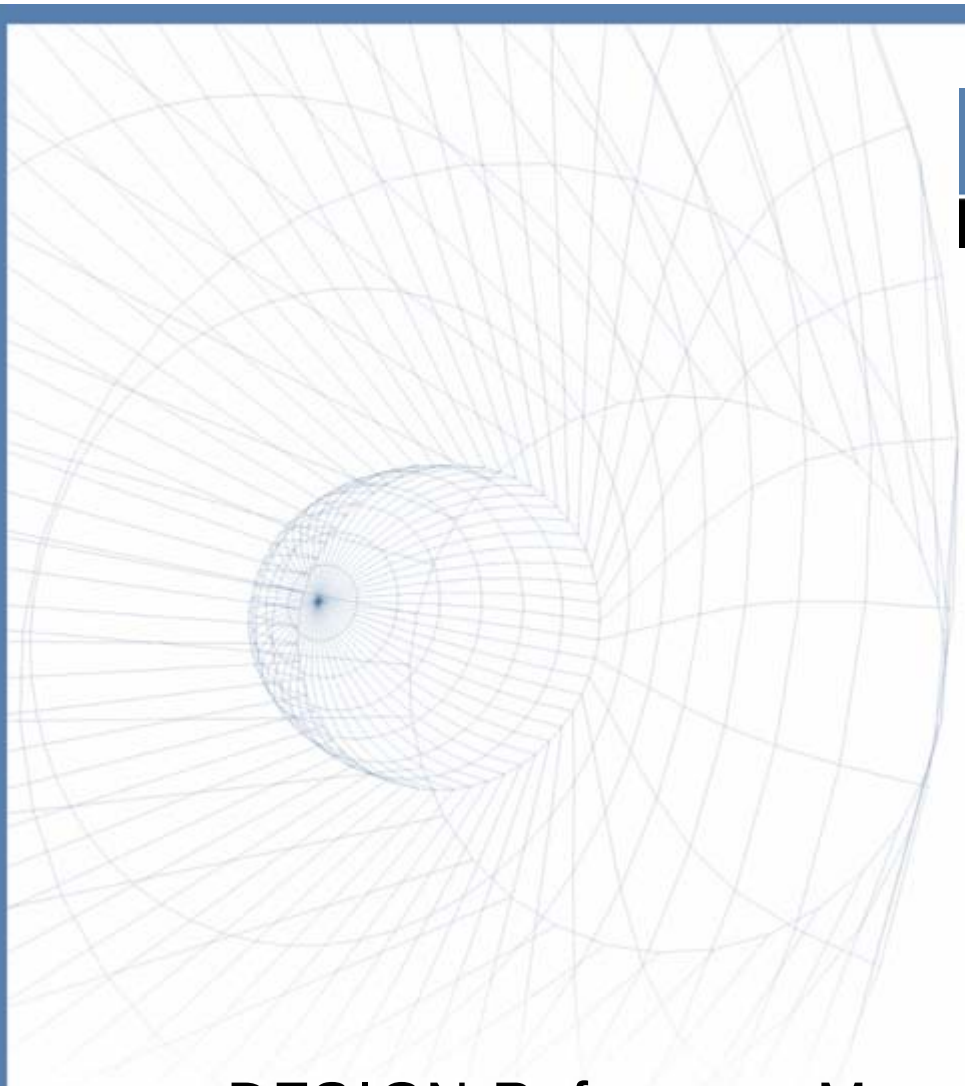


DESIGN Reference Manual

Utilities

www.aveva.com

Disclaimer

Copyright

First published September 2007

AVEVA Solutions Ltd, High Cross, Madingley Road, Cambridge, CB3 0HB, United Kingdom

Trademarks

Outfitting Design Reference Manual

Contents **Page**

Utilities

1 Introduction

1.1 About this Manual

The *Outfitting Design Reference Manual* describes all the Outfitting Design commands in detail. It also describes how the Outfitting Design database is structured, the Outfitting Design database elements and their attributes.

Outfitting Design is normally used interactively. The Graphical User Interface (GUI) provides discipline-based applications that help you to create, check and report on the model. How to use the applications is described in user guides and on-line help.

This manual is written for experienced users of Outfitting Design who need to use commands; for example, to write batch macros or to customise the GUI. If you are going to customise the GUI, you will also need to refer to the *Software Customisation Guide* and *Software Customisation Reference Manual* for information about PML, the AVEVA programming language.

1.2 Organisation of the Outfitting Design Reference Manual

The *Outfitting Design Reference Manual* has three parts:

- Part 1, *General Commands*, describes general Outfitting Design commands, which are used, for example, for setting up the display, and querying and navigating around the Outfitting Design database.
- In particular, it describes how to use the command **syntax graphs**, which are used to show all the options available for each command.
- Part 2, *Creating the Model,* describes the commands for creating database elements and setting their attributes.
- Part 3 (this volume) describes the Outfitting Design Utilities for data consistency checking and clash detection, and for exporting Outfitting Design data to programs such as Review.

For a comprehensive list of all Outfitting attributes and pseudo-attributes, see the *Software Customisation Reference Manual*.

1.3 Organisation of this Manual

This manual, Part 3, is divided into the following chapters:

- *Data Consistency Checking* explains how to check *Data Consistency.*
- *Clash Detection* describes the *Clash Detection* facilities.
- *Copying Model Data from Outfitting to Review* gives details of the EXPORT command, which *Copies Model Data from Outfitting to* Review.

2 Data Consistency Checking

This chapter describes the commands used for the Outfitting Design Data Consistency Checking Utility (DATACON). The checks include the following:

1. Adjacent items are connected and no gaps exist.
2. Connection types are compatible.
3. Connected components are not skewed with respect to one another.
4. Pipe bores or hanger rod diameters are consistent.
5. Branch and equipment connections are properly terminated.
6. Hangers are correctly connected to Fittings and Attas.
7. Tubes or rods joining components are not less than minimum acceptable lengths.
8. Angles of pulled Bends and Elbows fall within the limits set in the relevant Specifications.
9. Structural Sections and Joints are correctly positioned with respect to each other and are properly connected.
10. Lengths of structural Sections fall within predefined ranges.

The commands for checking for data inconsistencies fall into three general categories:

- Those which specify the design areas to be checked and how diagnostic messages will be output.
- Those which specify the maximum misalignments or positional errors allowed between adjacent components. No errors will be reported below these limits.
- Those which specify minimum lengths of tube between adjacent piping components (normally dependent upon the tube diameter) or for rod between adjacent hanger components, the maximum acceptable angle for pipe bends, and the minimum and maximum lengths of structural Sections.

2.1 Specifying the Checking Procedures

2.1.1 Specifying the Output Device for Diagnostic Messages

Keywords:

ALPHA FILE (APPEND OVERWRITE)

Description:

Before you initiate a data-checking operation you must specify whether the resulting output is to be sent to your terminal or to a file. The output device must be defined *before* the detailed checking is carried out, using the standard ALPHA FILE commands described in Part 1 of the *Outfitting Design Reference Manual.*

2.1.2 Checking Parts of the Design

Keywords:

CHECK ATTACHECK

Description:

The CHECK command initiates a full Component-by-Component data consistency check of specified parts of the Design. Up to 10 elements may be specified in a single CHECK command.

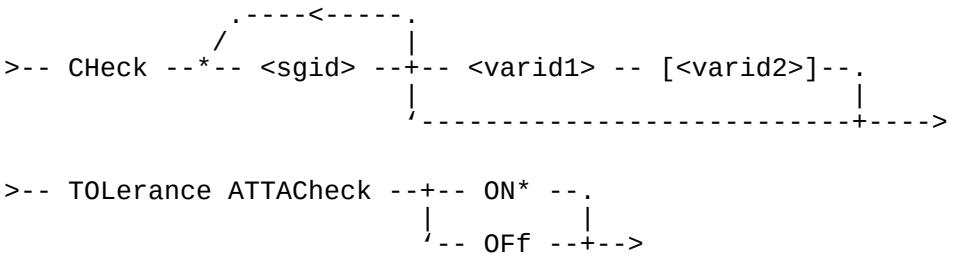
ATTAs will be checked in the same way as normal components if the ATTACHECK ON option is in force. They will be ignored if ATTACHECK OFF is in force.

You can store error references and (optionally) error codes in PML arrays, if required. If you do this, you can then use the ENHANCE command to highlight the problems on the display. See the *Software Customisation Guide* for more information on PML and arrays.

Examples:

`CHECK /ZONE-1`	Generates full consistency report on every component in the named Zone
`CHECK /BRAN1 / BRAN2 /BRAN3`	Generates selective report on the named Branches only
`TOL ATTACHECK OFF`	ATTAs will be ignored in subsequent checks

Command Syntax:



Where <varid1> is the PML array used to store references of any elements with errors, and the (option) <varid2> is the array that stores the error codes.

Querying:

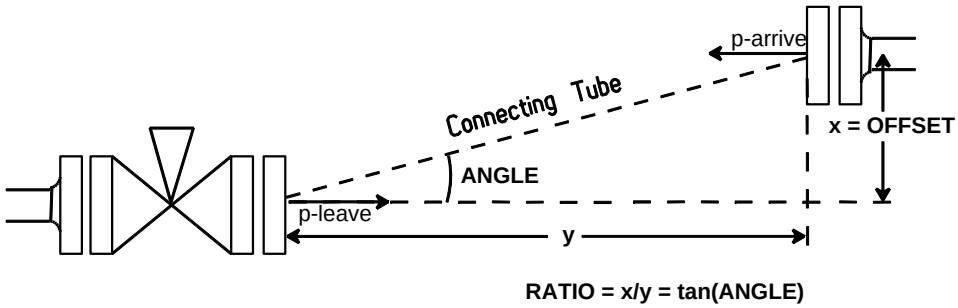
```
>-- Query TOLerance ATTACheck -->
```

2.2 Setting Geometric Error Limits

2.2.1 Piping/Hangers

Note: All references to pipe components and tube in this section apply equally to hanger components and rod.

The extent of the misalignment between two adjacent piping components may be measured using any of three parameters: the **offset** distance between their respective p-arrive and p-leave axes; the displacement **angle** between their respective p-arrive and p-leave axes; or the **ratio** of the offset to the projected distance between the arrive and leave p-points (which is equivalent to the tangent of the angle parameter). See Figure below.



Keywords:

TOLERANCE OFFSET ANGLE RATIO MAXANGLE

Description:

The TOLERANCE commands specify the maximum offset, angle or ratio misalignments that will be allowed between adjacent components before a diagnostic message is output.

Examples:

`TOL OFFSET 1 (Default)`	Maximum pipe misalignment is 1 mm offset
`TOL OFF 1/4 INCH`	Maximum pipe misalignment is 0.25 inch offset
`TOL ANGLE 0.0573 (Default)`	Maximum angular misalignment is 0.0573 degrees (i.e. 0.01 radians)
`TOL RATIO 0.001 (Default)`	Maximum offset/separation ratio is 0.001
`TOL ANG 1.5` `TOL RAT 0.0262`	These are equivalent angular settings, since 0.0262 is the tangent of the angle 1.5 degrees
`TOL DEF`	Resets all misalignment tolerances (Offset, Angle and Ratio) to their default values
`TOL MAXANG 90 (Default)`	Maximum permitted design angle for pulled bends and elbows can be set to values from 0° (straight tube) to 180° (U-bends).

Command Syntax:

```
>-- TOLerance --+-- OFFset --.
                |             |
                |-- ANGle ---|
                |             |
                |-- RATio ---+-- <uval> -->
                |
                |-- DEFault -->
                |
                  '-- MAXANGle -- value -->
```

Querying:

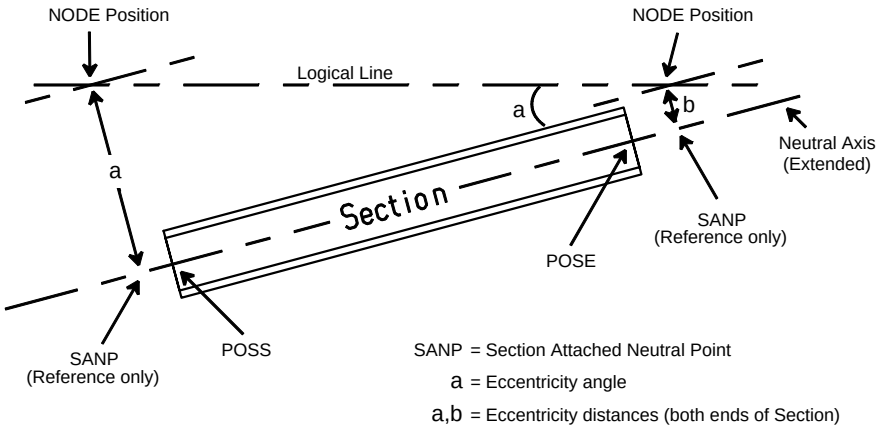
```
>-- Query TOLerance --+-- OFFset ----.
                      |             |
                      |-- ANGle -----|
                      |             |
                      |-- RATio -----|
                      |             |
                        '-- MAXANGle --+-->
>-- Query TOLerance OPTions -->
```

(also outputs ATTACHECK and TUBE settings)

2.2.2 Structural Sections

Misalignments of Sections in structural designs are defined in similar terms to those used for pipework, namely in terms of an offset **distance**, a misalignment **angle** or a **ratio** (the tangent of the angle). These are together referred to as the Section's **eccentricity**.

Distance and angular offsets are measured from the logical line between Nodes, while additional checks are made using the attached Neutral Points of the Section and the corresponding Nodes. The terminology is illustrated in the following Figure.



Keywords:

ECCENTRICITY SECTION DISTANCE ANGLE RATIO (TOLERANCE)

Description:

The ECCENTRICITY SECTION commands specify the maximum offset distance, angle or ratio misalignments that will be allowed between a Section's Neutral Axis and the Logical Line between Nodes before a diagnostic message is output.

Examples:

`ECC SECT DIST 1 (Default)`	Maximum offset for either end of Section is 1 mm
`ECC SECT DIST 0.5 INCH`	Maximum offset for either end of Section is 0.5 in
`ECC SECT ANGLE 0.0573 (Default)`	Maximum angular misalignment is 0.0573 degrees
`ECC SECT RATIO 0.001 (Default`	Maximum offset/node separation ratio is 0.001
`ECC SECT ANG 1.5`	These are equivalent angular settings, since
`ECC SECT RAT 0.0262`	0.0262 is the tangent of the angle 1.5 degrees
`ECC SECT DEF`	Resets all misalignment eccentricities (Offset distance, Angle and Ratio) to their default values

Command Syntax:

```
>-- ECCentricity -- SECTion --+-- DISTance --+-- <uval> -->
                              |               |
                              |               '-- DEFault -->
                              |
                              |-- ANGle --.
                              |           |
                              |-- RATio --+-- value -->
                              |           |
                              |           '-- DEFault -->
                              |
                              '-- DEFault -->
```

Querying:

```
Q TOLerance ECCentricity --- SECTion ---+--- DISTance ---.
                                        |               |
                                        |--- ANGle ------|
                                        |               |
                                        |--- RATio ------|
                                        |               |
                                        '----------------+--->
```

Note: The TOLERANCE keyword is used here to distinguish between the eccentricity limit that you have specified and the *actual* eccentricity derived from the structural model.

2.2.3 Structural Joints

The positional errors for Joints are measured as divergences of the Joint positions from their owning Nodes. The maximum permissible errors are defined, as for Sections, in terms of a Joint's **eccentricity**.

Note: Joint eccentricities are defined *separately* for Pjoint (Primary Joint) and Sjoint (Secondary Joint) elements.

Keywords:

ECCENTRICITY PJOINT SJOINT

Description:

The ECCENTRICITY PJOINT and ECCENTRICITY SJOINT commands specify the maximum permissible distance between the position of a Primary Joint or a Secondary Joint, respectively, and that of its owning Node before a diagnostic message is output.

Examples:

`ECC PJOINT 1 (Default)`	Maximum error in Primary Joint position is 1 mm
`ECC SJOINT 1 (Default)`	Maximum error in Secondary Joint position is 1 mm
`ECC PJOIN 0.1 INCH`	Maximum error in Primary Joint position is 0.1 inch
`ECC PJOIN DEFAULT`	Resets permissible Primary Joint error to 1 mm
`ECC SJOIN DEF`	Resets permissible Secondary Joint error to 1 mm

Command Syntax:

```
>--- ECCentricity ---+--- PJOInt --.
                     |              |
                     '--- SJOInt --+--- <uval> ----.
                                   |               |
                                      '--- DEFault -----+--->
```

Querying:

```
Q ECCentricity ---+--- PJOInt --.
                  |             |
                    '--- SJOInt ---+--->
```

2.2.4 Structural Fittings

The positional errors for Fittings are measured as divergences of the Fitting positions from the Neutral Points of the attached Sections. The maximum permissible errors are defined, as for Sections, in terms of a Fitting's **eccentricity**.

Keywords:

ECCENTRICITY FITTING

Description:

The ECCENTRICITY FITTING command specifies the maximum permissible distance between the position of a Fitting and that of its attached Section before a diagnostic message is output.

Examples:

`ECC FITTING 1 (Default)`	Maximum error in Fitting position is 1 mm
`ECC FITT 0.5 INCH`	Maximum error in Fitting position is 0.5 inch
`ECC FITT DEF`	Resets permissible Fitting position error to 1 mm

Command Syntax:

```
>--- ECCentricity --- FITTing ---+--- <uval> -----.
                                 |                |
                                 '--- DEFault ----+--->
```

Querying:

```
Q ECCentricity FITTing
```

2.3 Minimum Tube and Rod Lengths and Maximum Bend Angles

Note: All references to pipe components and tube in this section apply equally to hanger components and rod.

In order to warn you about potential problems in the practical assembly of pipework sections, the data consistency checks output a diagnostic message if the length of tube joining any pair of components is less than a prescribed minimum. The minimum allowed will normally be dependent upon the tube diameter. An incorrectly positioned ATTA (off the implied tube) may also give rise to a 'minimum tube length' error.

You will also be warned if the design angle for a variable-angle bend or elbow exceeds a prescribed maximum.

2.3.1 Specifying Minimum Lengths Irrespective of Bore

Keywords:

TOLERANCE TUBE

Description:

The TUBE command, used alone, specifies the minimum acceptable tube length which is applicable to all tube diameters which have not been more specifically set.

