

Design

Milkyway Menus & Forms

Manoz Palaparthi
10/17/03



> *Your Design Partner*

SYNOPSYS®

Milkyway Menus & Forms

The screenshot displays the Milkyway software interface. The main window has a menu bar with items: Tools, Library, Cell, Options, Views, Create, Modify, Select, Query, Tech File, Netlist In, Cell Library, Wire Tracks, CLF, ECO, Output, Hercules, and myflow. A context menu is open over the 'myflow' button, listing: Create Library ..., LEF in (highlighted with a black arrow), Open Library ..., BPV ..., and Close Library. A 'Hierarchy Walker Menu' dialog box is overlaid on the main window. It contains the following sections:

- Traverse Options:**
 - Traverse_In_Lib
 - Traverse_From_Hierarchy
 - Traverse_From_File
- Library Name** [text field]
- Cell Name** [text field]
- View Name** [text field]
- Output File Name** [text field]

Below these fields are sections for **Stop View Options** and **Applicable Functions**, each with several dropdown menus.

Agenda

1. Creating Menus and Forms

- i. What is Tk?
- ii. Some useful Tk functions

2. Scheme – Tk interface

- i. Creating menus using scheme
- ii. Calling Tk GUI from Milkyway GUI
- iii. Passing variables from Tk to scheme
- iv. Milkyway fonts

3. Built-in Milkyway menus

- i. Understanding Milkyway Menus
- ii. Creating new menus in Milkyway

Outline

1. **Creating Menus and Forms**
2. **Scheme – Tk interface**
3. **Built-in Milkyway menus**

What Is Tk?

- **What is Tk?**
 - **Is a toolkit for window programming**
 - **Provides a quick and fun way to generate user interfaces**
 - **An extension to Tcl**
- **Can be easily integrated with scheme**

Some useful Tk functions

- Tk provides some built-in applications apart from the standard widgets
- Browser application is one of such kind
- Commands to create a browser,
 - `tk_chooseDirectory` (to select a directory)
 - `tk_getOpenFile` (to select a file)

Example: myMenu.tcl (contd..)

```
#Proc for library name browser
```

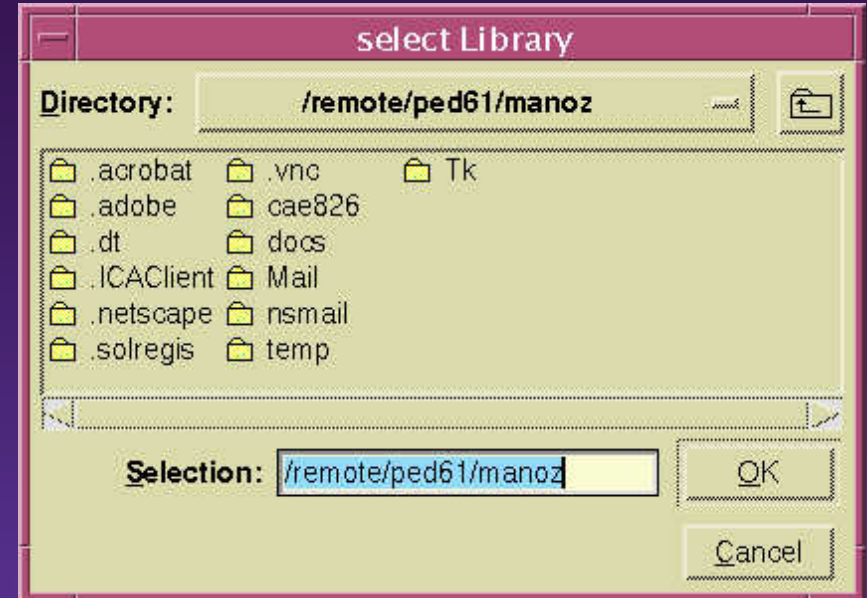
```
proc Browse {dir} {  
  global NewLibName  
  set NewLibName [tk_chooseDirectory -title \  
    "select Library" -initialdir $dir]  
}
```

```
#Proc for cellname browser
```

```
proc cellBrowse {cell} {  
  global CellName  
  set CellName [tk_getOpenFile -title "select cell" -initialdir $cell]  
}
```

```
#Create buttons for browser
```

```
button $w.frame2.alib.b1 -text "Browse..." -command {Browse .}  
button $w.frame2.cel.b1 -text "Browse..." -command {cellBrowse $NewLibName/CEL}
```



