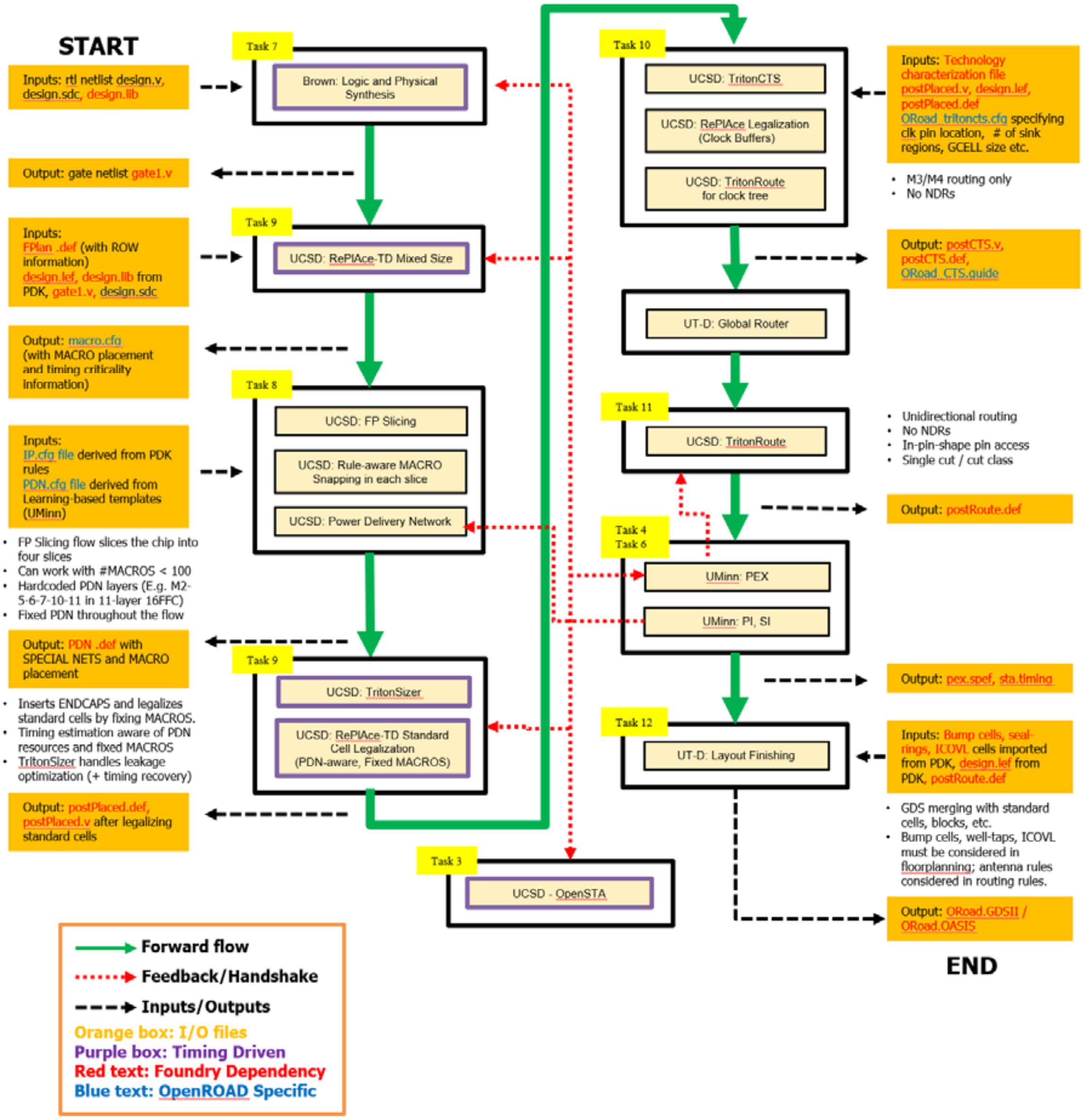
OpenROAD Flow and Notes, version 1.0, November 2018

https://theopenroadproject.org

Leading up to the first DARPA IDEA/POSH Integration Exercise (January 2019) and the planned Alpha release of OpenROAD tools (June 2019), we would like to share the following initial information about our project.