Examples:

`TOL TUBE 100 (Default)`	Minimum acceptable length is 100 mm
`TOL TUBE DEFAULT`	Minimum acceptable length is 100 mm
`TOL TUBE 6 INCH`	Minimum acceptable length is 6 inches

Command Syntax:

```
>-- TOLerance TUbe --+-- <uval> ---.
                     |              |
                     '-- DEfault --+-->
```

Querying:

```
>-- Query TUbe -->
>-- Query TOLerance OPTions -->
```

(also outputs ATTACHECK and misalignment settings; see *Checking Parts of the Design* and *Piping/Hangers*, respectively)

2.3.2 Specifying Bore-Dependent Minimum Lengths

Keywords:

TOLERANCE TUBE BORE MINIMUM

Description:

The TUBE BORE command allows you to specify different minimum tube lengths for different ranges of bore diameter. Each range of tube size is specified by a minimum and maximum bore (not necessarily in that order), followed by the minimum tube length allowed for that range.

Up to ten different ranges may be specified. If a tube diameter falls outside any specified range, then the current default length is applied *(*100 mm, unless this has been overridden by prior use of the TUBE <uval> command).

If two or more ranges overlap, you are warned but the ranges are not rejected. Tube length checks are applied in the order in which they have been specified. Thus, if the bore of a section of tube lies within more than one specified range, its length will pass or fail the tolerance test determined by the first valid range only.

Example:

`TOL TUBE BORE 15 25 MIN 150`	All tubes with bores in the range 15-25 mm must be at least 150 mm long
`TOL TU BO 25 15 150`	The same as the preceding example. The maximum bore is given *before* the minimum bore and the optional MINIMUM (length) command word has been omitted.
`TOL TU BO 2 IN 4 IN MIN 12 IN`	All tubes with bores in the range 2-4 inches must be at least 12 inches long
`INCH DIST` `TOL TU BO 2 4 MIN 12`	The same as the preceding example, except that the current distance units are now inches
`TOL TUBE DEFAULT`	Resets minimum length to 100 mm for *all* bore sizes regardless of any ranges previously defined separately

Complex Example:

The following sequence:

```
TOL TUBE  500
TOL TUBE  BORE  15  25  MINIMUM  100
TOL TU  BO  25  50  MIN  150  BO  50  100  MIN  300
TOL TU  BO  200  400  750
```

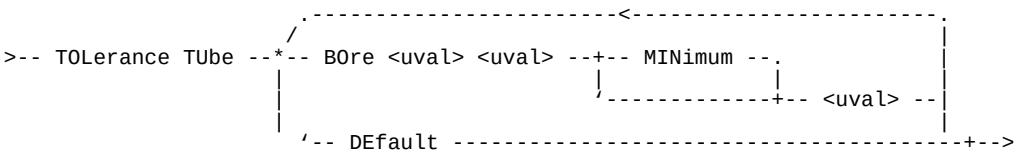
would cause subsequent checks to report tube lengths less than the permitted minima in the following circumstances:

For bores between 15 mm and 25 mm, if length is <100 mm

- For bores between 25 mm and 50 mm, if length is <150 mm
- For bores between 50 mm and 100 mm, if length is <300 mm
- For bores between 200 mm and 400 mm, if length is <750 mm
- For all other (unspecified) ranges, if length is <500 mm

The latter case would apply to bores less than 15mm, greater than 400mm, and within the range 100mm to 200mm.

Command Syntax:



Querying:

```
>-- Query TUbe -->
>-- Query TOLerance OPTions -->
```

(also outputs ATTACHECK and misalignment settings)

2.3.3 Specifying Maximum Angles for Bends and Elbows

Keywords:

MAXANGLE

Description:

The design angle for each pulled bend or elbow, taken from the setting of its ANGL attribute, is checked against a predefined maximum angle. A diagnostic message is output if the design angle is too large.

The default setting for the maximum permitted angle is 90°, but you may specify any limit within the range 0° (i.e. straight tube only) to 180° (i.e. U-bends allowed).

Example:

MAXANGLE 45

Command Syntax:

```
>--- MAXANGLE --- value --->
```

Querying:

```
Q MAXANGle
```

2.4 Combined Pipework Tolerance and Tube Length Options

Keywords:

TOLERANCE OPTIONS DEFAULT

Description:

The OPTIONS command allows you to reset defaults for, or query, all data consistency checking options for Tubes at the same time.

Example:

`TOL OPTIONS DEFAULT`	Resets the following data-checking parameters: • Offset tolerance to 1 mm • Angle tolerance to 0.0573 degrees • Ratio tolerance to 0.001 • Maxangle setting to 90 degrees • Minimum tube length for all bores to 100 mm • ATTACHECK option to ON

Command Syntax:

```
>-- TOLerance OPTions DEFault -->
```

Querying:

```
>-- Query TOLerance OPTions -->
```

Outputs the current settings for all data checking parameters. The parameters are output in the current units.

Querying Examples:

With all default settings in force, the output resulting from a Q TOL OPT command will show one of the following sets of values (depending upon the current units):

MM Bores and Distances:

```
Minimum tube length : 100 mm
Tolerance Offset    : 1 mm
Tolerance Angle     : 0.0573
Tolerance Ratio     : 0.001
Maximum Angle       : 90
Atta checking       : ON
```

INCH Bores and Distances:

```
Minimum tube length : 3.9 in
Tolerance Offset    : 0 in
Tolerance Angle     : 0.0573
Tolerance Ratio     : 0.001
Maximum Angle       : 90
Atta checking       : ON
```

FINCH Bores and Distances:

```
Minimum tube length : 3.15/16
Tolerance Offset    : 0.1/32
Tolerance Angle     : 0.0573
Tolerance Ratio     : 0.001
Maximum Angle       : 90
Atta checking       : ON
```

2.5 Specifying Minimum and Maximum Section Lengths

The derived lengths of structural Sections are checked against predefined minimum and maximum values and a diagnostic message is output if an actual length falls outside these limits.

Keywords:

SECTION DEFAULT RESET

Description:

The SECTION command allows you to specify minimum and maximum acceptable lengths for Sections.

You may set a single range of acceptable lengths for *all* unspecified types of Section by using the SECTION DEFAULT option, or you may set different ranges for one or more specific types of Section (up to a maximum of ten named types). The standard default range of permissible lengths for all unspecified types of Section is from zero minimum to 10000 mm maximum.

Examples:

`SECTION DEFAULT 1000 9500`	Lengths of all types of Section must be in the range 1000-9500 mm unless separately specified by a SECT *generic_type* option (as illustrated in the following examples)
`SECT COLUMN 1500 7500`	Column lengths must be in the range 1500-7500 mm; lengths of other types of Section must be within the current default range
`SECT BEAM 42 IN 9 FT`	Beam lengths must be in the range 42 inches to 9 feet; lengths of other types of Section must be within the current default range
`SECT COLUMN 0 12500 BEAM 8750 1000`	Column lengths must be in the range 0-12500 mm; Beam lengths must be in the range 1000-8750 mm; lengths of all other types of Section must be within the current default range

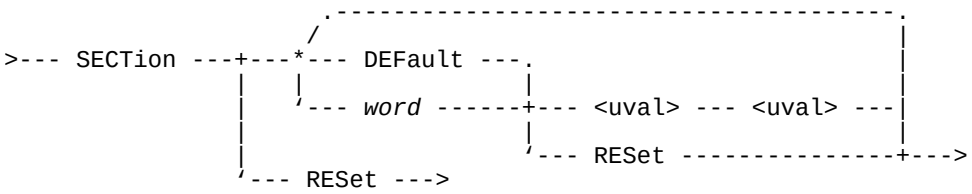
Note: As illustrated in the last example, you must always enter *two* values (a minimum and a maximum setting), even if one of these is a current default value (0 mm, say). You may, however, enter them in either order (i.e. minimum value first or maximum value first).

`SECT DEF RESET`	Lengths of all types of Section must be in the range 0-10000 mm unless separately specified by a SECT *generic_type* option (i.e. this command resets the original default settings)
`SECT COLUMN RESET`	Column lengths must be in the range 0-10000 mm; the current settings for other types of Section are not changed by this command

Examples:

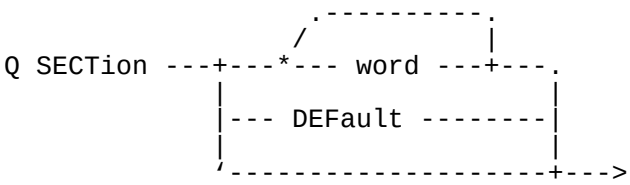
`SECT COL RESET BEAM RESET`	Column and Beam lengths must be in the range 0-10000 mm (Note that the RESET command word must be repeated for each type of Section)
`SECT RESET`	Lengths of *all* types of Section must be in the range 0-10000 mm (i.e. this resets the original default settings and cancels any individual settings for specific types of Section)

Command Syntax:



where *word* is any valid Section GTYPE which conforms with the Outfitting Design DB.

Querying:



Querying Examples:

`Q SECT DEFAULT`	Gives settings for current default range
`Q SECT BEAM`	Gives settings for Beams only
`Q SECT BEAM STRU COL`	Gives settings for the three named types of Section
`Q SECT`	Gives settings for any specifically-defined types plus default settings applicable to other Sections

2.6 Specifying User-Defined SKEYs Acceptable in Data Checks

The symbol used by ISODRAFT to represent a particular piping component on an isometric drawing is determined by its **symbol key** (SKEY attribute setting). (See the ISODRAFT *Reference Manual* for a full explanation of this concept.)

Keywords:

TOLERANCE CATALOGUE SKEY

Description:

When a data consistency check is carried out, the SKEY (if any) for each component is checked to see if it is of a standard type. This syntax lets you specify user-defined SKEYs so that they do not generate errors during data consistency checks.

Example:

`TOL CATA SKEY 'JIM' 'FRED' ...`	Components with the user-defined SKEYs 'JIM', 'FRED' etc. will not generate errors caused by unrecognised SKEYs.

Command Syntax:

```
                                .---------.
                               /           |
>-- TOLerance CATAlogue SKEY --*-- text ---+-->
```

Note: *Text* is case-sensitive; SKEYs are usually, but not necessarily, uppercase characters.

Querying:

```
>-- Query TOLerance CATAlogue SKEY -->
```

Outputs current list of user-defined SKEYs to be ignored during data consistency checks.

2.7 Diagnostic Messages from Data Consistency Checks

When the CHECK command is given, Outfitting Design scans the relevant DBs to extract the appropriate Design and Catalogue information and carries out the detailed checking operations described in the preceding sections. Whenever a design inconsistency or error is found, a diagnostic message is output to the screen (REQUESTS region) or to a file. It is these messages which are explained in this section.

Note: The diagnostic messages will often incorporate specific references (name, reference number etc.) to the elements found to be in error (although the true errors may be due to adjacent elements). **These specific references have generally been omitted from the example messages listed in the following subsections.**

If the checking procedures are completed without any errors being detected, the message

```
*--*  NO DATA INCONSISTENCIES  *--*
```

will be output.

2.7.1 Global Diagnostics

R 10 BAD OWNER REFERENCE

The owner reference refers either to a non-existent element or to one that does not contain the required element in its list part. The occurrence of this error implies that corruption of one or more DBs has taken place.

2.7.2 Branch-Specific Diagnostics

- **Branch Head Errors**

The following diagnostics apply only to the Head of a Branch:

A 10 HEAD REFERENCE NOT SET

The Head reference should only be unset (i.e. null) if the Head Connection Type HCONN is set to OPEN, VENT, CLOS or DRAN.

A 20 HEAD REFERENCE POINTS TO NONEXISTENT ELEMENT

This error would result from the deletion of a component, such as a Nozzle, to which the Head of the Branch was originally connected.

A 30 BAD HEAD RETURN REFERENCE

The Head is connected to an element that does not refer back to the Branch. This can occur when the Head of a Branch is connected to another Branch, implying that a Tee should be placed somewhere along the second Branch. The error can also occur when two or more branches are inadvertently connected to the same terminal.

A100 HEAD TERMINAL PROBLEM WITH SPREF

Either the Head terminal has an unset SPREF or the SPREF points to a non-existent SPCOM.

A110 HEAD TERMINAL PROBLEM WITH CATREF

Either the Head terminal has an unset CATREF or the CATREF points to a non-existent Catalogue component.

A120 HEAD TERMINAL PROBLEM IN ACCESSING P-POINTS

There is a Catalogue problem in accessing the p-points of the Head terminal.

A130 HEAD TERMINAL PROBLEM, P-POINT NO.n DOES NOT EXIST

A required p-point of the Head terminal does not exist in the set of p-points.

A140 HEAD TERMINAL PROBLEM INVALID FLOW ATTRIBUTE ON CONNECTED BRANCH

This is testing for flow with Branch to Branch connections. Branches have a FLOW attribute which can be set to FORW or BACK or NUL. If FLOW is not set correctly this results in an error

A150 HEAD TERMINAL PROBLEM, TERMINAL INCONSISTENT FLOW ACROSS BRANCH CONNECTION

The FORW flow means that the fluid in the Branch goes from Head to Tail with the reverse for BACK. The code then checks that the fluid flows correctly from one Branch to the next. To fix the problem check the network to see that all the flows are in the correct direction.

A200 DIRECTION HDIR NOT SAME AS TERMINAL DIRECTION

If the Head is connected to a terminal, such as a Nozzle or Tee, then the direction HDIR should always be identical to that of the appropriate p-point of the terminal.

A210 POSITION HPOS NOT SAME AS TERMINAL POSITION

If the Head is connected to a terminal, such as a Nozzle or Tee, then the position HPOS should always be identical to that of the appropriate p-point of the terminal.

A220 HBORE NOT SAME AS TERMINAL BORE

If the Head is connected to a terminal, such as a Nozzle or Tee, then the bore HBORE should always be identical to that of the appropriate p-point of the terminal.

A230 CONNECTION TYPE HCONN NOT SAME AS TERMINAL CONNECTION TYPE

If the Head is connected to a terminal, such as a Nozzle or Tee, then the connection type HCONN should always be identical to that of the appropriate p-point of the terminal.

A300 REFERENCE HSTUBE UNSET

There is more than 1mm of tube between the Head and the p-arrive of the first Component (or the Tail), but HSTUBE is unset.

A310 REFERENCE HSTUBE REFERS TO A NONEXISTENT SPCOM

This may occur if part of the Specification has been deleted.

A320 HSTUBE PROBLEM, CATREF IN SPCOM IS UNSET

This indicates an error in the Specification.

A330 HSTUBE PROBLEM, CATREF IN THE SPCOM REFERS TO NONEXISTENT Catalogue COMPONENT

This may occur if part of the Catalogue has been deleted or if the CATREF is unset.

A340 HSTUBE PROBLEM, GTYPE OF CATALOGUE COMPONENT IS NOT SET TO TUBE

The component pointed to by HSTUBE is not of type TUBE in the Catalogue.

A350 HSTUBE PROBLEM IN ACCESSING P-POINT

There is a Catalogue problem in accessing a p-point of the tube pointed to by HSTUBE.

A400 HBORE NOT SAME AS BORE OF HSTUBE

The bore of any tube leading from the Head, determined from the Catalogue, should always be identical to HBORE.

A410 HCONN NOT COMPATIBLE WITH CONNECTION TYPE OF HSTUBE

The connection type of any tube leading from the Head, determined from the Catalogue, should be compatible with HCONN.

A420 ISPEC REFERENCE POINTS TO NONEXISTENT ELEMENT

This error would occur if, for example, the Insulation Specification pointed to by ISPEC had been deleted.

A430 INSULATION CANNOT BE SELECTED USING HBORE

There is no suitable insulation in the Catalogue for the combination of temperature TEMP and bore HBORE.

- **Branch Tail Errors**

The following diagnostics apply only to the Tail of a Branch:

B 10 TAIL REFERENCE NOT SET

The Tail reference should only be unset (i.e. zero) if the Tail connection type TCONN is set to OPEN, VENT, CLOS or DRAN.

B 20 TAIL REFERENCE POINTS TO NONEXISTENT ELEMENT

This error would result from the deletion of a component, such as a Nozzle, to which the Tail of the Branch was originally connected.

B 30 BAD TAIL RETURN REFERENCE

The Tail is connected to an element that does not refer back to the Branch. This can occur when the Tail of a Branch is connected to another Branch, implying that a Tee should be placed somewhere along the second Branch. The error can also occur when two or more branches are inadvertently connected to the same terminal.

B100 TAIL TERMINAL PROBLEM WITH SPREF

Either the Tail terminal has an unset SPREF or the SPREF points to a non-existent SPCOM.

B110 TAIL TERMINAL PROBLEM WITH CATREF

Either the Tail terminal has an unset CATREF or the CATREF points to a non-existent Catalogue component.

B120 TAIL TERMINAL PROBLEM IN ACCESSING P-POINTS

There is a Catalogue problem in accessing the Tail p-points.

B130 TAIL TERMINAL PROBLEM, P-POINT NO.n DOES NOT EXIST

A required p-point of the Tail terminal does not exist.

B140 TAIL TERMINAL PROBLEM INVALID FLOW ATTRIBUTE ON CONNECTED BRANCH

This is testing for flow with Branch to Branch connections. Branches have a FLOW attribute which can be set to FORW or BACK or NUL. If FLOW is not set correctly, this results in an error

B150 TAIL TERMINAL PROBLEM, TERMINAL INCONSISTENT FLOW ACROSS BRANCH CONNECTION'

The FORW flow means that the fluid in the Branch goes from Head to Tail with the reverse for BACK. The code then checks that the fluid flows correctly from one Branch to the next. To fix the problem check the network to see that all the flows are in the correct direction.

B200 DIRECTION TDIR NOT SAME AS TERMINAL DIRECTION

If the Tail is connected to a terminal, such as a Nozzle or Tee, then the direction TDIR should always be identical to that of the appropriate p-point of the terminal.

B210 POSITION TPOS NOT SAME AS TERMINAL POSITION

If the Tail is connected to a terminal, such as a Nozzle or Tee, then the position TPOS should always be identical to that of the appropriate p-point of the terminal.

B220 TBORE NOT SAME AS TERMINAL BORE

If the Tail is connected to a terminal, such as a Nozzle or Tee, then the bore TBORE should always be identical to that of the appropriate p-point of the terminal.

