Common Innovus Commands

createRow
select_row
createSoftGuide
createGuide
createRegion
createFence
setInstancePlacementStatus

addHaloToBlock
deleteHaloFromBlock

addRoutingHalo
deleteRoutingHalo

createInstGroup
addInstToInstGroup

createExclusiveGroups
finishFloorplan

setTiewHiLoMode
addTieHiLo

addWellTap

addEndCap

setFillerMode
addFiller

addDecap

unplaceAllBlocks
writeFPlanScript

```
setAddRingMode -ignore_DRC true      -> addRing
setAddStripeMode -ignore_DRC true    -> addStripe
setViaGenMode -ignore_DRC true       -> editPowerVia
```

```
setSrouteMode   -> sroute
```

spare cells are in the netlist

    specifySpareGate
    setPlaceMode -place_global_module_aware_spare {true | false}
    reset_spare_insts

spare cells are NOT in the netlist

    createSpareModule
    placeSpareModule
    deleteSpareModule

specifyCellPad
specifyInstPad

reportCellPad
deleteCellPad

specifyJtag
traceJtag

setPlaceMode -place_detail_preroute_as_obs { }

place_connected

create_inst_space_group
delete_inst_space_group
checkPlace

setPlaceMode, setOptMode -> place_opt_design -expanded_views

deleteAllDensityAreas

concurrently placing macros and standard cells
set_macro_place_constraint
setPlaceMode -place_opt_run_global_place
place_design -concurrent_macros [-incremental]

specifySelectiveBlkGate
unspecifySelectiveBlkgGate

createPlaceBlockage

reportDensityMap

Report core utilization and density

```
    checkFPlan -reportUtil
```

report pin density for the entire core area or for a specified area

```
    queryPinDensity -area <x1 y1 x2 y2>
```

```
refinePlace
checkPlace
```

```
setOptMode -usefulSkewPreCTS {false | true}
```

```
scanReorder
deleteScanChain
```

Scan cells are identified in the timing library(.lib) If scan cells are not in the timing library, use this command to load information:
specifyScanCell

if you do not have a scanDEF
specifyScanChain [scanChainName]

A Design Exchange Format(DEF) file contains the design-specific information of a circuit and is representation of the design at any point during the layout process

A scanDEF file is a subsection of a DEF file. It is a pin-based format that describes the set of reorderable scan chains present in the scan-mapped and configured netlist

To generate a scanDEF file for your design, use the write_scandef command in Genus Synthesis Solution.

```
defOutBySection -noNets -noComps -scanChanins scan.def
```

setScanReorderMode ->   place_opt_design

Alternatively, you can run the scanReorder to reorder the scan chain

```
scanTrace [-lockup | -noLockup]
```

Reordering Scan Chains with Lockup Latches
Scans need to be linked across clock domatins with lockup latches

Display scan chains before and after reordering the scan:

```
    displayScanChain scanChainName
```

reporting scan chains

report_scan_chain -chain <chain_name>

deleteScanChain scanChainName

sroute
    power router
    connect the cells, blocks and pad power, and ground pins to global power and groud
earlyGlobalRoute
    early global router
    fast signal router needed for extraction and timing analysis
    useful for congestion analysis
    is not DRC or LVS clean
routeDesign
    NanoRoute Router
    Accurate timing and SI-aware router for tapeout
    Run before runing SI analysis, signoff timing analysis
fcroute
    route power and signal to bumps in a flip chip design
routeDesign -highFrequency
    router for mixed signal designs integrated into innovus and require additonal license

earlyGlobalRoute

Running the early global router
set the net attributes and the modes before running the early global router
1) set design mode for routing: setDesignMode
2) set the route attribute: setAttribute
3) set the early global route mode: setRouteMode
4) run early global route: earlyGlobalRoute

early global router
create actual wires, produce congestion map, DRC or LVS not clean, correlate to the detail routter(NanoRoute), not recommended to run SI analysis on an Early Global Router design

routing resource (honored by both NanoRoute and Early Global Route)
setDesignMode -topRoutingLayer <layerName> -bottomRoutingLayer <layerName>

setting the route attribue for specific nets (honored by NanoRoute and Early Global Route)
setAttribute
    -non_default
        select the non-default rule for the net
    -shield_net
        preserve the routing resource for shielding net
    -skip_routing

specifies that the specified net should not be routed

-preferred_extra_space

specifies the preferred extra spacing for net

-top_preferred_routing_layer

specifies the preferred highest layer for routing specified signal nets

-bottom_preferred_routing_layer

specifies the preferred lowest layer for routing specified signal nets

setRouteMode

Analyzing congestions and actions to consider

1) adjust block placement and/or orientation to make sure that connecting pins face each other
2) use a partial placement blockage to lower the congetsion in a specific area
3) early global route honors partial routing blockage

There are two types of congestion display: line style or diamond style

Interpret the routability stats by analyzing the overflow numbers
The purpose of earlyGlobalRoute is to quickly and accurately tell us how routable a design is.
When interpreting earlyGlobalRoute feedback, look at the following:

The final H and V overflow number

visual congestion map

EarlyGlobalRoute outputs two statistics:

one using its own engine to calculate the overflow number

overflow(H) = number of overflow Gcells for all layers(H) / sum of available gcells for all layers(H)

and another that uses Global Route’s calculation engine to calculate the numbers

overflow(H) = number of overflow for all layer(H) / sum of available gcells for all layers(H)

In above two cases, fully blocked Gcells do not appear in the sum of available Gcells value

reportCongestion

-overflow

Reports horizontal and vertical overflow.

-hotSpot

Reports the local hotspot score. It reports the maximum and total hotspot area.

-3d

Honors 3d congestion map. When specified, it enables the layer-based congestion reporting.

