vertical (y) axis:
(xfactor,xoffset,yfactor,yoffset)
This transformation is applied to one dimensional spectra.

FRAME

Routine arg. Input (*) BIN FIXED(31)

Command par. Integer replaceable

Number of the frame in which the overlays should be shown. If more than one frame is on the screen the frame has to be specified.

The type of this frame must be consistent with the specified spectrum or scatter-parameter. E.g. it is impossible to define scatter parameter for a spectrum frame.

NODE

Routine arg. Input CHAR(*) VAR

Command par. String replaceable default=*

Default node name

BASE

Routine arg. Input CHAR(*) VAR

Command par. String replaceable default=DB

Default Data Base name

SPEC_DIR

Routine arg. Input CHAR(*) VAR

Command par. String replaceable default=$SPECTRUM

Default spectrum Directory

PAR

Routine arg. Input CHAR(*) VAR

Command par. String replaceable default=DATA

Default parameter Directory

ADJUST

Routine arg. Input BIN FIXED(15), valid values 0 and 1

Command par. Switch

New spectrum is adjusted by cursor. This switch is only valid for 1 -dim spectra. If it is specified the cursor appears to define two points in the original spectrum. After that the specified frame is deleted and the spectrum which should be overlayed is displayed. Again the cursor appears to define two points in that spectrum. This cursor inputs define a linear transformation applied to the overlayed spectrum.

SAVE

Routine arg. Input BIN FIXED(15), valid values 0 and 1

Command par. Switch

Save the defined overlays for the specified frame in the picture which is actually displayed. To save the defined overlays for all frames give the command:

```
          OVERLAY /SAVE
```

To save the overlays for only one frame the frame number must be specified.

ERROR

Routine arg. Input BIN FIXED(15) valid values 0 or 1.

Command par. Switch negatable

Display statistical errors at each channel of a one dimensional spectrum.

/ERROR	Statistical errors (the square root of the count rate) are displayed.

/NOERROR	No errors are displayed

STYLE

Routine arg.	Input CHAR(*) VAR
Command par.	Set

Defines the display style of the spectrum data. For one and two dimensional spectra different styles are implemented.

One dimensional spectra:

/HISTO	Histograms are generated
/VECTOR	The spectrum contents are connected by polylines.
/MARKER	The spectrum contents are indicated by markers.

For two dimensional spectra:

/CONTOUR	Contour lines are displayed.
/CLUSTER	The spectrum count rates are indicated by clusters of a variable size and colour.
/ISO	A 3D isometric plot is generated.
/SCATTER	Scatter plot data are simulated. In each the countrate is indicated by a number of points, randomly distributed in the bin.

Function

Additional spectra or scatterparameters are added to one frame of the actually displayed picture.

For overlayed spectra (one and two dimensional) a transformation can be defined which is applied to the spectrum data before displaying them.

The two one-dim. spectra, which should be overlayed could be adjusted automaticly by setting '/ADJUST'. The cursor is then activated to get two points in the orginal spectrum. After that this frame will be deleted and the spectrum to be overlayed is shown. The cursor is activated again to get the corresponding points in that spectrum. With this coordinates a transformation in x-direction is defined which will be applied when overlaying the spectra.

Overlayed spectra are displayed in the modes as defined in the specified frame. The display style of the overlays can be modified, e.g:

OVERLAY spectrum /MARKER

Display the spectrum in marker mode. Futhermore it is possible to activate or deactivate the display of error bars.

To save the defined overlayed spectra or parameters for one frame in a Picture Data Element '/SAVE' has to be specified.

ATTENTION If in scatter frames additional scatter parameters should be displayed, specify first all parameters for all frames, then give the command:

```
OVERLAY/SAVE
```

which saves the additional scatter plots in your Data Base. After that the overlayed scatter plots are activated!

PLOT METAFILE

PLOT METAFILE file type command queue copies font

PURPOSE	Plot a metafile.
PARAMETERS	
file	VMS-file name for meta file. The default file type is ".MET".
type	The device type of the specified output device.
command	Optional print command (queue specification ignored) This argument is replaced. To clear it, specify " ".
queue	Queue-name of device at which the plotfile should be printed.
copies	Number of copies to produced. If a physical address is specified this argument is ignored.
font	Font for text in the picture
Caller	MDISP,MGOODISP
Author	W. Spreng

Example

1.) PLOT METAFILE test.met LN03 " " SYS$LN03_C
The metafile TEST.MET will be plotted on the LN03-PLUS Laser printer queue SYS$LN03_C.
2.) PLOT METAFILE test.met RP02 " "IBM::
Metafile test.met should be sent to IBM and plotted on the RP02 BENSON plotter.
3.) PLOT META test.met POST "PP A POST"
Format metafile to postscript format and print with PP command (DCL).

Remarks

Created by	D$DSPCM.PPL
File name	GOO$DISP:D$PLMET.PPL

Description

CALLING STS=D$PLMET(CV_file,CV_type,CV_command,CV_queue,
CV_copies,L_font)

COMMAND PLOT METAFILE file type command queue copies font

FILE

Routine arg. CHAR(*) VAR

Command par. String required

VMS file-name of the metafile, default file type is ".MET". Take care that the file is really a metafile.

TYPE

Routine arg. CHAR(*) VAR

Command par. String required

Device type of plotter. Supported types are:
1.) LN03 laser printer
2.) HP7550A3,HP7550A4 pen plotters
3.) POST postscript
4.) for plots send to IBM the following plotter
types are supported:
RP01 for the large BENSON plotter
RP02 for the small BENSON plotter
VP01 for the BENSON vector plotter.

COMMAND

Routine arg. CHAR(*) VAR

Command par. String optional

DCL command to print file. The picture is formatted into a file according type specification. Then this file is printed by command. The queue specification is ignored.
This argument is replaced. To clear it, specify " ".

QUEUE

Routine arg.	CHAR(*) VAR
Command par.	String required

Queue name or physical address of device. For available queue names at GSI use HELP PRINTER. It is possible to specify a physical address to copy the file to the device. In that case a colon has to be specified at the end, e.g. TXB4:

If a plot should be send to IBM the queue name has to be set to IBM::

COPIES

Routine arg.	CHAR(*) VAR
Command par.	String default = 1

Number of copies to produced. If a physical address is specified this argument is ignored.

FONT

Routine arg.	CHAR(*) VAR
Command par.	String default=0

Font number to change text fonts defined in the GKS-bundle tables which are used in case of hardware character requirement. The following GKS-font numbers are supported:

0	Hardware text font
-1...-11	proportional fonts.
-101..-111	proportional italics
-201..-211	mono spaced fonts
-301..-311	mono spaced italics

Function

This procedure plots a metafile on the specified plotter. First the device independent metafile is converted into a device dependent plotfile which is finaly sent to the specified queue or physical address. If in the queuename a colon is found it is assumed that a physical device address is specified and that the device is not spooled. Then plotfile is copied to the device. For the IBM a double colon is required.

PLOT PICTURE

```
PLOT PICTURE type command queue copies font file
                 /[NO]FLAG
                 /[NO]PRINT
```

PURPOSE	Send the current active picture to a plotter
PARAMETERS	
type	Device type of specified queue.
command	Optional print command (queue specification ignored) This argument is replaced. To clear it, specify " ".
queue	Optional queue-name of device at which the plotfile should be printed. Used, if no command was given.
copies	Number of copies to produced.
font	Font for text in the picture
font	Font for text in the picture
/[NO] FLAG	Print flag page
/[NO] PRINT	Avoid printing. A file should be specified.
Caller	MDISP,MGOODISP
Author	W. Spreng

Example

PLOT PICTURE ln03 queue=sys$ln03_a
sends one copy to LN03_plus plotter at queue
SYS$LN03_A.
PLOT PICTURE post "PS A POST"

Format postscript file and print with PS command
PLOT PICTURE color "PT A POST"
Format color postscript file and print with PT
PLOT PICTURE post "P A POST"
Format postscript file and print on LN03 printer A.
PLOT PICTURE lj250 "PI I 80"
Format inkjet file and print on LJ250 printer I.
PLOT PICTURE ln03 " " sys$ln03_a
The " " is necessary to clear a previous command.
PLOT PICTURE ln03 FILE=XXX.LN3 /NOPRINT
Store formatted picture in file without printing.