B230 CONNECTION TYPE TCONN NOT SAME AS TERMINAL CONNECTION TYPE

If the Tail is connected to a terminal, such as a Nozzle or Tee, then the connection type TCONN should always be identical to that of the appropriate p-point of the terminal.

- ### Plain Branch Errors

The following diagnostics can occur only for Branches with no piping components:

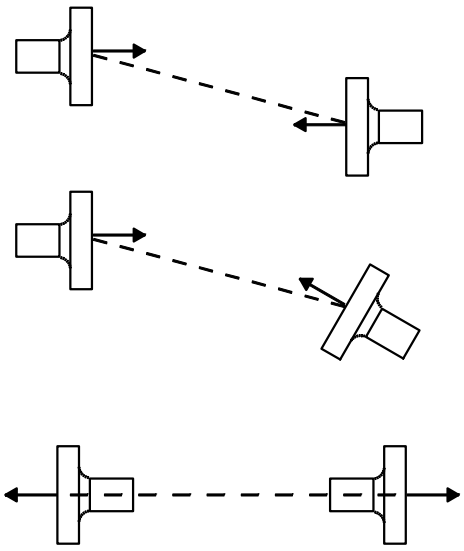
C500 TUBE BETWEEN HEAD AND TAIL LESS THAN TUBE MINIMUM

The distance between the Head position, HPOS, and the Tail position, TPOS, is greater than zero and less than the specified minimum tube length (default: 100mm).

C510 BAD HEAD TO TAIL GEOMETRY

Either the Head position, HPOS, does not lie at a positive distance along the line through TPOS in the direction TDIR or the Tail position, TPOS, does not lie at a positive distance along the line through HPOS in the direction HDIR.

The following illustration shows some typical examples:



C520 HBORE NOT SAME AS TBORE

When there are no components on the branch, the Head bore, HBORE, should be identical to the Tail bore, TBORE.

C530 HCONN IS NOT COMPATIBLE WITH TCONN

This implies that the Head is connected directly to the Tail with no Tube or piping components in between; hence the Head connection type, HCONN, must be compatible with the Tail connection type, TCONN.

C540 THIS BRANCH HAS NO COMPONENTS

This does not necessarily indicate an error. It is merely output as a warning to the designer.

2.7.3 Component-Specific Diagnostics

The following errors apply to individual piping components and, in some cases, to their adjacent connections. Some of the errors also apply to hanger components and/or to Nozzles.

- **All-Component Diagnostics**

These are applicable to any component, regardless of its position in the network:

D100 REFERENCE SPREF UNSET

This probably means that the designer has forgotten to select the piping component.

D110 REFERENCE SPREF REFERS TO A NON-EXISTENT SPCOM

This may occur if part of the specification has been deleted.

D120 SPREF PROBLEM, CATREF IN SPCOM IS UNSET

This indicates an error in the Specification.

D130 SPREF PROBLEM, CATREF IN THE SPCOM REFERS TO NON-EXISTENT CATALOGUE COMPONENT

This may occur if part of the Catalogue has been deleted or if the CATREF in the SPCOM is unset.

D140 SPREF PROBLEM, GTYPE OF CATALOGUE COMPONENT IS NOT SAME AS TYPE OF Outfitting Design DATA COMP0NENT

The GTYPE of the Catalogue component must be the same as the type of the piping Component in the design data.

D150 CATREF PROBLEM IN ACCESSING P-POINTS

An error has been found in accessing the p-points of the Catalogue component pointed to by CATREF.

D160 REFERENCE CATREF UNSET

This applies only to Nozzles, for which the CATREF must be set.

D170 REFERENCE CATREF REFERS TO A NON-EXISTENT CATALOGUE COMPONENT

This may occur if part of the Catalogue has been deleted. It applies only to Nozzles.

D200 ARRIVE P-POINT NOT IN PTSET OF CATALOGUE COMPONENT

The arrive p-point number does not exist in the PTSET of the Catalogue Component.

D210 LEAVE P-POINT NOT IN PTSET OF CATALOGUE COMPONENT

The leave p-point number does not exist in the PTSET of the Catalogue Component.

D300 REFERENCE CREF NOT SET

Multi-way Components may be left unconnected only if the connection type of the relevant p-point is OPEN, CLOS, VENT, DRAN or NULL.

D310 REFERENCE CREF POINTS TO NON-EXISTENT BRANCH

This may occur if the Branch that is pointed to by the CREF has been deleted.

D320 BAD CREF RETURN REFERENCE

This may occur if the Branch that is pointed to by the CREF has been reconnected to another terminal.

D330 REFERENCE CRFA n NOT SET

An entry in the connection reference array may only remain unset if the corresponding p-point in the Catalogue does not exist or if the connection type of the corresponding point is VENT, OPEN, CLOS or NULL.

D340 REFERENCE CRFA n POINTS TO NON-EXISTENT BRANCH

This may occur if the Branch pointed to by the nth reference in the connection reference array has been deleted.

D350 BAD CRFA n RETURN REFERENCE

This may occur if the element pointed to by the nth reference in the connection reference array has been reconnected to a third Branch.

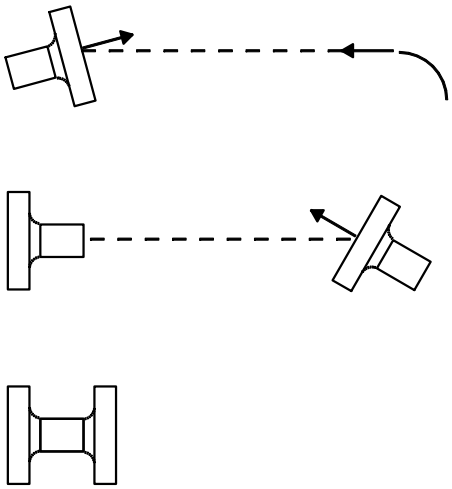
D400 ARRIVE TUBE [ROD] LESS THAN TUBE [ROD] MINIMUM. ACTUAL TUBE [ROD] LENGTH IS ...

The distance between the arrive p-point of this component and the leave p-point of the previous component (or Head) is greater than zero and less than the specified minimum tube [rod] length (default: 100mm).

D410 BAD ARRIVE GEOMETRY + details of geometric errors

The position and direction of the arrive p-point of this component are not correct with respect to the leave p-point of the previous component (or Head). The error could be caused by incorrect positioning of this component, the previous component (or Head) or both.

The following illustration shows some typical examples:



D420 BAD ARRIVE BORE [ROD DIAMETER]

The bore of the arrive p-point of this component is not equal to the bore of the preceding tube or, if this component is not preceded by tube, to the bore of the leave p-point of the previous component (or HBORE).

D430 BAD ARRIVE CONNECTION TYPE

The connection type of the arrive p-point of this component is not compatible with the preceding tube or, if this component is not preceded by tube, to the connection type of the leave p-point of the previous component (or HCONN).

D500 REFERENCE LSTUBE [LSROD] UNSET

The user has probably forgotten to select the piping/hanger Component.

D510 REFERENCE LSTUBE [LSROD] REFERS TO A NON-EXISTENT SPCOM

This may occur if part of the Specification has been deleted.

D520 LSTUBE [LSROD] PROBLEM, CATREF IN SPCOM IS UNSET

This indicates an error in the Specification.

D530 LSTUBE PROBLEM, CATREF IN THE SPCOM REFERS TO NON-EXISTENT CATALOGUE COMPONENT

This may occur if part of the Catalogue has been deleted or if the CATREF in the SPCOM is unset.

D540 LSTUBE PROBLEM, GTYPE OF CATALOGUE COMPONENT IS NOT SET TO TUBE

The component pointed to by LSTUBE is not of type TUBE in the Catalogue.

D550 LSTUBE [LSROD] PROBLEM IN ACCESSING PPOINTS

There is a Catalogue problem in accessing a p-point of the tube/rod pointed to by LSTUBE [LSROD].

D600 LEAVE BORE [DIAMETER] NOT SAME AS BORE [DIAMETER] OF LSTUBE [LSROD]

The bore of the leave p-point of this Component is not the same as the bore of the tube/rod following the Component.

D610 LEAVE CONNECTION TYPE NOT COMPATIBLE WITH CONNECTION TYPE OF LSTUBE [LSROD]

The connection type of the leave p-point of this Component is not compatible with the tube/rod following the component.

D620 INSULATION CANNOT BE SELECTED USING LEAVE BORE

There is no suitable insulation in the Catalogue for the combination of temperature TEMP and the bore of the leave p-point.

D630 ATTACHMENT TYPE INVALID - MUST BE ONE OF FLOW, XXXX, SSSS, CCCC, CCNN, INPP, WELD, HANG, PENI, NUL OR NULL

You have set an incorrect TYPE attribute for an ATTA.

D740 ANGLE OF COMPONENT IS GREATER THAN MAXIMUM ANGLE SPECIFIED IN SPREF

The designed angle of a Pulled Bend, Elbow or Weld is greater than is allowed by the SPECs.

D740 ANGLE OF COMPONENT IS LESS THAN MINIMUM ANGLE SPECIFIED IN SPREF

The designed angle of a Pulled Bend, Elbow or Weld is less than is allowed by the SPECs.

D740 ANGLE GREATER THAN m DEGREES. ACTUAL ANGLE IS n.

The designed angle of a Pulled Bend or Elbow is greater than the maximum angle specified by the user (see TOLERANCE MAXANGLE).

D800 Reference DETAIL unset.

This refers to the DETA attribute of the SPREF of the branch component.

Without it no SKEY will be available for that element within ISODRAFT.

D820 SKEY not set

D840 SKEY #N not known. Assumed to be user defined

(See the ISODRAFT *Reference Manual* for a list of valid SKEYs)

- **End-Component Diagnostics**

  These are applicable only to the last component in a Branch:

E700 LEAVE TUBE LESS THAN TUBE MINIMUM. ACTUAL TUBE LENGTH IS ...

The distance between the leave p-point of the current component and the tail position, TPOS, is greater than zero and less than the specified minimum tube length (default: 100mm).

E710 BAD LEAVE GEOMETRY

The position and direction of the leave p-point of this component are not correct with respect to the position, TPOS, and direction, TDIR, of the tail. The error could be caused by incorrect positioning of this component, the Tail, or both.

E720 LEAVE BORE NOT SAME AS TBORE

The bore of the leave p-point of this component is not the same as the tail bore, TBORE.

E730 LEAVE CONNECTION TYPE NOT COMPATIBLE WITH TCONN

The connection type of the leave p-point of this component is not compatible with the tail connection type TCONN.

2.7.4 Structure-Specific Diagnostics

The following abbreviations are used throughout this section to represent specific identifiers within the output messages:

struc_elem	A named structural element (a specific Section, Joint, or Fitting)
word	A qualifying Outfitting word; usually an attribute of a structural element in the current context

- **Catalogue/Connectivity Errors**

SC010 Catref/Specref of *struc_elem* unset

The SPREF for the named structural Section, Joint or Fitting does not point to a valid component in the Catalogue.

SC020 Pline *word* of *struc_elem* does not exist

The p-line of the given name cannot be accessed in the Catalogue description of the element. Either this p-line does not exist in the Catalogue description or the structural element points to the wrong Catalogue element.

SC030 POSL of *struc_elem* unset

The Positioning Line (POSL) attribute has not been set in the Outfitting Design DB.

SC040 POSL of *struc_elem_1* refers to non-existent Pline in *struc_elem_2*

The Positioning Line within the named Secondary Joint or Fitting does not refer to a valid p-line in the owning Section.

SC050 Incompatible Connection types between *struc_elem_1* and *struc_elem_2*

Self-explanatory.

SC060 Incompatible Connection references between *struc_elem_1* and *struc_elem_2*

Self-explanatory.

SC070 *struc_elem* is not connected

Self-explanatory.

SC080 Incompatible GTYPE of *struc_elem*: *word_1* in Design, *word_2* in Catalogue

The Generic Type used to classify the structural element in the Outfitting Design DB is not the same as that used in the referenced Catalogue description.

SC090 Jline of *struc_elem* unset

The named structural element does not have its Joining Line attribute set, so its position and/or orientation with respect to a connected component cannot be defined.

SC100 INCOMPATIBLE ROD DIAMETER BETWEEN *name* AND *name* FOR HANGER HEAD AND TAIL

The Fitting and the Atta between which the hanger is to be connected have incompatible diameters.

SC110 TBORE [HBORE] NOT SAME AS BORE OF *name*

The Head/Tail diameter of the hanger is not the same as that of the Fitting to which it is to be connected.

SC120 Catref/Specref of *struc_elem* refers to a non-existent element

The SPREF for the named structural Section, Joint or Fitting does not point to a valid component in the Catalogue.

Positional Errors

SP010 Discrepancy between ends of Jlines: *word* of *struc_elem_1* and *word* of *struc_elem_2*

The start/end of the Joining Line for the named Section does not have the same point location as the end of the Joining Line for the named Joint.

SP020 Positions of *struc_elem_1* and *struc_elem_2* misaligned

Self-explanatory.

SP030 *struc_elem_1* lies off the beginning or end of owning Section *struc_elem_2*

The named Joint or Fitting, that is meant to be connected to the named Section, is not positioned within the derived length of the Section.

- **Directional Errors**

SD010 Normal to Cutting plane *word* of *struc_elem_1* is perpendicular to Neutral axis of *struc_elem_2*

This is equivalent to saying that the end-cutting plane of a Section through the point of connection is effectively parallel to the neutral axis of that Section. This is not possible, since

the intersection point needed to define the position of the connection would then be at infinity.

SD020 Normal to Cutting plane *word* of *struc_elem_1* is perpendicular to Origin plane of *struc_elem_2*

This is equivalent to saying that the end-cutting plane of a Section through the point of connection is effectively parallel to the origin plane of that Section. This is not possible, since the intersection point needed to define the position of the connection would then be at infinity.

SD030 Cutting planes *word* of *struc_elem_1* and *word* of *struc_elem_2* misaligned

The start and end cutting planes of the named elements are not parallel.

SD040 Check Beta angles of *struc_elem_1* and *struc_elem_2*

The Beta Angle for the named Section should be the same as the Beta Angle for the connected Joint.

SD060 Normal to cutting plane *word* of *struc_elem* not in general direction of other end of section

The start and end cutting plane normals should point in the general direction of the other end of the Section.

- **Eccentricity Errors**

SE010 *Struc_elem* diverges from logical model: eccentricity out of range

The linear displacement between the Attached Neutral Points on the extended Neutral Axis of the named Section and the logical line joining the Node positions exceeds the permitted distance.

SE020 *Struc_elem* diverges from logical model: angle out of range

The angular deviation between the Neutral axis of the named Section and the logical line joining the Node positions exceeds the permitted amount.

- **Length Errors**

SL010 Length of *struc_elem* is out of range. Actual length is ...

The derived length of the named Section is not within the permissible range applicable to its type.

SL020 Logical length of *struc_elem* is zero

Self-explanatory.

3 Clash Detection

Outfitting Design's clash detection utility allows you to check any specified parts of the Outfitting Design database for clashes (interferences) between individual elements and to report on the results.

The types of clash identified by Outfitting Design depend on two factors:

- The **obstruction levels** of the clashing elements
- The current **touch** and **clearance** tolerances

1. **Obstruction Levels**

   All design primitives and all catalogue primitives have an obstruction level attribute (OBST) that has an integer value of 2, 1 or 0. The value of the OBST attribute defines the physical type of obstruction that the primitive represents.

   For positive primitives the effects are as follows:

   - **OBST = 2**

     A **hard obstruction**; the primitive represents a solid volume, such as a steel beam or a plant vessel, that has rigid and impenetrable surfaces.
   - **OBST = 1**

     A **soft obstruction**; the primitive represents a volume that is not solid but that needs to be kept clear for access purposes, such as an operating space around the control wheel of a valve.
   - **OBST = 0**

     No obstruction; the primitive represents a freely accessible volume, or is simply a representative symbol.

   In addition to the obstruction types defined by the OBST attributes, Insulation is treated as a special obstruction type in its own right.

2. **Extent of Clashing**

   As well as recognising the three types of clashing item (hard, soft and insulation), Outfitting Design recognises three classes of clash between them, depending upon the degree to which the two primitives intrude upon each other's allocated space. These classes are as follows:

   - Physical **clash**; the primitive volumes overlap by more than a specified amount.
   - A **touch**; the primitives *either* overlap by less than a specified amount *or* are separated at their closest point by less than a specified distance.
   - A **clearance**; the primitives are separated at their closest point by more than the amount necessary to constitute a touch but less than a specified clearance distance.

   These three classes are illustrated in *Figure 3:1.: Physical Clash, Touches and Clearances* for the clash specifications:

   Touch limits: 5 mm overlap to 2 mm gap

   Clearance limit: 8 mm

so that the following criteria apply:

- If the primitives overlap by more than 5 mm, a clash is reported
- If the primitives overlap by less than 5 mm, a touch is reported
- If the primitives do not overlap but are separated by less than 2 mm, a touch is reported
- If the primitives are separated by more than 2 mm but less than 8 mm, a clearance is reported
- If the primitives are separated by more than 8mm, no interference is found

Note: The clearance distance, if set, *must* be greater than the touch gap. Setting the clearance distance to zero switches off the clearance-checking function.

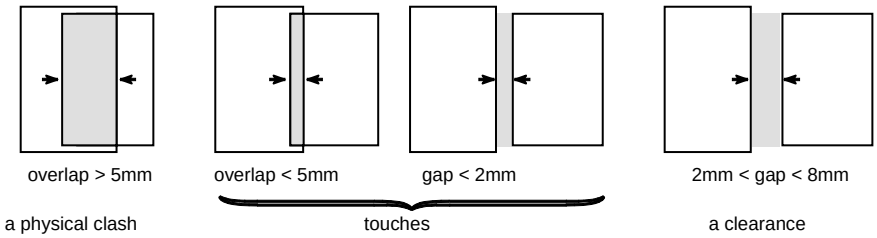


Figure 3:1. Physical Clash, Touches and Clearances

3. **Clash Detection Process**

   The locations and dimensions of all design elements are defined, for clash detection purposes, by reference to a **spatial map** that holds simplified descriptions of the component geometries (see *Section 4.7 Part 1 General Commands in Design Reference Manual*).

