;; hi lite shapes attached to gnd and vdd accordingly

procedure( hg( )

cv=geGetWindowCellView()

vbias=dbFindNetByName( cv "vss")

vbiasFigs=vbias->figs

foreach(fig vbiasFigs geSelectFig( fig ))

)

procedure( hv( )

cv=geGetWindowCellView()

vbias=dbFindNetByName( cv "vdd")

vbiasFigs=vbias->figs

foreach(fig vbiasFigs geSelectFig( fig ))

)

;; assumes you set variable findNet in CIW to net you want to find such as findNet="GND"

findNet="vss"

procedure( hiNet()

cv=geGetWindowCellView()

vbias=dbFindNetByName( cv findNet )

vbiasFigs=vbias->figs

foreach(fig vbiasFigs geSelectFig( fig ))

)

load("enc.il")

procedure( sw()

hiResizeWindow(window(1) list(223:3 1650:180))

)

;; reset window in good location for layout and CIW

procedure( rw()

(setf did nil layWin nil layWin2 nil )

layWin = setof(win hiGetWindowList()

rexMatchp("Layout" hiGetAppType(win)))

layWin2 = setof(win hiGetWindowList()

rexMatchp("Virtuoso X" hiGetAppType(win)))

if( layWin != nil then

hiResizeWindow((car layWin) list(223:3 1750:180))

;; IC5 only leSetLSWBBox( list( 1165:160 1275:1075))

)

;; if user was using Vxl then get that window name to resize

if( layWin2 != nil then

hiResizeWindow((car layWin2) list(43:113 1750:900))

)

)

;; procedure to update via enclosures as desigred by changed values while hitting the create via button in the layout

procedure( enc()

viaForm = hiGetCurrentForm()

;; change the left and right values to make it drc clean or your choice

viaForm->viaSingleModeScrollView->parameters->page1->layer1EncLeft->value = .03

viaForm->viaSingleModeScrollView->parameters->page1->layer1EncRight->value= .03

viaForm->viaSingleModeScrollView->parameters->page1->layer2EncLeft->value = .03

viaForm->viaSingleModeScrollView->parameters->page1->layer2EncRight->value= .03

;; change the top and bottom of of both layers to 0 to make minimum for tighter bus

viaForm->viaSingleModeScrollView->parameters->page1->layer1EncTop->value= .0

viaForm->viaSingleModeScrollView->parameters->page1->layer2EncTop->value= .0

viaForm->viaSingleModeScrollView->parameters->page1->layer1EncBottom->value= .0

viaForm->viaSingleModeScrollView->parameters->page1->layer2EncBottom->value= .0

)

;; update the via cut for now must selet the via first before running this procedure this updates to cut value of 2

procedure( vc()

q=geGetSelectedSet()

dbReplaceProp( (car q ) , "cutColumns" 'int 2 )

;; dbReplaceProp( (car q ) , "layer1Enc" 'list list(0.07 0.07) )

)

;; procedure to update the enclosure of the via must select the via first only one at a time

procedure( ev()

q=geGetSelectedSet()

dbReplaceProp( (car q ) , "layer1Enc" 'list list(0.03 0.03) )

)

;; commond ciw commands

commands = "ddsOpenLibManager(), dbReplaceProp, geGetEditCellView(), geGetSelSet(), cdfGetInstCDF((car k)), dbReplaceProp(), dbFindAnyInstByName(), geScrool() "

;; scroll quicklly side to side

procedure( sf()

(for i 1 10 geScroll( nil "e" nil))

)

procedure( fs()

( for i 1 10 geScroll( nil "w" nil))

)

;; increment the path width first by getting the path creation form and updating the width value

procedure( pu()

pForm = hiGetCurrentForm()

;; increment the path width

currWidth = pForm->wireWidth->value

pForm->wireWidth->value = currWidth + .01

)

;; increment the path to a list of common widths or hightly used

psPlace = 0

procedure( pathCycle()

;; increment the place holder slot of the list the nth number that the current place is set to ex 0 is the first value in the list pList

;; check if at the end of the list then reset back to beginning or else it won't cycle the list

;; the list of path widths want to cycle through can update as desired

pList = list( .09 .11 .13 .16 .5 1 2 )

listLength = length( pList )

;; first item of the list is at 0 so should decrement the list length by 1

if( psPlace == (listLength - 1 ) then

psPlace = 0

else

psPlace = psPlace + 1

)

pList = list( .09 .11 .13 .16 .5 1 )

pForm = hiGetCurrentForm()

pForm->wireWidth->value = ( nth psPlace pList )

;; output the current value of the path width

print( sprintf( nil "%f" (nth psPlace pList) ) )

)

;; to help change values by change a via using as a slider save is the obj id of the via to use as a slider l is the obj you want to update

;; dbReplaceProp( ( car l ), "cutColumns" 'int (car (cdar (save->viaHeader->overrideParams) ) ) )

;; procedure to replace staced vias enclosres

;; get the list from the selected via then do below and repeat for layer1Enc then put to bindkey