Outline

1. Creating Menus and Forms
2. Scheme – Tk interface
3. Built-in Milkyway menus

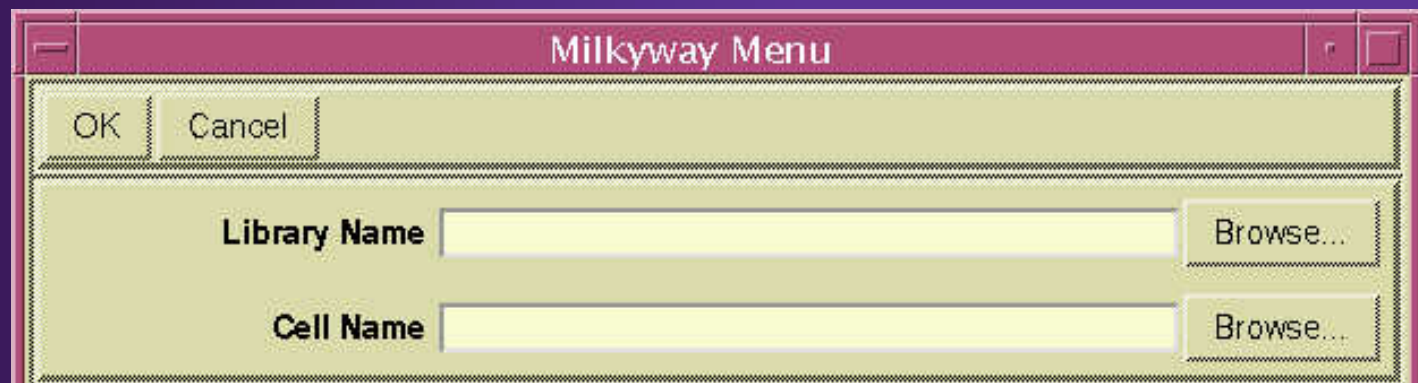
Scheme – Tk Interface

- **Calling Tk GUI from scheme**
- **Creating menus using scheme**
- **Calling Tk-GUI from Milkyway GUI**
- **Passing variables from Tk to scheme**
- **Milkyway fonts**

Calling Tk-GUI from scheme

- Tk-GUI can be called from Milkyway using the scheme command *loadUserForm*
- Example:

loadUserForm “myMenu.tcl”



Creating Menus Using Scheme

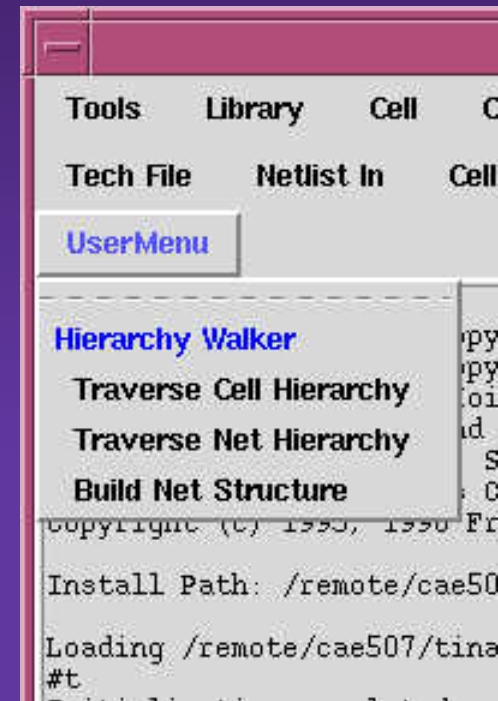
- Only drop down menus can be created using scheme
- Each menu item can be associated with a scheme function
- Command to add user menu is *addUserMenu*
- Command to remove a usermenu is *removeUserMenu*

Example –Menus Using Scheme

- Scheme code for creating menu

```
(addUserMenu "UserMenu" (  
  list "Hierarchy Walker"  
  (cons " Traverse Cell Hierarchy" "cellHier")  
  (cons " Traverse Net Hierarchy" "netHier")  
  (cons " Build Net Structure" "buildNetStruct")  
))
```