Hotspots are a better indicator of routability than overflow

If the max hotspot is greater than 100, it indicates a potential routing problem

If the total is too high, the routing will have detours

If the maximum area is small but the total area is high, it could indicate scattered hotspots, which could be fine for routing

create_library_set
The same library set can be referenced multiple times by different delay calculation corners

create_rc_corner
An RC corner object provides the software with all of information necessary to extract and use the RCs for delay calculation

To find out what the RC corners have been set to
    get_rc_corner

ceate_delay_corner
A delay calculation corner provides all of information necessary to control delay calculation for specific analysis view

To find out what the delays corner have been set to
    get_delay_corner

Creates a set of virtual operating conditions in the specified library without actually modifying the library
create_op_cond

get_op_cond

create_constraint_mode
The create_constraint_mode command is used to associate a Tcl list of .sdc files with a named mode

get_constraint_mode

create_analysis_view
The create_analysis_view command builds a top-level association of a delay calculation corner with a constraint mode

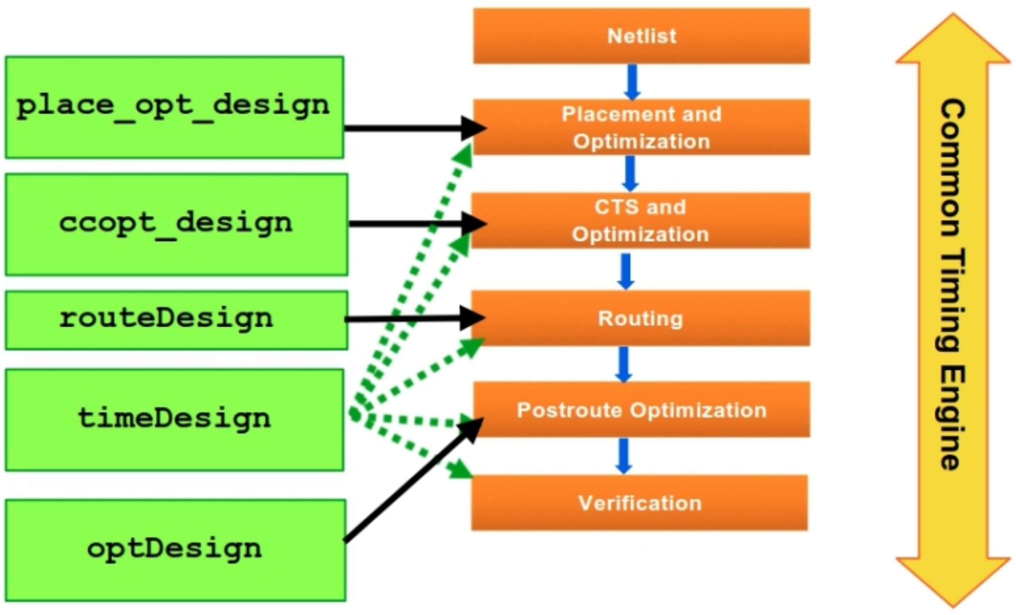
set_analysis_view
After creating analysis views, you must set which views the software should use for setup and hold analysis and optimization. These views are called "active" views
Libraries and data are loaded into the system as specified by the active views

timeDesign
The timeDesign command reports the results form each active analysis view, as well as an aggregated summary
By default, timeDesign creates only an aggregated summary.
Use the -expandedViews option reports timing of all views.
For Multi-Mode Muti-Corner analysis, the software creates a separate directory for each view.
For example, if your design has two analysis views, view1 and view2, the output reports are generated in the ./**timingReports**/view1
and ./timingReports/view2 directories

Extract the RCs in the design and then runing timing analysis for setup and hold

extractRc

timeDesign or report_timing



stage	command	analysis
Netlist		
placement & optimization	place_opt_design	timeDesign
CTS & optimization	ccopt_design	
Routing	routeDesign	
PostRoute optimization	optDesign	
Verification		

Extracting PreRoute RCs for 32nm and Below

For extracting preRoutes (Early Global Routes) at 32nm and below, a QRC extraction tech file is used

setDesignMode -process 32

setExtractRCMode -engine preRoute

extractRC

The extractRC command is executed automatically by the optDesign and timeDesign commands

Extracting PostRoute RCs for 32nm and Below

For extracting postRoute (NanoRoute detail router) at 32nm and below, a QRC extraction tech file is used

setDesignMode -process 32

setExtractRCMode -engine postRoute

extractRC

The extractRC command is executed automatically by the optDesign and timeDesign commands

Extraction Engines and Level of Accuracy

extraction method	accuracy	speed

preroute default(captable or QRC tech file)	low	fast
postroute detailed (2.5d extraction, captable or QRC tech file)		
Turbo Quantus QRC (QRC tech file)		
Integrated Quantus QRC (QRC tech file)		
Standalone/signoff Quantus QRC (QRC tech file)	high	slow

Correlating Extraction: Generating Scale Factors
Ostrich is a standalone utility to correlate the native extraction and sighoff extraction engines and generate the scaling factors. You can then set the scaling factors for the next extraction cycle

    generateRCFactor

Scaled OCV is a way to account for on-chip variation for delays by derating the early and late paths

    set_timing_derate

To take common clock path pessimism removal (CPPR) into account, specify:

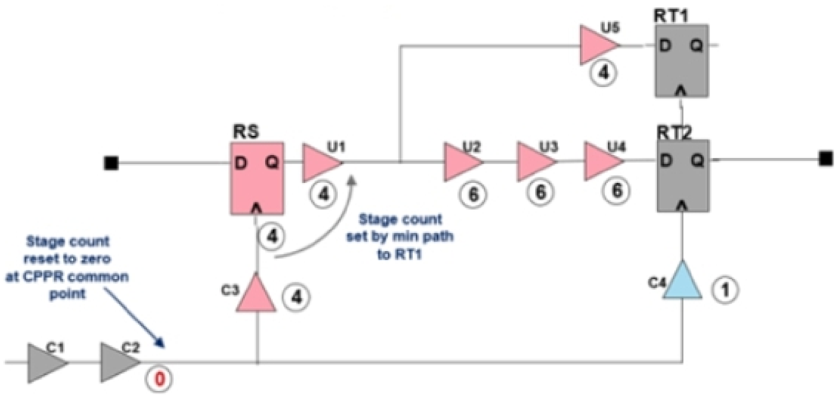
    setAnalysisMode -cppr both

AOCV accounts for on-chip delay variation by basing the cell’ s OCV delay multiplier on the depth of the paths
AOCV is primarily a means of reducting process pessimism when compared to derating

