DISCoVER Expedition Status (qPALACE v2)

THE USC-LED DISCOVER EXPEDITION AWARD INSTRUMENTAL ROLE IN DEVELOPING MORE EFFICIENT COMPUTING SOLUTIONS (PHOTO CREDIT: USC

D. Scott Holmes

University of Southern California

2025-06-17 ISEC, Erfurt, Germany

https://discoverexpedition.usc.edu/

IEEE-CSC, ESAS and CSSJ SUPERCONDUCTIVITY NEWS FORUM (global edition), Issue No. 60, Sept. 2025. Presentation given at ISEC 2025, Erfurt, Germany, June 2025.

DISCoVER Expedition

Design and Integration of Superconducting Computation for Ventures beyond Exascale Realization

Prof. Massoud Pedram
(Director)
USC

Prof. Eby Friedman
(Associate Director
and Thrust 2 Leader)
U. Rochester

Prof. Murali Annavaram
(Thrust 1 leader)
USC

Prof. Grace Xing
(Thrust 3 Leader)
Cornell U.

Prof. Yanzhi Wang
(Thrust 5 Leader)
Northeastern U.

Prof. Michael Hamilton
(Thrust 4 Leader)
Auburn U.

Oleg Mukhanov (SeeQC)

Ivan Nevirkovets (NWU)

Sasan Razmkhah (USC)

Roman Sobolewski (Rochester)

Scott Holmes (USC)

*Chris Ayala and Nobuyuki Yoshikawa (YNU)

Julie Albright (USC)

Mark Bocko (Rochester)

Darin Gray (USC)

Danielle Daniels (Rochester)

Selcuk Kose (Rochester)

Timothy Pinkston (USC)

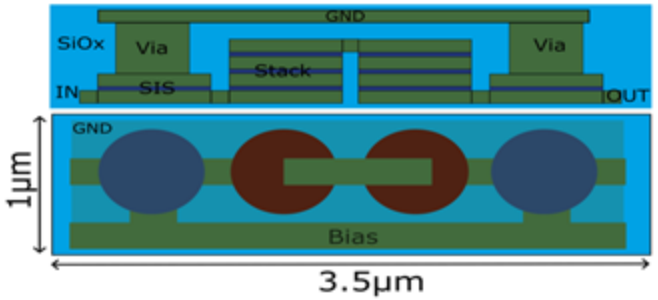
Ivan Komissarov (Rochester)

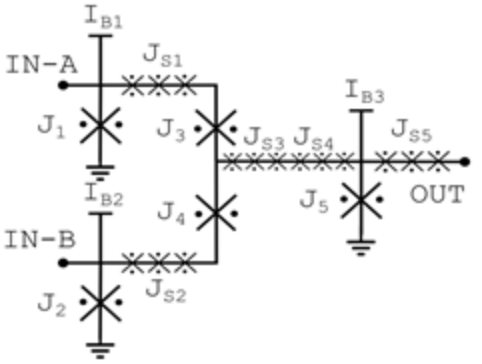
DISCoVER Expedition Thrusts

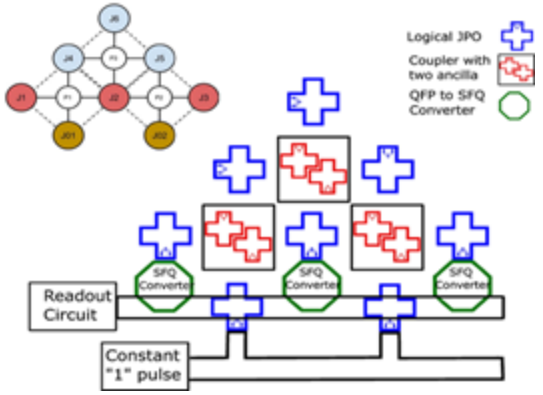
1. SuperSoCC Design and Applications		
SuperSoCC design and demonstration		
2. Circuits and Architectures (including EDA tools)		
All-JJ SFQ cell libraries and fast phase logic family		
3. Devices and Materials		
New barrier and magnetic materials for epitaxial growth		
4. Integration and Interfaces		
Electrical-thermal-mechanical optimization		
5. Environmental and Economic Impacts		
Carbon footprint of superconductor manufacturing		

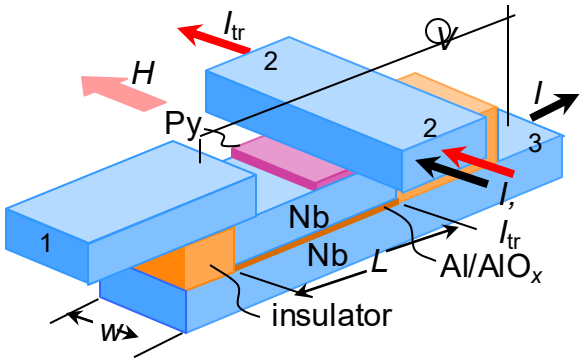
Accomplishments (1)

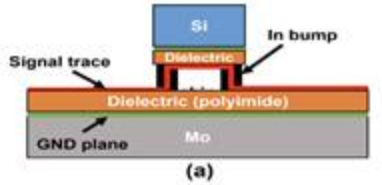
- Preliminary investigation of the material properties and JJ stack design for high-Jc, self-shunted 0-JJs and pi-JJs
- Development of a novel inductorless fast phase logic family with high Jc, self-shunted 0 and pi junctions to achieve 30 M JJ density per cm^2
- Introduction of a superconducting bistable memory cell with an in-memory computing feature
- Demonstration of a Gen-3 memory design with a magnetic layer
- Initial design of multiple modules of SuperSoCC, including SuperCPU, and SuperNN, and SuperIM design
- Suzuki and SQUID stack designs for interfacing and preamplification

