

EPICS Lecture @ KEK

Database Design with VisualDCT

Takashi Nakamoto
June 26th, 2013

Based on presentation by Nicholas Di Monte, APS

Database Design with VDCT

- Outline
 - **Introduction to VDCT**
 - **Using VDCT**
 - **Converting database files to VDCT**
 - **New features/options**
 - **Some VDCT examples**
 - **Known problems with VDCT**
 - **Resources**
 - **Acknowledgements**

What is VDCT?

- VDCT is Visual Database Configuration Tool
- Beta version released summer of 2002, funded by SLS
- Developed by *Cosylab*
- Replacement for text editor, DCT, JDCT, GDCT or Capfast
- VDCT developed to provide missing features in Capfast and GDCT.
- Supports hierarchical design
- Written in Java
 - **Therefore supported in various systems**
 - **Java Runtime Environment 2**
- Importing existing DB and DBD files

What is VDCT?

- VDCT features
 - **GUI features**
 - *Clipboard, undo, redo, object inspector, visual linking*
 - *Data flow arrows, not process flow*
 - **Supports hierarchal design**
 - *Based on the pvname separator*
 - *Grouping “grp1:grp2:test1AO”*
 - *VDCT templates can be used.*
 - **Separate VDB file as a template with ports and macros defined.**

What is VDCT?

- VDCT features
 - **Powerful DB parser**
 - *Supports existing DB's*
 - *Preserves DB comments, record/field order*
 - # normal comments
 - #! VDCT layout comments
 - *DB's can be edited manually*
 - **Single file which contains both DB and display data**
 - *GDCT created two separate files*

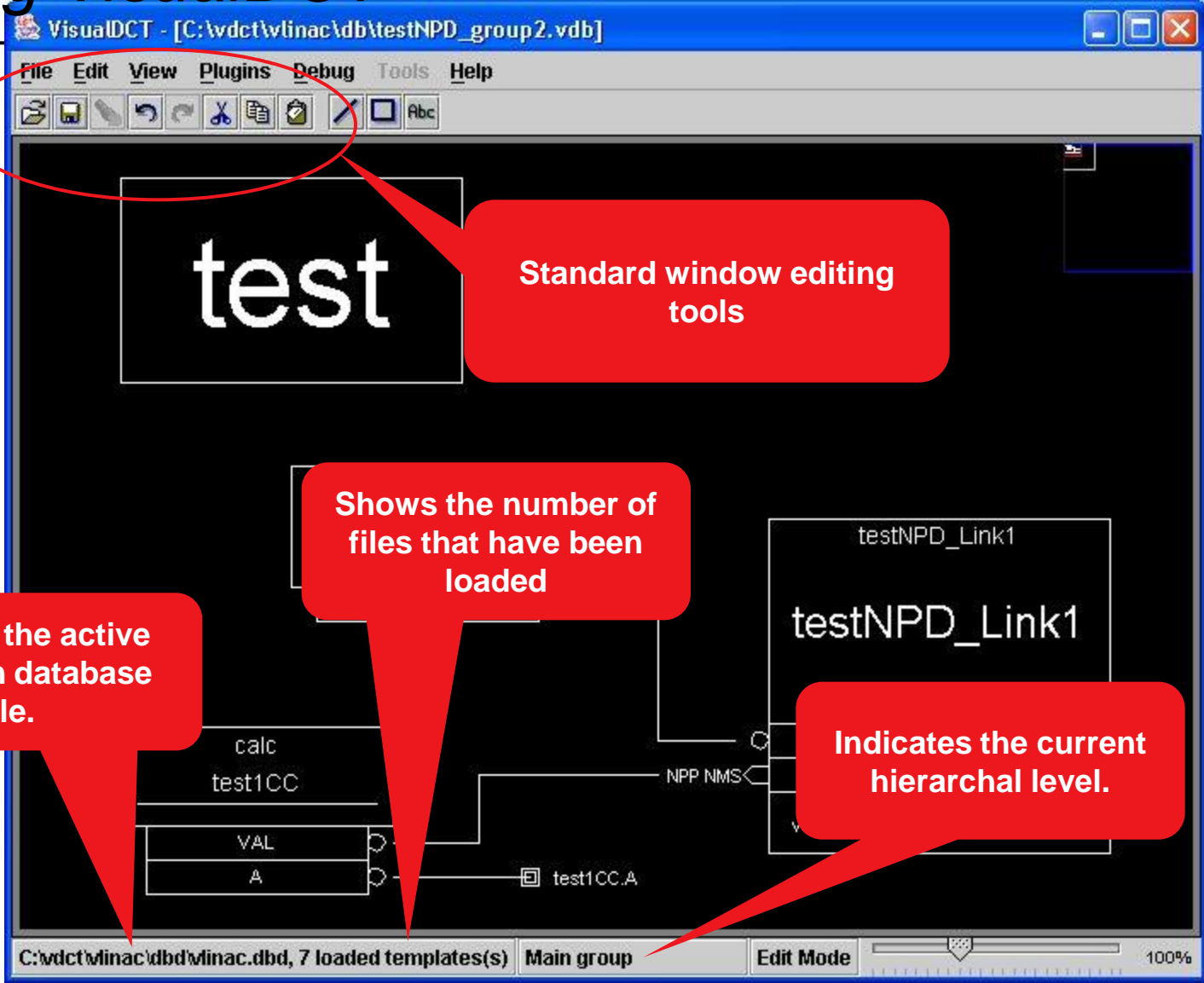
What is VDCT?