Comments and questions welcome: abk-openroad@eng.ucsd.edu

OpenROAD Flow and project task mapping (November 2018):


START
Inputs: rtl netlist design.v, design.sdc, design.lib
Task 7
Brown: Logic and Physical Synthesis
Output: gate netlist gate1.v
Task 9
Inputs: FPlan .def (with ROW information) design.lef, design.lib from PDK, gate1.v, design.sdc
UCSD: RePlAce-TD Mixed Size
Output: macro.cfg (with MACRO placement and timing criticality information)
Task 8
UCSD: FP Slicing
UCSD: Rule-aware MACRO Snapping in each slice
UCSD: Power Delivery Network
Inputs: IP.cfg file derived from PDK rules PDN.cfg file derived from Learning-based templates (UMinn)

- FP Slicing flow slices the chip into four slices
- Can work with #MACROS < 100
- Hardcoded PDN layers (E.g. M2-5-6-7-10-11 in 11-layer 16FFC)
- Fixed PDN throughout the flow

Output: PDN .def with SPECIAL NETS and MACRO placement
Task 9
UCSD: TritonSizer
UCSD: RePlAce-TD Standard Cell Legalization (PDN-aware, Fixed MACROS)

- Inserts ENDCAPS and legalizes standard cells by fixing MACROS.
- Timing estimation aware of PDN resources and fixed MACROS
- TritonSizer handles leakage optimization (+ timing recovery)

Output: postPlaced.def, postPlaced.v after legalizing standard cells
Task 3
UCSD - OpenSTA
Task 10
UCSD: TritonCTS
UCSD: RePlAce Legalization (Clock Buffers)
UCSD: TritonRoute for clock tree
Inputs: Technology characterization file postPlaced.v, design.lef, postPlaced.def ORoad_tritoncts.cfg specifying clk pin location, # of sink regions, GCELL size etc.

- M3/M4 routing only
- No NDRs

- Unidirectional routing
- No NDRs
- In-pin-shape pin access
- Single cut / cut class

Output: postRoute.def
Task 4
Task 6
UMinn: PEX
UMinn: PI, SI
Output: pex.spef, sta.timing
Task 12
UT-D: Layout Finishing
Inputs: Bump cells, seal-rings, ICOVL cells imported from PDK, design.lef from PDK, postRoute.def

- GDS merging with standard cells, blocks, etc.
- Bump cells, well-taps, ICOVL must be considered in floorplanning; antenna rules considered in routing rules.

Output: ORoad.GDSII / ORoad.OASIS
END
Forward flow
Feedback/Handshake
Inputs/Outputs
Orange box: I/O files
Purple box: Timing Driven
Red text: Foundry Dependency
Blue text: OpenROAD Specific

OpenROAD Flow Notes (November 2018)

OpenROAD RTL-to-GDS Expected Features at Alpha Release (June 2019):

- Netlist-to-GDS capable in TSMC 16FFC. Only designated foundry nodes will be supported (*Alpha: TSMC 16FFC; v1.0: TSMC16FFC, ST28FDSOI, TSMC65GP*)
- Field of use: Digital SOC only up through v1.0.
- Flow Inputs:
  - LEF 5.8, Verilog, Liberty (NLDM), SDC (only public versions, e.g., v1.4).
    - DEF, SPEF are supported in point tools
  - User inputs include PPA priority (power vs. performance vs. area), Power priority (static vs. dynamic)
    - *User must supply necessary config files and OpenROAD kit elements*
- Flow Outputs:
  - Post-route .def, tapeout-ready .gds, PPA estimates
- Feature (restriction) list:
  - Timing- and congestion-driven, “correct and safe by construction”
  - Integrated on commercial binary P&R database (if available in calendar 2018)
  - Timing models supported: only Liberty NLDM through v1.0.
    - *Timing analysis is non-SI at Alpha; SI loop added in v1.0.*
  - Power domains supported: 1 at Alpha; at most 3 (logic, memory, IO) with support of UPF 2.0 subset (LS, FP tasks) at v1.0.
    - *No power gating until v2.0.*
  - Clocks: At most 3 clocks at v1.0 (func, test, debug).
    - *Skew, insertion delay, buffer choice etc. are determined by the tool.*
- For developers and users: **we guarantee build on clean CentOS 6 VM.**
  - *Additional platforms will be announced at https://theopenroadproject.org*