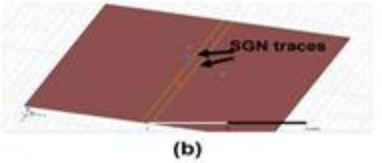


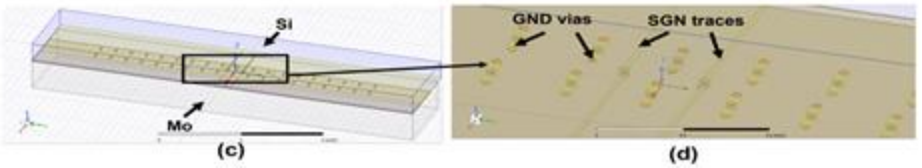








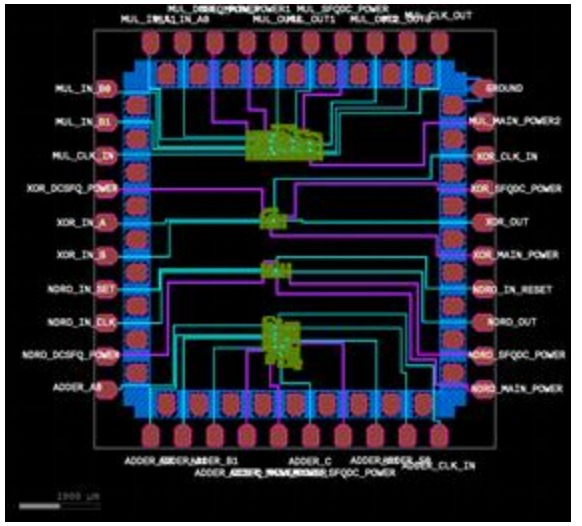




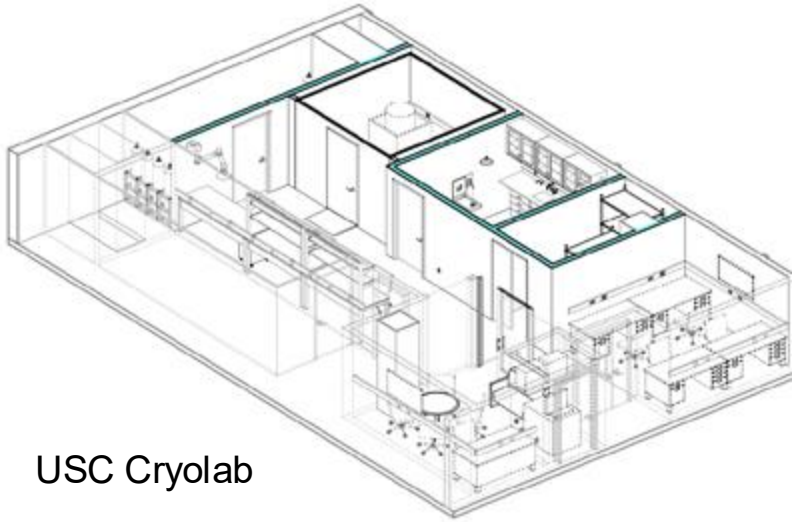
Accomplishments (2)

- Advanced current recycling and thermal modeling for superconductor logic families
- USC cryolab set up and first chips tested
- Chip fabrication at MIT LL; Testing at Auburn and USC cryolabs; empirical device demonstrations at NWU and Cornell
- Many publications (including a book)
- Several graduate students
- Monthly seminar series
- Significant education, outreach, and BPC activities