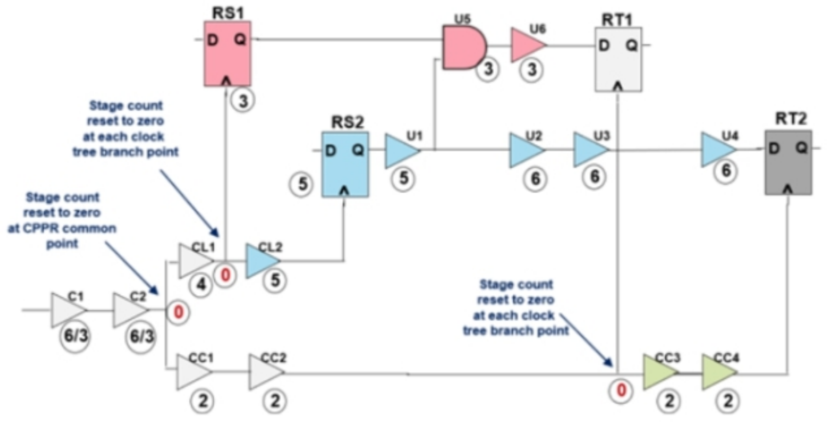
Getting the correct stage count for path is important in choosing the derating factor
Additional OCV factors can be embedded in the AOCV library
If the depth is high, the rationale is that the variations cancel out

Stage COunting: Graph-Based Analysis (GBA)

- Stage-counting always begins with a count of 0 at the CPPR common point.
- Default couting model is launch_capture
  - launch = clock + data count
  - capture = capture clock count


RT1
U5
D Q
4
RS
D Q
U1
4
U2
6
U3
6
U4
6
RT2
D Q
4
Stage count set by min path to RT1
Stage count reset to zero at CPPR common point
C3
4
C4
1
C1
C2
0

In graph-base mode, the algorithm requires resetting the stage count at clock branch points, which can lead to significant pessimism


RS1
D Q
U5
U6
RT1
D Q
Stage count reset to zero at each clock tree branch point
RS2
D Q
U1
U2
U3
RT2
D Q
U4
Stage count reset to zero at CPPR common point
CL1
CL2
C1
C2
6/3
6/3
CC1
CC2
Stage count reset to zero at each clock tree branch point
CC3
CC4

AOCV libraries are specified in the same way as Liberty timing libraries

setAnalysisMode -aocv true -analysisType onChipVariation -cppr both

create_library_set -aocv

update_library_set -aocv

Graph-Based Timing vs. Path-Based Timing

The difference between GBA and PBA is in how the slew are applied for PBA vs. GBA

GBA

For setup time, delay calculation applies the slowest slew value to all inputs

For hold time, delay calculation applies the fastest slew value to all inputs

Results in some pessimism

PBA

The actual slews are used for delay calculation

Generating an SDF file

select ideal clock if preCTS

write_sdf

timeDesign

run [Early Global Route], extraction and timing analysis

To run timing analysis only and to use the existing routing and extraction database, use

timeDesign -reportOnly

Report Delay Calculation for a Cell or Timing Arc

reportDelayCalculation

Report setup or hold timing or DRV violation

report_constraint

Generating a Timing Report

report_timing

Formatting Timing Reports

set_global report_timing_format {instance arc cell delay ...}

Setting path Groups for Timing and Optimization

- To explicitly create standard path groups
  - reg2reg reg2cgate
  - createBasicPathGroups
- To delete all created path groups
  - createBasicPathGroups -resest
- To explicitly create
  - reg2reg in2gre reg2out in2out
  - createBasicPathGroups -expanded
- To display the created path groups
  - get_path_groups

Creating and Reporting Path Groups

- setpathGroupOptions
- group_path
- report_path_groups

Run the setPathGroupOptions to guide timing optimization and timing analysis when path groups are used
The path groups are considered to be global in nature and apply to all the view, and are not supported on MMMC mode-specific basis. The path groups are persistent through a saveDesign to restoreDesign sequence.
To delete all path groups

- reset_path_group -all

To delete one path group

- reset_path_group -name <group_name>

check_timing
Performs a variety of consistency and completeness checks on the timing constraints specified for a design

dbIsInstDontTouch
dbIsCellDontTouch

What Is Optimization?
optimization is the process of iterating through a design such that it meets timing, area and power specification.
In general, optimization can be broken down into the following area

- timing
- signal integrity
- power
- area

Depending on the stage of the design, optimization can include the following operation:

adding buffer
resizing gates
restructuring netlist
remapping logic
swapping pins
deleting buffers
moving instances
applying usefull skew
layer optimization
track optimization

setOptMode
optDesign

place_opt_design
    run **setup optimization** along with placement

ccopt_design
    includes setup optimization

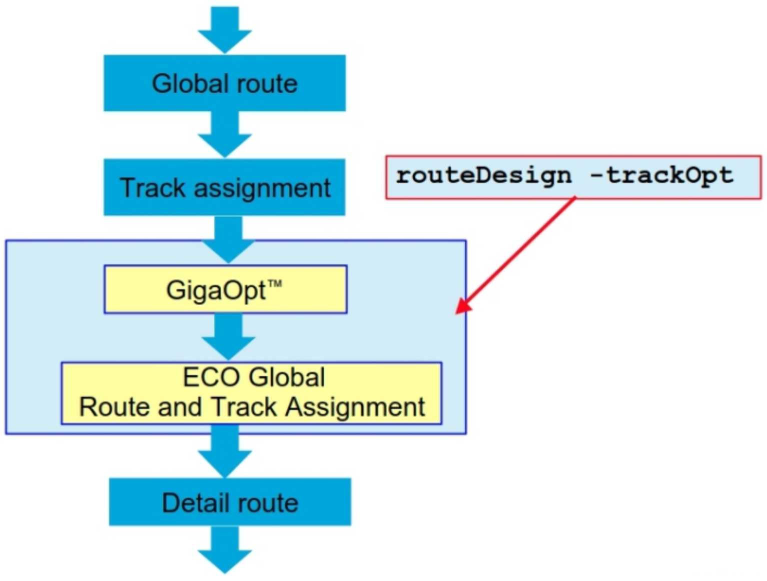
ccopt_design -cts
    just create a zero skew or skew balanced clock tree
If timing fails after building a zero skew clock tree, you will need to run
    optDesign -postCTS
    to optimize for setup time

To run hold optimization after CTS
    optDesign -postCTS -hold

run postRoute optimization for both setup and hold after ran routing with routeDesign
    optDesign -postroute -setup -hold

optimization with path groups
    If no path_groups are specified, the tool creates two high-effort path_groups:
        reg2reg reg2clkgate

Track optimization during routing
    routeDesign -trackopt



Track optimization steps

1. configure modes for extraction, SI analysis and delay calculation

```
setExtractMode -engine postRoute -effort medium
setDelayCalMode -SIAware true
setAnalysisMode -analysisType onChipVariation -cppr both
```