Remarks

Created by	GOO$DISP:D$DSPCM.PPL
File name	GOO$DISP:D$PLPIC.PPL

Description

CALLING	STS=D$PLPIC(CV_type,CV_command,CV_queue,CV_copies, L_font,CV_file,i_flag,i_print)
COMMAND	PLOT PICTURE type command queue copies font file /[NO]FLAG /[NO]PRINT

TYPE

Routine arg.	CHAR(*) VAR
Command par.	String required

Plotter type. Supported types are:
LN03 - Laser printer
POST - Postscript output format
COLOR - Color postscript output format
LJ250 - Color inkjet output format
SIXEL - Sixel output format
HP7550A3,HP7550A4 - colour pen plotter

COMMAND

Routine arg.	CHAR(*) VAR

Command par. String optional

DCL command to print file. The picture is formatted into a file according type specification. Then this file is printed by command. The queue specification is ignored.

This argument is replaced. To clear it, specify " ".

QUEUE

Routine arg. CHAR(*) VAR

Command par. String optional

Used, if no print command was specified. Queue name or physical address of device. If a colon (":") is found it will be assumed that a physical address is specified. Available queue names at GSI: HELP PRINTER

COPIES

Routine arg. CHAR(*) VAR

Command par. String default = 1

Number of copies to produced. If a physical address is specified this argument is ignored.

FILE

Routine arg. CHAR(*) VAR

Command par. String optional

Optional filename to store formatted picture. Normally one uses /NOPRINT to avoid printing.

FONT

Routine arg. BIN FIXED(31)

Command par. Integer default=0

Font number to change text fonts defined in the GKS-bundle tables which are used in case of hardware character requirement.

NOTE That means, hardware font must be enabled, i.e.:
DEFINE FRAME SETUP 10 0 (is default after start)

GTS-GRAL GKS The following GKS-font numbers are supported:

0	Hardware text font
-1...-11	proportional fonts.
-51	thick proportional font
-101..-111	proportional italics
-151	thick proportional italics
-201..-211	mono spaced fonts
-251	thick mono spaced font
-301..-311	mono spaced italics
-351	thick mono spaced italics

Recommended is font -51.

DEC GKS The standard DEC software fonts are:

0	Hardware text font
-1,1	Standard ISO font (Thin, ugly).
-15	Thick Roman (But underscore is arrow)

Recommended is font -15.
With Postscript format you may use one of the following fonts:

-101..-104	Times (Roman, italic, Bold, Bold italic)
-105..-108	Helvet. (Roman,italic,bold,bold italic)
-109..-112	Courier (Roman,italic,bold,bold italic)
-114..-117	Lubalin (Roman,italic,bold,bold italic)
-118..-121	School (Roman,italic,bold,bold italic)
-122..-125	Av.Garde (Roman,italic,bold,bold italic)
-126..-129	Souvenir (Roman,italic,bold,bold italic)

/FLAG

Routine arg. BIN FIXED(15) valid values 0 and 1

Command par. Switch replacable default=/noflag

If set, a flag page is printed

/PRINT

Routine arg.	BIN FIXED(15) valid values 0 and 1
Command par.	Switch replacable default=/print If set, picture is printed. /NOPRINT is used together with the file argument to store a picture in a file without printing.

Function

The current active GOOSY-picture is formatted in a file type 'type'. The file is printed on specified queue or by specified command. If a colon is found in the queuename it is assumed that a physical device is specified.

The data stored in the Workstation Independent Segment Storage (WISS) are associated to the plotter and a plotfile is generated. Finally this plotfile will be printed/copied to the specified device.

This command is not supported in the fast display version!

PLOT PLOTFILE

```
PLOT PLOTFILE file command queue copies
                 /[NO]DELETE
                 /[NO]FLAG
```

PURPOSE	Plot device specific plotfile
file	VMS-file name of plotfile.
command	Optional print command (queue specification ignored) This argument is replaced. To clear it, specify " ".
queue	Plotter queue name or physical device adress.
copies	number of copies to be made.
/[NO] DELETE	Delete plotfile after printing.
/[NO] FLAG	Print Flag page.
Caller	MDISP,MGOODISP
Author	W. Spreng

Example

PLOT PLOTFILE X.ln03 queue=sys$ln03_a
sends one copy to LN03_plus plotter at queue
SYS$LN03_A.
PLOT PLOT X.PS "PS A POST"
Print X.PS with PS command
PLOT PLOT X.PS "P A POST"
Print file on LN03 printer A.
PLOT PLOT X.ln03 " " sys$ln03_a
The " " is necessary to clear a previous command.

Remarks

Created by D$DSPCM.PPL

File name GOO$DISP:D$PLPFI.PPL

Description

CALLING
```
STS=D$PLPFI(CV_file,CV_command,CV_queue,CV_copies,
            I_delete,i_flag)
```

COMMAND
```
PLOT PLOTFILE file command queue copies
                /[NO]DELETE
                /[NO]FLAG
```

FILE

Routine arg. CHAR(*) VAR

Command par. String required

Plotfile name. Default file type is ".PLT" Wildcards are supported at any position in the file specification.

COMMAND

Routine arg. CHAR(*) VAR

Command par. String optional

DCL command to print file. The picture is formatted into a file according type specification. Then this file is printed by command. The queue specification is ignored. This argument is replaced. To clear it, specify " ". At GSI, there are the commands P, PP, PS and PI. Typing these DCL commands gives more information.

QUEUE

Routine arg. CHAR(*) VAR

Command par. String optional

Used, if no print command was specified. Queue name or physical address of device. If a colon (":") is found it will be assumed that a physical address is specified. Available queue names at GSI: HELP PRINTER

COPIES

Routine arg.	CHAR(*) VAR
Command par.	String default=1 Number of copies to be produced. If a physical address is specified this argument is ignored.

/DELETE

Routine arg.	BIN FIXED(15) valid values 0 and 1
Command par.	Switch default=/delete If set, the Plotfile will be deleted

/FLAG

Routine arg.	BIN FIXED(15) valid values 0 and 1
Command par.	Switch replacable default=/noflag If set, a flag page is printed

Function

If in the GOOSY-display process a plotter has been allocated as a spooled device the generated plotfile "file" could be sent to the device "queue".

If in the queuename a colon is found it is assumed that a physical device address is specified and that the device is not spooled. Then plotfile is copied to the device. Instead of a queue the complete print command can be specified. When the print command is P, PP, PS or PI, the print is done without spawning (GSI queues assumed). Otherwise the print command is spawned.

PRINT

```
PRINT command printer form file
                    /DELETE
```

PURPOSE	Plot device specific plotfile
command	GSI print command (P, PP, PS, PT or PI).
printer	GSI printer letter (A,B,C...)
form	GSI style (like in DCL commands)
file	File to be submitted or printed.
/DELETE	Qualifier, i.e. /DELETE

Description

CALLING	@CALL U$PRINT(CV_command,CV_printer,CV_form ,CV_file,CV_qualifier)
COMMAND	PRINT command printer sform file /DELETE

COMMAND

Routine arg.	CHAR(*) VAR
Command par.	String GSI print command (P, PP, PS, PT or PI). Instead of specifying printer and form separately, they may be specified with command, i.e. "P A 80"

PRINTER

Routine arg.	CHAR(*) VAR

Command par. String

GSI printer letter (A,B,C...) A list of available printers at GSI can be obtained by DCL command HELP PRINTER

FORM

Routine arg. CHAR(*) VAR

Command par. String

GSI style (like in DCL commands) A list of available forms at GSI can be obtained by DCL commands P PP PS or PI, respectively.

FILE

Routine arg. CHAR(*) VAR

Command par. String required

File to be printed.

/DELETE

Routine arg. CHAR(*) VAR

Command par. qualifier

/DELETE delete file after print

Function

FUNCTION Print file.

EXAMPLE

```
@CALL U$PRINT('P','A','POST','TEST.PS','/D')
  @CALL U$PRINT('P','A','80','TEST.TXT','')
  @CALL U$PRINT('P A 80','','','TEST.TXT','')
  PRINT P A 80 test.txt
```

PROJECT

```
PROJECT spectrum target window dimension node base spec_dir
          /ADD/SUB/CLEAR [=ACTION]
          /POLYGON
```

PURPOSE	Project window in 2-dim spectrum
PARAMETERS	
spectrum	Source spectrum.
target	Target spectrum
window	Limits of the projection window.
dimension	Dimension onto which the projection has to be performed.
node	Default node name
base	Default Data Base name
spec_dir	Default spectrum Directory
ACTION	Add or subtract result of projection.
/ADD	Add projection
/SUB	Substract projection
/CLEAR	Clear source spectrum
/POLYGON	Interprete the window as a polygon name.