DISCoVER Workshop, ISCA 2022



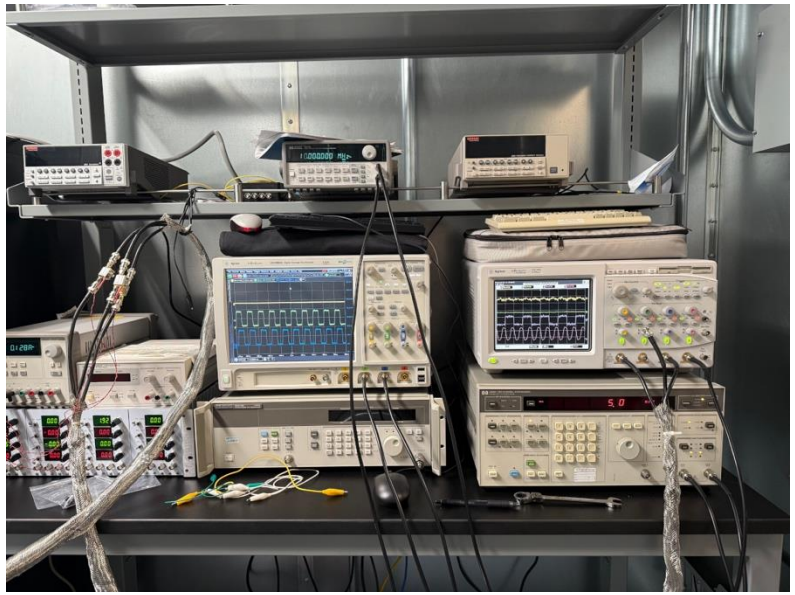
Chip Fabrication at MIT LL

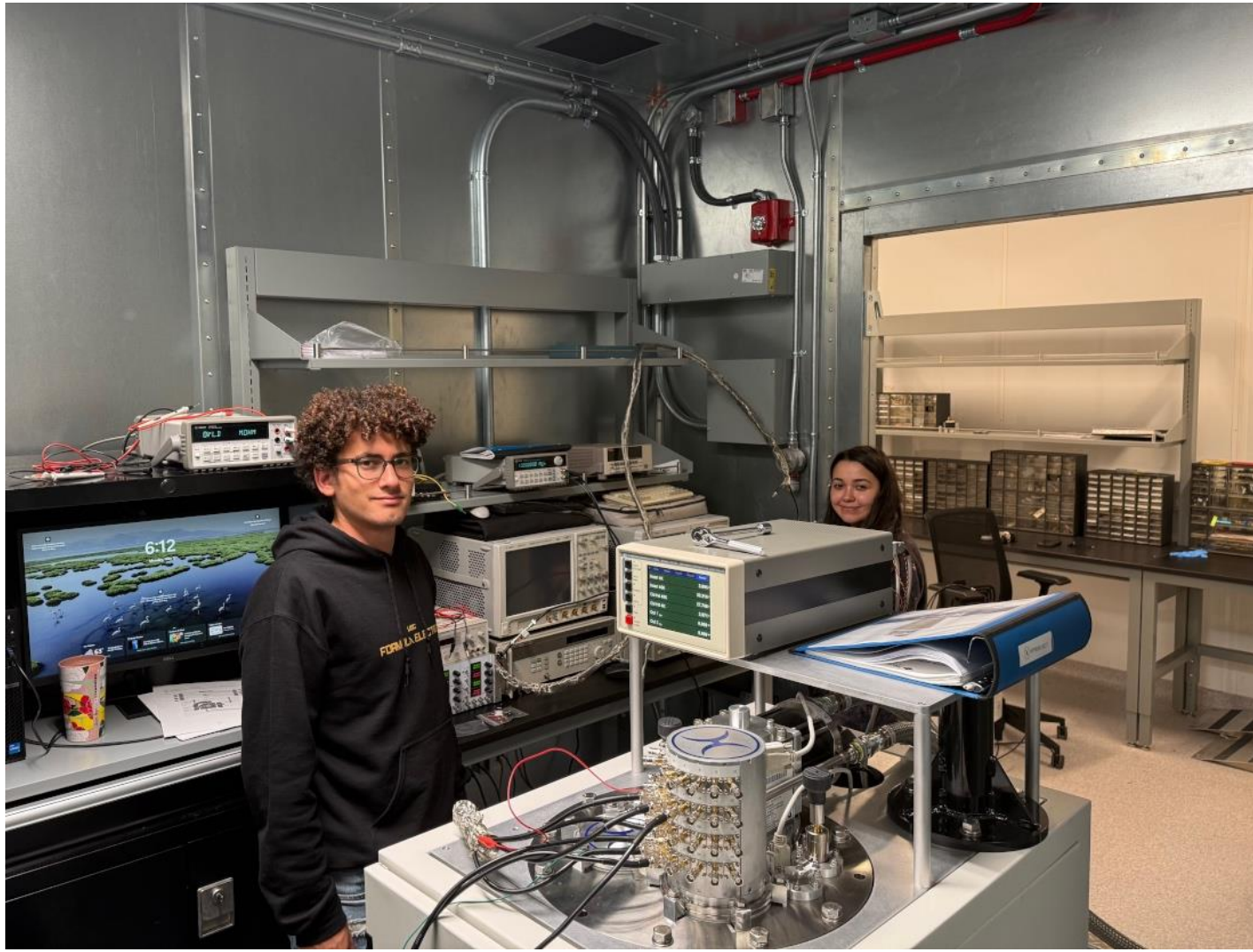


USC Cryolab

Cryogenic Testing at USC CryoLab

- Used to measure and characterize SFQ chip designs.
- Includes: RF-shielded area, work benches, clean area for packaging, chip design area, utility and server rooms, and storage room.

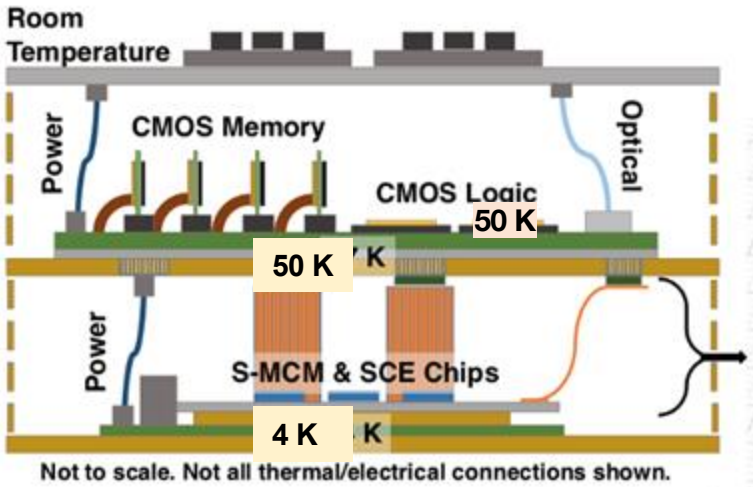


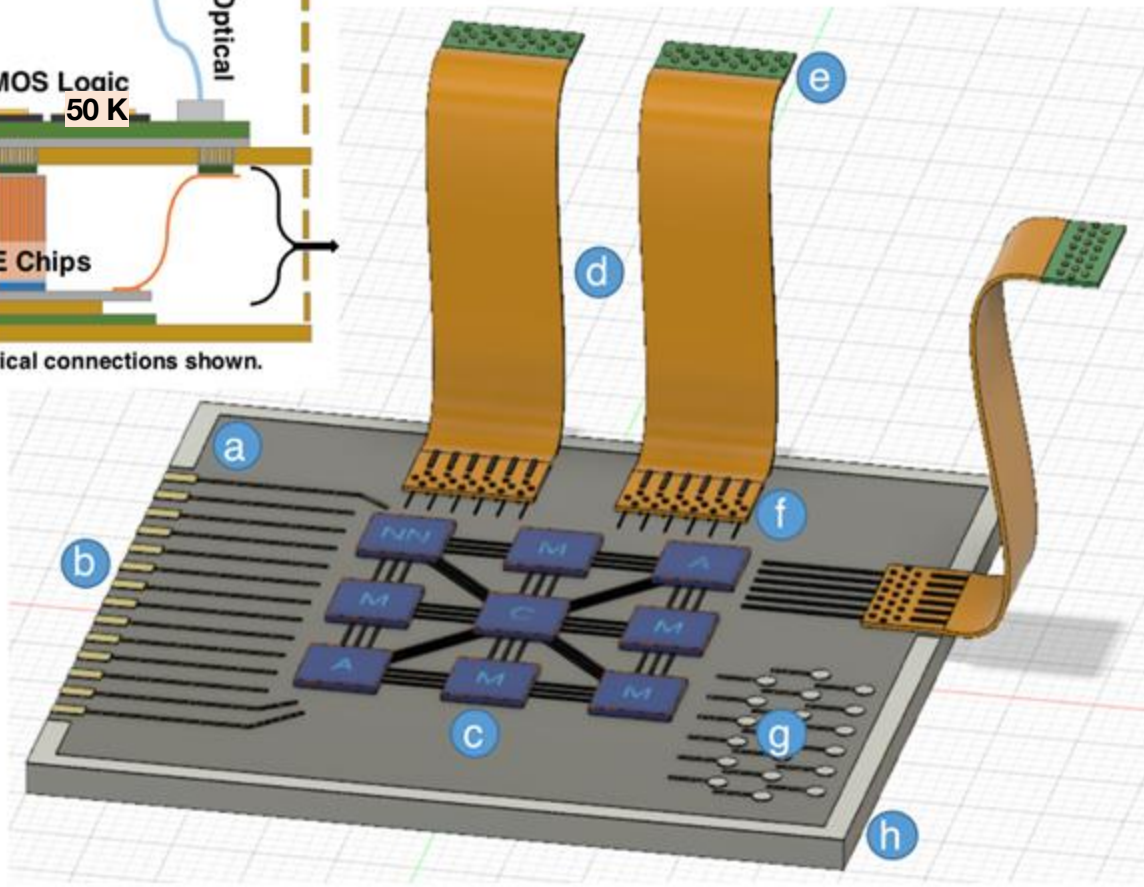


HYPRES ICE-T

1. SuperSoCC Design and Applications

SuperSoCC Testbed





(a) S-MCM

(b) Edge contacts (power)