2. Route with track optimization using the tQuantus engine for extraction

```
routeDesign -trackOpt
```

Setting the optimization mode to use non-default routes

```
setOptMode -ndrAwareOpt <ndr_list>
```

optimizing all negative slack paths

```
setOptMode -allEndPoints true
```

Run total negaive slack optimization in addition to the worst negative slack optimization
This feature is automatically activated whild runing the high-effore timing-closure flow

reporting hold unfixed paths

```
optDesign -hold -holdVioData <filename>
```

additional reporting has been added to the optDeisng command to make it easier to see why the hold violation could not be fixed

Design Mode setting for power-dirven placement and optimizaiton

```
setDeisgnMode -powerEffort high|low|none
```

What are the components of power?

power = leakage power + dynamic power

Leakage to Dynamic ratio for power optimization

```
setOptMode -leakageToDynamicRatio [0-1]
```

balanced leakage / dynamic optimization: 0.5

Standalone Power optimization

```
    optPower -effortLevel
```

When combined with the setOptMode it optimizes leakage and dynamic power

```
    setOptMode -leakageToDynamic
    optPower -preCts
```

power view definition

```
    set_analysis_view -setup <VIEW> -hold <VIEW> -leakage <VIEW> -dynamic <VIEW>
```

Providing an activity file for power optimization

```
    read_activity_file <filename> -format (VCD | TCF | SAIF | FSDB)...
```

setting the switching activity for power optimization

```
    set_default_switching_activity
```

Set the default acivity to avoid overly pessimistic activity for unannotated nets or all nets in vectorless case
Best applied to inputs, sequential outpusts, and clock gates

```
    set_default_switching_activity
        -input_activity
        -seq_activity
        -clock_gates_output
```

reporting power

```
    report_power -outfile <FILE>
```

This command is used internally during optimization and as a standalone command for reporting power
read an activity file during each iteration

reporting instance power

```
    report_instance_power <instance_name>
```

Electromigration is a phenomenon where high AC currents flowing through resistive interconnect causes extreme temperatures
The current limits are specified in the LEF file with the ACCURRENTDENSITY parameter and in the Quantus QRC technology file

Fixing Electromigration(EM) Violations

```
    start(postRoute db)
    verify_drc(optional)
    verifyACLimit
    fixACLimitViolation
    verify_drc(optional)
    verifyACLimit
```

The verifyACLimit detects and saves the violation data in a report
By default the fixACLimitViolation widens wires with awareness of DRCs

Setting the Switching Activity Before verifying the AC limit
setting the switching activity will generate more accurate results
a) For example, set the switching activity with a TCF file
```
read_activity_file -format TCF -set_net_freq true test.tcf
```
b) To generate a TCF file if you do not have a TCF file
```
set_default_switching_activity -reset
set_default_switching_activity -input_activity 0.2 -seq_activity 0.2
propagate_activity
write_tcf test.tcf
read_activity_file -format TCF -set_net_freq true test.tcf
```
You can read activity files in one of several formats VCD TCF SAF FSDB, etc

Reporting and Fixing Electromigration violations
First generate an EM analysis report and next, use the generated file to fix the violations
1. Run the verifyACLimit to generate an EM analysis report. set the corrrect switching activity for analysis:
```
verifyACLimit -use_db_freq -report em1.report
```
2. Next, fix the violation by runing the command:
```
fixACLimitViolation -useReportFile em1.report
```
   Either an Innovus or a Voltus report can be used for fixACLimitViolation
   By default, only wire widening sis attempted for datapath nets.
   specify the number of iterations by specifying fixACLimitViolation -maxiter {1-3}
      Iteration 1 (default) : wire widening
      Iteration 2: wire widening, downsizing
      Iteration 3: wire widening, downsizing, bufffering

Specifying and Using NDR for EM Fixing
   Net-based NDR are specified with the setAttribute
   fixACLimitViolation finds the best NDR ofr the net and applies it based on the violation
   setAttribute -em_ndr_dist applies NDRs up to distance
```
add_ndr -name auto_rule_em_test
setAttribute -net n_3023 -em_ndr_rule auto_rule_em_test
setAtrribute -net n_3023 -em_ndr_dist 14.7695
...
ecoRoute
extractRC
```

```
setDesignMode -powerEffort
setOptMode -usefulSkewPreCTS
setOptMode -usefulSkewCCOpt
setOptMode -usefulSkewPostRoute
```

What is clock tree synthesis?

clock tree synthesis is the process of inserting buffers in the clock path, with the goal of minimizing clock skew and latency to optimze timing

clocks paths are one of the following:

1) ideal
   equal delay from clock source to sinks
2) traditional skew balanced
   delay from clock source to sinks equal within some tolerance
3) paths with useful skew
   a) clock path delays adjusted to assist timing slack optimization
   b) strict balancing not required and the skew can be exploited to save clock area and power

Traditional skew balanced methodology

suboptimal clock and datapath implementation

1) unneceesary
   no fundamental timing requirement that clock need to be balanced
2) expensive
   clock buffer explosion to minimize skew
   other expensive options (for example, mesh, spine, and so forth)
3) severe IR drop
   all flops/RAMs forced to trigger at the same time
4) critical path
   maty iterations, excessive run time, area explosion, higher leakage

paths with useful skew withccopt technology

superior clock and datapath implementation

1) MM/MC/OCV
   useful-skew takes into all timing aspects including MM, MC, OCV, setup, hold
2) efficient
   significant reduction in clock buffers(no explicit requirement to balance tree)
3) lower IR drop
   flops/RAMs triggered at different times
   critical and non-critical sinks are skewed
4) time borrowing
   faster timing closure
   higher performantce, low area, lower leakage

What is time borrowing in the ccopt?

time borrowing is the process of moving slacks by optimizing noncritical paths to the critical chain to create spare slack clock tree vs. skew group

1) a clock tree is subset of circuit on which cts can operate and contains physical constraints

    determine physical characteristics of the clock

2) a skew group contains balancing constraints, for example, skew and insertion delay

    determine balancing or initial balancing

create_ccopt_clock_tree
create_ccopt_skew_group

Each skew group is superimposed on the clock tree graph, a skew group is created per SDC clock per constraint mode - 1-1 mapping

skew group definitions
skew group defines groups of sinks to balance
skew group name: clockName/mode