Examples

PROJECT two one_x
The two dimensional spectrum ”two” is projected onto dimension 1 (default). The cursor appears to specify the projection window at the y-axis. The result is stored in spectrum one_x.

```
PROJECT two one_x win=* /ADD
```
Project total range of dimension 2 onto dimension 1 and add result to spectrum "one_x."

```
PROJECT two one_y win=(100,200) dim=2/SUB
```
Project specified window (on dimension 1) onto dimension 2 and subtract result from spectrum "one_y".

```
PROJECT two one_y WIN=polygon dim=2/POLYGON
```
The projection range is "polygon" and the projection occurs onto the y-axis.

Remarks

REMARKS	Up to now only projections of two dimensional spectra are supported.
Created by	D$DSPCM.PPL

Description

CALLING STS=D$PROJ(CV_spectrum, CV_target, CV_window, L_dimension, CV_node, CV_base, CV_dir, CV_action, I_polygon)

COMMAND
```
PROJECT spectrum target window dimension node base spec_dir
           /ADD/SUB/CLEAR [=ACTION]
           /POLYGON
```

SPECTRUM

Routine par. Input CHAR(*) VAR

Command par. Required, replaceable
Multidimensional spectrum which should be projected onto a one dimensional spectrum.
=> Up to now only 2.-dim spectra are supported!<=

TARGET

Routine par. Input CHAR(*) VAR

Command par. Required, replaceable
Target spectrum. Spectrum for projection. This has to be a one dimensional spectrum. If it does not exist, it will be created with the attributes of "spectrum" in dimension "dimension".

WINDOW

Routine par. Input CHAR(*) VAR

Command par. Optional, replaceable

Limits of the projection window. This is a one dim. window. Specify it like:
(min,max) - minimum and maximum window limit
* - the whole spectrum range
The upper limit is exclusiv; e.g for a binsize 1 spectrum the window (1,2) is one bin; but (1,2.1) is the same as (1,3) which projects two bins (because all values >= 2 and <3 belong to the bin with the value 3)!

You can specify a polygon as a projection region. In that case this parameter is interpreted as a name of the polygon and the "/POLYGON" must be set. All bins for which the middle of the bin is in the polygon are considered during the projection.

DIMENSION

Routine par. Input BIN FIXED(31)

Command par. Integer Default=1

Dimension onto which the projection has to be performed.

NODE

Routine par. Input CHAR(*) VAR

Command par. Global replaceable default=*

Default node name

BASE

Routine par. Input CHAR(*) VAR

Command par. Global replaceable default=DB

Default Data Base name

SPEC_DIR

Routine par. Input CHAR(*) VAR

Command par. Global replaceable default=$SPECTRUM

Default spectrum Directory.

POLY_DIR

Routine par. Input CHAR(*) VAR

Command par. Global replaceable default=$POLYGON

Default polygon Directory.

ACTION

Routine par. Input CHAR(*) VAR

Command par. Set default= /CLEAR

Add or subtract result of projection.

/ADD	The result is added to the contents of the target spectrum
/SUB	The result is subtracted from the contents of the target spectrum.
/CLEAR	The result of the projection is directly stored in the target spectrum. Old spectrum data are lost.

POLYGON

Routine par. Input BIN FIXED(15) valid values 0 and 1

Command par. Switch

If set the specified window is interpreted as the name of a polygon data element. The projection range is the area enclosed by this polygon.

Function

A two dimensional spectrum is projected onto a one dimensional spectrum. The axis onto which the projection should be performed could be specified with 'dimension'. The projection range could be specified in the following way:

(min,max) : minimum, maximum window limits

* : for the whole spectrum range

If no window is specified the cursor appears to select the limits in the specified spectrum.
The source and target spectrum have to be of compatible data types, i.e.:
D->D
R->R
L->L
I->I
I->L
Specifying the /POLYGON switch the window is interpreted as the Data Element name of a polygon. In that case the projection range is the area enclosed by the polygon. All spectrum bins for which the middle of the bin is in the polygon are considered during the projection.
With the projection data several actions could be performed

/ADD	The result is added to the contents of the target spectrum
/SUB	The result is subtracted from the contents of the target spectrum.
/CLEAR	The result of the projection is directly stored in the target spectrum. Old spectrum data are lost.

REFRESH

```
REFRESH frame
   /[NO]UPDATE
   /[NO]WINDOW
   /[NO]OVER
```

PURPOSE	Refresh picture as displayed
PARAMETERS	
frame	Number of frame or spectrum name. To refresh the whole picture specify frame=*.
/[NO] UPDATE	Get new spectrum copy
/[NO] WINDOW	Redraw windows too
/[NO] OVER	Redraw overlayed spectra
Caller	MDISP,MGOODISP
Action rout.	D$REFR
Author	W. Spreng

Example

1.) REFRESH *
Redraw the current picture to get rid of informations not stored in segments. Windows and overlays are lost.
2.) REFRESH
Redraw the whole picture with new spectrum contents.
3.) REFRESH 3/upd/over
Redraw frame 3 with new spectrum contents and redraw the overlayed spectra too. The other frames on the screen are not affected.

Remarks

File name D$REFR.PPL

Created by D$DSPCM.PPL

Description

CALLING STS=D$REFR(CV_frame,I_update,I_window,I_over,B_mask)

COMMAND

```
REFRESH frame
   /[NO]UPDATE
   /[NO]WINDOW
   /[NO]OVER
```

FRAME

Routine arg. Input CHAR(*) VAR

Command par. String replaceable

Frames which should be refreshed. Valid inputs are:

```
     n - a single frame number as shown on screen
   n:m - a range of valid frames
     * - for all frames on screen
```

UPDATE

Routine arg. Input BIN FIXEWD(15) valid values are 0 and 1

Command par. Switch.

If specified the spectra in the refreshed frames are displayed with the actual spectrum contents. Otherwise the spectra are redrawn as actually displayed on the screen.

WINDOW

Routine arg. Input BIN FIXEWD(15) valid values are 0 and 1

Command par. Switch.

The integration windows, the marked points and the fitted functions are redrawn in the refreshed frames.

OVER

Routine arg.	Input BIN FIXEWD(15) valid values are 0 and 1
Command par.	Switch.
	The overlayed spectra are redrawn with their actual spectrum contents.

FUNCTION

Refresh a single frame or several frames of the current picture.

It is possible to perform a refresh getting the actual spectrum contents, then /UPDATE have to be set. If /WINDOW and /OVER is not specified the displayed windows and overlayed spectra are lost after the refresh.

If a scatterframe is refreshed the scatterpoints are deleted. Therefore this command can be used for an effective and fast refresh of the scatter frames. It is much faster than a new DISPLAY PICTURE or DISPLAY SCATTER command, because an initialization of the scatterframes is not necessary.

REPLACE CONDITION WINDOW

```
REPLACE CONDITION WINDOW condition limits dimension cond_dir
base node
   /CHANN/CALIBR [=CALIBR]
   /XAXIS/YAXIS [=AXIS]
```

PURPOSE Set or replace window condition by cursor

PARAMETERS

condition	Name of window condition
limits	Limits for condition window
dimension	Dimensions for multi-dimensional condition windows which should be set.
cond_dir	Default condition Directory
base	Default Data Base name
node	Default node name
CALIBR	Specifies units of input

/CHAN	limits specified in channels
/CALIB	limits specified in calibrated units

AXIS	Axis at which the condition limits should be set with cursor

/XAXIS	x-coordinates of input used
/YAXIS	y-coordinates of input used

Example

SET CONDITION WINDOW c1 100,400 5
Lower limit 100; upper limit 400 is set in dimension 5 of condition C1.
SET CONDITON WINDOW c1 100,400 *
Lower limit 100; upper limit 400 is set in all dimensions of condition C1.
SET CONDITON WINDOW c1 100,400 3:5
Lower limit 100; upper limit 400 is set in dimension 3 to 5 of condition C1.
SET CONDITON WINDOW c_array 100,400 3:5
Lower limit 100; upper limit 400 is set in dimension 3 to 5 of condition C_ARRAY(1).
SET CONDITON WINDOW c_array(2:4) 100,400 3:5 /CALIB
The specified limits are given in calibrated units they are transformed into spectrum units and then they are set in dimension 3 to 5 of condition C_ARRAY(2) to (4)
SET CONDITON WINDOW c_array(*) dim=*
The cursor appears to specify the limits set in all dimensions of all condition array members.