(c) Core SCE chips (w/ **Devices**)

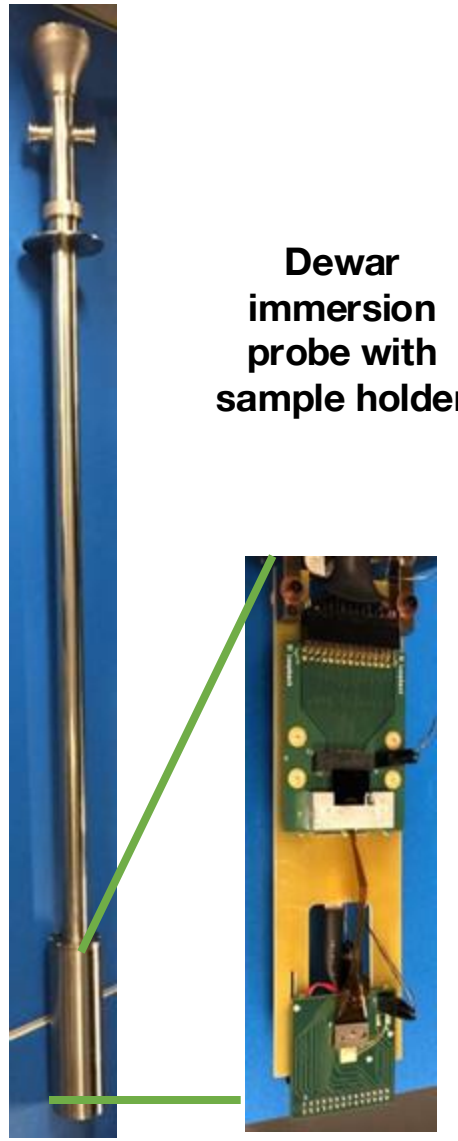
(d) SC flex cables (**Interconnects**)

(e) Interface @ 4 K

(f) Interface @ S-MCM

(g) Area array contacts (signal)

(h) Thermalization approach



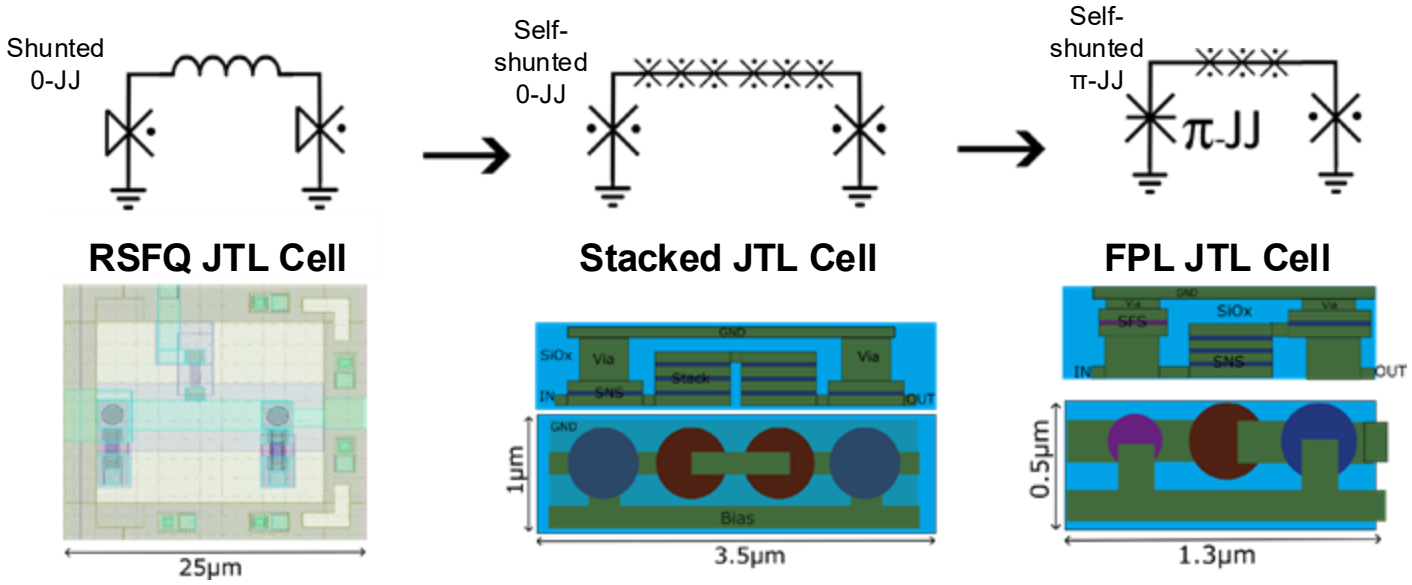
Dewar immersion probe with sample holder

2. Circuits and Architectures (including EDA tools)

Fast Phase Logic (FPL)

Fast Phase Logic (FPL)

- Increasing the critical current will make JJs self-shunted, which reduces the layout area
- Replacing inductors with stacked JJs reduces the layout area and makes cell design much easier
- Smaller layout and no inductor phase delay result in faster signal propagation and lower latency

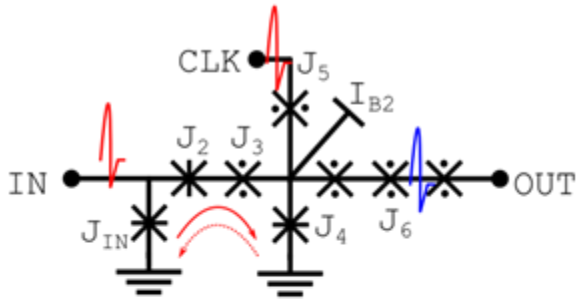


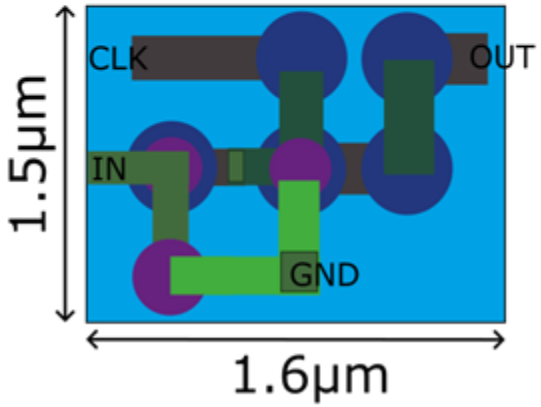
* S. Razmkhah and M. Pedram. "High-density superconductive logic circuits utilizing 0 and π Josephson junctions." Eng. Res. Express, 6(1) 015307, 2024