   Each element that is to be checked for clashes has its own geometry checked against that of all other elements that are specified by a current **obstruction list**. Items that are not in the obstruction list are ignored during the clash checking operations.

3.1 Entering Clash Detection Mode

Keywords:

DESCLASH EXIT

Description:

The DESCLASH command puts you into Clash Detection mode. While in Clash Detection mode, all commands that you enter are interpreted as being specific to the clash checking and reporting functions, rather than as general Design mode commands.

To return from Clash Detection mode to Design mode, use the EXIT command.

Note: All clash-detection option settings are stored globally, so that they remain in effect from one DESCLASH session to another. Once defined, you need not respecify any of the option settings unless you wish to change them (or unless you leave and then return to Outfitting Design).

3.2 Defining the Obstruction List

Keywords:

OBSTRUCTION EXCLUDE REMOVE

Description:

The list of obstructions, defining those items in the spatial map against which clashes are to be checked, may be built up in stages. You may add items to, or remove items from, the current list in any of the following ways:

- By adding one or more specified Design elements
- By adding implied tube or rod between piping or structural components
- By excluding one or more members owned by elements in the list
- By removing items from the current obstruction list and/or from the current exclusion list

Note: The current obstructions and exclusions are stored as two separate lists, the *effective* obstruction list being the difference between the two. The OBSTRUCTION and EXCLUDE commands add specified items to those lists, but do not overwrite any existing contents. To remove items from the obstruction and/or exclusion lists, you must do so explicitly and separately for each list by using the REMOVE command.

When you add any element to the obstruction list, all elements and primitives below the specified item (that is, all of its members) are automatically incorporated into the list. By default, the obstruction list contains *all* design elements in the current MDB.

If a Branch (or higher) element is added to the obstruction list, implied tube within the Branch is treated as part of the obstruction. If, however, individual piping components are added to the list, implied tube connected to those components is not included automatically and must be added specifically if required. The same principles apply to implied rod, and also to implied tube or rod within Groups.

Only items that have previously been added to the obstruction or exclusion lists may be specified in a REMOVE command. Removing an obstruction does not automatically remove any exclusions which were specified when that obstruction was added to the list.

Examples:

`OBST ALL`	Adds every item from every Outfitting Design DB in current MDB to obstruction list (default)
`OBST /PUMP1 /PUMP2 /VESS1`	Adds individually identified elements to obstruction list.
`OBST /GROUP.MOD2`	Adds all elements in named group to obstruction list.

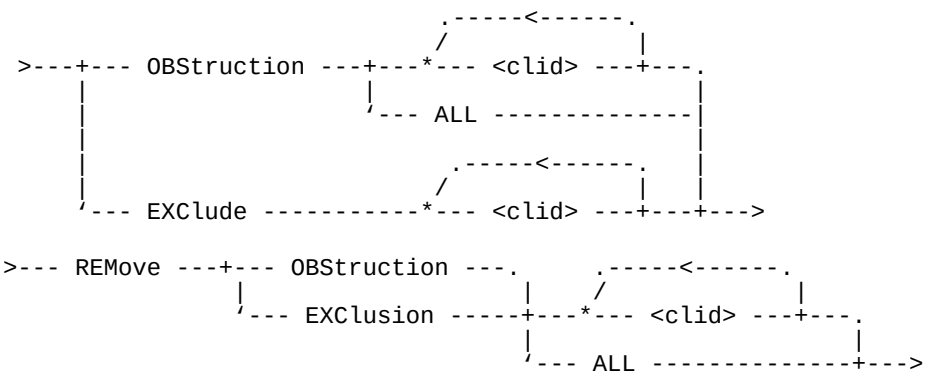
Examples:

`OBST LEAVE TUBE FROM /FLAN2`	Adds individual lengths of implied tube or rod to obstruction list.
`OBST HEAD ROD OF /HANG1`	
`OBST LEAVE /VALV3 IARRIVE /VALV5 TAIL /BRAN1`	
`EXCL /BRAN3 /BRAN5`	Excludes individually identified elements from current obstruction list (by adding them to exclusion list).

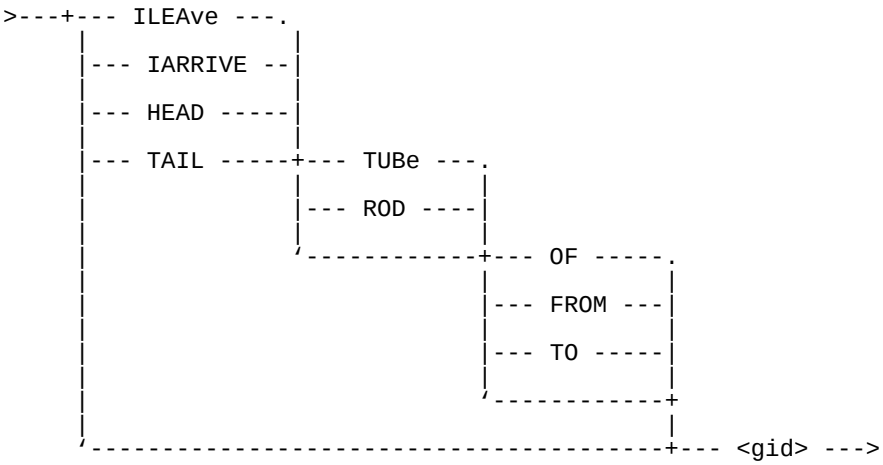
Note: The named elements must be members of items already in the obstruction list.

`EXCL LEAVE TUBE FROM /FLAN2`	Excludes individual lengths of implied tube or rod from current obstruction list.
`EXCL HEAD ROD OF /HANG1`	
`EXCL LEAVE /VALV3 IARRIVE /VALV5 TAIL /BRAN1`	
`REM OBST ALL`	Empties obstruction list.
`REM OBST /PUMP2 /VESS1`	Removes individually identified elements from obstruction list.
`REM OBST LEAVE TUBE FROM /FLAN2`	Removes individual lengths of implied tube or rod from obstruction list.
`REM OBST HEAD ROD OF /HANG1`	
`REM OBST LEAVE /VALV3 IARRIVE /VALV5 TAIL /BRAN1`	
`REM EXCL ALL`	Empties exclusion list.
`REM EXCL /BRAN3 /BRAN5`	Removes individually identified elements from exclusion list.
`REM EXCL LEAVE TUBE FROM /FLAN2`	Removes individual lengths of implied tube or rod from exclusion list.
`REM EXCL HEAD ROD OF /HANG1`	
`REM EXCL LEAVE /VALV3 IARRIVE /VALV5 TAIL /BRAN1`	

Command Syntax:



where <clid> (clashing item identifier) is



Querying:

`Q CLASH OBStruction`	Lists current obstruction list.
`Q CLASH EXClusion`	Lists current exclusion list.

3.3 Defining the Region of Interest

Keywords:

LIMITS AUTOLIMITS

Description:

By default, all parts of the design model relevant to the current obstruction list will be checked for interferences during subsequent clash-checking runs. If you do not want to check the entire design, you may define a restricted region of interest represented by a **3D limits box**.

For a clash to be reported, both items involved in the clash must lie wholly or partly within the limits box. A clash between items that lie partially within the limits box will always be reported, even if the point at which the clash occurs lies outside the box.

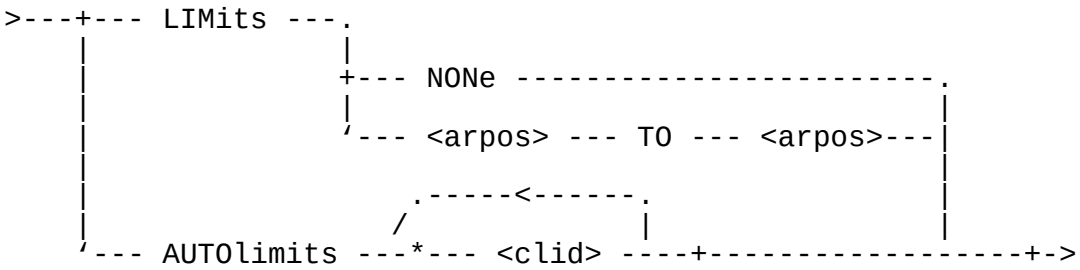
You may set up a limits box in one of two ways:

- **Explicitly**, using the **LIMITS** command, by defining the positions of two diametrically opposed corners (either as explicit co-ordinates or as the origins of specified elements).
- **Implicitly**, using the **AUTOLIMITS** command, by identifying one or more elements and allowing Outfitting Design to calculate the appropriate box that includes them. The limits box is calculated so as to include the *whole* of each named element (including touch and clearance values); it is not determined by the origins of the elements.

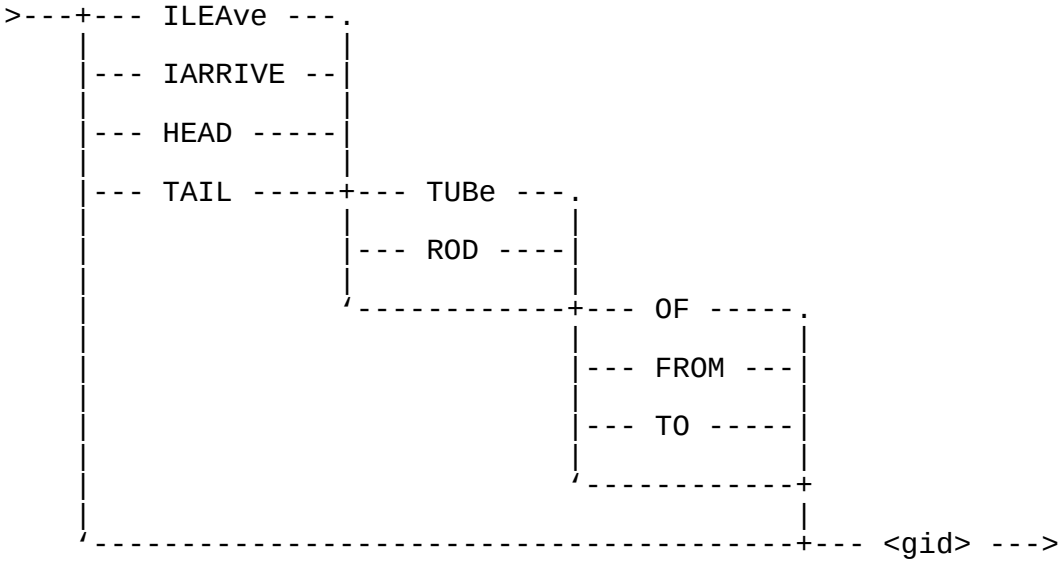
Examples:

`LIMITS N2560 E12150 U2150 TO S6530 W2135 U4200`	Sets corners of box to explicit coordinates.
`LIMITS /STR6 TO / PUMP1`	Sets corners of box to origins of named elements.
`LIMITS S0 E1300 U0 FROM /NOZZ3 TO N650 E175 D50 FROM /VESS2`	Sets corners of box to points referenced from origins of named elements.
`AUTO /PUMP1 /PUMP2 /VESS2`	Calculates box enclosing named elements.
`AUTO /GROUP1`	Calculates box enclosing all elements in named group.
`AUTO HEAD OF / BRAN1-2 TAIL OF / BRAN3-1` `AUTO /VESS1 /VESS2 LEAVE TUBE FROM / FLAN2`	Calculates box enclosing named elements and implied tube or rod.
`LIMITS NONE`	Removes any current limits box, thus restoring default state in which whole design model is checked.

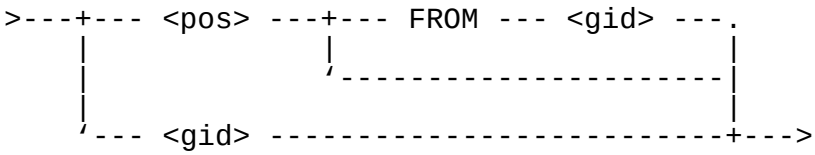
Command Syntax:



where <clid> (clashing item identifier) is



and <arpos> (absolute/relative position) is



Querying:

`Q CLASH LIMits`

Gives coordinates for current limits box.

`Q VOLume` *`element_id(s)`*

Gives coordinates for limits box enclosing specified design item(s).

`Q WINdow <arpos> TO <arpos>`

Gives names of all elements within limits box specified by the two positions; for example:

```
Q WIN N2560 E12150 U2150 TO S6530 W2135 U4200

Q WIN /STR6 TO /PUMP1

Q WIN S0 E1300 U0 FROM /NOZZ3 TO N650 E175 D50 FROM /VESS2
```

3.4 Clash Detecting Design or Production Data

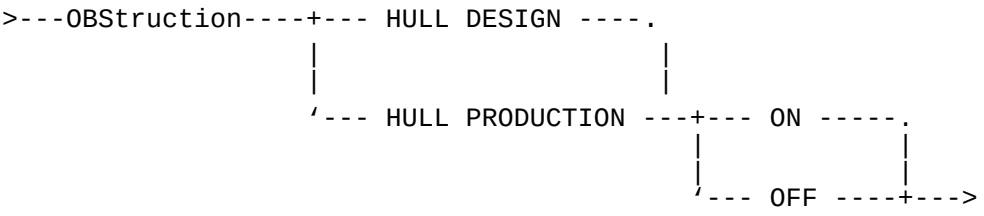
In Outfitting there are structural design data, termed Outfitting Design and detailed design data, termed PRODUCTION. These two sets of data represent the same model item and occupy the same 3D space. In clasher you only want to clash against one of the sets of data.

The facility provides you with the option of switching the “obstruction” between Design and Production. The option will only be applied when ‘OBST ALL’ is in place, i.e. if the user has specified an explicit list of obstructions, then the settings will be ignored.

Example:

OBSTRUCTION HULL DESIGN OFF	Will not clash against hull Design items
OBSTRUCTION HULL DESIGN ON	Will clash hull Design items - this is the default
OBSTRUCTION HULL PRODUCTION OFF	Will not clash against hull Production items - this is the default
OBSTRUCTION HULL PRODUCTION ON	Will clash against hull Production items

Command Syntax:



Querying

Q CLASH OBSTRUCTION HULL DESIGN

Returns true/false

Q CLASH OBSTRUCTION HULL PRODUCTION

Returns true/false

3.5 Defining Touches and Clearances

Keywords:

TOUCHING CLEARANCE

Description:

These commands jointly define the tolerances that determine whether any given clash is reported as a physical clash, a touch or a clearance (see *Figure 3:1.: Physical Clash, Touches and Clearances*). Outfitting Design reports a **touch** if two primitives either:

- Overlap by less than a specified **touch overlap** distance
- Are separated by less than a specified **touch gap** distance

The touch overlap setting must be positive: the touch gap must be positive and *less* than the current setting for the clearance distance. The default settings are for a touch overlap of 2mm and a touch gap of zero.

Outfitting Design reports a **clearance** if two primitive volumes are separated at their closest point by more than the currently defined touch gap but by less than a specified **clearance**

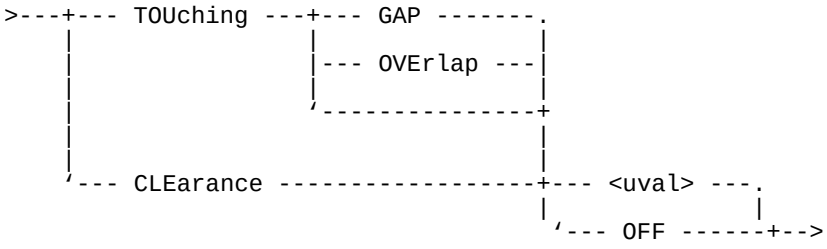
distance. The clearance distance must be positive and *greater* than the current setting for the touch gap. By default the clearance distance is undefined, so that no clearances will be found.

Outfitting Design reports a **clash** if two primitives overlap by more than the touch overlap distance.

Examples:

`TOUCH OVER 3`	Sets touch overlap; current units assumed to be mm.
`TOUCH OVER 0.2 INCH`	Sets touch overlap in specified units.
`TOUCH OVER OFF`	Effectively sets touch overlap to zero.
`TOUCH GAP 2`	Sets touch gap; current units assumed to be mm.
`TOUCH GAP 0`	Resets the touch gap to its default value.
`TOUCH GAP OFF`	Effectively sets touch gap to zero.
`CLEARANCE 10`	Sets clearance; current units assumed to be mm.
`CLEARANCE 0.25 INCH`	Sets clearance in specified units.
`CLEARANCE 0`	Unsets clearance distance
`CLEARANCE OFF`	(default state)

Command Syntax:



Querying:

```
Q CLASH TOUching OVErlap
Q CLASH TOUching GAP
Q CLASH TOUching
Q CLASH CLEarance
```

3.6 Ignoring Clashes Within Specified Element Types

Keywords:

NOCHECK WITHIN

Description:

By default, no checks are made for clashes between items owned by the same Structure, Substructure, or Equipment. In addition, you may use the NOCHECK and WITHIN commands to tell Outfitting Design to ignore all clashes within one or more other specific types of element.

All clashes below each element of the specified types will be ignored during the checking operation, whatever the hierarchic level of the clashing items. Clashes specified in this way are ignored during the actual clash-checking operation and are not therefore available in memory for inclusion in subsequent output reports.

Examples:

`NOCHECK WITHIN BRAN`	Ignores clashes within individual Branches (but still reports clashes between items in different Branches).
`NOCHECK FRMW`	Ignores clashes within individual Frameworks (but still reports clashes between items in different Frameworks).
`WITHIN FRMW`	Restores clash checking within Frameworks; that is, it cancels a previous NOCHECK FRMW command.
`WITHIN EQUI`	Restores clash checking within individual items of Equipment (which are not checked by default).

Command Syntax:

```
>---+--- NOCheck ---+--- WIThin ---.
    |               |              |
    |               '--------------+   .----<------.
    |                              | /             |
    '--- WIThin -------------------+---*--- <sig> -----+--->
```

where <sig> (significant element) is any of the following:

SITes	ZONes	PIPes	BRAnches	EQUipments
STRuctures	SUBStructures	HANGers	RESTRaints	
PTRacks	FRMWorks	SBFRMworks		

3.7 Ignoring Clashes at Steelwork Junctions

Keywords:

INCLUDE IGNORE CONNECTIONS ALL WITH SPECIFICATIONS

Description:

This facility allows you to control checking at steelwork junctions. (Clashes between sections and attached joints etc. are ignored automatically.)