Description

CALLING STS=D$SCWIN(CV_cond,CV_limits,CV_dim,CV_dir,CV_base,
cv_node,CV_cal,CV_Axis)

COMMAND REPLACE CONDITION WINDOW condition limits dimension cond_dir base node
/CHANN/CALIBR [=CALIBR]
/XAXIS/YAXIS [=AXIS]

CONDITION

Routine arg. Input CHAR(*) VAR

Command par. String required

Name of condition window. Valid inputs are:
COND - for a single condition window
COND(3) - for a single member in a condition array
COND(1:4) - for a several members of a condition array
COND(*) - for all members of a condition array
Wildcards in the condition and directory name are not supported.

LIMITS

Routine arg. Input CHAR(*) VAR

Command par. String

Limits for condition window, only one pair of limits is supported. The specified limits are set for all dimensions.

DIMENSION

Routine arg. Input CHAR(*) VAR

Command par. String default=*

Dimension in which the limits should be set. Possible input values:
n - single number
n:m - range
* - set all dimensions

COND_DIR

Routine arg. Input CHAR(*) VAR

Command par. String global replaceable default=$CONDITION

Default condition Directory

BASE

Routine arg. Input CHAR(*) VAR

Command par. String global replaceable default=DB

Default Data Base

NODE

Routine arg. Input CHAR(*) VAR

Command par. String global replaceable default=*

Default node for Data Base

CALIBR

Routine arg. Input CHAR(*) VAR

Command par. Set default=/CHAN

Possible inputs are:

/CHAN	Limits are specified in original spectrum coordinates (channels).
/CALIB	Limits are specified in calibrated units.

Limits specified in calibrated units will be recalculated into original spectrum units, before they are set in the condition windows.

AXIS

Routine arg. Input CHAR(*) VAR

Command par. Set default=/XAXIS

Specifies the axis at which the graphical input should occur.

/XAXIS	Limits are taken from X-axis.
/YAXIS	Limits are taken from Y-axis.

Function

Set limits in the specified window condition. For multi-dimensional conditions it is necessary to specify the dimensions which should be set. If the window limits are specified the are set in all specified dimensions. If no window limits are specified they are set via graphical cursor input. Therefore a graphical input device has to be allocated.

REPLACE POLYGON

```
REPLACE POLYGON polygon frame xpoints ypoints
                    poly_dir base node poly_pool
              /DELETE/MODIFY [=MODE]
```

PURPOSE	Set points in polygon
PARAMETERS	
polygon	Name of polygon data element
frame	Frame in which the polygon should be edited.
xpoints	X-Points of the polygon.
ypoints	Y-Points of the polygon.
poly_dir	Default condition Directory
base	Default Data Base name
node	Default node name
poly_pool	Default Pool for polygons.
mode	Switch from modification to deletion of existing points.

/MODIFY	Existing points are modified.
/DELETE	Existing points are ignored.

Caller	MDISP,MGOODISP,D$DSPCM
Author	W. Spreng

Remarks

File name	D$SPOLY.PPL
Created by	D$DSPCM.PPL

Description

CALLING STS=D$SPOLY(CV_polygon, L_frame, RA_xpoints, RA_ypoints, CV_poly_dir, CV_base, cv_node, CV_poly_pool, I_XY, CV_mode LA_array)

COMMAND REPLACE POLYGON polygon frame points poly_dir base node poly_pool
/IXYPOINTS
/DELETE/MODIFY [=MODE]

POLYGON

Routine arg. Input CHAR(*) VAR

Command par. String required replaceable

Name specification for the polygon, which should be set or modified. If the polygon Data Element does not exist, it will be created.

FRAME

Routine arg. Input BIN FIXED(31)

Command par. Integer, default=0

Number of frame in which the polygon should be set or modified. This frame is used to display the polygon points if an existing polygon should be modified (/MODIFY switch set).

XPOINTS

Routine arg. Input (*) BIN FLOAT(24)

Command par. Real array

Array of x-values for the polygon. If this points are specified the corresponding y-points have to be defined, too. If no points are given a graphical polygon editor is envoked to set or modify the polygon points.

YPOINTS

Routine arg. Input (*) BIN FLOAT(24)

Command par. Real array

Array of y-values for the polygon. If this points are specified the corresponding x-points have to be defined, too. If no points are given a graphical polygon editor is envoked to set or modify the polygon points.

POLY_DIR

Routine arg.	Input CHAR(*) VAR
Command par.	String global replaceable default=$POLYGON Default directory for polygons.

BASE

Routine arg.	Input CHAR(*) VAR
Command par.	String global replaceable default=DB Default Data Base name.

NODE

Routine arg.	Input CHAR(*) VAR
Command par.	String global replaceable default=* Default node name for the Data Base.

POLY_POOL

Routine arg.	Input CHAR(*) VAR
Command par.	String global replaceable default=$PIC_POOL Default Pool in which the polygon should be created.

XYPOINTS

Routine arg.	Input BIN FIXED(15)
Command par.	Switch If this switch has been specified the values in XPOPINTS are interpreted as successive (x,y)-pairs. E.g.: XPOINTS=100,200,300,400 are the points (x,y)1=100,200 and (x,y)2=300,400.

MODE

Routine arg.	Input CHAR(*) VAR

Command par.	Switch default=/MODIFY If the specified polygon has been defined earlier, the existing points can be modified or ignored

/MODIFY	The existing points are displayed and can be modified.
/DELETE	The existing points should be deleted and all polygon points have to be set again.

Function

Set or modify points in the specified polygon Data Element. If the polygon does not exist it will be created.

If no points are specified a graphical polygon editor is envoked, which allows to delete, insert or to append new points in the existing polygon. At the moment up to 64 points can be graphicly handled. If you need more please contact the GOOSY group.

The points could be specified directly in "xpoints" and "ypoints", then all existing points in the polygon are ignored. No changes of existing points is possible. If "points" is specified you have to define the whole polygon-points!

SAVE DISPLAY

SAVE DISPLAY file directory

PURPOSE	Save displayed picture in a metafile.
PARAMETERS	
file	File name for the metafile.
directory	VMS-directory for the file
Caller	MDISP,MGOODISP,D$DSPCM
Author	W. Spreng

Example

1. SAVE DISPLAY spectrum
   Creates a metafile with the file specification SYS$LOGIN[.METAFILE]SPECTRUM.MET
   2. SAVE DISPLAY spectrum []
   Saves the picture in the metafile "spectrum.met" in the actually used directory.
   3. SAVE DISPLAY []spectrum
   Is an invalid input!!!

Remarks

File name	D$SAVE.PPL
Created by	GOO$DISP:D$DSPCM.PPL
REMARKS	This command is not supported in the fast display version.

Description

CALLING	STS=D$SAVE(CV_file,CV_directory,B_mask)
COMMAND	SAVE DISPLAY file directory

FILE

Routine arg. CHAR(*) VAR

Command par. String required

VMS-file name of the metafile to be created. The default file extension is ".MET".

DIRECTORY

Routine arg. CHAR(*) VAR

Command par. String replaceable default=[username.METAFILE]

VMS-directory name in which the metafile should be stored. If the directory does not exist, it will be created.

Function

The displayed picture is saved in a GKS-metafile. The directory for the metafile will be created if it does not exist.

A GKS output metafile is allocated and the picture is saved in that file, except scatterdata and spectrum updates all graphical data are stored in the created file.

A metafile is a device independent plotfile and it can be redisplayed by DISPLAY METAFILE command or on a plotter by the PLOT METAFILE command.

This command is not supported in the fast version of the GOOSY display process.

SET CALIBRATION FIXED

```
SET CALIBRATION FIXED name unit start step input calib uncalib
parameters polynom module image cal_dir base node
   /[NO]FILE
   /FIT/MODULE/PARAMETER/PROMPT/TABLE [=ACTION]
```

PURPOSE	Set table for a fixed-type calibration.
PARAMETERS	
name	Name of calibration
unit	Description of the calibration units.
start	Uncalibrated value for the first entry in the calibration table.
step	Stepwidth in uncalibrated units between two calibrated table entries.
input	Specifies a file which can be used to read the uncalibrated and calibrated values
parameters	Parameters for linear transformation
calib	Calibrated values
uncalib	Uncalibrated values.
module	User module which should be dynamicly linked out of a sharable image.
image	Sharable image which contains the specified user module.
cal_dir	Directory for calibration data elements.
base	Data Base name
node	Node name for Data Base file
/[NO] FILE	The required inputs should be read from an input file.
ACTION	Action which should be performed.