- VDCT features
 - **Rapid database development**
 - **Simple mouse-clicks**
 - **Visualization of record instances**
 - *Easier to understand*
 - **Yet no field description as with DCT, JDCT & GDCT**
 - *Detect errors faster*
 - **e.g. broken links shown with a cross**
 - **Database can be split into logical blocks (grouping)**
 - *e.g. hierarchical design*
 - **Printing ?**

Using VisualDCT

- To start VisualDCT in Windows
 - Execute (double click) “VisualDCT2.4.1253.jar”
 - Or, use command line options
 - *VisualDCT2.4.1253.jar [<DBDs>] [<DB>]*
- Load DBD file(s)
 - Recommend selecting save option in “DBD Manager”
- Load DB or VDB file.
- Save work with a VDB extension. (recommended)
- Once a VDB file is created and saved, no need to specify DBD files, DBD files will be included at the beginning of a VDB file.
 - **#! DBDSTART**
 - **#! DBD(“ ../.. /dbd/vlinac.dbd”)**

Using VisualDCT



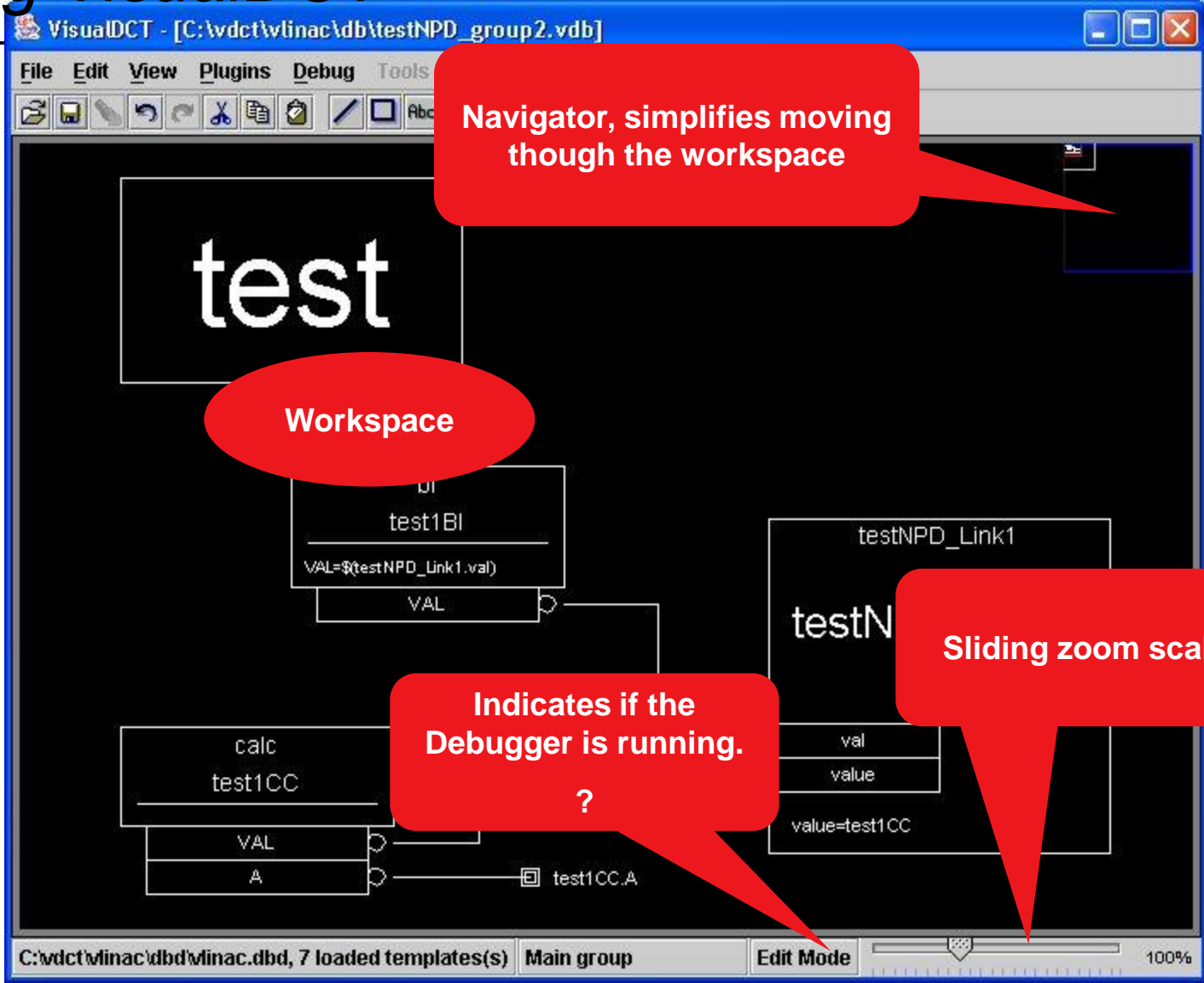
Standard window editing tools

Shows the number of files that have been loaded

Displays the active definition database file.

Indicates the current hierarchal level.