Frequently, you may wish to leave end preparations at steelwork joints until late in the design process. If you do this, you can inhibit clash reporting using the commands described here.

Examples:

`INCLUDE CONNECTIONS`	Includes all steelwork connections during subsequent clash checking.
`IGNORE CONNECTIONS`	Ignores steelwork connections during checking.
`IGNORE CONNECTIONS WITH SPECIFICATIONS`	Ignores only those connections which have specifications.

Command Syntax:

```
>-- INClude -- CONnections -->

 >-- IGNore --+-- ALL --.
              |         |
              '---------+-- CONnections -->

 >-- IGNore -- CONnections -- WIth -- SPEcifications -->
```

3.8 Ignoring Touches

Keywords:

IGNORE INCLUDE TOUCHES

Description:

Even though the current touch overlap setting may be non-zero, you may tell Outfitting Design to ignore all touches during subsequent clash checks.

Touches ignored in this way are not available in memory for inclusion in subsequent output reports. If you are likely to want to check touches later, it is better to include them in the clash-checking operation (which is the default situation) and then to inhibit their inclusion in the report if necessary.

Examples:	
`IGNORE TOUCHES`	Ignores all touches during subsequent clash checking.
`INCLUDE TOUCHES`	Restores the default situation, where touches are detected and stored with the current clash list.

Command Syntax:

```
>---+--- IGNore ----.
    |               |
    '--- INclude -----+--- TOUches --->
```

Querying:

```
Q CLASH IGNore
```

3.9 Controlling the Reported Clash Position

Keywords:

MIDPOINT

Description:

By default, the reported position for a clash depends on which part of the overlapping region is first detected by the checking process: in most cases, this somewhat arbitrary position identifies the clash sufficiently accurately. The MIDPOINT option lets you specify that the reported position is always at the centre of a box surrounding the overlapping region, giving more reproducible (but slower) results.

Command Syntax:

```
>-- MIDpoint --+-- ON ---.
               |         |
                 '-- OFF --+-->
```

Querying:

```
Q CLASH MIDpoint
Q CLASH OPTions
```

3.10 Specifying How Branches Are Checked

Keywords:

BRANCH ACHECK BCHECK

Description:

Assuming that you have *not* specified NOCHECK BRANCHES (see *Ignoring Clashes Within Specified Element Types*), you may check for clashes within pipe branches in either of two ways:

- As a full primitive-by-primitive check of every component within each branch - known as a **Type A check** (or **ACHECK**)
- As a simplified check which ignores the possibility of clashes between certain pairs of components within the branches - known as a **Type B check** (or **BCHECK**)

  (Clashes between adjacent components and attachments within a Branch are ignored automatically.)

The purpose of the BCHECK option is to eliminate from the clash report spurious clashes that result when zero-length components (such as welds and olets) separate other components or tubing. If you specify a BCHECK, the warning message

```
*** BRANCH CHECKING - OPTION B ***
```

will be output as part of the report header. This is because, although most of the clashes ignored by Type B checking really are spurious, some genuine clashes may also be ignored. The latter, examples of which are given in *Rules Applicable During Type B Checks*, will be mostly due to design errors, many of which should have been diagnosed at an earlier stage by means of a data consistency check (see *Data Consistency Checking*).

The **default** check is the ACHECK option, since this is a fail-safe check for *all* potential clashes.

Example:

```
BRANCH B

BRANCH A
```

Command Syntax:

```
>--- BRANCh ---+--- Acheck ---.
               |               |
               '--- Bcheck ----+--->
```

Querying:

```
Q CLASH CHECK
Q CLASH OPTions
```

3.10.1 Rules Applicable During Type B Checks

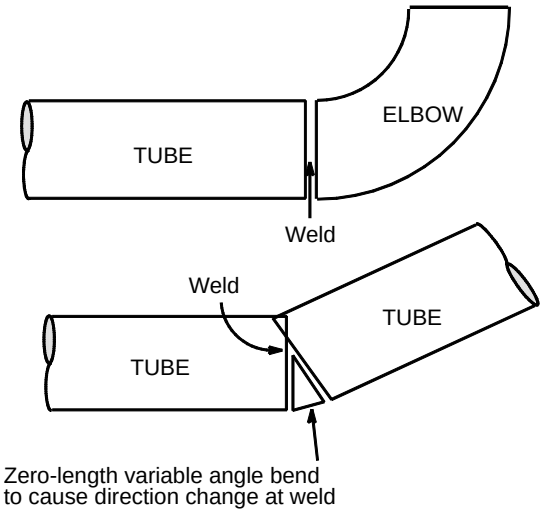
The following sections define the rules that Outfitting Design applies during the clash-checking process when the BCHECK option is in force. They include examples of the sorts of design errors that would remain undetected in the clash report. The term 'clash' is used in these examples to refer to all types of interference; in practice, some of these may be reported as touches, depending upon the touch specifications in force.

Clashes Within a Single Branch

Rule: If the leave point (p-leave) of an upstream component or tube coincides with the arrive point (p-arrive) of a downstream component, and there is no intervening Tube, the two components will not be checked for clashes between them.

This rule is intended to suppress clashes when two components or tubes within a branch are separated by one or more zero-length components that have no Geomset.

For example:



ACHECK: Tube/elbow clash

BCHECK: No clashes

ACHECK: Tube/tube clash
(the weld and the zero-length bend have no geomsets)

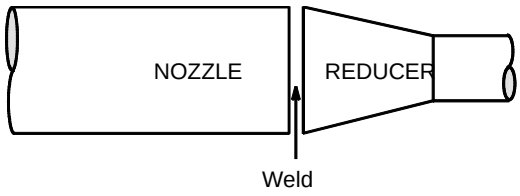
BCHECK: No clashes

Clashes Between a Branch and a Nozzle

Rule: If a branch head or tail and a nozzle are properly connected *and* the p-arrive or p-leave of the final component coincides with the branch head or tail, then no clash will be reported.

This rule is intended to suppress clashes when a nozzle and a pipe component or tube are separated by one or more zero-length components.

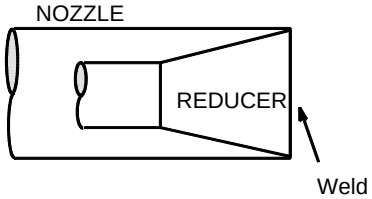
For example:



ACHECK: Nozzle/reducer clash

BCHECK: No clashes

The rule would, however, cause the following (unlikely) clash to be ignored:



Note: Reducer wrongly orientated

ACHECK: Nozzle/reducer clash
Nozzle/tube clash

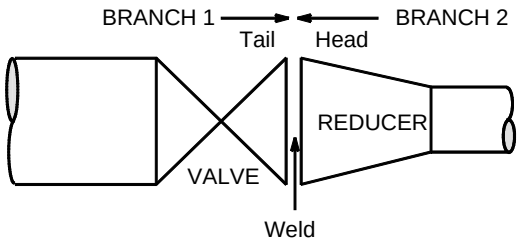
BCHECK: Nozzle/tube clash only
Nozzle/reducer clash i ignored

Clashes Between Two Branches

Rule 1: If two branches are connected together end-to-end *and* the p-arrive/p-leave of two components or tubes coincide with their respective branch head/tail (whichever is the connected end), then no clash will be reported.

This rule is intended to suppress clashes similar to those described for clashes within a single branch, except that the relevant components or tubes may now be from different, but connected, branches.

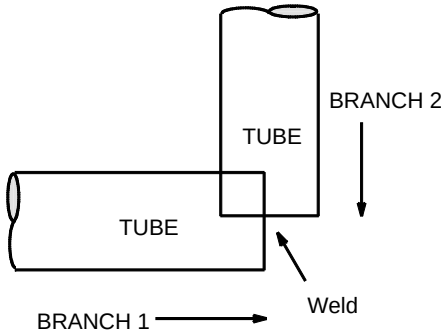
For example:



ACHECK: Valve/reducer clash

BCHECK: No clashes

Rule 1 would, however, allow some clashes due to routing errors to be ignored. For example:



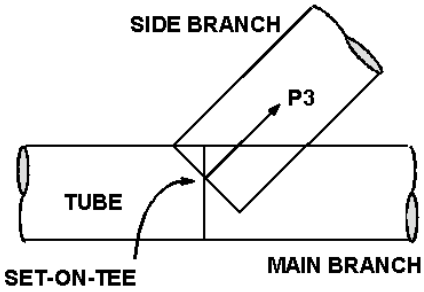
ACHECK: Tube/tube clash

BCHECK: No clashes

Rule 2: If the head/tail tube of one branch is connected to a set-on tee or olet (having no geometry other than a sphere) in a second branch *and* the p-arrive/p-leave of the connected component in the main branch coincides with the p-arrive/p-leave of the tee'd component in the side branch *and* the latter point is also the HPOS or TPOS of the side branch, then no clashes will be reported between the head/tail tube of the tee'd component and the tube on either side of the tee/olet in the main branch.

This rule is intended to suppress clashes when a side branch is connected to a zero-length component in another branch.

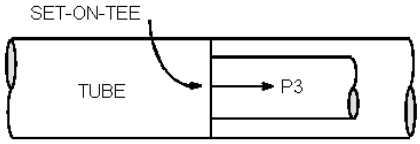
For example:



ACHECK: Two tube/tube clashes at tee position (between side branch and tube each side of tee in main branch)

BCHECK: No clashes

Rule 2 would, however, allow the following clash to be ignored:



Note: Side branch on tee has wrong DDANGLE

ACHECK: Tube/tube clash at tee position

BCHECK: No clashes

3.11 Checking For Clashes

3.11.1 The Principles

Having set up the obstruction list and defined any non-default clash specifications (touch and clearance settings, restricted region of interest, Branch BCHECK option, etc.), you can now tell Outfitting Design which design items you want it to check (the **check list**).

The clash-checking process is carried out in two stages, as illustrated in the following Figure:

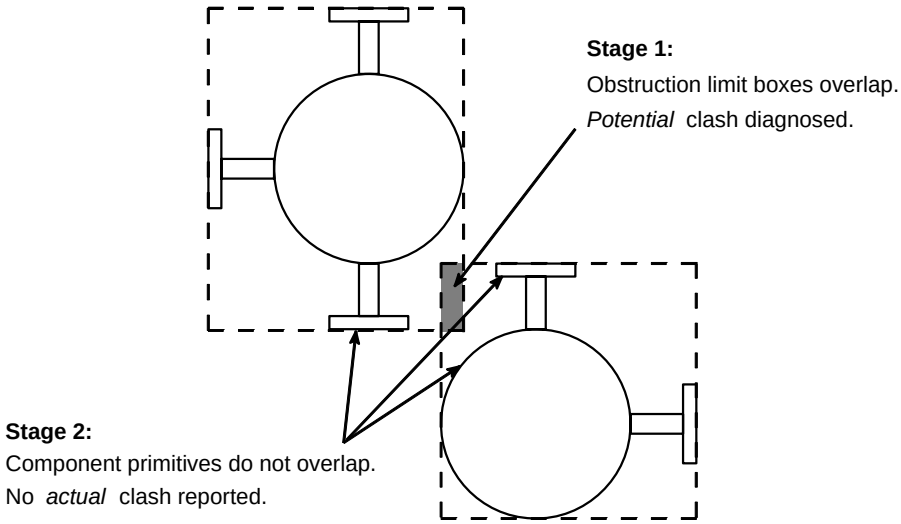


Figure 3:2. Diagnosis of potential/real clashes

- In **Stage 1**, the obstruction limit boxes that enclose the individual design elements (as represented in the spatial map) are checked for overlapping. If no overlap occurs between the obstruction limit boxes of two elements, then no clash can exist. If, however, the boxes do overlap, then a *potential* clash exists and the second stage of checking is carried out.
- In **Stage 2**, the detailed geometry of the elements within overlapping obstruction limit boxes (as represented in the Geometric Modelling Library) is checked to see if any of the constituent primitives overlap. If they do, then an appropriate clash is reported.

To confirm the absence of clashes in a proven design, or to run a superficial check on a new design, you can carry out just the first stage of the checking process, known as a **box check**. Note, however, that if you run only a box check on an unproven design, you are likely to generate a report containing many spurious clashes resulting from situations such as the one illustrated in Figure 3-2. The extra time taken to analyse the output report can outweigh the time saved by running the simplified checking procedure, so use this option with care.

3.11.2 Running a Full Component Check

Keywords:

CHECK

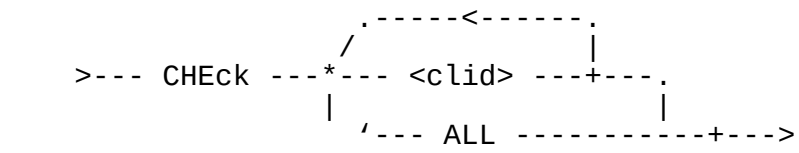
Description:

The CHECK command initiates a full two-stage check for clashes between specified items (the check list) and the current obstruction list.

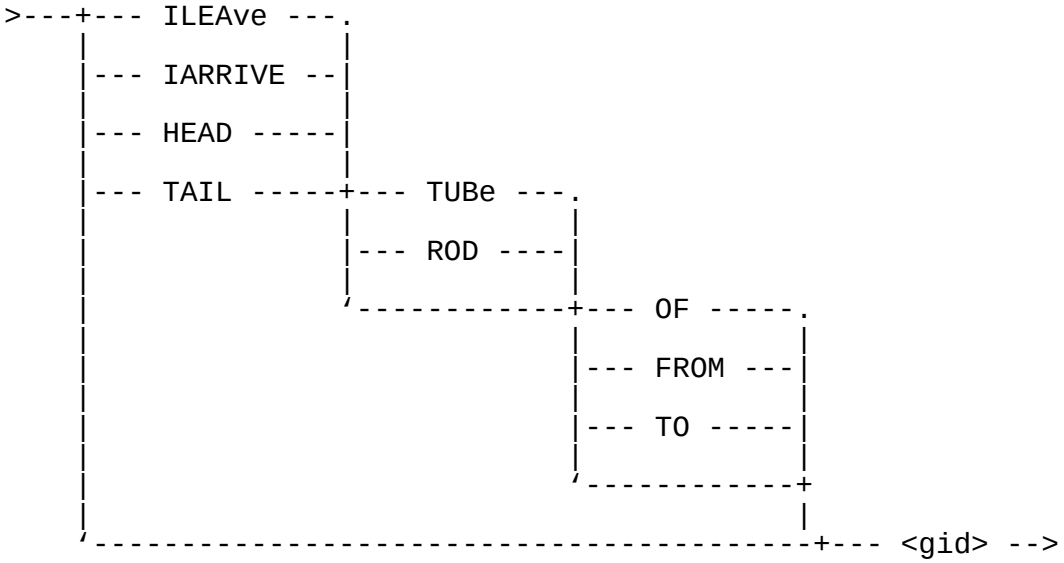
Examples:

`CHECK ALL`	Checks all items in the design model (within any restrictions defined as in Sections 3.3 to 3.9) against the obstruction list.
`CHECK /ZONE1.PIPES`	Checks only specified items against the obstruction list.
`CHECK /PUMP1 / PUMP2 /VESS2`	
`CHECK /GROUP.MOD2`	
`CHECK LEAVE TUBE FROM /FLAN2 IARRIVE TUBE TO / VALV5`	

Command Syntax:



where <clid> (clashing item identifier) is



3.11.3 Running an Obstruction Box Check

Keywords:

BOXCHECK

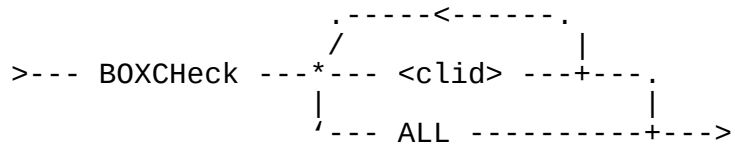
Description:

The BOXCHECK command initiates only a simplified (Stage 1) check for clashes between specified items (the check list) and the current obstruction list.

Examples:

`BOXCHECK ALL`	Checks all items in the design model (within any restrictions defined as in Sections 3.3 to 3.9) against the obstruction list.
`BOXCHECK / ZONE1.PIPES`	Checks only specified items against the obstruction list.
`BOXCHECK /PUMP1 / PUMP2 /VESS2`	
`BOXCHECK / GROUP.MOD2`	
`BOXCHECK LEAVE TUBE FROM /FLAN2 IARRIVE TUBE TO / VALV5`	

Command Syntax:



where <clid> (clashing item identifier) has the syntax shown in *Running a Full Component Check*.

3.11.4 Adding Checked Items to the Obstruction List

Keywords:

CHECKADD

Description:

The CHECKADD command builds up the obstruction list in a progressive way as the checking process proceeds.

The first item specified in the checklist is added to the current obstruction list (which may initially be empty) and that item is then checked against the cumulative obstruction list thus created. This process is repeated for each item in the checklist in turn.

(You can achieve similar results by including the required items in both the initial obstruction list and the checklist when using the CHECK command.)

Example:

```
CHECKADD /A /B /C /D
```

Example:

This builds up the obstruction list in four separate stages and checks the next specified element against the current list at each stage, thus:

Stage 1:

/A is added to the (empty) obstruction list and /A is checked against it, thus checking for clashes between the elements:

```
/A/A
```

Stage 2:

/B is added to the obstruction list, which then comprises /A /B, and /B is checked against this new list, thus checking for clashes between the following pairs of elements:

```
/A/B    /B/B
```

Stage 3:

/C is added to the obstruction list, which then comprises /A /B /C, and /C is checked against this new list, thus checking for clashes between the following pairs of elements:

```
/A/C    /B/C    /C/C
```

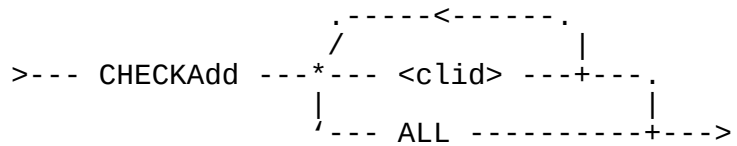
Stage 4:

/D is added to the obstruction list, which then comprises /A /B /C /D, and /D is checked against this new list, thus checking for clashes between the following pair of elements:

```
/A/D    /B/D    /C/D    /D/D
```

Note: This has the same overall effect as the command sequence:
`OBSTRUCTION /A /B /C /D`
`CHECK /A /B /C /D`
which creates the obstruction list /A /B /C /D and the check list /A /B /C /D and checks for clashes between the pairs
/A/A /A/B /A/C /A/D /B/B /B/C /B/D /C/C /C/D /D/D.)