/FIT	The given calibrated and uncalibrated values are fitted with a polygon.
/MODULE	The calibration table must be filled by a user written procedure.
/PARAMETER	The parameters for a polygon are given and the table is created with that values.
/PROMPT	The given calibrated values are used and the uncalibrated values are specified by cursor inputs. Then a fit is performed.
/TABLE	All specified calibrated values are put directly into the calibration table

Caller	MDISP,MGOODISP
Author	W. Spreng

Example

1.) SET CALIBRATION FIXED name energy -30 2.0 PARA=(100.0,30.0,0.5,0.001) /PARAMETER

With the specified polynom parameter the calibration table is calculated. The first table entry corresponds to an uncalibrated value of "-30.0" the stepwidth between two subsequent entries is "2.0".

2.) SET CALIBRATION FIXED name energy -30 2.0 cal=(100.0,200,400) uncal=(0,200,300) poly=2/FIT

A polynom of the 2nd power is fitted to the listed calibrated and uncalibrated values. The resulting parameters are used to calculate the calibration table.

3.) SET CALIBRATION FIXED name energy -30 2.0 file /FILE/FIT

As above, but the list of calibrated and uncalibrated values are read from file.

4.) SET CALIBRATION FIXED name energy -30.0 2.0 calib=(0.0,20.0,30.0,40.0,50.0) /TABLE

The specified calibrated values are set directly in the table.

5.) SET CALIBRATION FIXED name energy -30.0 2.0 module=X$calib imange=USERSHR /MODULE

The module X$CALIB is called, it has to be linked into the sharable image USERSHR.

Remarks

Created by	D$DSPCM.PPL

Description

CALLING STS=D$SFICA(CV_NAME,CV_UNIT,R_START,R_STEP,CV_INPUT, RA_CALIB, RA_UNCAL, RA_PARAMETER, L_POLY, CV_MODULE, CV_IMAGE, CV_CAL_DIR, CV_BASE, CV_NODE, I_FILE, CV_ACTION)

COMMAND SET CALIBRATION FIXED name unit start step input calib uncalib parameters polynom module image cal_dir base node
/[NO]FILE
/FIT/MODULE/PARAMETER/PROMPT/TABLE [=ACTION]

NAME

Routine arg. Input CHAR(*) VAR

Command par. String required

Name of fixed calibration which should be set.

UNIT

Routine arg. Input CHAR(*) VAR

Command par. String required

Description for the calibration units. This title appears later on as the lettering at the calibrated spectrum axis.

START,STEP

Routine arg. Input BIN FLOAT(24)

Command par. Float required

For the calibrations of type FIXED the range of the calibration table is determined by the uncalibrated value corresponding to the first table entry, the stepwidth between two table entries and the total length of the calibration table.

"START" specifies the uncalibrated value for the first entry in the calibration table. "STEP" determines the Stepwidth of uncalibrated values between two subsequent table entries.

INPUT

Routine arg. Input CHAR(*) VAR

Command par. String

Specifies a file from which the calibrated and/or the uncalibrated values are read. Input from the specified file is possible in all cases, for which the calibrated and/or uncalibrated values for a fit are used. For details see the function description.

If a file name is specified the /FILE switch is required.

CALIB

Routine arg. Input (*) BIN FLOAT(24)

Command par. Float array

Input of calibrated values (e.g the energies of a calibration source) which should be used to determine the parameter of a polynom performing a fit with the specified calibrated and uncalibrated values. Therefore for each calibrated value the corresponding uncalibrated value is required.

If calibrated values are given the list of uncalibrated values can be specified directly in "UNCALIB" parameter (this mode is activated by the /FIT switch) or they can be specified via graphical inputs if /PROMPT has been specified!

Futhermore the specified calibrated values are put directly into the calibration table if the /TABLE switch is set.

See the /ACTION parameter for a detail description.

UNCALIB

Routine arg. Input (*) BIN FLOAT(24)

Command par. Float array

Input of uncalibrated values which should be used to determine the parameter of the linear polynom performing a fit with the specified uncalibrated and calibrated values. Therefore for each uncalibrated value the corresponding calibrated value is required.

This list of values is used if the /FIT switch has been specified. See the /ACTION parameter for a detail description.

PARAMETER

Routine arg. Input (*) BIN FLOAT(24)

Command par. Float array

Array containing the parameter for a polynom, used to determine the calibration table. The parameters should be specified like:

```
          R_0,R_1,...,R_n n is arbitrary
The calibration table is calculated with these parameters:
     cal = R_0 + R_1*uncal + ... + R_n*(uncal**n)
```

If the polynom parameter should be used specify the /PARAMETER switch. See the /ACTION parameter for a detail description.

POLYNOM

Routine arg. Input BIN FIXED(31)

Command par. Integer default=0

Largest exponent of the polynom which should be fitted to the calibrated values.

MODULE

Routine arg. Input CHAR(*) VAR

Command par. String

User module which should be linked dynamicly from a user defined sharable image. The user routine must set the calibration table. If a user module should be called the /MODULE switch has to be set and the logical name of the sharable image has to be specified. The user-module is called with the following parameter:

```
  L_status=user(R_start,R_step,L_entries,RA_table)
          R_start - Start value of table.
                    BIN FLOAT(24) INPUT
          R_step - Stepwidth
                    BIN FLOAT(24) INPUT
          L_entries - Number of table entries
                    BIN FIXED(31) INPUT
          RA_table- Array which contains the final
                    calibrated values.
```

(*) BIN FLOAT(24) OUTPUT

A module "user" can be linked into the sharable image "SHARE.EXE" by:

```
LSHARE user.obj share.exe /share=usershr
```

"USERSHR" is the logical image name, which can be inserted into the "IMAGE" parameter of this command.

IMAGE

Routine arg. Input CHAR(*) VAR

Command par. String

Sharable image in which the user module can be found.

CAL_DIR

Routine arg. Input CHAR(*) VAR

Command par. String global replaceable default=$CALIB

Default directory for calibration Data Elements.

BASE

Routine arg. Input CHAR(*) VAR

Command par. String global replaceable default=DB

Default Data Base name

NODE

Routine arg. Input CHAR(*) VAR

Command par. String global replaceable default=*

Default node name.

FILE

Routine arg. Input BIN FIXED(15) valid values 0 and 1

Command par. Switch negatable default=/NOFILE

If /FILE has been specified all input necessary to perform the fit are read from the specified file in the parameter "INPUT".

The list of calibrated and uncalibrated values should be read from file if this switch together with /FIT has been specified.

The list of calibrated values is read from file and the corresponding values are prompted via cursor inputs if this switch has been specified together with /PROMPT.

The values of all table entries are read from file if the /TABLE switch is set.

See the /ACTION parameter for a detail description.

ACTION

Routine arg. Input CHAR(*) VAR

Command par. Set default=/PARAMETER

Specifies which input should be used and which should be performed . Possible inputs are:

/FIT	The given calibrated and uncalibrated values are fitted with a arbitrary polygon, which is then used to fill the calibration table.
/MODULE	The calibration table must be filled by a user written procedure.
/PROMPT	The given calibrated values are used and the uncalibrated values are specified via cursor inputs. Then a polynom fit is performed, to get polynom parameter used to fill the calibration table.
/PARAMETER	The parameters for a polygon are given and the table is created with that values.
/TABLE	All specified calibrated values are put directly into the calibration table.

Function

The calibration table for a FIXED-type calibration is set. In a fixed-calibration table the stepwidth between two uncalibrated values is fixed. The corresponding calibrated values are keept in a table for each step. Therefore the range of the calibration table is determined by the uncalibrated value for the first table entry "start", the stepwidth between two entries "step" and total range of the calibration table.

To set the values in the calibration table different modes are implemented. In the most cases it is sufficient to fit a sequence of calibrated values (e.g. energies from a calibration source) and uncalibrated values (normaly the spectrum channels of the corresponding energies) by a polynom, which is used to calculate the calibration table. But sometimes the calibration function should be anything, in that case the calibration table can be set in a user module or by directly specifying all table entries.

Modes The following modes are implemented:

1. /FIT The calibrated and uncalibrated values are specified and a least square fit should be performed with the data using an arbitrary polynom with the largest exponent given by the parameter "polynom". This polynom is used to calculate the calibration table. In that mode the input from a file is possible.

2. /PROMPT The calibrated values and the maximum polynom power are specified. The corresponding uncalibrated values are prompted by cursor inputs. A fit is performed to get the polynom parameters which are used to create the calibration table. In that mode the calibrated values can be read from file.

3. /TABLE If the calibration table has been calculated earlier it is possible to put the calibrated values directly into the table by the /TABLE switch. In that case the number of specified values and the number of table entries has to be identical. In that mode the calibrated values can be read from file.