Automatically Generating a CTS spec file

Before running CTS, automatically generate a clock specification file from the SDC files

create_ccopt_clock_tree_spec

a) analyze muti-mode timing graph
b) create clock trees and skew groups, and configure them

ouptut saved in the database or to a file(-file <filename>) which contains:

clock trees, skew groups and property setttings

Automatically generating clock trees and skew groups

The generated clock spec file will contain clock trees and skew groups

1. read in post-CTS SDC but without set_propagated_clock command for IO latency updates
2. Interactively set the mode to ideal to avoid modifying the sdc file by running:

```
set_interactive_constraint_modes [all_constraints_modes -active]
    reset_propagated_clock [get_clocks *]
set_interactive_constraint_modes   ""
```

sink of generated clock are balanced with sinks of master

checks before runing CTS

    check_design -type cts

cells and Library Trimming
The recommendation is to only specify LVT cells for the clock

reports why CTS rejected cells which you specified by running:

    report_ccopt_cell_filtering_reasons

Setting up ccopt properties and running the command flow

This flow describes how to set properties and run commands for ccopt

commands	note
pre-CCTS Innovus Database	
Load post-CTS timing constraints	avoid set_propagated_clock
set preferred setOptMode settings	Not needed for "ccopt_design -cts"
create_route_type -name CLK_NDR .. set_ccopt_property route_type -net_type {leaf, truck, top} -clock_tree <name>	set NDR data type
set_ccopt_property buffer_cells ... set_ccopt_property invert_cells .. set_ccopt_property clock_gating_cells ...	specify CTS cells
set_ccopt_property_target_max_trans set_ccopt_property target_skew ...	
create_ccopt_clock_tree_spec	generate CCOpt clock trees from .sdc file
ccopt_design [-cts]	

clock tree synthesis flow with ccopt			
	place_opt_design		
ccopt is by default a concurrent CTS and timing-optimization engine. It can be run in either useful skew mode or skew-balanced mode	ccopt_design (optDesign built-in)	ccopt_design -cts optDesign -postCTS	runing ccopt with -cts in skew-balanced mode requires an additional optDesign step for datapath optizaiton
	optDesign -postCTS -hold		
	routeDesign		
	optDesign -postRoute -setup -hold		

configuing ccopt before runing cts

The generated clock tree and skew groups may need to be customized

It' s recommended to not edit the automatically generated spec file but to instead override the values by setting properties

create_ccopt_clock_tree_spect	create_route_type	set_ccopt_property
derive clock tree structure (clock trees, skew groups, stop/ignore pins)	configure clock tree routing	configure ccopt setting

optimization setting for useful skew

setOptMode -usefulSkewCCOpt	cts	post-csts optimization
None	skew balanced	without useful skew

standard	skew balanced	with useful skew
extreme	useful skew	with useful skew, hold aware

what are ccopt route type for ndrs

ccopt route type let you categorize types of clock routes so that you can set different non-default routing rules (NDR) for detail routing the clock tree.

leaf: nets connected to leaf cell

trunk: all other nets

top: user defined as net connected to > x fanouts

```
set_ccopt_property -clock tree <clocktree> routing_top_min_fantout
```

```
create_route_type -name top_rule -non_default_rule CTS_2W2S -top_preferred_layer M4 -bottom_preferred_layer M3 -shield_net VSS
```

Note

that the top routing rules will not be used unless the routing_top_min_fanout property is also set:

```
set_ccopt_property route_type -net_type top top
set_ccopt_property route_type -net_type trunk trunk
set_ccopt_property route_type -net_type leaf leaf
```

specifying NDRs for clock routes

Non-default routing rules have to be first defined via LEF or by using the add_ndr command

get_ccopt_skew_groups

This command returns a list of skew groups whose names match the specified pattern.

get_ccopt_clock_trees

This command returns the names of the clock trees whose names match the specified pattern.

report_ccopt_skew_groups

report_ccopt_clock_trees

balancing skews of unrelated clocks

```
create_ccopt_skew_group
    -balance_skew_groups
    -target_skew
```

specifying cells for ccopt

set_ccopt_property

Specifies whether clock tree synthesis should prefer to use inverters rather than buffers when balancing the clock tree. If set to true, CTS will use inverters for the clock tree balancing process. If set to false, CTS will use the minimum number of levels of inverters required to maintain logical correctness. If set to auto (the default) CTS will use what it considers to be the best combination of buffers and inverters to get optimal quality of results.

Valid values: auto true false

Default: auto

Optional applicable arguments: "-clock_tree <name>".

clock tree halo support

set_ccopt_property cell_halo_x -cell <cell> -clock_tree <clock_tree> <x_spacing>

set_ccopt_property cell_hao_y -cell <cell> -clock_tree <clock_tree> <y_spacing>

To check the halos:

report_ccopt_cell_halo_violation

setting stop and ignore pins for clock tree synthesis

set_ccopt_property sink_type {stop, ignore ...} -pin <name>

stop pin:

clock spec creation stops tracing at the pin

the pin is considered a sink to be balanced in any skew groups which reach it, even the pin is not identified as a clock pin

ignore pin:

clock spec creation will trace to this pin, even if SDC clocks do not

the pin is not balanced

To report the clock tree(not skew group) in a text file, run the following command:

report_ccopt_clock_tree_structure

To report clock timing, run the command:

report_clock_timing

How to stop CCOpt-CTS at different stages for debugging

CTS goes through five major stages by default when ccopt_design is run:

1. Clustering
2. Virtual Balancing
3. Balancing
4. Routing
5. Post Conditioning

You can stop CTS after any of these intermediate stages for debugging. But the flow **cannot** be restarted from any intermediate stage.

stage	command	comment
1. clustering	set_ccopt_property balance_mode **cluster** create_ccopt_clock_tree_spec -file spec.tcl source spec.tcl ccopt_design -cts [or ccopt_design]	
2. virtual balancing	set_ccopt_property balance_mode **trial** create_ccopt_clock_tree_spec -file spec.tcl source spec.tcl ccopt_design -cts [or ccopt_design]	
3. Full Balancing	set_ccopt_property use_estimated_routes_during_final_implementation true create_ccopt_clock_tree_spec -file spec.tcl	

	source spec.tcl ccopt_design -cts [or ccopt_design]	
4. routing	set_ccopt_property post_conditioning **false** create_ccopt_clock_tree_spec -file spec.tcl source spec.tcl ccopt_design -cts [or ccopt_design]	
5. post conditioning		