- Creates a Menu “UserMenu”

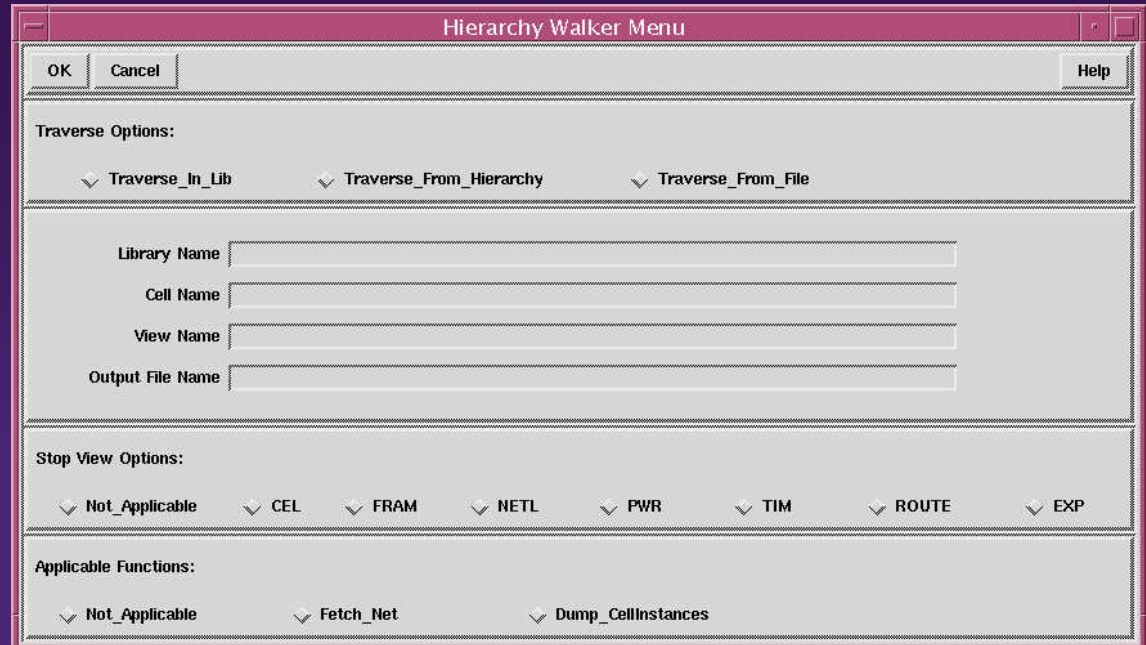


Calling Tk-GUI From Milkyway GUI

- We can integrate Tk-GUI and Scheme-GUI
- Each entry in a scheme menu should be associated with a scheme function
- If the scheme function is a procedure which loads the tcl file, then a click on that menu entry will bring up the Tk-GUI
- The proc for the command “cellhier” should be,

```
(define cellHier
  (lambda()
    (begin
      (loadUserForm “cellhier.tcl”)
    )
  ))
```

Example



```
))  
(addUserMenu "UserMenu" (  
  list "Hierarchy Walker"  
  (cons " Traverse Cell Hierarchy" "cellHier")  
  (cons " Traverse Net Hierarchy" "nethier")  
  (cons " Build Net Structure" "buildNetStruct")  
))
```



Passing Variables From Tk to Scheme

- Values of variables can be passed from Tk to scheme using the Tk command *userEvalScm*
- It takes one argument which can be a scheme command or function
- Syntax:
`userEvalScm "<scheme command>"`

Passing Variables From Tk to Scheme

- **Example**

```
userEvalScm "geOpenLib \"$libName\""
```

- **The Tk command above when executed opens a library whose name is defined by the Tk variable \$libName. (If the library exists!)**

Milkyway fonts

- Tk GUI default fonts & color are different from Milkyway fonts&color
- To get the Milkyway fonts,

```
set avnt_default_font *adobe*helvetica*bold-r-normal*12*
set avnt_menulabel_font *adobe*helvetica*bold-o-normal*12*
set avnt_entry_font *adobe*helvetica*medium-r-normal*12*
set avnt_text_font *adobe*courier*medium-r-normal*12*
set avnt_tag_font *adobe*courier*bold-r-normal*12*
option add *font $avnt_default_font
option add *background #d9d9d9
option add *foreground #000000
option add *activeBackground #ecec
option add *activeForeground #000000
option add *selectBackground #c3c3c3
option add *disableForeground #a3a3a3
option add *troughColor #c3c3c3
option add *highlightBackground #d9d9d9
```