Using VisualDCT



Navigator, simplifies moving though the workspace

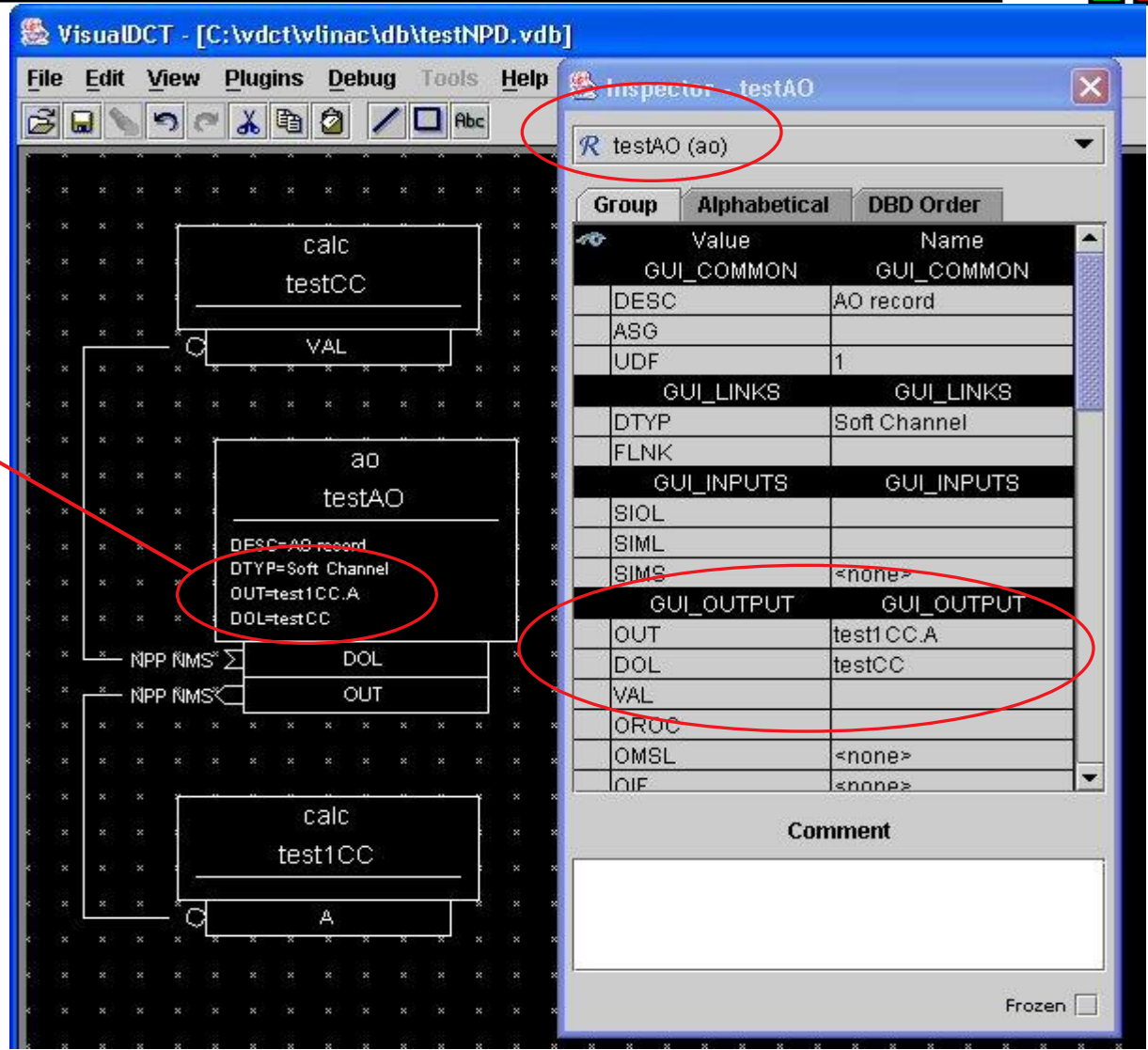
Workspace

Indicates if the Debugger is running.
?

Sliding zoom scale.