;; foreach(mapcar x k dbReplaceProp( x "layer2Enc" 'list list(0.03 0 0 0.03) ))

;; menu for vias fixs won't update via5\_4 since min enc is .02um for M5

procedure( ens()

sel = geGetSelectedSet()

;; for each via in the list change the enclosure property with the new values

foreach(mapcar x sel if( x->viaHeader->viaDefName != "M6\_M5" then

dbReplaceProp( x "layer1Enc" 'list list(0.03 0 0 0.03) )

dbReplaceProp( x "layer2Enc" 'list list(0.03 0 0 0.03) )

else

)

)

)

procedure( ee()

sel = geGetSelectedSet()

;; for each via in the list change the enclosure property with the new values

foreach(mapcar x sel if( x->viaHeader->viaDefName != "M6\_M5" then

dbReplaceProp( x "layer1Enc" 'list list(0.0 .03 .03 0) )

dbReplaceProp( x "layer2Enc" 'list list(0.0 .03 .03 0) )

else

)

)

)

;; fixed the via auto according no matter orientation of col and rows to fix enclosure for drc. add for loop later

procedure( fV()

(setf rowCount 0 colCount 0 viaCount 0 )

;; get the via or group of vias only works for one now though.

l =geGetSelectedSet()

vParams = (car l)->viaHeader->overrideParams

colCount=(cadr ( cadr k))

rowCount=(cadr ( car k ))

if( rowCount == 1 && colCount == 2 then ee()

else if( rowCount == 2 && colCount == 1 then ens() )

)

)

procedure( winPop()

hiDisplayMenu(

;;; creating a dynamic popup menu

hiCreateSimpleMenu(

'trWindowsPopUp

"Windows"

foreach( mapcar wid hiGetWindowList()

hiGetWindowName( wid )

) ; foreach

foreach( mapcar wid hiGetWindowList()

sprintf(

nil

"hiRaiseWindow( window( %d ))"

wid->windowNum

)

) ; foreach

) ; hiCreateSimpleMenu

) ; hiDisplayMenu

) ; procedure

;;; Setting the F6 function key to display the popup menu

hiSetBindKey( "Command Interpreter" "<Key>F6"

"winPop()" )

procedure( ring()

( setf v a 0)

)

procedure(XmenuAddMenu(\_args)

let( (item1 item2)

;; create a couple of menu items

item1 = hiCreateMenuItem( ?name 'item1 ?itemText "ViaEnclosureFix"

?callback "ev()"

)

item2 = hiCreateMenuItem( ?name 'item2 ?itemText "viacut2"

?callback "vc()"

)

item3 = hiCreateMenuItem( ?name 'item3 ?itemText "cyclePathWidth"

?callback "pathCycle()"

)

item4 = hiCreateMenuItem( ?name 'item4 ?itemText "HilightGND"

?callback "hg()"

)

item5 = hiCreateMenuItem( ?name 'item5 ?itemText "HiLightVDD"

?callback "hv()"

)

item6 = hiCreateMenuItem( ?name 'item6 ?itemText "ResetWindows"

?callback "rw()"

)

item7 = hiCreateMenuItem( ?name 'item7 ?itemText "LibManager"

?callback "ddsOpenLibManager()"

)

;; create a menu that includes the menu items and return a list of the

;; pulldown menus

list(hiCreatePulldownMenu('IndieHelperTools "XTC LayoutTools" list(item1 item2 item3 item4 item5 item6 item7 )))

);let

); procedure

;; create a userMenuTrigger trigger that automatically adds the menu

deRegUserTriggers("maskLayout" nil 'XmenuAddMenu)

deRegUserTriggers("maskLayoutXL" nil 'XmenuAddMenu)

procedure( swapInst( inst1 inst2)

( let ( dv id1 id2 temp)

cv=geGetWindowCellView()

id1=dbFindAnyInstByName( cv inst1)

id2=dbFindAnyInstByName( cv inst2)

if( id1 && id2 then

temp=id1->xy

id1->xy =id2->xy

id2->xy = temp

printf( "%s and %s have been swapped \n" inst1 inst2)

else

printf("Inst not found\n")

) ;if

);let

)

procedure( swapInstWLabel( inst1 label)

( let ( dv id1 labId temp)

cv=geGetWindowCellView()

id1=dbFindAnyInstByName( cv inst1)

labId = lable

if( id1 && labId then

temp=id1->xy

id1->xy =labId->xy

labId->xy = temp

printf( "%s and %s have been swapped \n" inst1 labId )

else

printf("Inst not found\n")

) ;if

);let

)

;; bindkeys

hiSetBindKey("Layout" "<Key>F12" "rw()")

hiSetBindKey("Layout" "CtrlShift<Key>V" "enc()")

hiSetBindKey("Layout" "CtrlShift<Key>V" "vc()")

hiSetBindKey("Layout" "CtrlShift<Key>F" "pathCycle()" )

hiSetBindKey("Layout" "a" "ens()" )