Milkyway fonts

Before



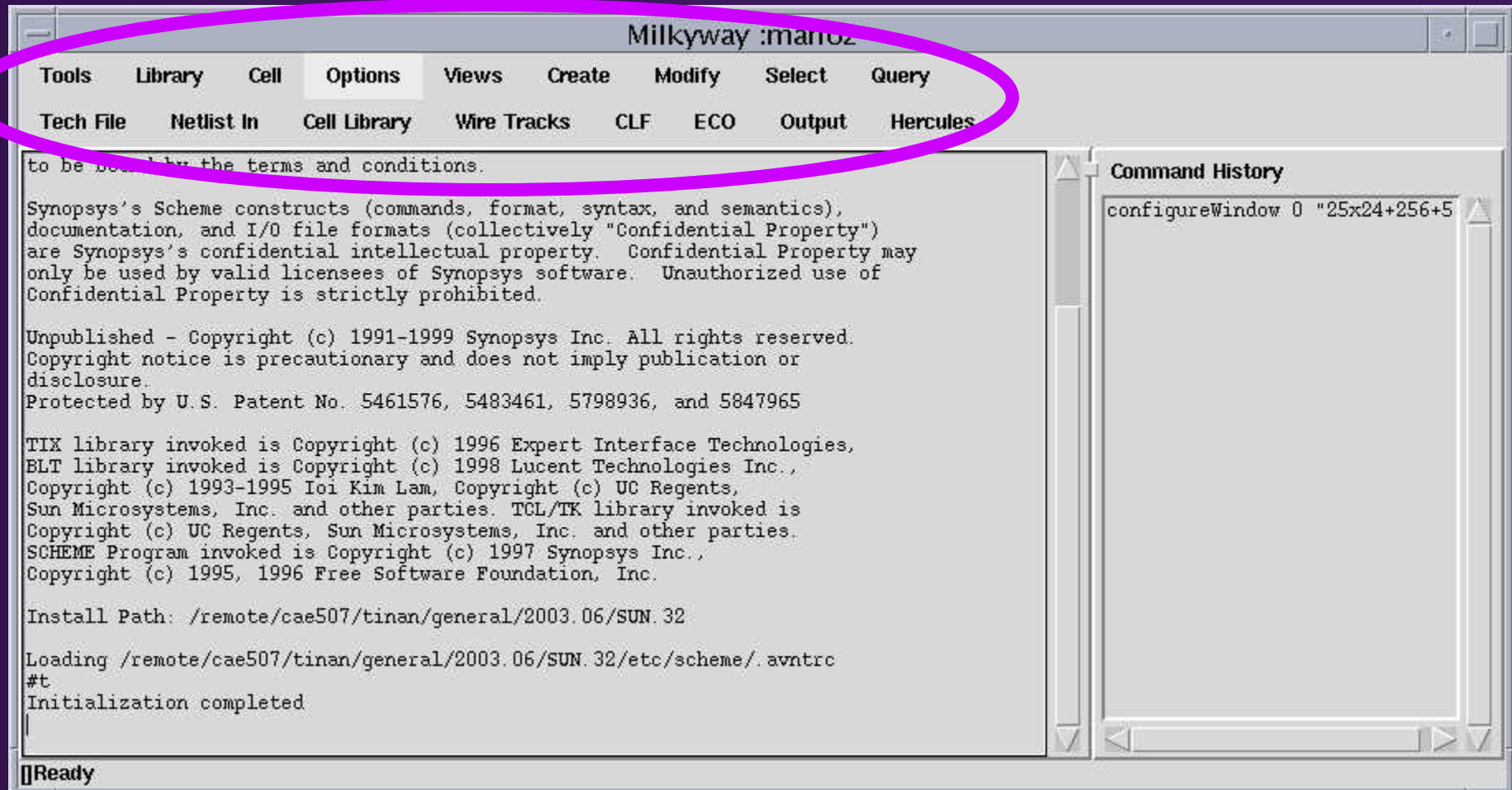
After



Outline

1. Introduction to Tk
2. Scheme – Tk interface
3. **Built-in Milkyway menus**

Built-in Milkyway Menus



Built-in Milkyway Menus

- Menus are installed in etc/menus directory
- All files have extension *.menu*

Milkyway.menu

libcell_mw.menu

- Each files defines a menu in Milkyway

Built-in Milkyway menus

- *Milkyway.menu* defines menu for the tool