Using VisualDCT

- Inspector
- Records
 - Fields
 - *Visible*
- Links
 - Data flow

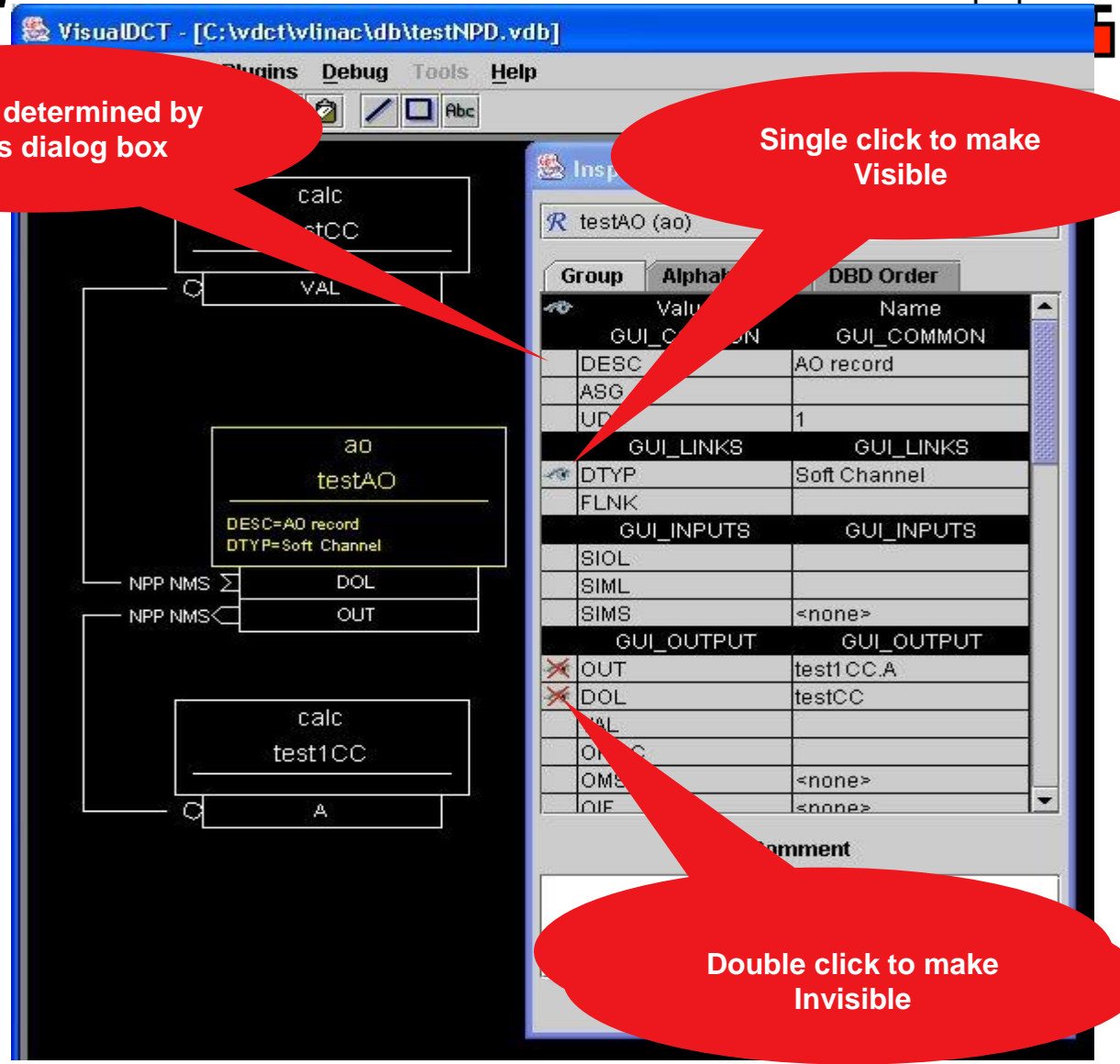


The screenshot shows the VisualDCT interface with a data flow diagram on the left and an 'Inspector' window on the right. The diagram includes blocks for 'calc testCC', 'ao testAO', and 'calc test1CC', with various input and output fields like 'VAL', 'DOL', 'OUT', and 'A'. The 'Inspector' window is titled 'testAO' and displays a table of fields and their values.

Value	Name
GUI_COMMON	GUI_COMMON
DESC	AO record
ASG	
UDF	1
GUI_LINKS	GUI_LINKS
DTYP	Soft Channel
FLNK	
GUI_INPUTS	GUI_INPUTS
SIOL	
SIML	
SIMS	<none>
GUI_OUTPUT	GUI_OUTPUT
OUT	test1CC.A
DOL	testCC
VAL	
OROC	
OMSL	<none>
OIF	<none>



Using VisualDCT



Default is determined by Settings dialog box

Single click to make Visible

Double click to make Invisible

Field Visibility Values

-0: **NON_DEFAULT_VISIBLE**

-Blank for build 1249

-0: **VISIBILITY_SELECT**

-Blank for build 1250

-1: **ALWAYS_VISIBLE**

-Eye

-2: **NEVER_VISIBLE**

-Eye w/Red X

Group	Alpha	DBD Order
GUI_COMMON		
Value	Name	
GUI_COMMON	GUI_COMMON	
DESC	AO record	
ASG		
UD	1	
GUI_LINKS		
DTYP	Soft Channel	
FLNK		
GUI_INPUTS		
SIOL		
SIML		
SIMS	<none>	
GUI_OUTPUT		
<input checked="" type="checkbox"/>	OUT	test1CC.A
<input checked="" type="checkbox"/>	DOL	testCC
<input type="checkbox"/>	VAL	
OMC		
OMS	<none>	
OIF	<none>	

Using VisualDCT

Visibility text in vdb file

#! Visibility("testAO.DTYP",1)

#! Visibility("testAO.OUT",2)

#! Visibility("testAO.DOL",2)

Visibility Defined:

#! Visibility("fieldname", visibility)

Where visibility:

- **0 – NON_DEFAULT_VISIBLE**
 - **Build 1249 and earlier.**
- **0 – VISIBILITY_SELECT**
 - **Build 1250 and later.**
- **1 – ALWAYS_VISIBLE**
- **2 – NEVER_VISIBLE**

Using VisualDCT

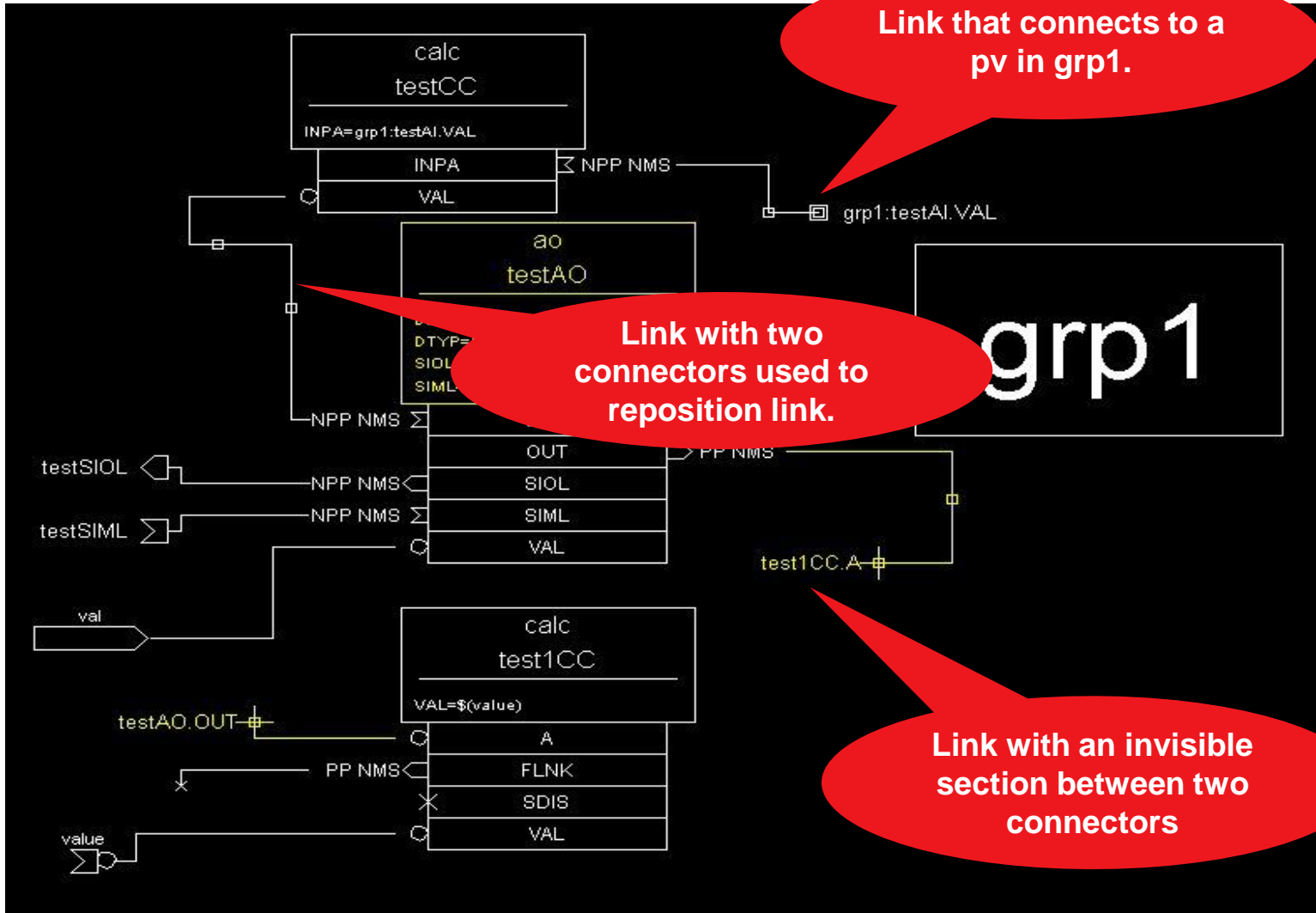
Visibility text in vdb file – why Option #0 should be NON_DEFAULT_INVISIBLE for build 1249