FPL DFF and AND2 Gates

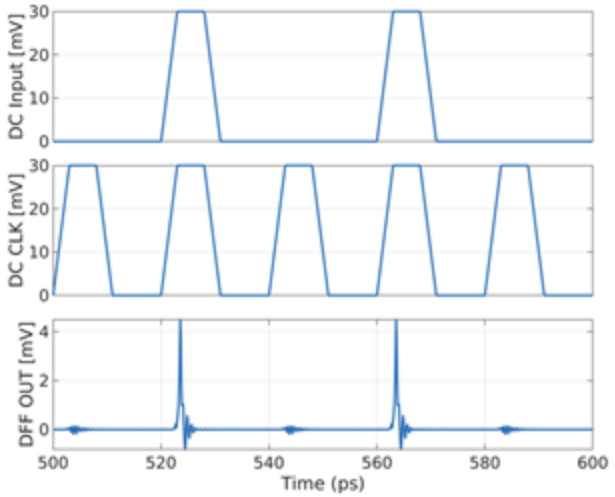
- DFF cell design with high JC 0-JJs and π-JJs
  - Input pulses change the direction of the current inside the 3π loop
  - HFQ clock arrival triggers an output pulse

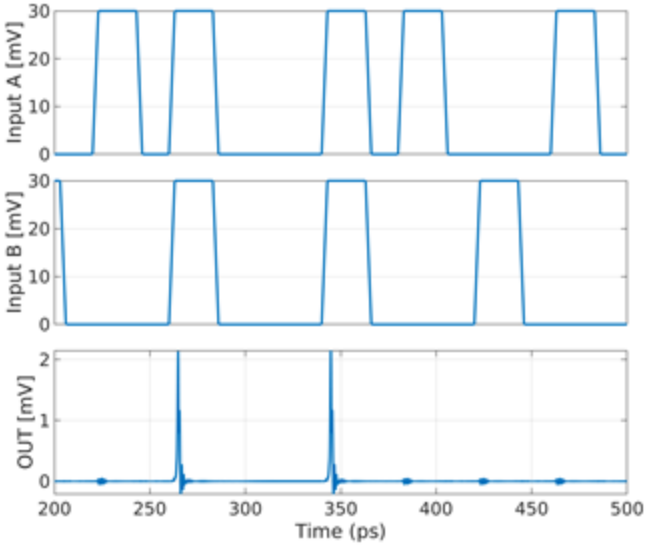
D-Flip Flop in the FPL cell library

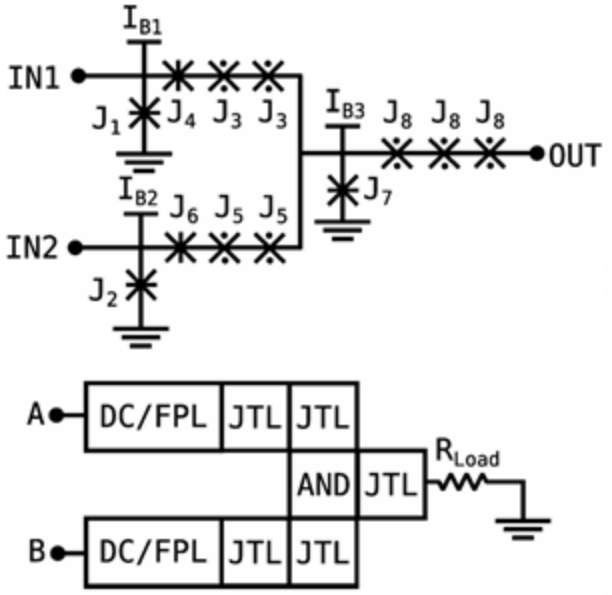




πDFF simulation result and test circuit

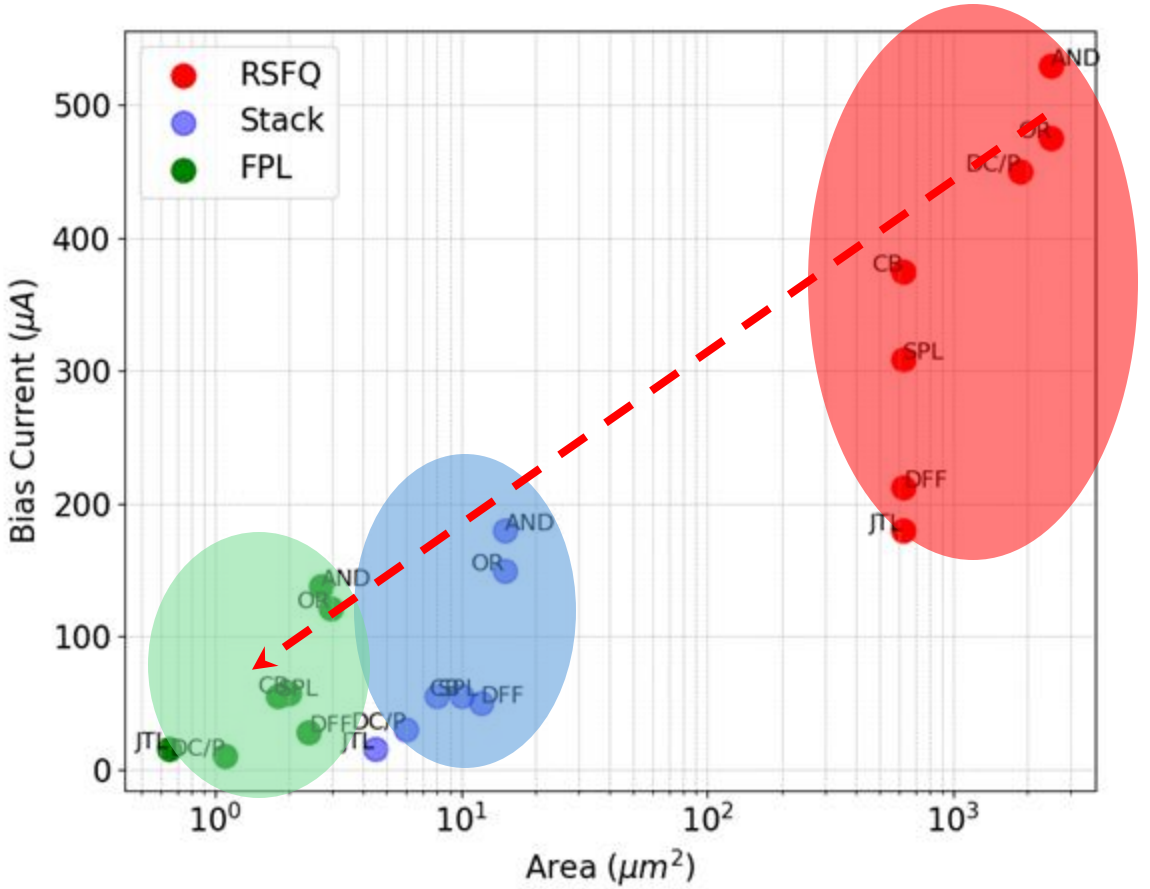






Gate Size Comparison

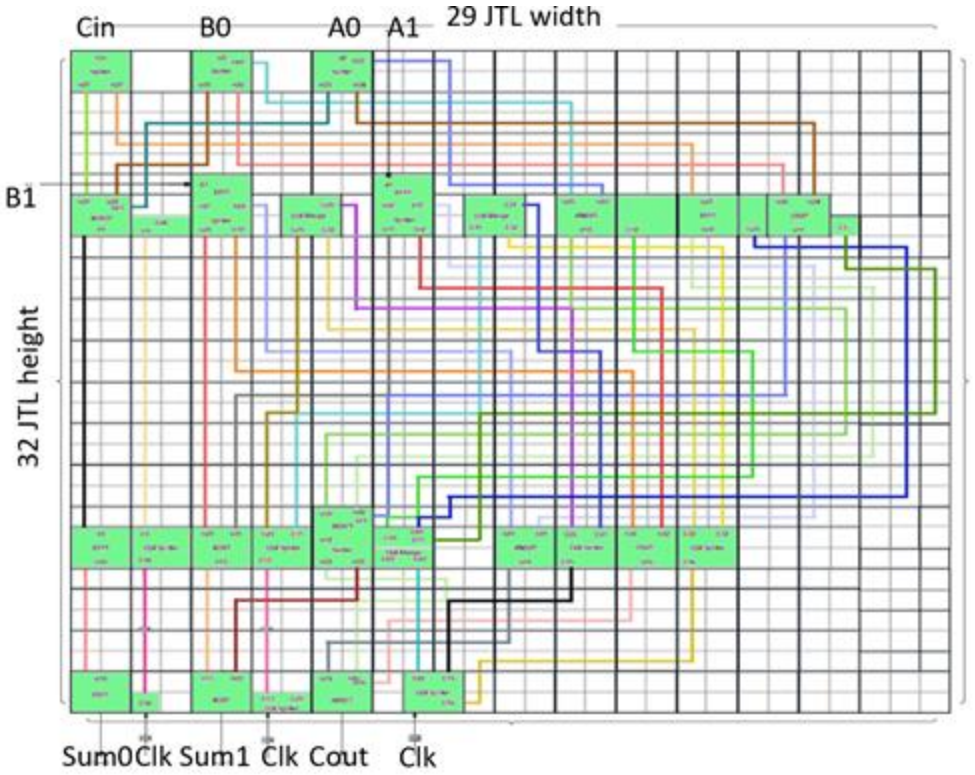
Cell	RSFQ [μm²]	JJ Stack [μm²]	FPL [μm²]
JTL	625	3.5	0.65
DFF	625	12	2.4
Merger	625	10	2.0
Splitter	625	8	1.8
OR	2500	15	2.95
AND	2500	15	2.7