Identifying dont touch and dont use object

Identifying dont touch and dont use object attribute helps you prevent and debug unexpected results during clock tree synthesis

```
report_preserves
    [-help]
    [-computed]
    [-obj_type {design | hinst | module | inst | net | hnet | pin | hpin | base_cell}]
    [-dont_touch | -dont_use]
```

report the attributes of objects on the whole design, for example, dont touch, dont use setting

Allowing resizing of dont touch objects

```
dbSet [dbGetInstByName name].dontTouch sizeOk
```

Run the command to allow resizing of objects with dont touch attributes

The command override the default behavior of the CTS tool which does not size objects with dont touch attributes

```
innovus #> dbSchema inst dontTouch
=======================================================
inst: Instance - canonical (flat), equivalent to DEF COMPONENT. Points to a libCell or ptnCell.
-------------------------------------------------------------------------------------------
dontTouch(settable): enum(constPropDeleteOk constPropSizeDeleteOk deleteOk false mapSizeOk none
```

sizeDeleteOk **sizeOk** sizeSameFootprintOk sizeSameHeightOk true), This attribute defines the user preservation status of an instance during optimization.

Fixing CCOpt PostRoute

You may need to fix the clock tree postRoute because detail routing clock nets may introduce DRV, as a result of routing detours and via

CTS	ccopt_design
1. Detail route clock nets	
2. Post-conditioning (DRV fixing)	
3. Post-CTS optimization	
4. Full design routing routeDesign	
optDesign -postRoute -setup -hold Note: The command ccopt_pro is included in the postRoute optimization	

ccopt_pro repairs clock DRV and optionally clock skew

What is a library exchange format file

The library exchange format(LEF) file contains layer, via, and macro definitions, which are required by placement and routing

Technology section

Cell/Macro section

Routing Flow

stage	comment
Global Route	Design divded into GCells and routes assigned to tracks Congestion analyzed
Detail Route	creates the routes applies all DRC rules Gvie priority to critical nets
Search and Repair	surgical fixes for individual vias and wires search area increases by 3x3 5x5 and 7x7 in GCell areas Deletion and reroute as a lst resort if no other solution is found

Repairing Antenna

Antenna repair uses layer hopping by default, specify antenna diode cells to allow the router to use antenna cells to repair violation

Approach	Rationale	Impact
layer hopping	Top metal complete connection	Longer routes more RCs more congestion
net diode insertion	discharge path through diode	added capacitance area penalty longer routes(place and rotue to diode
diode in cell	cell protected by construction	added capacitance area penalty

diode insertin is enabled when fix antenna and insertion diodes are selected

select insert diodes to define the diode cell name of the cells to insert

```
setNanoRouteMode -drouteFixAntenna true
setNanoRouteMode -routeInsertAntennaDiode true
setNanoRouteMode -routeAntennaCellName  “antenna”
```

By default, the NanoRoute engine searchs the LEF file and insert the first Macro with
CLASS CORE ANTENNACELL it finds with the appropriate SITE definition

Specifying Attribtes

when you set attributes for nets, the NanoRoute routes the design according to the attribtes you set and saves the attribtes with the database

```
setAttribute -net net1 -shield_net VDD -shield_side two_sides
```

Timing and SI-Driven Routing
setting NanoRoute mode for timing and SI-Driven routing guides the router to produce routes with the best Quality of Results(QoR)
setNanoRouteMode -routeWithSiDriven true -routeWithTimingDriven true

setDesignMode
    -topRoutingLayer layer
        Specifies the highest lef layer name for global and detail routing.
        Default: ""
    -bottomRoutingLayer layer
        Specifies the lowest lef layer name for global and detail routing.
        Default: ""

power-driven routing
    setDesignMode -powerEffort high

Yield optimization: postRoute wire spreading
    setNanoRouteMode
        -droutePostRouteSpreadWire
            Specify one of the following:
                auto: Spreading is on if the timing effort is set to high (setDesignMode –flowEffort high). Otherwise it is off.
                true: 0.5 / 1 track spreading.
                Set -droutePostRouteSpreadWire to true if you want to spread.
                false: no spreading
                0.25: 0.25 / 0.5 / 1 track spreading
                0.5: 0.5 / 1 track spreading
                1: 1 track spreading

Yield Optimization: postROute wire widening
Widen wires where resources are available without adding DRC and antenna violations or affecting timing. wire widening uses NONDEFAULT rules
    setNanoRouteMode -droutePostRouteWidenWireRule ruleName
    setNanoRouteMode -droutePostRouteWidenWire widen
    use the following commands to control the effect on timing (default 0.0 ns):
    setNanoRouteMode -drouteMinSlackForWireOptimization <slack>
    routeDesign -wireOpt

Yield Optimization: muticut via swap
PostRoute single-cut via to multicut via swap for 16nm and below nodes
    setNanoROuteMode -droutePostRouteSwapVia multiCut
    setNanoRouteMode -drouteMinSlackForWireOptimization <slack>
    routeDesign -viaOpt

Reporting Wires and Nets

To report route statistics, run the command report_route

    report_route [-clock] [-em] [-multi_cut] [-ndr] [-net]