- **The visual graphics should convey the logic of the design.**
- **When importing an old text database, almost all fields have been defined with at least the default value.**
- **The designer should highlight the fields of interest to clarify the logic flow.**
- **Limited work space in the graphical window.**

Build 1250 added an option to change this:

- **Defined in: View => Settings => Visual TAB**
- **Uncheck “Show value of fields when it is not default”**
 - *All fields are now hidden except for those fields set to visible*

Using VisualDCT (links)



Using VisualDCT (links)

Link/Connector text in vdb file

```

#! Field("testAO.OUT",255,1,"testAO.OUT")
#! Link("testAO.OUT","testAO/OUT2")
#! Connector("testAO/OUT2","testAO/OUT1",660,340,255,"",0)
#! Connector("testAO/OUT1","testAO/OUT",620,380,255,"",1)
#! Connector("testAO/OUT","test1CC.A",220,480,255,"",0)

```

Using VisualDCT (links) – Last slide

Link/Connector text in vdb file

#! Field("fieldname", color, rotated, "description")

Where rotated: (*not documented*)

- 0 – Left side of field box
- 1 – Right side of field box

#! Link("fieldname", "inLinkID")

#! Connector(" inLinkID ", " outLinkID ", x, y, color, "desc", option)

Where option: (*not documented*)