NDRO	2500	15	3.0



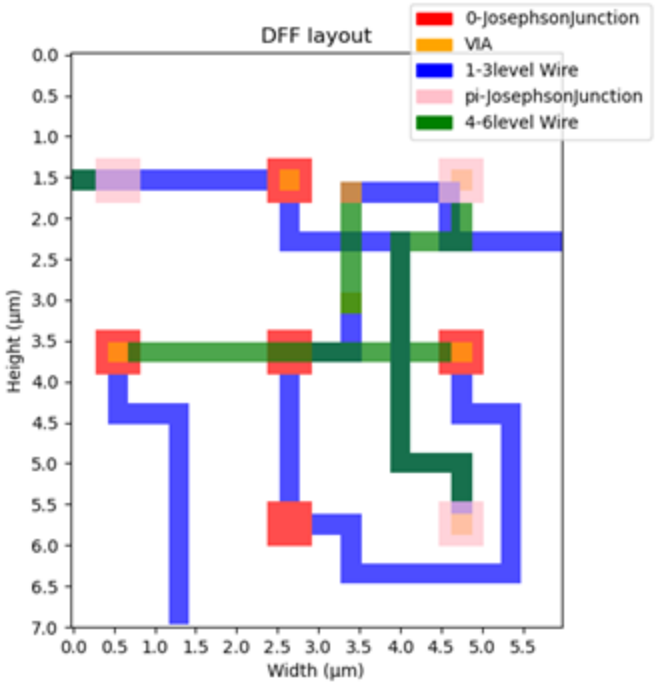
* S. Razmkhah and M. Pedram. "High-density superconductive logic circuits utilizing 0 and π Josephson junctions." Eng. Res. Express, 6(1) 015307, 2024

Automated FPL cell design

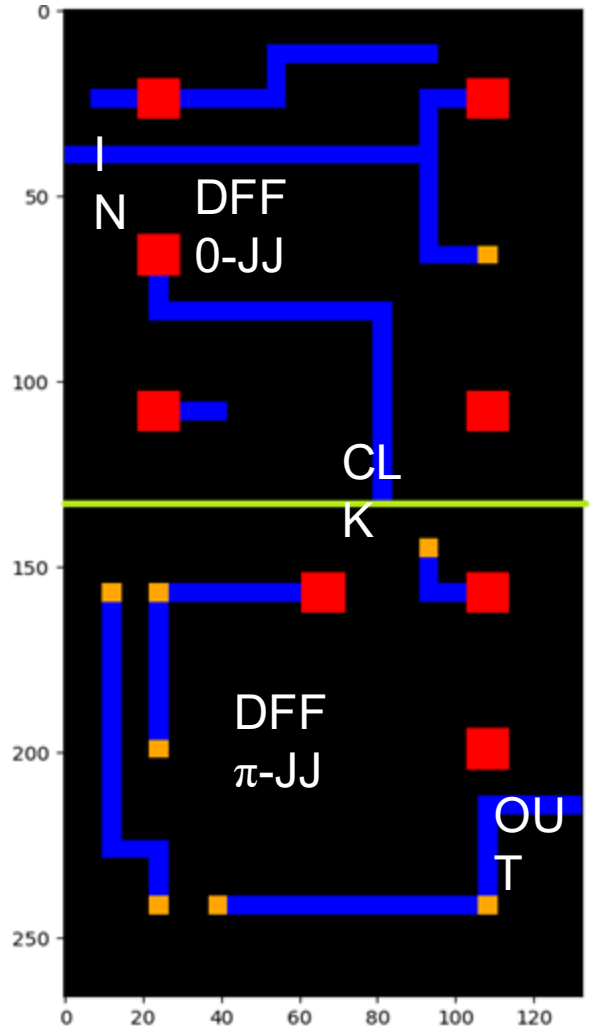
- Using multi-phase and delay line clocking and creating “supercells” to reduce the path balancing overhead.
- JTL router with counter-dataflow clocking, which is better for SFQ logic since it does not require path-balancing DFFs or uniform cell heights.
- No inductors, so no inductance extraction is needed.
- Place and route gates and supercells directly.