Reporint wire segment paths

    reportWirePath -start <term> -end <term>

Adding partial routing blockages

    createRouteblk -partial

overflow or OverCon = (Deman - supply) per Gcell

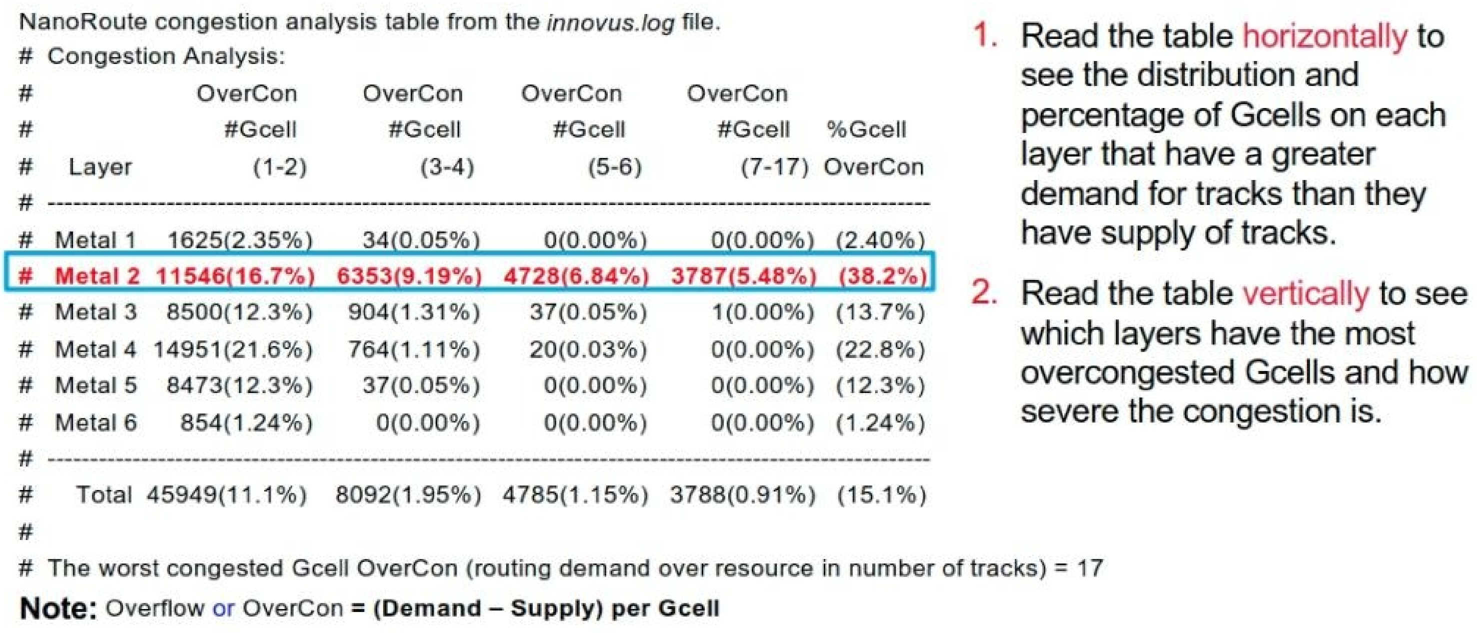
reportCongestion

    -overflow

        Reports horizontal and vertical overflow.

    -3d

        Honors 3d congestion map. When specified, it enables the layer-based congestion reporting.



NanoRoute congestion analysis table from the *innovus.log* file.

```
#  Congestion Analysis:
#                 OverCon       OverCon       OverCon       OverCon
#                  #Gcell        #Gcell        #Gcell        #Gcell    %Gcell
#    Layer          (1-2)         (3-4)         (5-6)        (7-17)  OverCon
#  ---------------------------------------------------------------------------
#   Metal 1   1625(2.35%)     34(0.05%)      0(0.00%)      0(0.00%)  (2.40%)
#   Metal 2  11546(16.7%)   6353(9.19%)   4728(6.84%)   3787(5.48%)  (38.2%)
#   Metal 3   8500(12.3%)    904(1.31%)     37(0.05%)      1(0.00%)  (13.7%)
#   Metal 4  14951(21.6%)    764(1.11%)     20(0.03%)      0(0.00%)  (22.8%)
#   Metal 5   8473(12.3%)     37(0.05%)      0(0.00%)      0(0.00%)  (12.3%)
#   Metal 6    854(1.24%)      0(0.00%)      0(0.00%)      0(0.00%)  (1.24%)
#  ---------------------------------------------------------------------------
#     Total  45949(11.1%)   8092(1.95%)   4785(1.15%)   3788(0.91%)  (15.1%)
#
#  The worst congested Gcell OverCon (routing demand over resource in number of tracks) = 17
```

Note: Overflow or OverCon **= (Demand – Supply) per Gcell**

1. Read the table horizontally to see the distribution and percentage of Gcells on each layer that have a greater demand for tracks than they have supply of tracks.
2. Read the table vertically to see which layers have the most overcongested Gcells and how severe the congestion is.

Rules of thumb (excluding the pin acess layer):

    Total % of overCong < 2% -> easy to route

    Total % of overCong between 2% and 6% -> difficult to route

    Total % of overCong > 6% -> high number of DRC violations

Fixing Routing Violations

    editDelete -regular_wire_with_drc

    routeDesign -globalDetail

selecting and editing nets

setEditMode
editSelect
editAddRoute

The editSelect Command
select wires and vias for wire editing with the editSelect command

To deselect wires
editDeselect

Deleting Routes interactively
editDelete

Cutting and Resizing Wires
cut a wire by drawing a line through the wire or drawing a box around a wrire with command
editCutWire
-selected
Specifies that only the selected wires, bus guides, or rectangles should be cut.

change a wire’ s length or width interactively or with the command
editResize

What are path wires?
path wires are small amounts of metal created by the detail router to fix DRCviolations
path as a rectangle on the routing alyer in the NETS section of a DEFpathes are commonly used to add a small amount of metal to fix minimum-area vilolation, or fill in notch violations

Interactively Replacing Vias
You can replace a selected via with another via in the design that has the LEF rule

What are the types of noise that can impact a routed design?
There are two major types of noise that can affect signal integrity(SI) in a routed design:
1. Glitches can cause functional failures
2. Crosstalk can cause timing problems

Fixing signal integrity issues with routing fixes
Because signal integrity(SI) is an interconnect-centric issue, it is most effective to address it at the routing stage when the wires meet the gates.

Glitch fixing includes fast, on-the-fly extraction, timing, and analysis
1. The analysis engine combines glitch peaks from all aggressors if there are overlapping timing windows
2. the router re-route nets whose combined glitch exceeds the noise threshold(peak, width)

balances	wire spacing
SI	net ordering

/Timing /Routability	wire topology control
	layer selection to reduce coupling
	layer selection to reduce resistance
	minimizing parallel long wires
	shielding
	buffer insertion

Fixing Crossstalk postroute

The software uses an advanced crosstalk repair algorithm that features:

- victim-driver upsizing
- buffer insertion
- aggressor downsizing
- aggressor rip up
- victim spacing and protection

```
setDelayCalMode -engine aae -SIAware
optDesign -postRoute
```

Verify a design for the following:

- connectivity
- metal density
- DRC rules
- process antenna
- well taps
- ac limit
- wire gap

what are DRC and LVS?

Design-rule checks(DRCs) are checks that are performed with respect to the rules provided by a foundary for a particular technology and process node

Layout-vs-schematic(LVS) violations occur when there is mismatch in connectivity between what is in the placed-and-routed design and the verilog netlist or shorted nets.