- 0 – Visible
- 1 – Invisible
- 2 – External Input
- 3 – External Output

Using VisualDCT

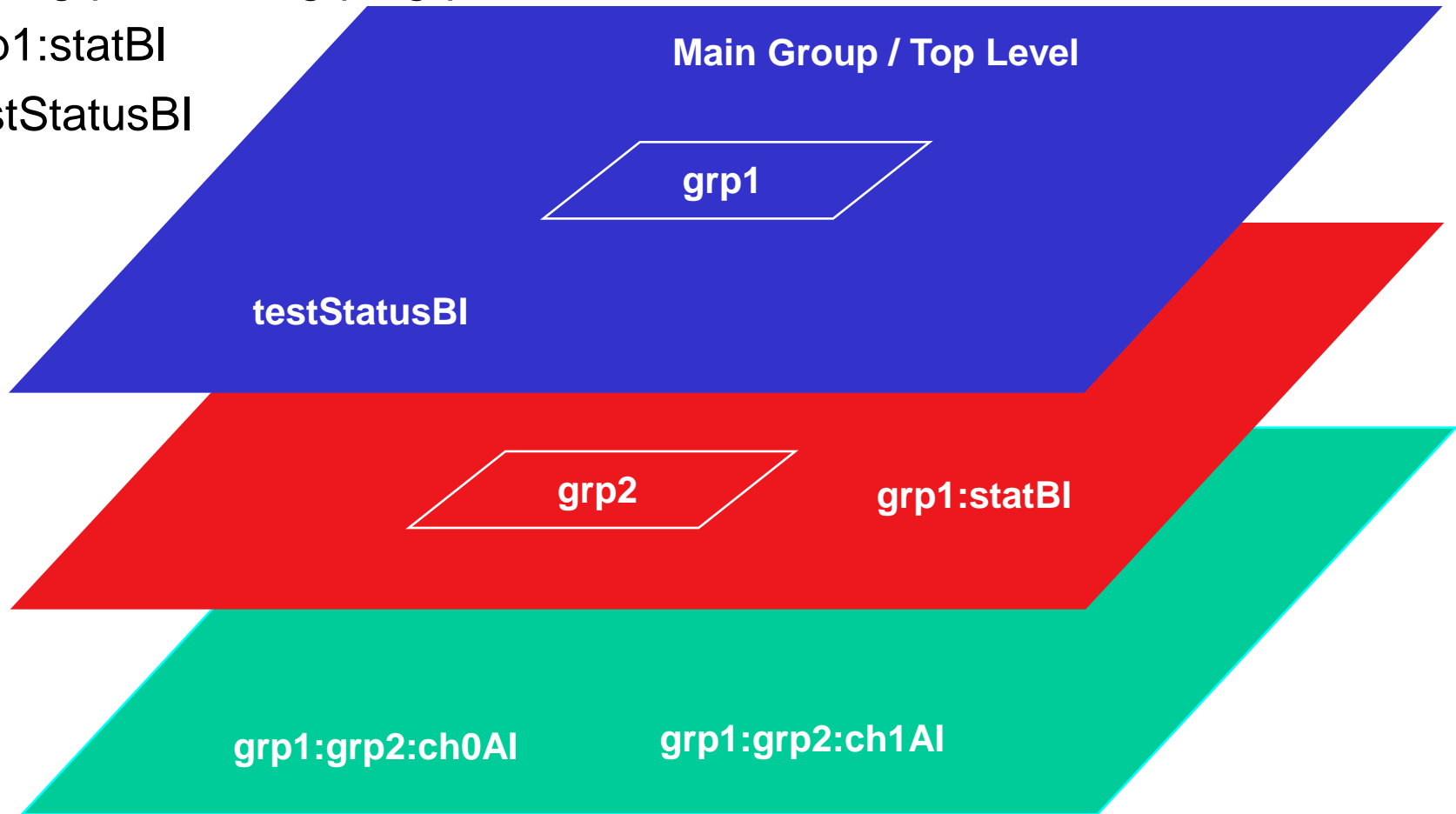
Hierarchy Support

- Based on the pvname separator
- Grouping “grp1:grp2:test1AO”
 - **This will create three levels**
- Grouping must be enabled before loading DB
 - **Separator must also be defined**
- Support templates
 - **Use [Generate...](#) command to flatten vdb with templates**
 - ***Macros* pass information into a template**
 - ***Ports* pass information upwards out of a template**
 - **Use import command to add template**

Using VisualDCT

Hierarchy

- grp1:grp2:ch0AI, grp1:grp2:ch1AI
- grp1:statBI
- testStatusBI



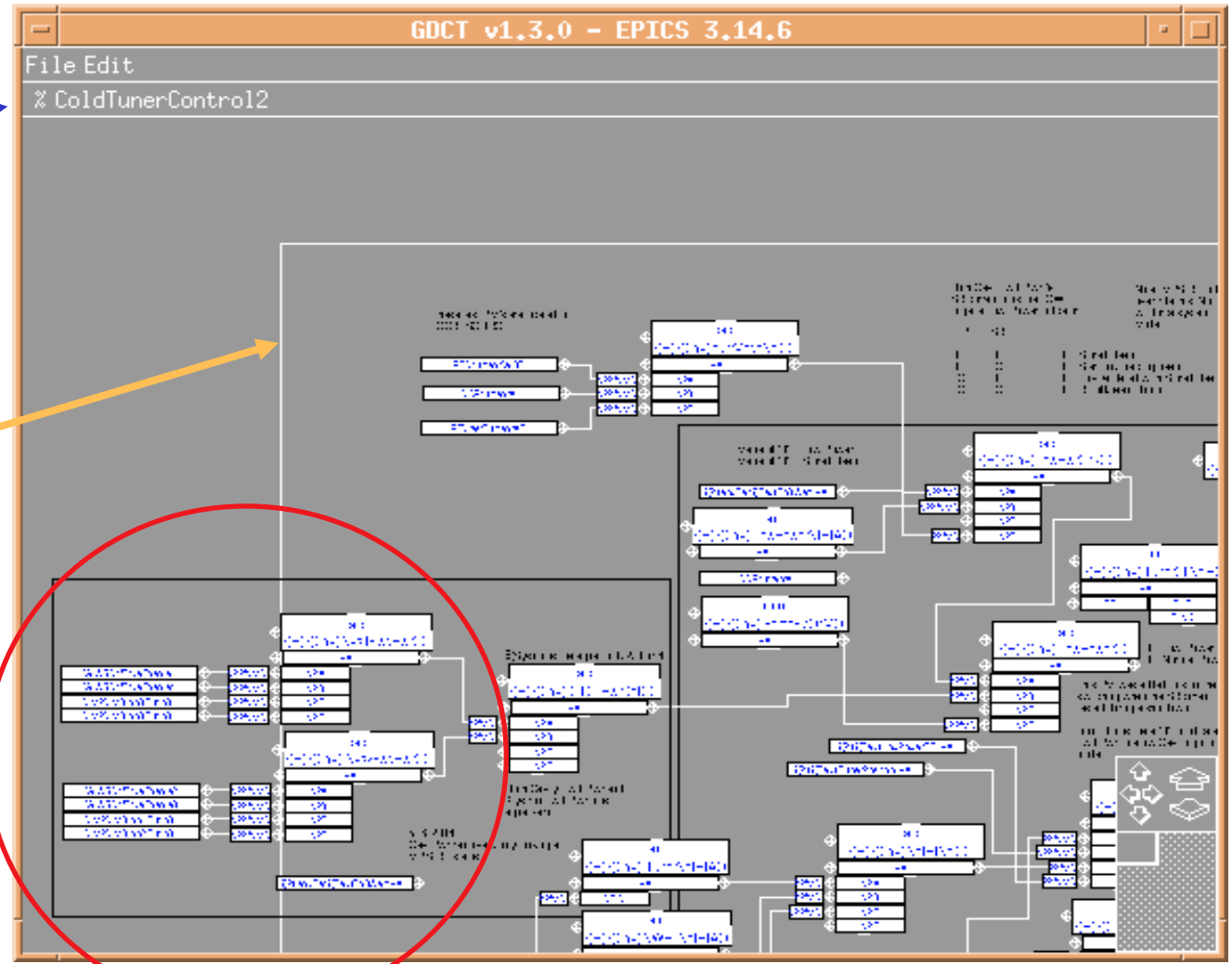
Converting files

- Converting from GDCT313 to VisualDCT
 - From the File menu, select “**Save As VDCT...**”
 - Minor touch up maybe needed.
 - All graphic items must be in the defined workspace outlined by the white border in GDCT