Command Syntax:



where <clid> (clashing item identifier) has the syntax shown in *Running a Full Component Check*.

3.12 Reporting the Clashes Found

3.12.1 Principles

A report is sent automatically to the Request region each time you run a clash check; that is, each time you enter a CHECK, BOXCHECK or CHECKADD command. You can change the format for such a report before running the clash check if necessary. In addition, you may send output to a file by using the ALPHA FILE or ALPHA LOG commands in the usual way.

The default report format comprises the following three parts:

- **The report header**: Details of the program version in use; the types of clash reported; any non-default checking options and limits; the touch and clearance limits; any special reporting options in use.
- **The main body**: Details of the clashes found, including the clash type and extent and the identifiers of the two design items involved. The clashes are grouped into **sections**, one for each significant element that contains an interference. Where space permits, each clash is reported on a single line.
- **The clash summary**: Lists the total number of clashes of each type found; the total number of elements checked during the run covered by the report; the number of elements found to be free of any interferences.

All data resulting from a clash-checking run is held in the computer's memory until overwritten by data from a later run (or until you change modules). This allows you to generate further reports derived from the same data, possibly using different reporting options from those in force for the original report.

3.12.2 Customising the Report Header

Keywords:

REPORT HEADER

Description:

The standard header comprises the following:

- The program version and the date and time at the start of the check.
- The types of clashes being reported and the elements specified in the check-initiation command.
- The touch and clearance definitions, the current Branch checking option, and any non-default options which may be in force.

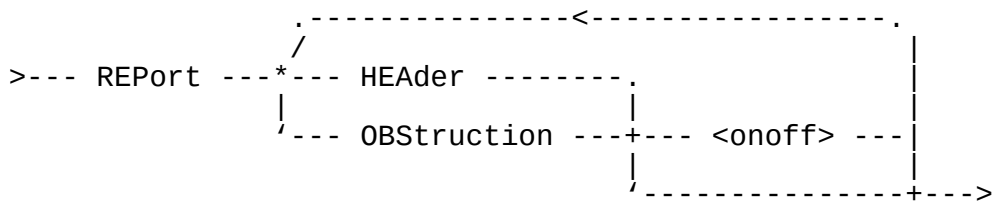
Examples:

`REPORT HEADER OFF`	Suppresses entire header
`REPORT HEADER` `REPORT HEADER ON`	Reinstates header if previously suppressed (ON is the default)

Examples:

`REP OBSTRUCTION` `REP OBSTRUCT ON`	Outputs obstruction list as part of header
`REP OBSTR OFF`	Suppresses obstruction list in header (the default)

Command Syntax:



Querying:

`Q CLASH REPort HEAder`
`Q CLASH REPort OBStruction`
`Q CLASH REPort`
	List all report settings (header + main + summary)

3.12.3 Customising the Main Body of the Report

Keywords:

REPORT MAIN SECTION POSITION REF
NUMBER PRIMARY SIGNIFICANT FIRST SECOND BOTH REMOVE

Description:

The standard format shows each clash on a separate line, with full details of the clashing items and the nature of the clash. The reported clashes are grouped into **sections**, each of which lists all clashes within a single significant element, the name of which is usually shown only at the beginning of the section to avoid excessive repetition of data (the name of the second clashing element is always shown in full to avoid any ambiguity).

The following details may also be included in the report:

- The clash **position**, in either Site or World co-ordinates (to control how this position is calculated, see *Controlling the Reported Clash Position*).
- The Outfitting **reference numbers** of the clashing elements, as well as their names.
- Sequential **clash numbers**, used to identify individual clashes in other commands (such as when approving clashes).
- By default, all types of clash, touch and clearance are reported. You may restrict the report to one or more specified clash types (e.g. hard/hard only, touches only, etc.).

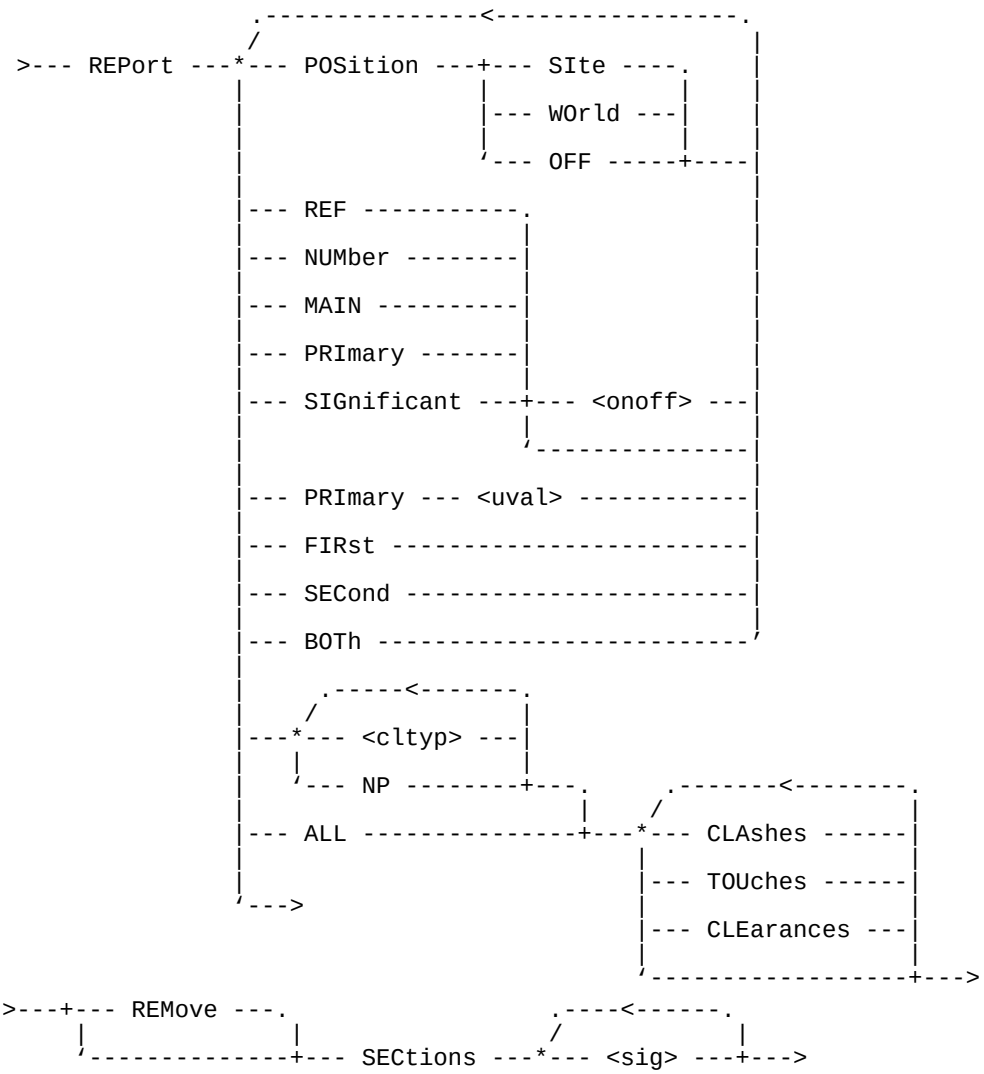
Examples:

`REPORT MAIN OFF`	Suppresses main body of report
`REPORT MAIN` `REPORT MAIN ON`	Reinstates main body if previously suppressed (ON is the default)
`REPORT POSITION SITE`	Shows clash positions in Site coordinates
`REPORT POSITION WORLD`	Shows clash positions in World coordinates
`REPORT POSITION OFF`	Suppresses clash positions (the default)
`REPORT REF` `REPORT REF ON`	Outputs reference numbers of clashing items
`REPORT REF OFF`	Suppresses reference numbers (the default)
`REPORT NUMBER` `REPORT NUMBER ON`	Outputs sequential clash numbers.
`REPORT NUMBER OFF`	Suppresses clash numbering (the default)
`REPORT FIRST`	List clash only under first item in DB hierarchy (the default)
`REPORT SECOND`	List clash only under second item in DB hierarchy
`REPORT BOTH`	List clash twice, once under each item in DB hierarchy
`REPORT PRIMARY` `REPORT PRIMARY ON`	Reports only first or highest priority clash found between two significant elements (i.e. suppresses multiple clashes) but also shows actual number of clashes which would have been reported if this option were not in force.
`REPORT PRIMARY 500`	Suppresses multiple clash reports if clash positions are less than 500 mm apart (if current units are mm)
`REPORT PRIMARY OFF`	Reports all clash occurrences, including those between different primitives of the same pairs of significant elements (the default)
`REPORT SIGNIFICANT` `REPORT SIGNIFICANT ON`	Lists all significant elements which have been checked, not just those for which clashes have been detected.
`REPORT SIGNIFICANT OFF`	Reports only those significant elements which involve clashes (the default)

Examples:

REMOVE SECTIONS SUBS STRUC	Suppresses the labelling of those sections of the report which represent Substructures and Structures. Clashing items in these sections will be identified by their full name on each line (in the same way that the second clashing element is always shown)
SECTIONS SUBS STRU	Reinstates labelling of specified sections

Command Syntax:



where <sig> (significant element) is any of the following:

Querying:

```
Q CLASH REPort MAIN
Q CLASH REPort POSition
Q CLASH REPort REF
Q CLASH REPort NUMber
Q CLASH REPort PRImary
Q CLASH REPort SIGnificant
Q CLASH REPort DUPlication
```

Shows under which sections clashes will be reported (i.e. First, Second or Both)

```
Q CLASH REPort LEVel
```

Lists clash types to be reported on

```
Q CLASH REPort
```

List *all* report settings (header + main + summary)

3.12.4 Customising the Report Summary

Keywords:

REPORT SUMMARY

Description:

The standard summary, output at the end of the clash report, comprises a list showing:

- The total number of clashes found of each type
- The total number of significant elements checked during the run covered by the report

  The number of elements found to be free of any interferences

  This summary is headed *** ACTUAL CLASH SUMMARY ***.

  If REPORT PRIMARY ON has been specified (see *Customising the Main Body of the Report*), *two* report summaries will be produced; one headed *** PRIMARY CLASH SUMMARY *** and one headed *** ACTUAL CLASH SUMMARY ***.

Examples:

REPORT SUMMARY OFF	Suppresses output of report summary
REPORT SUMMARY REPORT SUMMARY ON	Reinstates report summary if previously suppressed (ON is the default)

Command Syntax:

```
>--- REPort --- SUMmary ---+--- <onoff> ---.
                           |                |
                           '----------------+--->
```

Querying:

```
Q CLASH REPort SUMmary
Q CLASH REPort
```

List *all* report settings (header + main + summary)

3.13 Querying Clash Data Directly

You may query some of the clash data directly, without generating a report.

3.13.1 Querying Individual Clashes

Keywords:

FIRST SECOND TYPE POSITION

Description:

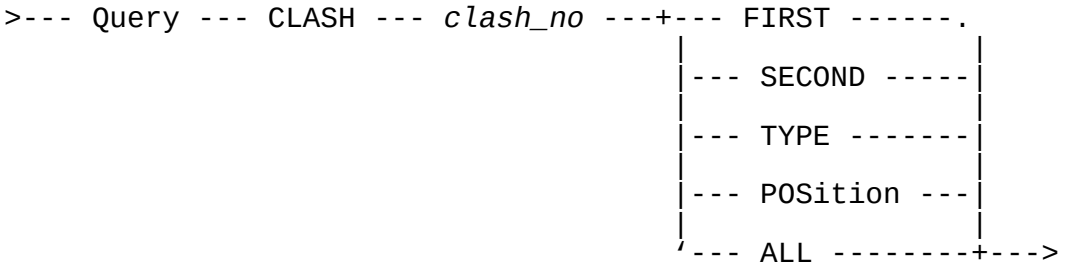
These options allow you to query individual parts of specified clashes. The clashes are identified in each case by their clash numbers.

Examples:

`Q CLASH 2 FIRST`	Outputs name of first clashing element (the 'clasher') for clash number 2
`Q CLASH 2 SECOND`	Outputs name of second clashing element (the 'clashee') for clash number 2
`Q CLASH 2 TYPE`	Outputs type of clash (e.g. HH TOUCH)
`Q CLASH 2 POSITION`	Outputs position of clash in coordinate system currently set for reporting (i.e. Site or World)

`Q CLASH 2 ALL`	Outputs all of the preceding data; for example:
	`Clash number: 2`
	` Clasher: CONE 1 of EQUI /VESS1`
	` Clashee: SLCY 1 of STRU /V1.PL`
	` Type: HH TOUCH`
	` Position: W 10000mm N 18000mm U 2200mm`

Command Syntax:



3.13.2 Querying Clash Statistics

Keywords:

COUNT CLASHES TOUCHES CLEARANCES NOTPROVEN

Description:

These options allow you to query the total number of clashes of each type (excluding approved clashes).

Example:

```
Q CLASH COUNT CLASHES
Q CLASH COUNT TOUCHES
Q CLASH COUNT CLEARANCES
Q CLASH COUNT NOTPROVEN
CLASH COUNT ALL
```

Outputs number of clashes of each type; for example:

```
Total Clashes: 30
 Total Touches: 116
 Total Clearances: 218
 Total Not proven: 37
```

Command Syntax:

```
>--- Query --- CLASH --- COUNT ---+--- CLASHes ------.
                                  |                   |
                                  |--- TOUCHes ------|
                                  |                   |
                                  |--- CLEARances ---|
                                  |                   |
                                  |--- NOTProven ----|
                                  |                   |
                                  '--- ALL ----------+--->
```

3.14 Reporting Without Further Clash Checking

All clash data derived during a clash-checking run is retained in the computer's memory until overwritten by data from a subsequent run (or until you change modules). This allows you to output further reports derived from that data, using modified reporting options and/or a different output device if required.

3.14.1 Generating a Report from Existing Clash Data

Keywords:

OUTPUT

Description:

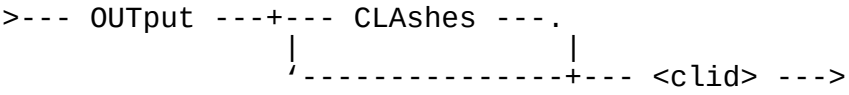
The report generated in response to an OUTPUT command has exactly the same format, determined by any current reporting options which you have set, as that generated in response to a CHECK command. The difference is that the check options, touch and clearance values, obstruction list etc. which apply to the Output report are those current *when the clash run was carried out*; these need *not* be current when the OUTPUT command is given.

Examples:

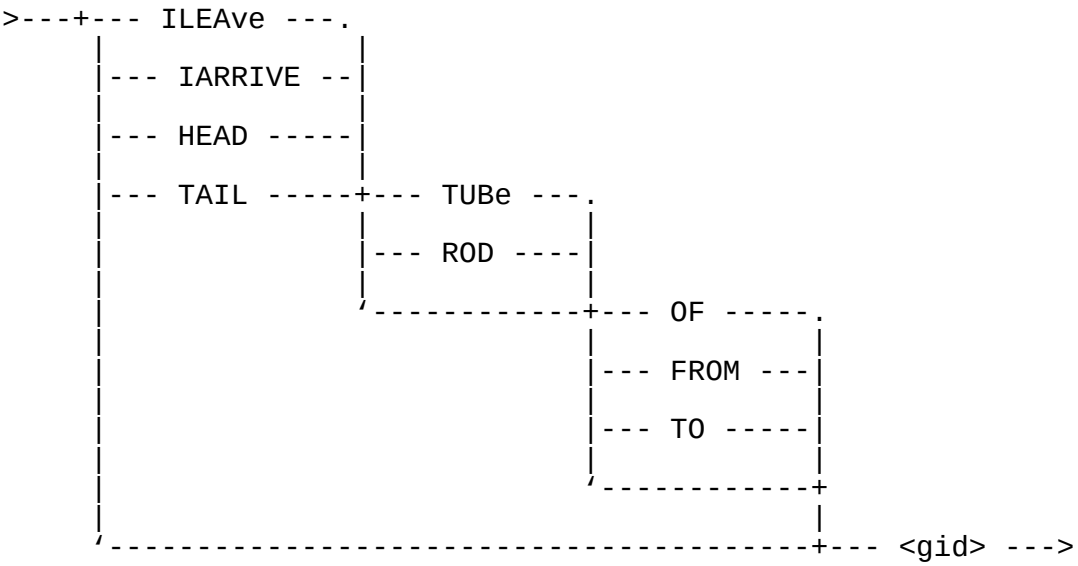
```
OUTPUT CLASHES /ZONE1.PIPES
OUTPUT /PUMP1 /PUMP2 /VESS2
OUTPUT /GROUP.MOD2
OUTPUT LEAVE TUBE FROM /FLAN2 IARRIVE TUBE TO /VALV5
```

	These commands output reports of known clashes involving the specified items in each case
OUTPUT	Outputs a complete report of all clash data currently held in memory

Command Syntax:



where <clid> (clashing item identifier) is



3.14.2 Suppressing Clash Repetition in Reports

Keywords:

REPORT FIRST

Description:

When used before an OUTPUT command, the REPORT FIRST option allows you to generate a sequence of reports from a single set of clash data such that each clash is reported once only throughout the complete sequence.

Example:

`REPORT FIRST ALL`
`OUTPUT /ZONE.PIPES`
`OUTPUT /`
`ZONE.STEELW`
	Assume that you have just run a clash check which includes piping and steelwork items among the elements checked. Then this sequence generates two separate reports; the first (from OUTPUT /ZONE.PIPES) shows all clashes involving pipework elements, including those between pipework and steelwork; the second (from OUTPUT /ZONE.STEELW) shows all clashes involving steelwork elements except those which were included in the first report (i.e. the second report omits clashes between pipework and steelwork.

Command Syntax:

```
>--- REPort FIRst ALL --->
```

3.15 Approving Clashes

3.15.1 The Principles

You may specify one or more clashes which, although real interferences, are to be omitted from subsequent clash reports. Such clashes are referred to as **approved clashes**.