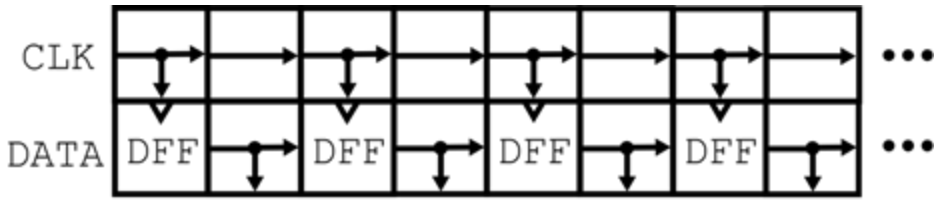
Auto-routed supercell (4-bit KSA) with JTLs



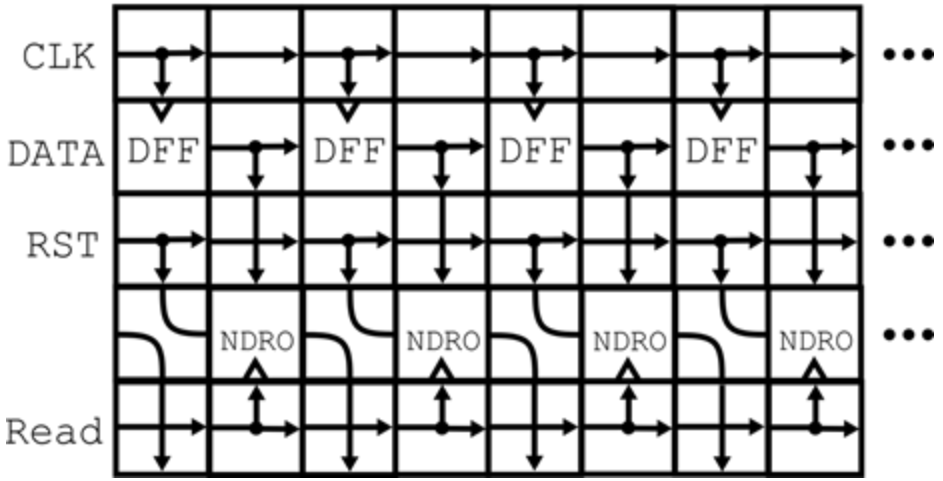
PhaseConnect (ASC 2024)