4. /MODULE The calibration table could be set with a user written procedure which must be linked into a sharable image. The names of the user module and the sharable image have to be specified in the "MODULE" and "IMAGE" parameter. The user-module is called with the following parameter:

```
L_status=user(R_start,R_step,L_entries,RA_table)
          R_start - Start value of table.
                    BIN FLOAT(24) INPUT
          R_step - Stepwidth
                    BIN FLOAT(24) INPUT
          L_entries - Number of table entries
                    BIN FIXED(31) INPUT
          RA_table- Array which contains the final
                    calibrated values.
                (*) BIN FLOAT(24) OUTPUT
```

If the return status is not success the calibration will not be modified.

To link a module into a shareable image a DCL command is provided:

```
      LSHAR user image.exe/share=usershr
```

which linke the module "USER" into the executeable shareable image "IMAGE.EXE" and the image is assigned to the logical name "USERSHR", which can be used for the required "IMAGE" parameter.

5./PARAMETER The parameters of an arbitrary polynom, which should be used to calculate the calibration table, can be specified in "PARAMETER":

```
            R_0,R_1,...,R_n n-arbitrary
```

The calibration table is calculated with these parameters:

```
      cal = R_0 + R_1*uncal + ... + R_n*(uncal**n)
```

File input Futhermore there are two input possibilities for the required calibrated and/or uncalibrated values. The first one is to specify them directly in the corresponding parameter. The second one is to read the values from file by specifying the "INPUT" and the /FILE switch.

The input format of this file is very simple:

A "!" in the first column is interpreted as a comment line. The calibrated an uncalibrated values should be order in columns separated by blank or by comma. In each record one or two values are read, depending if only calibrated or additionally uncalibrated values should be read. If two values are required the first value is uncalibrated the second is calibrated.

An example for file input if "/FIT" is specified:

```
   ! Comment line
      100.0 200.0
      150.0 301.0
   ! another comment line
      300.0,605.0
      400.0, 790.0
   ! end of data
```

SET CALIBRATION FLOAT

```
SET CALIBRATION FLOAT name unit input module image
                cal_dir base node
                /[NO]FILE
```

PURPOSE	Set table for a float-type calibrations.
PARAMETERS	
name	Name of calibration
unit	Description of the calibration units. This string is shown at the calibration axis.
input	Specifies a file which can be used to read the uncalibrated and calibrated values
module	User module which should be dynamicly linked out of a sharable image.
image	Sharable image which contains the specifeid user module.
cal_dir	Directory for calibration data elements.
base	Data Base name
node	Node name for Data Dase file
/[NO] FILE	The required inputs should be read from an input file. If this switch is not specified the inputs should be specified in the command.
Caller	MDISP,MGOODISP
Author	W. Spreng

Example

1.) SET CALIBRATION FLOAT name energy file/FILE
The complete table (a list of calibrated and uncalibrated values is read from file.
2.) SET CALIBRATION FLOAT name energy module=X$CALIB image=USERSHR
The module X$CALIB is called, it has to be linked into the sharable image USERSHR.

Remarks

Created by D$DSPCM.PPL

File name GOO$DISP:D$SFLCA.PPL

Description

CALLING

```
STS=D$SFLCA(CV_NAME,CV_UNIT,CV_INPUT,CV_MODULE,
            CV_IMAGE,CV_CAL_DIR,CV_BASE,CV_NODE,
            I_FILE)
```

COMMAND

```
SET CALIBRATION FLOAT name unit input module image
           cal_dir base node
           /[NO]FILE
```

NAME

Routine arg. Input CHAR(*) VAR

Command par. String required

Name of float calibration which should be set.

UNIT

Routine arg. Input CHAR(*) VAR

Command par. String required

Description for the calibration units. This title appears later on as the lettering at the calibrated spectrum axis.

INPUT

Routine arg. Input CHAR(*) VAR

Command par. String

Specifies a file from which the calibrated and/or the uncalibrated values for the calibration table are read. If a file name is specified the /FILE switch is required.

MODULE

Routine arg. Input CHAR(*) VAR

Command par. String

User module which should be linked dynamicly from a user defined sharable image. The user routine must set the calibration table. The user-module is called with the following parameters:

```
    L_status=user(L_entries,RA_uncal,RA_cal)
        L_entries - Number of table entries
                BIN FIXED(31) INPUT
        RA_uncal - Array which contains a list of
                uncalibrated values.
             (*) BIN FLOAT(24) OUTPUT
        RA_cal - Array which contains the final
                calibrated values.
             (*) BIN FLOAT(24) OUTPUT
```

A module "user" can be linked into the sharable image "SHARE.EXE" by:

```
    LSHARE user.obj share.exe /share=usershr
```

"USERSHR" is the logical image name, which can be inserted into the "IMAGE" parameter of this command.

IMAGE

Routine arg. Input CHAR(*) VAR

Command par. String

Sharable image in which the user module can be found.

CAL_DIR

Routine arg. Input CHAR(*) VAR

Command par. String global replaceable default=$CALIB

Default Directory for calibration Data Elements.

BASE

Routine arg. Input CHAR(*) VAR

Command par.	String global replaceable default=DB Default Data Base name

NODE

Routine arg.	Input CHAR(*) VAR
Command par.	String global replaceable default=* Default node name.

FILE

Routine arg.	Input BIN FIXED(15) valid values 0 and 1
Command par.	Switch negatable default=/NOFILE If this switch is set the values of all table entries are read from file.

Function

The calibration table for a FLOAT-type calibration

Modes The following modes are implemented:

1. /FILE The whole table is read from the specified file. The input format of this file is very simple:

A "!" in the first column is interprted as a comment line. The calibrated an uncalibrated values should be order in columns separated by blank or by comma. In each record two values are read. The first value is uncalibrated the second is calibrated.

2. Module A user written subroutine is called which fills the calibration table. The routine has to be linked into a sharable image. The procedure is called with three the following parameters:

```
    L_status=user(L_entries,RA_uncal,RA_cal)
        L_entries - Number of table entries
                    BIN FIXED(31) INPUT
        RA_uncal - Array which contains a list of
                    uncalibrated values.
             (*) BIN FLOAT(24) OUTPUT
        RA_cal - Array which contains the final
                    calibrated values.
             (*) BIN FLOAT(24) OUTPUT
```

If the return status is not success the calibration will not be modified.

To link a module into a shareable image a DCL command is provided:

```
        LSHAR user image.exe/share=usershr
```

which linke the module "USER" into the executeable shareable image "IMAGE.EXE" and the image is assigned to the logical name "USER-SHR", which can be used for the required "IMAGE" parameter.

SET CALIBRATION LINEAR

```
SET CALIBRATION LINEAR name unit input parameters
            calib uncalib cal_dir base node
            /FILE
            /FIT/PROMPT/PARAMETER [=/ACTION]
```

PURPOSE	Set parameters for a linear calibration.
PARAMETERS	
name	Name of calibration
unit	Description of calibration units.
input	Specifies a file which can be used to read the uncalibrated and calibrated values
parameters	Parameters for linear transformation
calib	Calibrated values
uncalib	Uncalibrated values.
cal_dir	Directory for calibration Data Elements.
base	Data Base name
node	Node name for Data Base file
/[NO] FILE	The required inputs should be read from an input file.
ACTION	Action which should be performed. Possible inputs are:

/FIT	Perform a fit with given calibrated and uncalibrated values.
/PROMPT	The uncalibrated values are picked with cursor inputs.
/PARAMETER	The specified parameters are used.

Caller	MDISP,MGOODISP
Author	W. Spreng
Dataset	D$SLICA.PPL

Example

1.) SET CALIBRATION LINEAR name energy (100.0,0.5) /PARAMETER
The parameter (100.0 offset, factor 0.5) are put into the calibration Data Element. If a spectrum is displayed with these calibration the description at the axis is "energy".
2.) SET CALIBRATION LINEAR name energy calib=(100.0,200,400) uncalib=(0,200,300) /FIT
A fit is performed with the list of calibrated and uncalibrated values. The resulting polynom parameter are stored in the calibration.
3.) SET CALIBRATION LINEAR name energy file /FILE /FIT
The list of calibrated and uncalibrated values are read from file.
4.) SET CALIBRATION LINEAR name energy calib=(100.0,200,400) /PROMPT
The uncalibrated values are prompted via cursor after that the fit is performed.
5.) SET CALIBRATION LINEAR name energy file/FILE /PROMPT
The calibrated values are read from file.