```
MENU_Bar      0
MENU_Group    "Tools"    0
MENU_Button   "Quit"      menuQuit
MENU_GroupEnd

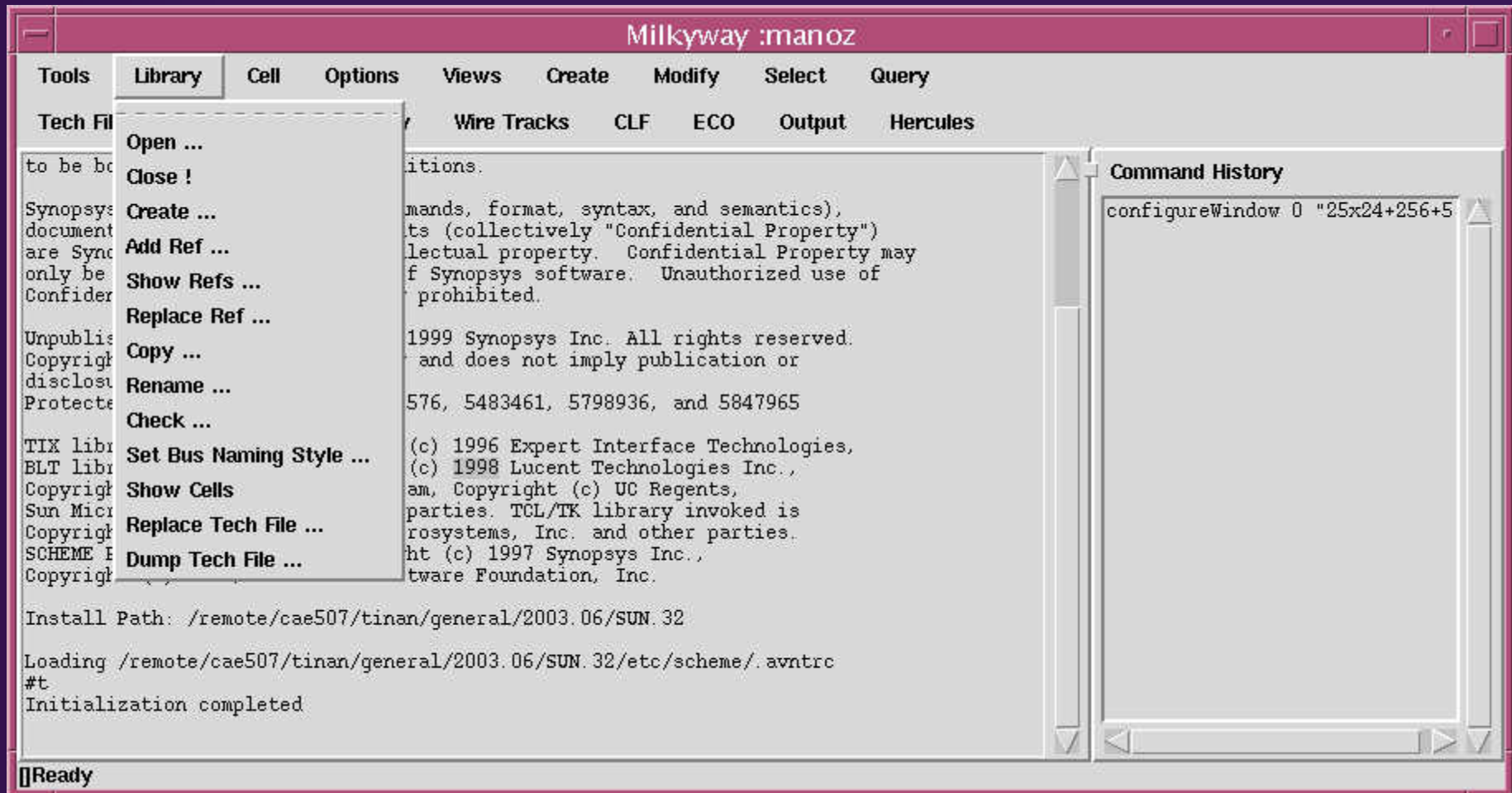
MENU_File     "libcell_mw"
MENU_File     "options_mw"
MENU_File     "views_mw"
MENU_File     "edit_mw"
MENU_File     "select_mw"
MENU_File     "query_mw"
MENU_Bar      1
MENU_File     "techfile_mw"
MENU_File     "dataprep_mw"
MENU_File     "output_mw"
MENU_File     "hercules"
```