DRC checks using place-and-route data

All cells are modeled with a LEF(or abstarct). No layout available

pros	**cons**
Fast	The checks are as accurate as the abstracts
small database	No checks at different hierarchy levels
problems can be debugged and fixed fast	connection to pins could have violations

verifying connectivity

To report open nets, antennas, loops, and partial routing, for all nets or specified nets in your design

```
verifyConnectivity
```

verifying metal density
To check wheter the metal density for each routing layer is within the minimum and maximum density values specified in the technology file

```
verifyMetalDensity
```

verifying DRC
Toe check for rule violations as specified in the tech.lef

```
verify_drc
```

commands for verifying DRCs

```
set_verify_drc_mode
verify_drc
```

Run the set_verify_drc_mode and verify_drc commands to verify DRC rules, for example, spacing for all nets, special nets,etc.
A routing halo is an area around a macro that prevents routing other than for direct pin access

```
set_verify_drc_mode
    -check_routing_halo
        true
    -max_wrong_way_halo
        0.1
verify_drc
```

verifying DRCs for Cells

```
set_verify_drc_mode
    -check_only {all | regular | special | selected_net | selected | cell | default}
```

check violations within a cell or between two different cells by

```
set_verify_drc_mode -check_only cell
verify_drc
```

Ignoring cell blockage DRCs

```
set_verify_drc_mode
    -ignore_cell_blockage true (default flas)
```

Disabling DRC Rules

```
set_verify_drc_mode
    -disable_rules
```

The following commands disable eol spacing check:

```
set_verify_drc_mode -disable_rules {eol_spacing}
verify_drc
```

The following commands check enclosure and cut spacing rules only:

```
set_verify_drc_mode -disable_rules {eol_spacing min_cut color min_step protrusion min_area ...}
verify_drc
```

verifying the process antenna

```
verifyProcessAntenna
```

Identify process antenna violations in your design and output an antenna.lef for IO pins in a block
Maximum floating area violations appear in the antenna report(.rpt) file.

verifying well taps
Run verifyWellTap to check the design after running the addWellTap command to highlight row areas that contain potential violations if cells area placed in those areas

```
verifyWellTap -cell *WELL* -rule .02
```

If a list of cells is not specified, the CLASSS CORE WELLTAP is used from the LEF file to determine the cell masters

set_well_tap_mode
addWellTap
deleteFiller

Verifying AC Limit(Electromigration)

calculates the root mean square current(Irms) at the driver output and compares it to the ACCURRENTDENSITY tables in the LEF file that contain the Irms limits for routing layers.

Use this form to check for AC current violations on signal nets

Verifies the continuity of **follow pin and stripes**. Use this command to identify power strap failures due to broken power wires in the power planning stage
verifyWireGap

-wireToWire value

Specifies the wire-to-wire gap distance. Use this check to ensure that wires do not stop short of another wire of the same net or layer

Engineering Change Orders

What is an ECO
An Engineering Change Order(ECO) is written when a change has to be made to an implemented design

ECO Commands in Innovus

ecoDesign - super-command

ECO commands

- ecoAddRepeater
- ecoChangeCell
- ecoCloneFlop
- ecoCompareNetlist
- ecoDefIn
- ecoDeleteRepeater
- ecoPlace
- ecoRoute
- ecoSplitFlip
- ecoSwapSpareCell
- attachTerm

detachTerm

addInst

deleteInst

deleteBufferTree

setting the ECO mode

setEcoMode

-LEQCheck true

enable functionality checking when swapping cells. During ECO, the software can swap cells that are logically equivalent

-spreadInverter true

distribute an inverter pair evenly between the driver and the first bifuraction point. The design must be routed before you use this parameter

-honorPowerIntent

used to perform power-domain-aware checks before adding removing or sizing cells

-batchMode

- Make sure that -batchMode is set to false before saving the design. If it is set to true, any other setEcoMode commands will not be observed. setEcoMode commands such as -refinePlace and -updateTiming should always be set prior to, or at the same time as, setting -batchMode true.
- Make sure that you exit batch mode (setEcoMode -batchMode false) prior to running timing analysis. Otherwise, report_timing will report the following:

```
No constrained timing paths found.
Paths may be unconstrained (try '-unconstrained' option).
```

Speeding up run time of interacti...

-updateTiming <true|false>

control whether the eco* command recompute timing. If false, the commands do not recompute timing and later you can run report_tiing -recompute timing

-refinePlace {true|false}

Legalizes placement whenever ecoAddRepeater or ecoChangeCell are used. If false, you must run refinePlace after ECO if you want to legalize placement.

Interactive ECO

ecoAddRepeater adds a single buffer or two inverter on a net, If specified, a downstream sink point is buffered at a particular location.

ecoAddRepeater

-net <netname>

-cell <cellname>

-loc {x1 x2}

-noPlace

Specifies that the inserted cells should not be placed. Only the logical change in connectivity will be made.

ecoChangeCell

To upsize specified instance

```
        ecoChangeCell -inst Top_Inst1 -upsize
    To downsize specified instance
        ecoChangeCell -inst Top_Inst2 -downsize
    To modify specified instance to specific cell
        ecoChangeCell -inst Top_Inst3 -cell BUFX20
```

Note: To prevent ecoChangeCell from running refinePlace, use setEcoMode -refinePlace flase

Deltes buffers or inverter pairs

To delete the specified buffer, where one buffer is A/inst1

```
    ecoDeleteRepeater -inst A/inst1
```

To delete an inverter pair, where one inverter is A/inst2 and another is A/inst3

```
    ecoDeleteRepeater -invPair {A/inst2 A/inst3}
```

Run interactive ECO to:

- add a buffer
- add a instance
- change a cell for a particular instance
- delete a buffer
- display a buffer tree

To save the interactive ECO operations to a replay file(ECO Directives), before starting the ECOs, run the following commands

```
    intECO <eco_file> # logs ECO commands
        <Interactive ECO operations>
    endECO # Ends logging of ECO commands
```

LoadECO

reads a file containing ECO directives and applies changes to the current netlist. Asummary report is generated

Note:

The ECO directive in the file read by the loadECO command are not TCL commands and ttherefore cannot be run directly from the Innovus command line or sourced form a file.

All the command/directives in the file are uppercase

attachTerm

Attaches a terminal to a net. If the terminal already connects to a different net, the software first detaches the terminal from the current net, then attaches it to the new net.

detachTerm

Disconnects a terminal from a net.