The obstruction limits boxes for all approved clashes, as represented in the spatial map, are identified in a separate list known as the **approval list**. This enables the program to check which items have been moved within the design after approval, and therefore to alert you to the fact that some clash approvals may no longer be valid.

3.15.2 Adding Clashes to the Approval List

Keywords:

APPROVE

Description:

Adds clashes to the approval list in any of the following ways:

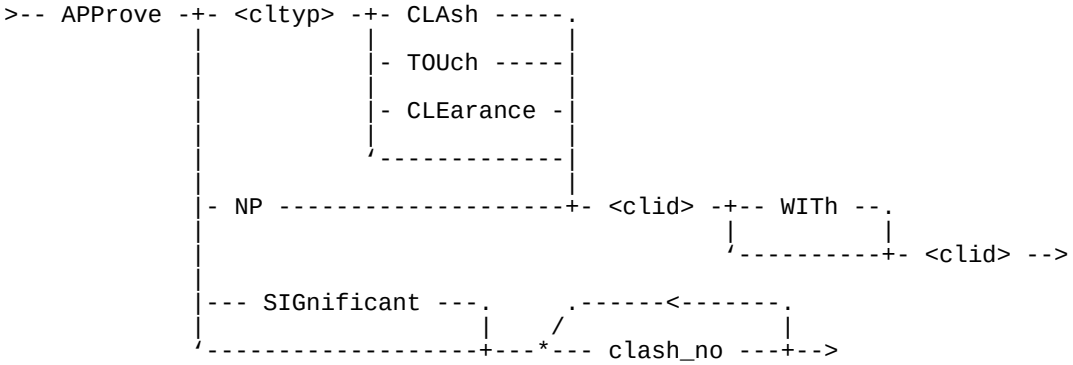
- By specifically identifying a known clash between two named items
- By specifically identifying a known clash by means of its reference number in the latest clash report
- By generally referring to actual or potential clashes between named items; either before or after running a check to see what clashes exist
- By specifying actual or potential clashes within a single named element

  Approved clashes will be omitted from clash reports regardless of which way round the interfering items are specified in the obstruction list and the check list in subsequent clash-checking runs.

Examples:

`APPROVE HS TOUCH BOX1 OF /EQUI1 WITH /GASK1 OF / BRAN2`	Approves hard/soft touch between named items (and any lower level touches; e.g. HI, SS etc.)
`APPROVE 5`	Approves the fifth clash in most recent clash report
`APPROVE SIGNIF 5`	Approves fifth clash in most recent clash report at significant element level, rather than at primitive level; where the significant elements are: Sites, Zones, Pipes, Branches, Equipments, Structures, Substructures, Hangers, Restraints, Ptracks, Frameworks, and Subframeworks
`APPROVE II CLASH / BRAN1-1 WITH / ZONE.EQUIP`	Approves all Insulation/Insulation clashes between any elements below /BRAN1-1 and any elements below / ZONE.EQUIP
`APPROVE / ZONE.EQUIP WITH / ZONE.STEELW`	Approves *all* types of clash between members of the named zones
`APPROVE HS CLASH / ZONE.EQUIP WITH / ZONE.EQUIP`	Approves all hard/soft clashes between members of / ZONE.EQUIP (and any lower level clashes, touches or clearances)

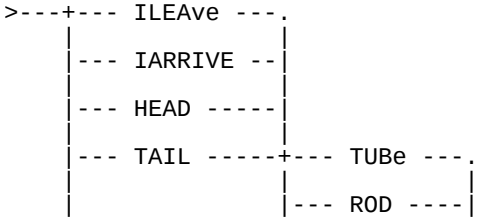
Command Syntax:

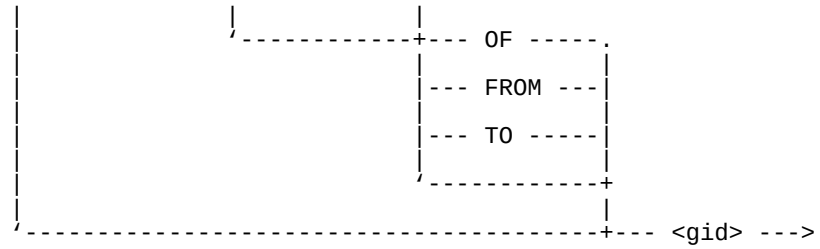


where <cltyp> (clashing type) is one of the following:

`HH HS HI SS SH SI II IH IS NP`

and <clid> (clashing item identifier) is





Note: Clashing type H(ard) automatically includes S(oft) and I(insulation). CLASH automatically includes TOUCH and CLEARANCE.

3.15.3 Reapproving Moved Clashes

Keywords:

REAPPROVE

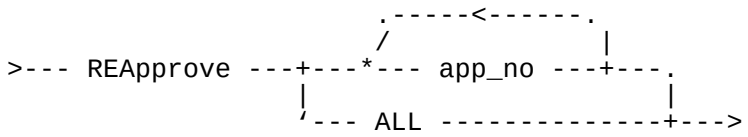
Description:

When an item involved in an approved clash has been moved within the design, the clashes involving that item may be reapproved (if you are sure that such approval is still valid) without the need to reenter the full clash details. The result is that the new obstruction boxes for those items, in the spatial map, are stored with the existing approved clash details.

Examples:

REAPPROVE 5 6 12	Reapproves clashes specified by their numbers in the approved clash list.
REAPP ALL	Reapproves the whole approval list.

Command Syntax:



3.15.4 Removing Clashes from the Approval List

Keywords:

REMOVE

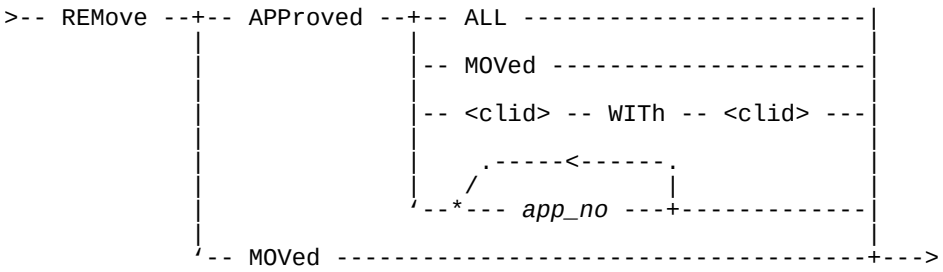
Description:

Removes specified clashes, or clashes between items which have been moved in the design since their approval, from the approval list.

Examples:

`REMOVE APPROVED / ZONE.EQUIP WITH / ZONE.STEELW`	Removes approval of clashes between named elements.
`REM APP 5 6 12`	Removes approval of clashes specified by their numbers in the approved clash list.
`REM APP MOVED` `REMOVE MOVED`	Removes approval of clashes for which one or both clashing items have been moved in the design.
`REMOVE APPROVED ALL`	Empties the approval list.

Command Syntax:



3.15.5 Outputting the Approval List

Keywords:

OUTPUT REPORT APPROVED MOVED

Description:

Outputs the current approval list in a choice of formats (which you set as for the clash report listings summarised in *Running an Obstruction Box Check*).

Examples:

`OUTPUT APPROVED`	Outputs full approval list
`OUTPUT MOVED` `OUTP APPR MOV`	Outputs only those approved clashes which are affected by items which have moved in the design since approval.
`OUTP APPR / ZONE.EQUIP`	Outputs list of approved clashes involving elements or primitives within specified Zone

Examples:

`OUTP MOVED / ZONE.EQUIP` `OUTP APPR MOV / ZONE.EQUIP`	Outputs approved clashes involving only elements or primitives in specified Zone which have moved in the design since approval
`REPORT APPROVED` `REPORT APPROVED ON`	Causes summary of approved clashes to be appended to all subsequent clash reports
`REPORT APPROVED OFF`	Suppresses summary of approved clashes in subsequent reports (the default)

Command Syntax:

```
>-- OUTput --+-- APProved --+-- MOVed --.
             |              |           |
             |              '-----------+--.
             |                             |
             '-- MOVed --------------------+-- <clid> --.
                                           |            |
                                           '------------+-->

>-- REPort -- APProved --+-- <onoff> --.
                         |             |
                         '-------------+--->
```

3.16 Saving and Restoring Clash Data

You may save the results of the current clash run to a file and then restore that data for use later in the same session or during another session.

3.16.1 Saving Data to a File

Keywords:

SAVE

Description:

The SAVE options allow you to save any or all of the following types of data to a named file. In each case you can overwrite an existing file by appending the OVER command to the file name in the usual way.

- The current setup parameters:
  - The obstruction list
  - The limits box co-ordinates
  - The touch and clearance settings
  - Whether or not the midpoint positioning option is in force
  - Any checking options currently specified (e.g. NOCHECK, IGNORE, ACHECK/BCHECK options)
  - Any reporting options currently specified
- The **clash details** resulting from the most recent clash-checking run, including the relevant checking options and obstruction list for inclusion in future reports

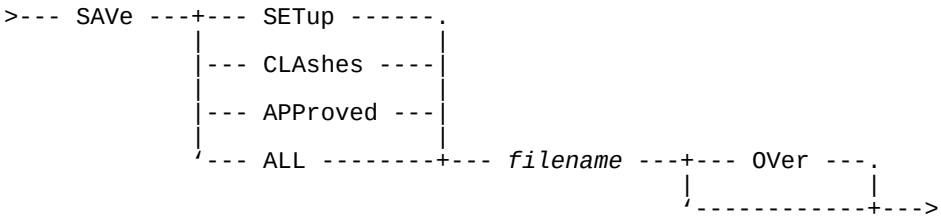
- The current list of **approved clashes**

Examples:

`SAVE SETUP /CLASH1` `SAVE SET /CLASH1 OVER`	Saves setup parameters, as listed in the above description, to file /CLASH1
`SAVE CLASHES /CLASH2` `SAVE CLA /CLASH2 OVER`	Saves clash details, including checking options and obstruction list, to file /CLASH2
`SAVE APPROVED /CLASH3` `SAVE APP /CLASH3 OVER`	Saves current approval list to the file /CLASH3
`SAVE ALL /CLASH4` `SAVE ALL /CLASH4 OVER`	Saves setup parameters, clash details and approval list to file /CLASH4

Note: While the SAVE ALL option is often convenient, bear in mind that you cannot later restore only *part* of the data without affecting the rest. This could mean that when the file is restored you will overwrite some settings that you wish to retain.

Command Syntax:



3.16.2 Restoring Data from a File

Keywords:

RESTORE

Description:

The RESTORE command allows you to read back clash data from a file. Data restored in this way is available for further reference as though generated during the current Outfitting Design session.

The effects of the three types of data that may be restored are as follows:

- **Setup data** *overwrites* any current clash parameter settings. The restored data applies to all subsequent clash checking and reporting operations.
- **Clash data** *overwrites* all current clash information. The original clash numbers, saved with the data, are retained.

- **Approvals data** is *added* to the current approvals list. Approved clashes added in this way are given new approved clash numbers.

Example:

```
RESTORE /CLASH4        Restores all clash-related data from the named file
```

Command Syntax:

```
>--- REStore --- filename --->
```

3.17 Creating Macro Files Incorporating Clash Data

You can create a file incorporating clash data in a format suitable for input as a macro to another Outfitting module, such as Outfitting Draft.

3.17.1 The Principles

The process operates by merging the clash data with information read from a pre-prepared **template file** and then outputting the result of this merger to the required macro file. The template file must contain all necessary commands for the module to which the data is to be transferred, with coded **keywords** marking the locations into which the clash data is to be loaded. Outfitting Design then substitutes its clash report data for these keywords.

The keywords to be used in the template file, each of which is identified by **$**...**$** markers, are as follows:

$CLA1$	First clashing item
$CLA2$	Second clashing item
$CLATYPE$	Clash type
$CLAPOS$	Clash position in World coordinates
$CLATEXT$	Clash description as output in a standard report; for example `HH TOU BOX 1 of /VESS1 with HEAD TUBE of /BRAN1`
$CLANUM$	Clash number
$CLAOWN1$	Significant element owning first clashing item
$CLAOWN2$	Significant element owning second clashing item
$$	Used to represent a single $ character in the macro file

Note: The use of the $ character in this context identifies these keywords as escape codes, as defined in Part I of the *Software Customisation Reference Guide*.

3.17.2 Generating Macro-Style Output

Keywords:

REPORT MACRO

Description:

The command for specifying macro-style output from Outfitting Design's Clash Detection mode is an extension of the REPORT command options described in *Reporting the Clashes Found*.

The REPORT MACRO command must be followed by the name of a valid template file. If the named file cannot be read by Outfitting Design, or if there is an error in the formatting of its keyword content, then the MACRO option is ignored and subsequent reports will be output in the standard way.

To generate a macro file, having first given a valid REPORT MACRO *template_file* command, use the ALPHA FILE syntax to direct your output to the required macro file name. Then output your clash report in any of the usual ways (that is, by using a CHECK, BOXCHECK, CHECKADD or, more probably, OUTPUT command).

When a report is output in macro mode, the following conditions apply:

- The header and summary are not output, so that only the main body data is merged with the template file
- Section identifiers are not output
- The REF and NUMBER options, if in force, are included in the $CLATEXT$ locations
- The BOTH/FIRST/SECOND and PRIMARY options, if in force, are taken into account when working out which clashes to output

Examples:

This example illustrates how you might create an Outfitting Draft input macro for plotting clashing items identified by Outfitting Design. It assumes some understanding of the use of Outfitting Draft, although you need not understand the purpose of all of the Outfitting Draft commands in order to follow the basic principles.

A template file containing the necessary commands for Outfitting Draft to display and plot four views of clashing items might be as follows:

```
$$( $CLANUM$: $CLATEXT$ $$)
 $$( NEW DEPT /DEPT-1
   NEW REGI /REGI-1
    NEW DRWG /DRWG-1
     NEW LIBY /LIBY-1
      NEW DLLB /DLLB-1
       NEW RPLB /RPLB-1
        NEW STYL /STYL-1
          TU ON CL OFF
          DLEV6
        NEW RRST /RRST-1
         NEW RRUL /RRUL-1
         USE /STYL-1 FOR ALL
        $$)

 /DLLB-1
     NEW IDLI /IDLI-$CLANUM$
     ADD $CLAOWN1$ $CLAOWN2$

 /DRWG-1
 NEW SHEE /CLASH-SHEET1-$CLANUM$
 SIZE A4
  NEW VIEW /VIEW1-$CLANUM$
```

```
VREG XR 0.46 YR TO XR 0.9 YR 1
IDLN /IDLI-$CLANUM$
VTYPE UNIV
RRSF /RRST-1
VSCA 1/40
THPOS $CLAPOS$
DIR N

NEW VIEW /VIEW2-$CLANUM$
VREF XR 0 YR 0.1 TO XR 0.44 YR 0.55
IDLN /IDLI-$CLANUM$
VTYPE UNIV
RRSF /RRST-1
VSCA 1/40
THPOS $CLAPOS$
DIR E 0.01 D

NEW VIEW /VIEW3-$CLANUM$
VREG XR 0.46 YR 0.1 TO XR 0.9 YR 0.5
IDLN /IDLI-$CLANUM$
VTYPE UNIV
RRSF /RRST-1
VSCA 1/40
THPOS $CLAPOS$
LOOK D

NEW VIEW /VIEW4-$CLANUM$
VREG XR 0 YR 0.6 TO XR 0.44 YR 1
IDLN /IDLI-$CLANUM$
VTYPE UNIV
RRSF /RRST-1
VSCA 1/40
THPOS $CLAPOS$
ISO 3
SHEE
NEW NOTE
NEW TEXP
AT X137 Y32
ALIGN HB JUST C CHEI 4 FONT 1
BTEX '$CLANUM$: $CLATEXT$'
SHEE
UPDATE DESI

PLOT SHEE FILE /$CLASHPLOT$CLANUM$
$$(-------------------------------------------------$$)
```

You could name this template file /Outfitting Draft.TEM and then merge existing clash data with it to create an Outfitting Draft macro input file /Outfitting Draft.MAC, thus:

```
REPORT MACRO /   specifies macro mode and template file
OUTFITTING
DRAFT.TEM

ALPHA FILE /     specifies merged output file (the macro)
OUTFITTING
DRAFT.MAC

$P '$$           ( OUTFITTING DRFAT macro created using OU
                 TFITTING DRAFT.TEM $$)'
```

```
$P ''            leaves blank line after header

OUTPUT           outputs all current clash data to macro

$P '$$.'         adds macro end marker $.

APLHA FILE END   closes output file

REPORT MACRO OF  reverts to normal output
F
```

The resulting macro /Outfitting Draft.MAC will comprise multiple copies of the Outfitting Draft commands with the appropriate data substitutions for each clash output.

If you run this macro, the specified four views will be plotted for each pair of clashing items diagnosed and output by Outfitting Design. Each set of views will be sent to a plotfile named /CLASHPLOT*n*, where *n* is the clash number allocated by Outfitting Design.

Command Syntax:

```
>--- REPort --- MACro ---+--- template_filename ---.
                         |                         |
                         '--- OFF -----------------+--->
```

Querying:

```
Q CLASH REPort MACro
 Gives name of template file (or OFF)
```

3.18 Displaying Clashes Visually

Keywords:

REPORT GRAPHICS COLOUR CLASH OBST RESETHIGHLIGHT

Description:

In addition to alphanumeric reporting of clash data (to your terminal or to a file), Outfitting Design can show the locations of clashes graphically by highlighting the clashing elements on the display.

When graphical reporting is switched on, for each clash found, the element in the obstruction list (the 'clashee') is displayed in the CLASH colour and the element in the check list (the 'clasher') is displayed in the OBST colour.

If a clashing element is already in the drawlist (i.e. already displayed), it will be highlighted by a change to the appropriate colour. If the element is not currently displayed, it will be added to the drawlist automatically (in the default visible colour) and will then be highlighted in the appropriate colour.

The element stays highlighted until another clash check is run, or until you remove all graphical highlighting specifically by using the RESETHIGHLIGHT command.