Milkyway.menu

- **MENU_Group "Tools" 0**
Specifies the text label “Tools” for menu bar
- **MENU_Button "Quit" menuQuit**
Specifies text label “Quit” and scheme command “*menuQuit*” for menu button
- **MENU_GroupEnd**
Ends the definition of the Tools
- **MENU_File “libcell_mw”**
Calls the menu file *libcell_mw.menu*
- **MENU_Bar –**
Makes a new line

libcell_mw.menu



libcell_mw.menu

```

MENU_Group    "Library"    0
MENU_Button   "Open ..."  geOpenLib
MENU_Button   "Close !"    geConfirmCloseLib
MENU_Button   "Create ..." cmCreateLib
MENU_Button   "Add Ref ..." cmRefLib

```

```

MENU_GroupEnd

```

```

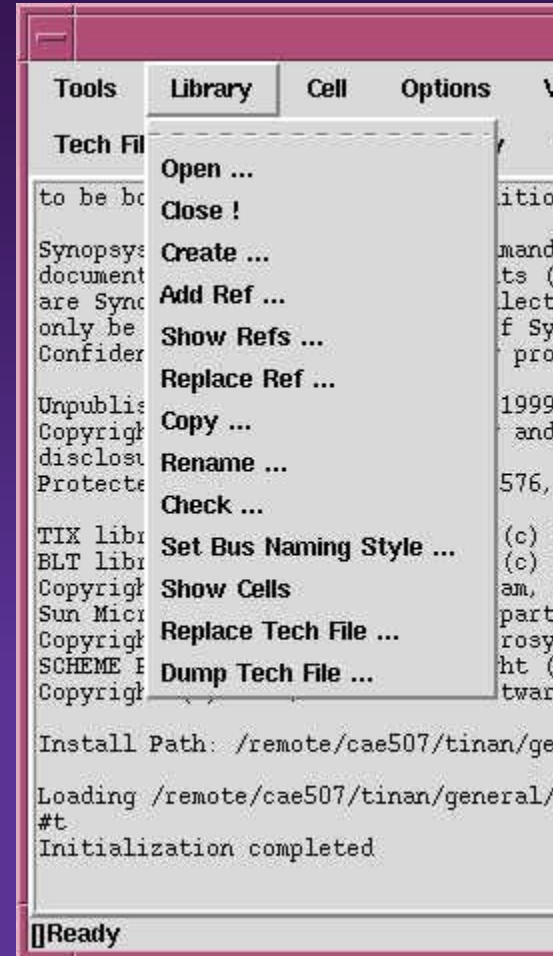
MENU_Group    "Cell" 0
MENU_Button   "Open ..."  geOpenCell
MENU_Button   "Close ..." geCloseWindow