Converting files

GDCT

Workspace border



Converting files

- Converting a DB text file to VisualDCT
 - In VisualDCT select View menu, then Settings
 - *Then select Visual Tab*
 - Uncheck “**Show value of fields when it is not default**”
 - Load DB file
 - Rearrange display for clarity
 - Save with a .vdb extension (recommended)
 - *The Generate command will only create a .db file*

Converting files

- Tools not to use on VDB files, JDCT & DCT313
 - **They remove all display formats**
- Caution when using “vi” or text editor

Some more recent command/options

- Morph command
 - **Allows a record to be converted from AI to AO, etc.**
- Speed option
 - **Silhouette when moving a record**
- Field Visibility
 - **Option to change the default visibility**
- Window Pan Direction
 - **Push or pull while moving around workspace.**
- Spreadsheet View
 - **Edit large number of similar records without interesting linking**

JCA Debug Plugin (1 / 2)

- **Sponsored by SLS**
- **Enable “live mode” - view real PV values in your template**
- **Macros are obtained when a substitution file is provided**
- **Value is displayed inside the template instance symbol together with its timestamp**
- **Fields to be displayed are selected with the inspector**

The screenshot displays the JCA Debug Plugin interface. On the right, a 'live mode' window shows the following data:

```

ao
slide1:speed
-----
0.200
07:50:56.576
-----
OUT=sm1:speed.VAL NPP NMS
  
```

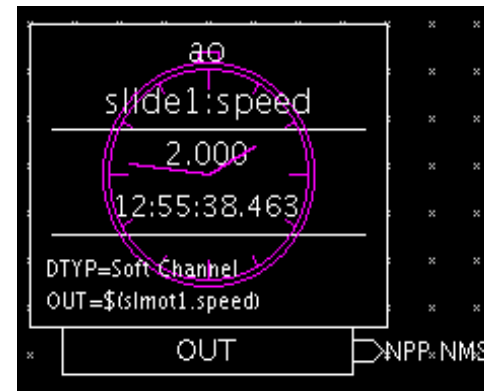
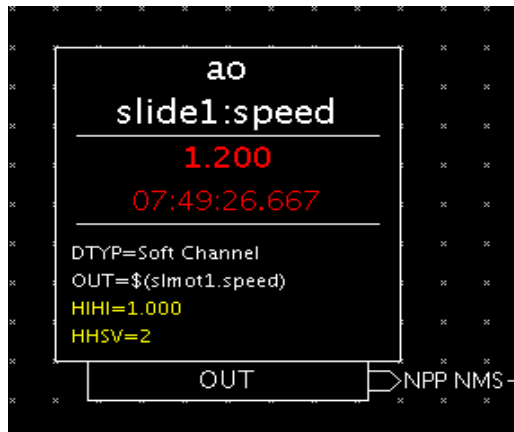
Below this window is the 'Inspector' panel for the object 'slide1:speed (ao)'. It shows a table of fields:

Field	Value	Variable
GUI_COMMON		GUI_COMMON
DESC		
ASG		
GUI_LINKS		GUI_LINKS
DTYP		Soft Channel
FLNK		
GUI_INPUTS		GUI_INPUTS
SIOL		
SIML		
SIMS		<none>
GUI_OUTPUT		GUI_OUTPUT
OUT		sm1:speed.VAL NPP...
VAL	0.200	

Below the table is a 'Comment' field and a 'Frozen' checkbox.

JCA Debug Plugin (2 / 2)

- Alarms are displayed with different colors
- Timeouts are indicated with a clock over the template instance symbol
- Change to another template instance (other substitution file line) on the fly



Hierarchies: Concepts

- **Substitute for having only flat database**
- **Allowing two-way communication among templates of different hierarchical order**
- **“Data Encapsulation”**
- **Similar to OOP**
- **New keywords will be introduced:**
 - expand
 - template
- **Current EPICS versions are unable to load these databases – database flattening**

Hierarchies: Expand

- **Used by higher level database file when instantiating a template**
- **Fills in values for macros**
- **Unexpanded macros can still be expanded at load time (substitution files)**

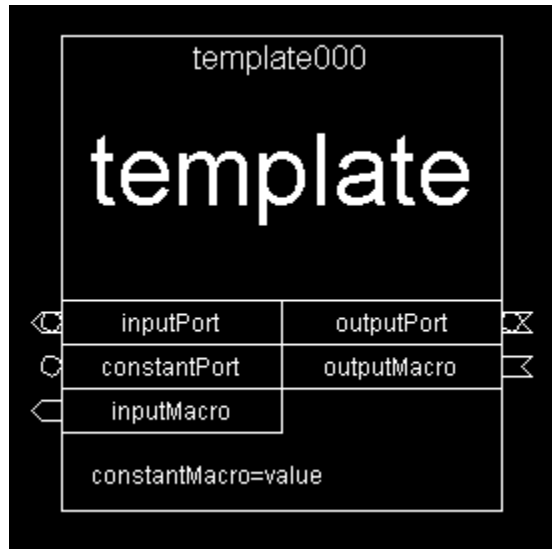
```
expand("slideMotor.vdb", slmot1) {
    macro(name, "sm1")
    macro(address, "4")
    macro(demand, "slide1:demand.VAL")
}
```