Remarks

Created by	D$DSPCM.PPL
File name	GOO$DISP:D$SLICA.PPL

Description

CALLING STS=D$SLICA(CV_NAME,CV_UNIT,CV_INPUT, RA_PARAMETER, RA_CALIB, RA_UNCALIB, CV_CAL_DIR, CV_BASE, CV_NODE, I_FILE, CV_ACTION)

COMMAND
```
SET CALIBRATION LINEAR name unit input parameters
            calib uncalib cal_dir base node
            /[NO]FILE
            /FIT/PROMPT/PARAMETER [=ACTION]
```

NAME

Routine arg. Input CHAR(*) VAR

Command par.	String required Name of the calibration which should be set.

UNIT

Routine arg.	Input CHAR(*) VAR
Command par.	String required Description for the calibration units. This title appears later on as the lettering at the calibrated spectrum axis.

INPUT

Routine arg.	Input CHAR(*) VAR
Command par.	String Specifies a file from which the calibrated and/or the uncalibrated values are read. Input from the specified file is possible in all cases, for which the calibrated and/or uncalibrated values for a fit are used. For details see the function description. If a file name is specified the /FILE switch is required.

PARAMETERS

Routine arg.	Input (*) BIN FLOAT(24)
Command par.	Float array default = (0.0,1.0) Array containing the parameter of the linear polygon used for the calibration. If these parameters should be used specify the /PARAMETER switch. See the /ACTION parameter for a detail description.

CALIB

Routine arg.	Input (*) BIN FLOAT(24)
Command par.	Float array Input of calibrated values which should be used to determine the parameter of the linear polynom by performing a fit with the specified calibrated and uncalibrated values. Therefore for each calibrated value the corresponding uncalibrated value is required.

If calibrated values are given the list of uncalibrated values can be specified directly in "UNCALIB" parameter (this mode is activated by the /FIT switch) or they can be specified via graphical inputs if /PROMPT has been specified! See the /ACTION parameter for a detail description.

UNCALIB

Routine arg. Input (*) BIN FLOAT(24)

Command par. Float array

Input of uncalibrated values which should be used to determine the parameter of the linear polynom performing a fit with the specified uncalibrated and calibrated values. Therefore for each uncalibrated value the corresponding calibrated value is required.

This list of values is used if the /FIT switch has been specified. See the /ACTION parameter for a detail description.

CAL_DIR

Routine arg. Input CHAR(*) VAR

Command par. String global replaceable default=$CALIB

Default Directory for calibration Data Elements.

BASE

Routine arg. Input CHAR(*) VAR

Command par. String global replaceable default=DB

Default Data Base name

NODE

Routine arg. Input CHAR(*) VAR

Command par. String global replaceable default=*

Default node name.

FILE

Routine arg. Input BIN FIXED(15) valid values 0 and 1

Command par. Switch negatable default=/NOFILE

If /FILE has been specified all input necessary to perform the fit are read from the specified file in the parameter "INPUT".

The list of calibrated and uncalibrated values should be read from file if this switch together with /FIT has been specified.

The list of calibrated values is read from file and the corresponding values are prompted via cursor inputs if this switch has been specified together with /PROMPT. See the /ACTION parameter for a detail description.

ACTION

Routine arg. Input CHAR(*) VAR

Command par. Set default=/PARAMETER

Specifies which input should be used and which should be performed . Possible inputs are:

/FIT	The given calibrated and uncalibrated values are fitted with a linear polygon.
/PROMPT	The given calibrated values are used and the uncalibrated values are specified via cursor inputs. Then a fit is performed.
/PARAMETER	The specified parameters are used.

Function

The two parameter of a linear polygon are stored in a calibration Data Element. The polygon parameter can be determined in different modes:

Modes	The following modes are implemented:
1./PARAMETER	The parameters are specified and should be copied into the Data Element. In that case /PARAMETER must be set.
2. /FIT	The calibrated and uncalibrated values are specified and a least square fit should be performed through the data. This mode is activated with /FIT.

3. /PROMPT Only calibrated values are specified and the corresponding values should be specified via cursor input. If you like this specify /PROMPT.

File input There are two possibilities to specify the required calibrated and uncalibrated values. They can be specified directly in the command or they are read from the specified inputfile if the /FILE switch is set! The input from file is convenient if e.g. the energies of the same calibration source should be used to calibrate different spectra. In that case the corresponding peaks can be picked by cursor in each spectrum for each energy value found in the specified file.

The input format of this file is very simple:

A "!" in the first column is interprted as a comment line. The calibrated an uncalibrated values should be order in columns separated by blank or by comma. In each record one or two values are read, depending if only calibrated or additionally uncalibrated values should be read. If two values are required the first value is uncalibrated the second is calibrated.

An example for file input if "/FIT" is specified:

```
! Comment line
! uncalibr.—calibr.
    100.0 200.0
    150.0 301.0
! another comment line
    300.0,605.0
    400.0, 790.0
! end of data
```

SET DEVICE COLOR

```
SET DEVICE COLOR name index r g b
     /UPDATE
```

PURPOSE	Allocate a graphical device
PARAMETERS	
name	Logical device name use TT for current terminal.
index	Color index (0-max)
R,G,B	Color values for read, green, blue (0-100)
/UPDATE	update color setup
Caller	MDISP,MGOODISP,D$DSPCM
Author	W. Spreng

Example

Remarks

File name	D$SECOL.PPL
Created by	GOO$DISP:D$DSPCM.PPL

Description

CALLING	STS=D$SECOL(CV_dev,I_index,R_r,R_g,R_b,I_update)
COMMAND	SET DEVICE COLOR name index r g b /UPDATE

NAME

Routine par. Input CHAR(*) VAR

Command arg. String required

Logical or physical device name. For a graphical Terminal it is the VMS-Terminal address; e.g.:

TXA6	for a terminal connected directly to one VAX.
LTA999	for a terminal connected to a LAT server.
TT	for the terminal of the session.

For spooled devices (all supported plotters) you can specify a file name with or without a file type.

For Metafiles a file name or a logical name which refers to a file is required.

For MICRO-VAX II and GPX devices the device name is arbitrary, it is displayed at the generated graphic window.

INDEX

Routine par. Input BIN FIXED(31)

Command arg. Integer, Default=1

Specifies the color index to be changed. Valid values are 0-max, where 0 is the background color.

1	picture lettering
2	spectrum name
3	axes and lettering
4	lettering info
6	spectrum

R

Routine par. Input BIN FLOAT(24)

Command arg. Float

Percentage of red color admixture. Valid values aree 0-100.

G

Routine par. Input BIN FLOAT(24)

Command arg. Float

Percentage of green color admixture. Valid values 0-100.

B

Routine par. Input BIN FLOAT(24)

Command arg. Float

Percentage of blue color admixture. Valid values 0-100.

UPDATE

Routine par. Input BIN Fixed(15)

Command arg. Switch

There are some devices which are not possible to update theire color map dynamicaly. In that case a change of the color map can be make visible by redrawing the whole picture. To improve the performance of the color map changes in that case the update is suppressed until it will be specified.

Function

If you do not agree with the default color map of GOOSY, you have the possibility to define you own color lay out by specifying the color index and the RGB values of that color.

SET DISPLAY MODE

SET DISPLAY MODE
/STANDARD/FAST [=VERSION]

PURPOSE	Select display version.
PARAMETERS	
VERSION	Version which should be used.

/STANDARD	Standard display version with full functionality.
/FAST	Reduced version which needs less CPU.

Caller	MDISP, MGOODISP
Author	W. Spreng

Remarks

Created by	GOO$DISP:D$DSPCM.PPL
File name	GOO$DISP:D$DVERS.PPL

Description

CALLING	STS=D$DVERS(CV_VERSION)
COMMAND	SET DISPLAY MODE /STANDARD/FAST [=VERSION]

VERSION

Routine arg.	CHAR(*) VAR

Command par.	Set default=/STANDARD Change actual display version. Possible inputs are:

/STANDARD	Standard display version is used. The whole display functionality is supported.
/REDUCED	The reduced display version is activated. The internal graphic storage is not used, this reduces the CPU usage to 60% of the standard version. But not all commands are supported and several are only available with reduced functionality.

FUNCTION

Up to now you can select between two display versions. The STANDARD implementation, which supports all commands and the FAST version which works without any intermediate graphic storage.

The fast version saves CPU-time, it needs about 60% of the standard version CPU-time. It is recommended to use this reduced version when GOOSY should run on small VAX's like the 750 or a MICRO-VAX II and a CPU intensive analysis is active. Futhermore the fast display version do not need any disk storage for the intermediate graphic storage.

This advantages have to be payed with a reduced functionality:

1.) The PLOT PICTURE command is not supported.