```

```

MENU_GroupEnd

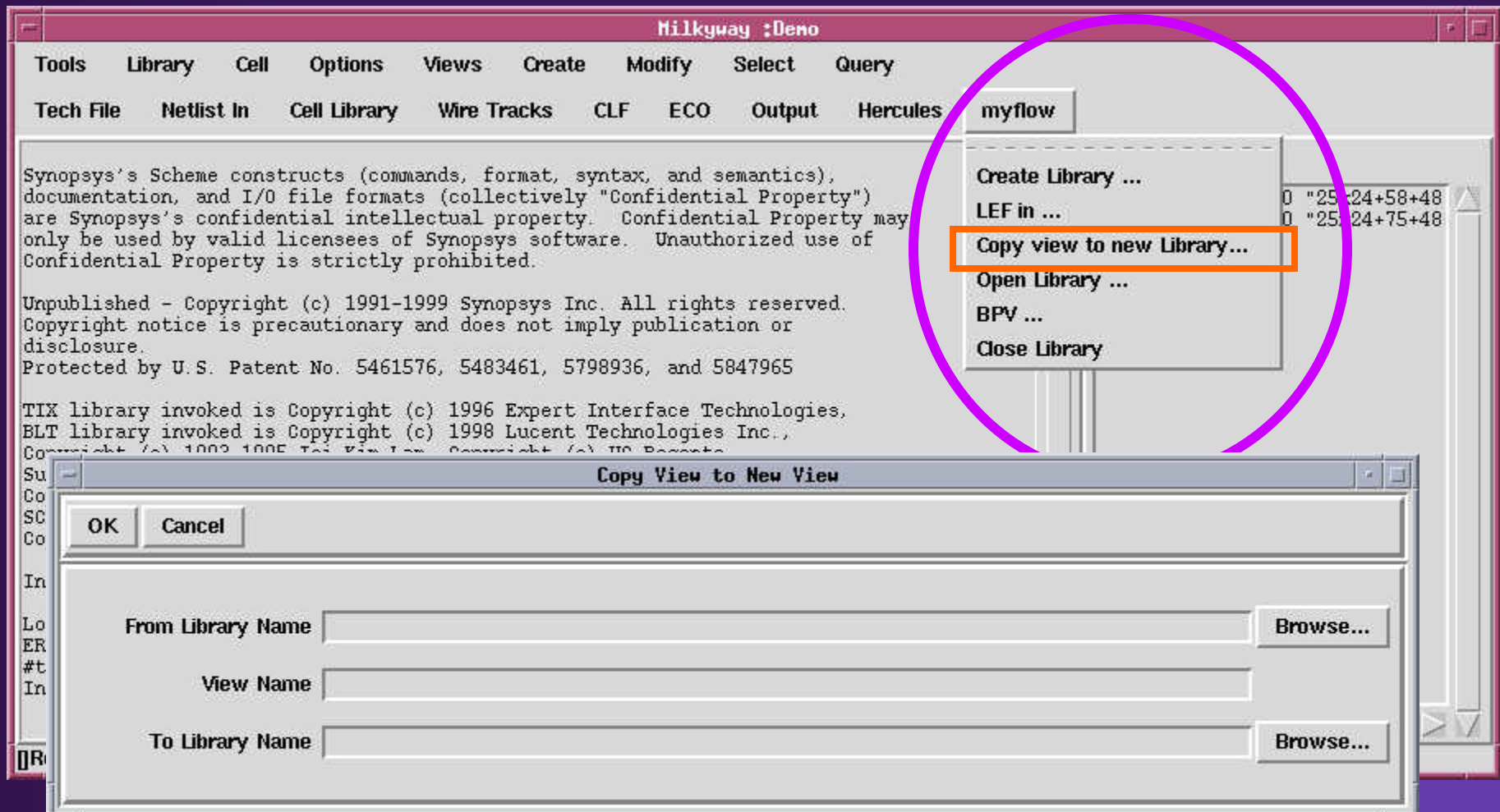
```



Customizing the Milkyway Menu bar

- Appearance of Milkyway GUI can be changed by modifying these files.
- By modifying these files, we can
 - Move default menus to other part of GUI
 - Add menus customized to a flow
 - Remove menus disabling some functions
 - Change the text labels to look more apt

Example: Adding a new menu



Example: Adding a new menu

- Creates a new menu that drives a whole flow
- New file *etc/menus/myFlow.menu*

```
MENU_Group    "myflow"          0
MENU_Button   "Create Library ..." cmCreateLib
MENU_Button   "LEF in ..."      auNLIApi
MENU_Button   "Copy view to new view .." copyView
MENU_Button   "Open Library ..." geOpenLib
MENU_Button   "BPV ..."      auExtractBlockagePinVia
MENU_Button   "Close Library"   geConfirmCloseLib
MENU_GroupEnd
```

- *Milkyway.menu* should be appended with
MENU_file "myflow"

Example: Adding a new menu

- Scheme/.avntrc should be appended with

```
(define copyView  
  (lambda ()  
    (begin  
      (load "copyview.tcl")  
    )))
```

Thank U!!