Hierarchies: Template

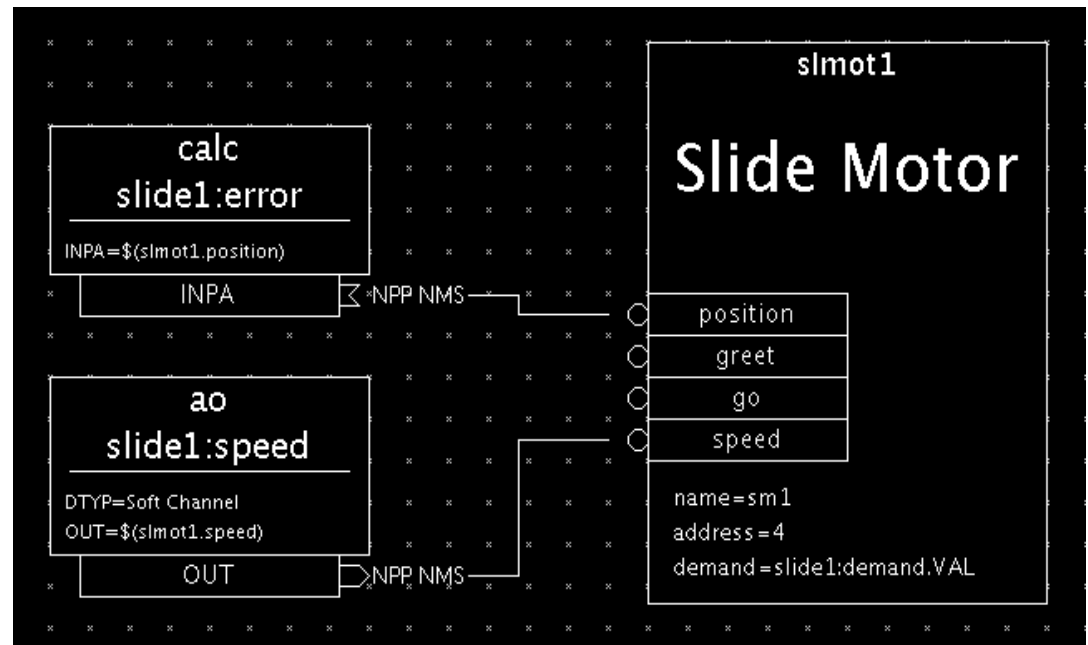
- Lower level files “export” their “public” information (usually names of PVs of public importance) - ports.
- Referred to as $\$(instance_name.port_name)$ by the higher level file.

```
# This is a template comment...
template("Slide Motor") {
    port(position, "$(motor.position)", "Current position of the slide")
# this is a ports section comment
    port(greet, "Hello, world!", "Just being friendly...")
    port(go, "$(name):startmoving", "Forward link to this to cause
movement")
    port(speed, "$(name):speed.VAL", "Record to set motor speed mm/sec")
}
```

Everything already works with Visual DCT!

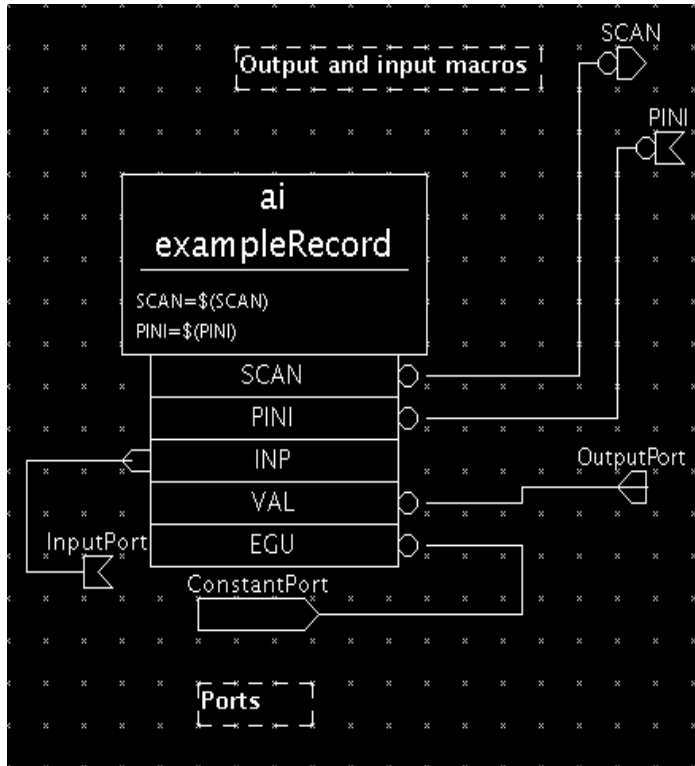


- One template block hides a database of arbitrary complexity!
- All that is seen are the ports and macros (input/output parameters)

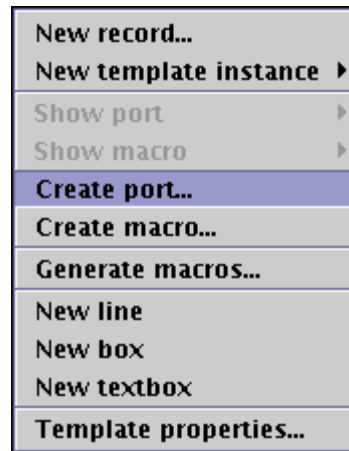


- Connect the template into your database!
- Enter the instantiated template by SHIFT+click

Lower level (template) file

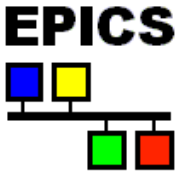


- All macros and ports can be created with a mouse click
- Input/Output/Constant attributes serve for better readability



Macros can be generated from all different occurrences of \$()

Database Flattening Tool



- **Integral part of Visual DCT**
- **Product is a flat EPICS db file.**
- **All flattening tools should use comments that would refer blocks of code to the original files.**
- **Two passes (port names may be used before they are defined in the database).**

- Cosylab VisualDCT Project Page
 - **Provides related articles and presentations, links to Builds, demo, bug reports, etc.**
 - **Download the latest builds, documentation, examples and plugins**
 - <http://visualdct.cosylab.com/builds/VisualDCT/>