2.) The SAVE DISPLAY command is not supported.

3.) The REFRESH command works only like REFRESH * /UPD.

4.) When you allocate one device which could not delete a specified section on the screen, the display surface of all devices will be cleared during the EXPAND command! E.g if an plotter or a metafile is allocated this will happen.

After the FAST version is selected the standard version could be activeted at any time, e.g. to plot a picture. But then first a new DISPLAY command must be given first!

SHOW DEVICES

SHOW DEVICE

PURPOSE	Show all allocated user devices

Remarks

File name	D$SHALL.PPL

Description

CALLING	STS=D$SHALL()
COMMAND	SHOW DEVICE

Function

Show all allocated graphical devices. The information on the devices allocated in the current session is fetched from the device description table and displayed. The logical device names, the physical address, the device type and category (input or output) and the GOOSY main device type are listed.

SHOW DISPLAY GLOBALS

```
SHOW DISPLAY GLOBALS
    /SPECTRUM
    /PICTURE
```

PURPOSE	Show global display parameter.
PARAMETERS	
/SPECTRUM	Global parameter for spectra are listed.
/PICTURE	Global parameter for pictures are listed.
Caller	MDISP,MGOODISP,D$DSPCM
Author	W. Spreng

Remarks

File name	D$SDIGL.PPL
Created by	GOO$DISP:D$DSPCM.PPL

Description

CALLING STS=D$SDIGL(I_spectrum,I_picture)

COMMAND

```
SHOW DISPLAY GLOBALS
    /SPECTRUM
    /PICTURE
```

SPECTRUM

Routine arg. Input BIN FIXED(15) valid values are 0 and 1.

Command par. Switch

The global parameters for spectra, define by the DEFINE DISPLAY SPECTRUM command are listed.

PICTURE

Routine arg.	Input BIN FIXED(15) valid values are 0 and 1.
Command par.	Switch

The global parameters for spectra, define by the DEFINE DISPLAY PICTURE command are listed.

FUNCTION

The global display parameters are listed as defined with the DEFINE DISPLAY SPECTRUM and DEFINE DISPLAY PICTURE command.

SHOW MAPPING

SHOW MAPPING base area

PURPOSE	Show mapping of a Data Base
PARAMETERS	
base	Data Base name required common default
area	optional area string optional
Caller	M$SMAPCM
Author	H.G.Essel
File name	M$ASHMAP.PPL
Dataset	-
EXAMPLE	SHO MAP DB

DESCRIPTION

CALLING	STS=M$ASHMAP(cv_dbnam,CV_area)
ARGUMENTS	
cv_dbnam	I Name of Data Base. A logical name translation will be performed. CHARACTER (*) VAR
CV_area	I optional area string. All areas containing the string will be listed. CHARACTER (*) VAR
FUNCTION	The routine will show the mapping of all areas. When an area is mapped, the name is displayed. The access (read or write) is displayed. No change in mapping is done.
REMARKS	Names will be converted to upper case characters only.

START SCATTER

```
START SCATTER
    /SYNCHRONOUS /ASYNCHRONOUS [=/MODE]
```

PURPOSE	Start scatter plots for actual picture
PARAMETERS	
MODE	Transfer mode of scatter data.

/SYNCHRONOUS The analysis waits for request from the display process.

/ASYNCHRONOUS The analysis does not wait for a request from the display process.

Caller	MDISP, MGOODISP
Author	W. Spreng

Remarks

Created by	D$DSPCM.PPL
File name	D$GO.PPL

Description

CALLING	STS=D$GO(CV_mode,B_mask)
COMMAND	START SCATTER /SYNCHRONOUS/ASYNCHRONOUS [= MODE]

/MODE

Routine arg.	Input CHAR(*) VAR

Command par. Switch default = /SYNCHRONOUS

Determines the transfer mode of the scatter data from the analysis to the display process.

/SYNCHRONOUS The analysis waits for request from display process and continues if the scatter data are accepted by the display. This modes slows down the analysis.

/ASYNCHRONOUS Analysis continues when a buffer has been sent to the display process. The next buffer will be sent after the request from the display process. In that mode the scatter plot does not influence the analysis.

FUNCTION

To start the scatter plots in the picture which is actually displayed on the screen. The freeze bit in the picture header has to be resetted.

STOP SCATTER

STOP SCATTER

PURPOSE	Stop scatter plots for actual picture
Caller	MDISP, MGOODISP
Author	W. Spreng

Remarks

Created by	D$DSPCM.PPL
Dataset	D$STOP.ppl

Description

CALLING	STS=D$STOP()
COMMAND	STOP SCATTER

Function

To stop the scater plots in the picture which is actually displayed on the screen. The freeze bit in the picture header has to be set.

UPDATE FRAMES

```
UPDATE FRAMES frames seconds
                  /REFRESH
```

PURPOSE	Update frames in the actual picture on screen
PARAMETERS	
frames	Array of frame numbers, e.g: 1,5,3,8
seconds	time difference betwen two updates
/REFRESH	refresh the spectrum data.
Caller	MDISP, MGOODISP
Author	W. Spreng

Example

UPDATE FRAMES 1,4,3,8,5 10

The contents of the spectra in frame 1,3,4,5,8 are displayed every 10-seconds.

Remarks

File name	D$DUPD.PPL
Created by	GOO$DISP:D$DSPCM.PPL

Description

CALLING	STS=D$DUPD(LA_frames,l_time,I_refresh,La_array, B_mask)
COMMAND	UPDATE FRAMES frames seconds /REFRESH

FRAMES

Routine arg. (*) BIN FIXED(31)

Command par. Integer array required

List of frames for which an update should be performed.

SECONDS

Routine arg. Input BIN FIXED(31)

Command par. Integer required

Time in seconds between two updates.

REFRESH

Routine arg. Input BIN FIXED(15)

Command par. Switch

If set the specified frames are refreshed. That means the old spectrum is deleted and the new data are drawn into the existing frame.

Function

The spectrum contents in the current active picture are updated every 'seconds'. Specified Scatterframes or two dimensional spectra are ignored. The new spectrum contents are overlayed into the existing frames.

ZOOM FRAME

```
ZOOM FRAME frame picture dyn_scat
           pic_dir dyn_dir buffer_size base node
```

PURPOSE	Zoom one frame of a picture.
PARAMETERS	
frame	Frame number as displayed on screen
picture	Name of picture which contains the frame to be zoomed.
dyn_scat	Name of dynamic list for scatter plot
pic_dir	Default picture Directory
dyn_dir	Default Directory name for dynamic list
buffer_size	Number of scatter points in scatter buffer
base	Default Data Base
node	Default node name
Caller	MDISP, MGOODISP
Author	W .Spreng

Remarks

File name	GOO$DISP:D$ZOOM.PPL
Created by	GOO$DISP:D$DSPCM.PPL

Description

CALLING STS=D$ZOOM(L_frame,CV_picture,CV_dyn_scat,
CV_pic_dir,CV_dyn_dir,CV_buffer_size
,CV_base,CV_node)

COMMAND ZOOM FRAME frame picture dyn_scat
pic_dir dyn_dir buffer_size base node

FRAME

Routine arg. Input BIN FIXED(31)

Command par. Integer required

Number of the frame which should be zoomed.

PICTURE

Routine arg. Input CHAR(*) VAR

Command par. String required

Name of the picture from which the specified frame should be zoomed. If not specified the actual active picture is used.

DYN_SCAT

Routine arg. Input CHAR(*) VAR

Command par. String global replaceable default=$SCATTER

Name of dynamic list used to create the entries necessary for the scatter plot. The SCATTER entry in this dynamic list is necessary to tell the analysis which scatterplots are requested by your display.

ATTENTION The dynamic list has to be attatched before a scatterplot could be created.

PIC_DIR

Routine arg. Input CHAR(*) VAR

Command par. String global replaceable default=$PICTURE

Default Directory for pictures.

DYN_DIR

Routine arg. Input CHAR(*) VAR

Command par. String global raplacable default=$DYNAMIC

Default Directory name for dynamic list

BUFFER_SIZE

Routine arg. Input CHAR(*) VAR

Command par. String global replace default=1000

Number of scatter points in scatter buffer.

BASE

Routine arg. Input CHAR(*) VAR

Command par. String global replaceable default=DB

Default Data Dase.

NODE

Routine arg. Input CHAR(*) VAR

Command par. String global replaceable default=*

Default node for Data Base.

Function

The specified frame in the specified picture is displayed on the screen. It is not necessary that the picture is displayed on the screen. If the picture is omitted the current active picture is assumed.

Index

Contents