Examples:

`REPORT GRAPHICS ON`	Highlights clashes on the graphical display Clashes not highlighted on the graphical display (the default)
`REPORT GRAPHICS OFF`	
`RESET`	Removes graphical highlighting from all Volume views

Command Syntax:

```
>--- REPort --- GRAphics --- <onoff> --->
>--- COLour -+- CLASH -.
              |         |
              '- OBST --+- colour_name --------------.
                        |                            |
                        '- MIX RED n GREen n BLUe n -+->
>--- RESEThighlight --->
```

Querying:

```
Q CLASH REPort GRAphics
```

3.19 Automatic (Continuous) Clash Detection

As an alternative to intermittent clash checking on demand, as assumed by the preceding command syntax in this chapter, you may check for clashes automatically each time you modify the design.

3.19.1 Specifying Automatic Clash Detection Mode

Keywords:

AUTOCLASH

Description:

When automatic clash checking is switched On, a clash check is carried out at the end of every command line in which an element has been modified in some way that could cause a clash to occur; that is:

- When a new element has been created
- When an element's position and/or orientation has been changed
- When an element's geometry has been changed

Each clash check is carried out using the current option settings (obstruction list, limits box, etc.), the modified element being included automatically in an implied DESCLASH command (see *Entering Clash Detection Mode*).

The results of each clash check replace those of any previous checks, so that any reported clashes *must* result from the actions of the last command. Your attention will already be focused on the current element, so that it is usually most convenient to rely on graphical highlighting to show the clash (as explained in *Displaying Clashes Visually*), rather than to output the clash data to a file. You can use the OUTPUT command to see more details of the clash if required.

Automatic clash checking is switched Off by default.

Command Syntax:

```
>--- AUTOCLASH --- <onoff> --->
```

Querying:

```
Q AUTOCLASH
```

Note: In order to avoid spurious clash reports when a new Branch is created, the last section of implied tube in a Branch is checked only if the Branch LTAI attribute is set to True. (The LTAI attribute is set automatically by Outfitting Design when the Branch Tail is positioned.)

Note: The clash-checking rules are as follows:

Note: If the current element is a Piping Component *and* is the last component in the Branch, then its leave tube is checked only if the Branch LTAI attribute is set to True.

Note: If the current element is a Branch which has no members, then the tube which constitutes the Branch is checked only if the Branch LTAI attribute is set to True.

3.19.2 Logging the Clashes

Keywords:

CLASHLIST

Description:

In order to provide an audit trail of the effect of the current session, the system keeps a list of all elements for which it has carried out an automatic clash check. You may review the effect of your design changes by rerunning a clash check on all the elements in this list.

Note: The results of the checks derived from the clash list in this way will be based on the current option settings, which may not be the same as those in force when the original checks were made.

Examples:

`CLASHLIST DISPLAY`	Reruns a clash check on all elements in the clash list
`CLASHLIST EMPTY`	Removes all elements from the clash list

3.20 Error Messages in Clash Detection Mode

The following error messages are those that are specific to Outfitting Design's Clash Detection mode.

(69:4) May only specify tube on piping or hanger components

You have tried to specify implied tube connected to an inappropriate type of element (see *Defining the Obstruction List* for examples of the syntax).

(69:5) Must be a BRANCH or HANGER to specify HEAD or TAIL tube

You have tried to specify a head or tail tube connected to an inappropriate type of element (see *Defining the Obstruction List* for examples of the syntax).

(69:7) *name/refno* has not been fully mapped

The specified element has not had its positional data updated in the spatial map.

(69:8) *name/refno* is already in the obstruction list

You have tried to add an element to the obstruction list that is already present in the list. Use the Q CLASH OBSTRUCTION command to check the current list if necessary.

(69:9) *name/refno* is not in the spatial map

The specified element has not had its positional data updated in the spatial map.

(69:11) *name/refno* is not in the obstruction list

You have tried to remove or exclude an element from the obstruction list that is not present in the list. Use the Q CLASH OBSTRUCTION command to check the current list if necessary.

(69:12) Approved clash number *integer* is not in approval list

You have tried to remove or reapprove a clash by referring to an approved clash id that is not present in the list. Use the OUTPUT APPROVED command if necessary to check the current list.

(69:13) Element type *word* is not currently a section header

You have tried to remove a section from the main body of the report that is not present in the list of sections. Use the Q CLASH REPORT MAIN command to check the current sections if necessary.

(69:16) Maximum number of element types for the NOCHECK option exceeded

The maximum number of element types that you may specify in a NOCHECK command is 20.

(69:17) Element type *word* is not in the list of those set for NOCHECK

You have tried to use the WITHIN command to reinstate an element type which has not been previously specified in a NOCHECK command.

(69:18) Clearance value must be positive

You have specified a negative argument in a CLEARANCE command. You must specify either a positive value (which must be greater than the current touch gap) or zero (which switches the clearance option off).

(69:19) Clearance value must be greater than touching gap
(69:20) Touching gap must be positive
(69:21) Touching gap may not be greater than current clearance
(69:22) Touching overlap must be positive
(69:23) Invalid element type for an approved clash

See *Adding Clashes to the Approval List* for the valid ways of specifying approved clashes.

(69:24) Approved clash already exists

You have tried to add a clash to the approval list that is already present in the list. Use the OUTPUT APPROVED command to check the current list if necessary. See *Adding Clashes to the Approval List*.

(69:25) *integer* is not a valid approved clash number

You have tried to remove or reapprove a clash by referring to an approved clash id that is not present in the list. Use the OUTPUT APPROVED command if necessary to check the current list. See *Adding Clashes to the Approval List*.

(69:26) There are no items in the obstruction list

You cannot run a clash check until you have added at least one element to the obstruction list.

(69:27) Maximum number of element types for the SECT option exceeded

The maximum number of element types that you may specify in a REMOVE SECTIONS command is 20.

(69:28) Clashes cannot be saved as no clash run has been made

The SAVE CLASHES command can only save clash data currently held in memory from the most recent clash-checking run. There is no valid data in memory.

(69:29) Error in file format

You have probably tried to restore data from a file that was not written to by an Outfitting Design (or CLASHER) SAVE command.

(69:33) No clash run has been made

The SAVE CLASHES command can only save clash data currently held in memory from the most recent clash-checking run. There is no valid data in memory.

(69:34) File *filename* already exists. Output defaulted to terminal

You can only overwrite the contents of an existing file by using either the FFILE command or the FILE ... OVER option. Your FILE command has been ignored as a protection against accidentally overwriting the named file.

(69:35) File *filename* is in use. Output defaulted to terminal

You have specified a file that is already open (or which is otherwise inaccessible for writing to). Your output will therefore be sent to your terminal until you specify a valid file name.

(69:36) File *filename* does not exist

The file from which you are trying to read is not accessible from the OS directory from which you entered Outfitting. Do you need to give the full pathname for the file

(69:37) File *filename* already exists

You can only overwrite the contents of an existing file by using either the FFILE command or the FILE ... OVER option.

(69:38) File *filename* is in use

You have specified a file that is already open (or which is otherwise inaccessible for writing to).

(69:39) Page width may not be less than 50 characters

A page width of less than 50 characters will be too narrow to accommodate the clash data lines in the report.

(69:40) Page length must be between 20 and 200 lines

The page length must be of a suitable length to accommodate a sensible top and bottom margin and a reasonable number of report lines.

(69:41) Page margin must be between 0 and 20

The margin width must be positive and must allow enough remaining line length to accommodate a typical report line.

(69:43) No spatial map exists for DB containing *name/refno*

The specified element forms part of a DB for which a spatial map has not yet been created. The positional data for the element is therefore unavailable for clash checking.

(69:47) The leave tube for *name/refno* is not in the spatial map

The specified element has not had its positional data updated in the spatial map since it was connected to the next downstream component.

(69:48) No approved clashes exist between *name/refno* and *name/refno*

You have tried to reapprove one or more clashes between named items for which no approved clashes have been specified.

(69:56) Unable to open file *filename*

The specified file cannot be found. Check that you have specified its correct pathname and that the directory containing it is accessible from the directory from which you entered Outfitting.

(69:57) Line *integer* of macro template *filename* is too long

The specified command line in your template file is of such a length that, when the clash-checking keywords have been replaced by the corresponding clash data, it exceeds the maximum permitted length for an Outfitting command line.

(69:58) Line *integer* of macro template *filename* does not have matching dollar signs

Each keyword in a macro template file must be enclosed between a pair of **$** escape characters. The **$** characters in the specified file do not form properly matched pairs.

(69:59) Line *integer* of macro template *filename* has unrecognised keyword

See *Creating Macro Files Incorporating Clash Data* for a list of valid keywords for use in macro template files.

(69:60) Macro template *filename* has no keywords

There are no identifiable locations in the template to which clash data can be transferred. Outfitting Design cannot check the command syntax in the template and it is most likely that the enclosing **$** characters have simply been omitted from the keywords. See *Creating Macro Files Incorporating Clash Data*.

(69:61) Spatial map marked as incomplete (for DB *integer*)

The positional data has not been updated in the spatial map since the specified DB was last accessed from a design module. Entry to a design module with map maintenance off will cause the map to be flagged as incomplete if *any* design change is made, even if no clashes are affected.

(69:66) No obstruction list. Use 'OBS ALL' or 'OBS id1 id2 ... idn'

You cannot run a clash check until you have added at least one element to the obstruction list.

(69:67) Currently OBS ALL. Specifying additional obstructions has no effect

Since the obstruction list already comprises all relevant elements in the Outfitting Design DB, there is no point in trying to add specifically named elements to the list.

(69:68) No template file has been specified for macro output

When using the REPORT MACRO option you *must* specify the name of the template file from which the structure of the macro is to be read. This is quite independent of the specification of the output file (FILE or FFILE command) which defines where the final macro is to be sent.

(69:79) ALL not valid for EXCLUDE command

The members of obstruction list elements that are to be excluded from the list must be individually specified. EXCLUDE ALL would effectively eliminate the entire obstruction list (use REMOVE OBSTRUCTION ALL if you really want to do this).

(69:80) *name/refno* is already in the exclusion list

You have tried to add an element to the exclusion list that is already present in the list. Use the Q CLASH OBSTRUCTION command to check the current list if necessary.

(69:81) *name/refno* is not in the exclusion list

You have tried to remove an element from the exclusion list that is not present in the list. Use the Q CLASH OBSTRUCTION command to check the current list if necessary.

4 Copying Model Data from Outfitting to Review

This chapter tells you how to use the Outfitting Design EXPORT command to identify a list of objects which are to be reviewed graphically (using AVEVA's Review product range) and to define how they are to be represented. EXPORT extracts from the Outfitting Design database the relevant data for the primitives which will make up the display, including the Outfitting Design hierarchy, and stores it in an intermediate file (a **model file**) for use by Review.

4.1 Model File

To specify the name of the model file, to which design data is to be copied, use one of the commands:

EXPORT FILE *filename*

EXPORT FILE *filename* **READ**

Opens an existing file in read-only mode, allowing its content to be queried but not overwritten.

EXPORT FILE *filename* **OVERwrite**

Overwrites an existing file of the same name.

EXPORT FILENote *text*

Adds a line of user-supplied text to the model file header.

For example,

```
EXPORT FILE /TESTFILE
EXPORT FILE /TESTFILE READ
EXPORT FILE /TESTFILE OVER
EXPORT FILENOTE 'Platform PA2 REVIEW Data'
```

To query the name of the current model file, use the command

Q EXPORT FILE
(similarly **Q EXPORT FILENote** to query the header text)

4.2 Object Selection and Colour Setting

Elements that are to be copied to the model file are held in a **draw list**. Elements may be specified explicitly or by using a PML expression to define **selection criteria**. For information about PML, see the *Software Customisation Reference Manual*.

Only **significant elements**, such as BRAN, EQUI, SUBS, HANG, STRU, PTRA, etc., may be added to or removed from the draw list. If you try to add or remove an element below a

significant element (that is, a primitive), the whole of its owning significant element will be added to or removed from the draw list.

Colours to be used to display the different element types can be specified explicitly or by using the **Autocolour** selection rules - see the AUTOCOLOUR command in Part 1 of the *Outfitting Design Reference Manual.*

Examples:

```
EXPORT /VESS1 COLOUR 2
```

Specifies colour of exported element directly.

```
EXPORT AUTOCOLOUR  ALL BOXES WITH (XLEN GT 100)  COLOUR 10
```

Specifies using AUTOCOLOUR rules.

```
EXPORT AUTOCOLOUR  ALL BRAN MEMBERS COLOUR (:ICOLOR OF SPREF)
```

Specifies using AUTOCOLOUR rules.

The colour number, whether given as an integer or as an expression, refers to the colour number to be used in Review.

The order in which rules are given is important, because they are evaluated in this order until a rule is encountered for which the selection criteria are satisfied. This is the rule from which the colour is taken. If no rule is satisfied, or if no colour rules have been given, or if the selection is invalid for some reason, then colour 0 is used. Rules may be reordered, removed and controlled by the following commands:

```
EXPORT AUTOCOLOUR ON
```

Turns the use of Autocolour in EXPORT mode on. The rules will be ignored until turned on.

```
EXPORT AUTOCOLOUR OFF
```

Turns the use of Autocolour in EXPORT mode off.

```
EXPORT AUTOCOLOUR RESET
```

Clears the current selection by removing all rules.

```
EXPORT AUTOCOLOUR REMOVE 4
```

Removes rule 4.

```
EXPORT AUTOCOLOUR REORDER 4 TO 99
```

Reorders rule 4 to position 99.

Querying:

A maximum of 200 Autocolour rules are allowed at present. The following queries are also available:

```
Q EXPORT AUTOCOLOUR NUM
```

Returns the number of rules.

```
Q EXPORT AUTOCOLOUR MODE
```

Returns the mode state (on or off).

`Q EXPORT AUTOCOLOUR` *integer*

Returns the selection criteria for rule *integer.*

`Q EXPORT AUTOCOLOUR FOR` *identifier*

Returns colour to be selected for identified element.

4.3 Controlling the Representation of Holes

Holes may be represented realistically, with colour shaded inner surfaces, and may be 'seen through'. To control the appearance of holes, use the commands

EXPORTHOLEs ON
HOLEs OFF(default)
HOLEs SOLID

EXPORT HOLES SOLID represents holes as solids in colour 1, which by default is black. Solid holes cannot be seen through.

The state of the hole settings may be queried using

Q EXPORT HOLES

Note: Since holes are constructed using facets which involve a large amount of geometric data, switching holes 'on' will increase the computational overhead when processing. The more holes in the model, the longer the model file will take to generate, and the more slowly it will be drawn in Review.

Note that other representation settings, such as Tube, Centreline, Obstruction, Insulation and Drawing Level, are taken from the current Outfitting Design settings.

4.4 Text Encoding

The format used to encode text to be exported can be set using the **EXPORT ENCODING** syntax. Text may be exported using Unicode™ UTF-8 character encoding in order to support foreign language character sets.

EXPORT ENCODING EUC- set EUC encoding
EXPORT ENCODING UTFEight- set UTF-8 encoding
Q EXPORT ENCODING - gives the currently selected format

4.5 Copying Data to the Model File

To transfer data about the primitives owned by all elements in the current draw list into the current model file, use the command

EXPORT FINish

This command does not change the current draw list.

4.6 Exporting Implied Tubing

Four distinct policies exist for exporting implied tubing from Outfitting to Review. Together, they offer great flexibility in allowing users to specify the way implied tube is managed when it is exported.

The policies are:

- Export **all** implied tube items in the same branch to a **single** container.
- Export **each** implied tube item in a branch to a **separate** container.
- Export **each** implied tube item in a branch into the container of the immediate **upstream** element in the branch.
- Export **each** implied tube item in a branch into the container of the immediate **downstream** element in the branch.

The Outfitting Design/EXPORT commands to select each of these policies are:

```
EXPORT IMPLIED TUBE [INTO] SING/LE [CONT/AINER]
EXPORT IMPLIED TUBE [INTO] SEP/ARATE [CONT/AINERS]
EXPORT IMPLIED TUBE [INTO] UP/STREAM [CONT/AINERS]
EXPORT IMPLIED TUBE [INTO] DOWN/STREAM [CONT/AINERS]
```

Where keywords enclosed in [] are optional and '/' indicates possible abbreviations.

For the SINGLE case the single container inherits the name of the owner branch.

For the SEPARATE case the naming convention 'TUBE n of BRANCH name' has been used to name the separate tube containers.

For the UPSTREAM case the container inherits the name of the immediate upstream element.

For the DOWNSTREAM case the container inherits the name of the immediate downstream element.

The default mode is SINGLE.

4.7 Exporting Insulation/Obstruction Translucency and Colours

Users can export design colours and translucent insulation/obstructions to REVIEW using the syntax:

```
EXPORT REPR/esentation ON/OFF
```

EXPORT REPR ON turns the command REPR/esentation ON allowing design colours and translucent insulation/obstructions to be exported.

EXPORT REPR OFF turns the command OFF and design colours and translucent insulation/obstructions will not be exported.

4.8 EXPORT Command - Full Syntax

```
>- EXPORT -+-- HOLES --+-- ON -----.
           |           |           |
           |           |-- OFF ----|
           |           |           |
           |           '-- SOLID --+-->
           |
           |-- FILENote text -->
           |
           |-- FILE name---+-- TEXT --.
           |               |          |
           |               '----------+-- READ -------.
           |                          |               |
           |                          |-- OVERwrite --|
           |                          |               |
           |                          '---------------+--->
           |
           |                .-------<----.
           |               /             |
           |-- <selatt> --*-- <selatt> --'
           |              |
           |              |-- COLour integer -->
           |              |
           |              '-->
           |
           |-- ENCODING --+-- EUC -----.
           |              |            |
           |              '–UTFEight --+→
           |
           |- IMPLIED TUBE-+ -------.
           |               |        |
           |               '- INTO- +-- SINGle ----.
           |                        |              |
           |                        |-- SEParate --|
           |                        |              |
           |                        |-- UPstream --|
           |                        |              |
           |                        '- DOWNstream -+ - CONTainers->
           |                                       |
           |                                       '-------------->
           |
           |-- REPResentation --+-- ON -----.
           |                    |           |
           |                    '-- OFF ----+-->
           |
           |-- FINish -->
           |
           '-- <expcol> -->
```

Note: <selatt> is the general selection syntax.
<expcol> is the AUTOCOLOUR command

For more information, see *Part 1 of the Outfitting Design Reference Manual*.

Index

R

S

T