Level-Setting for OpenROAD Tool/Flow Users:

- The handoff from IDEA TA-2 to TA-1 has not yet been specified by the IDEA Design Flow Working Group. OpenROAD relies on handoff into TA-1 being “realizable”.
- In light of “No Human In Loop”, only a limited Tcl (v8.5) scripting interface is provided.
- OpenROAD tools will not be foundry-qualified.
  - We require “config files” and other one-time generation of OpenROAD kit elements.
  - OpenROAD tools and tool developers will not be able to read encrypted advanced-node PDKs (e.g., .qrctechfile, .tluplus, Calibre .svrf, etc.). A key challenge is to be safe and correct by construction given this limitation.
  - We believe that “ConfigFile” and “precharacterized models / abstractions” must be available per enablement.
    - *For example: FoundryX may hand an “OpenROAD kit” to its 7FF customers. This “kit” will need to be produced by a FoundryX customer running its licensed commercial EDA tools (e.g., Raphael, HSPICE, etc.) with OpenROAD-published “calibration scripts”. We see no other workaround for our inability to see the foundry enablement.*
- OpenROAD will not have golden verifiers.
  - OpenROAD provides analyses and estimators for timing, parasitics and power/signal integrity. However, these will not be golden, “signoff” verifiers. Our analysis reports will be heavily guardbanded, and will leverage models of analysis miscorrelations (cf. calibrations, “kit” elements). Tapeout without use of golden verifiers is up to the user; foundry acceptance of tapeout is between foundry and user.
    - *Note: OpenROAD will not develop formal verification tools; again, correct-by-construction transforms only*
- OpenROAD tools are being developed and released by non-commercial entities.
  - A commercial EDA tool vendor receives bug/enhancement requests accompanied by a testcase that exhibits the bug or behavior at issue. In the OpenROAD/IDEA world, bug reports will likely <u>not</u> be accompanied by testcases due to blocking NDA / IP restrictions. We believe that bug fixing can be slow to impossible in this regime.

Additional Notes

- Terminology
  - “PDK” refers to pcell, SPICE model, parasitic model, sealring, DRM, …
  - “Enablement” refers to IPs and stdcell libraries (+ reference flow in commercial context)
- On “config files”
  - For any given target foundry PDK, OpenROAD tools will assume availability of precharacterized models / abstractions in OpenROAD format, generated by OpenROAD utilities (run by some foundry customer). We refer to these as “config files” and/or as elements of an “OpenROAD kit” that must be available to our tools, for a given foundry PDK and enablement.
  - Examples of utilities and kit elements:
    - SPICE-based delay calibration
    - BEOL PEX calibration
    - Safe electrical design methodology: (i) bounds on loads, slews; (ii) interconnect tuning: NDRs, buffering; (iii) PBA, SI delay and slew timing margining from non-SI GBA STA
    - (Stitchable) PDN templates
    - Library filtering: don’t-use cell list (difficult pin access), forbidden cell abutments
  - Example config file elements in floorplan space (for given PDK and IPs):
    - Macro placement halos
    - Lowest-common-multiple grid for “snapping” of macro placements
  - Example “kit” elements that must be supplied by OpenROAD users:
    - Package stackup, layout rules
    - Sealring .lef (produced by Virtuoso flow from pcell in foundry PDK)
    - Bump .lef
    - Abstraction of foundry dummy fill rules (all layers)
- Current repositories
  - Placement: https://github.com/abk-openroad/RePlAce
  - CTS: https://github.com/abk-openroad/TritonCTS
  - STA: https://github.com/abk-openroad/OpenSTA
  - Gate Sizing: https://github.com/abk-openroad/TritonSizer
  - Private as of November 2018:
    - Floorplan: https://github.com/abk-openroad/TritonFPlan
    - Project GitHub: https://github.com/The-OpenROAD-Project/The-OpenROAD-Project