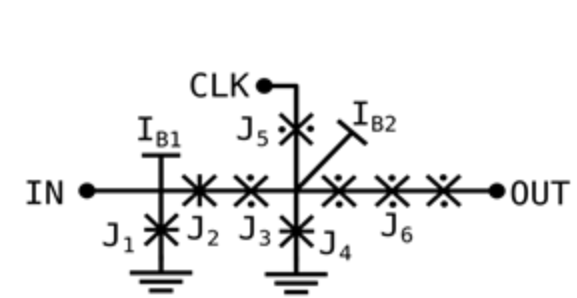
FPL Memory Design

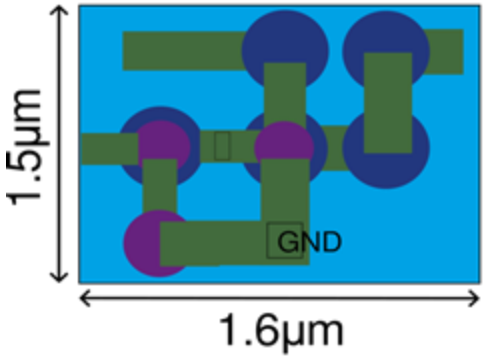


Shift register base memory

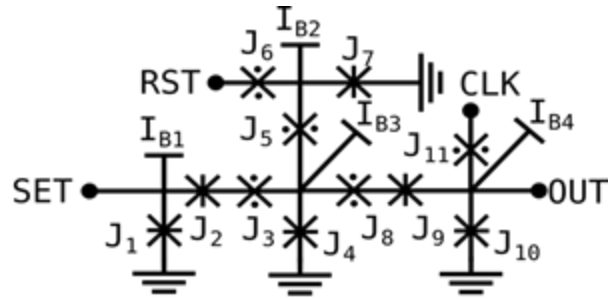


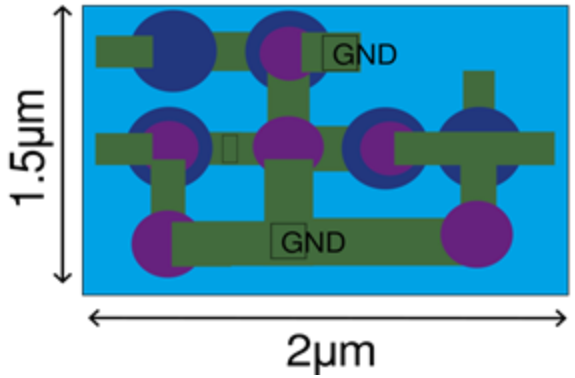
Non-destructive memory





FPL DFF Cell





FPL NDRO Cell

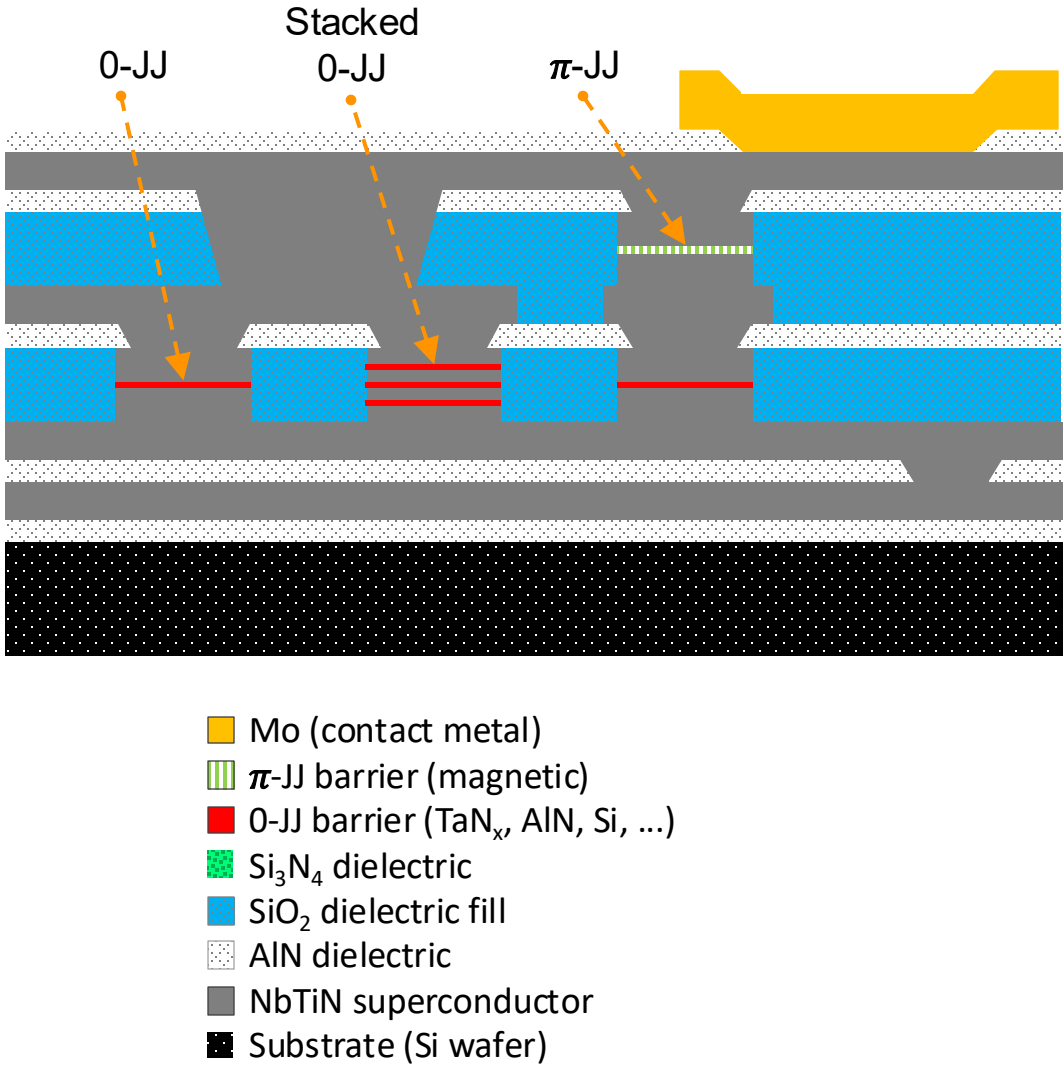
With FPL memory, 64 kb can be placed in 5×5 mm^2

3. Devices and Materials

Materials and Devices: Goals

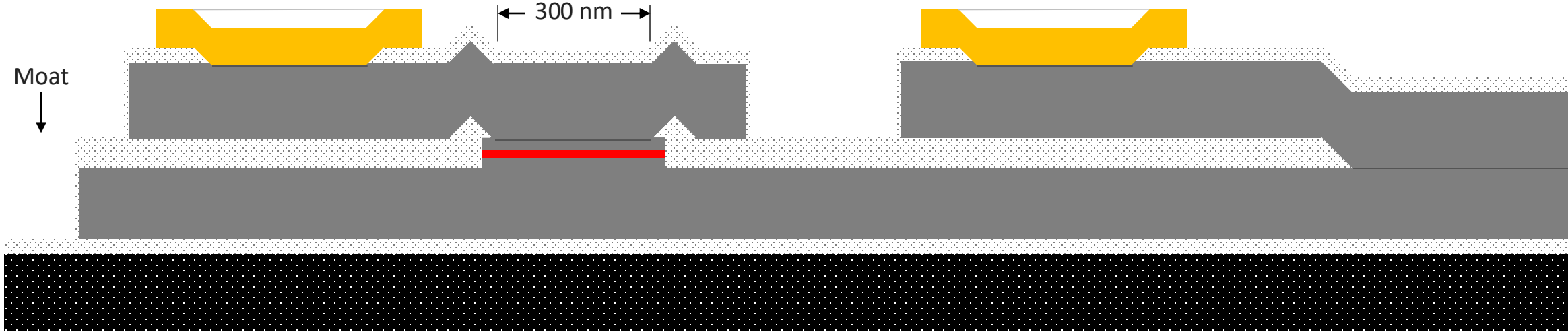
Fabrication stack

- **0-JJs switch** (NbTiN superconductor)
  - J_C ~ 600 μA/μm^2, variation within a few %
  - Compact, self-shunted devices
- **Stacked 0-JJs provide inductance**
  - Non-switching, so J_C is less important
  - 3-JJs per stack is typical
- **π-JJs provide π phase shift**
  - Non-switching, so J_C is less important
  - Resistive barrier provides some damping; possibly enough for SSBF (sustainable superconductive ballistic fluxon) circuit styles
  - Magnetic materials needed



Stackup for initial junction testing (v7)

Junctions with grounded base electrodes, ~ equal thickness junction electrodes, and NbTiN protected from oxygen



- Requirements:
  - Si substrate: high resistivity, no oxide
  - AlN
    - buffer layer on substrate to match NbTiN crystal structure
    - Dielectric insulation between superconductor layers
  - NbTiN superconductor
    - Always over AlN or NbTiN for best crystal structure
    - Thickness > 50 nm for best superconducting properties
  - Junction barrier (TaN_x, AlN, Si, …)
    - Crystal structure compatible with NbTiN
    - Low chemical reactivity with surrounding materials
    - Deposited as a trilayer without breaking vacuum
    - 0-JJ: Semiconductor with uniform properties
    - π-JJ: Ferromagnet, ferrimagnet, or altermagnet
  - Contact metal (Mo)
    - Ohmic contact with NbTiN
    - Oxidation resistant

Stackup process flow (v7)

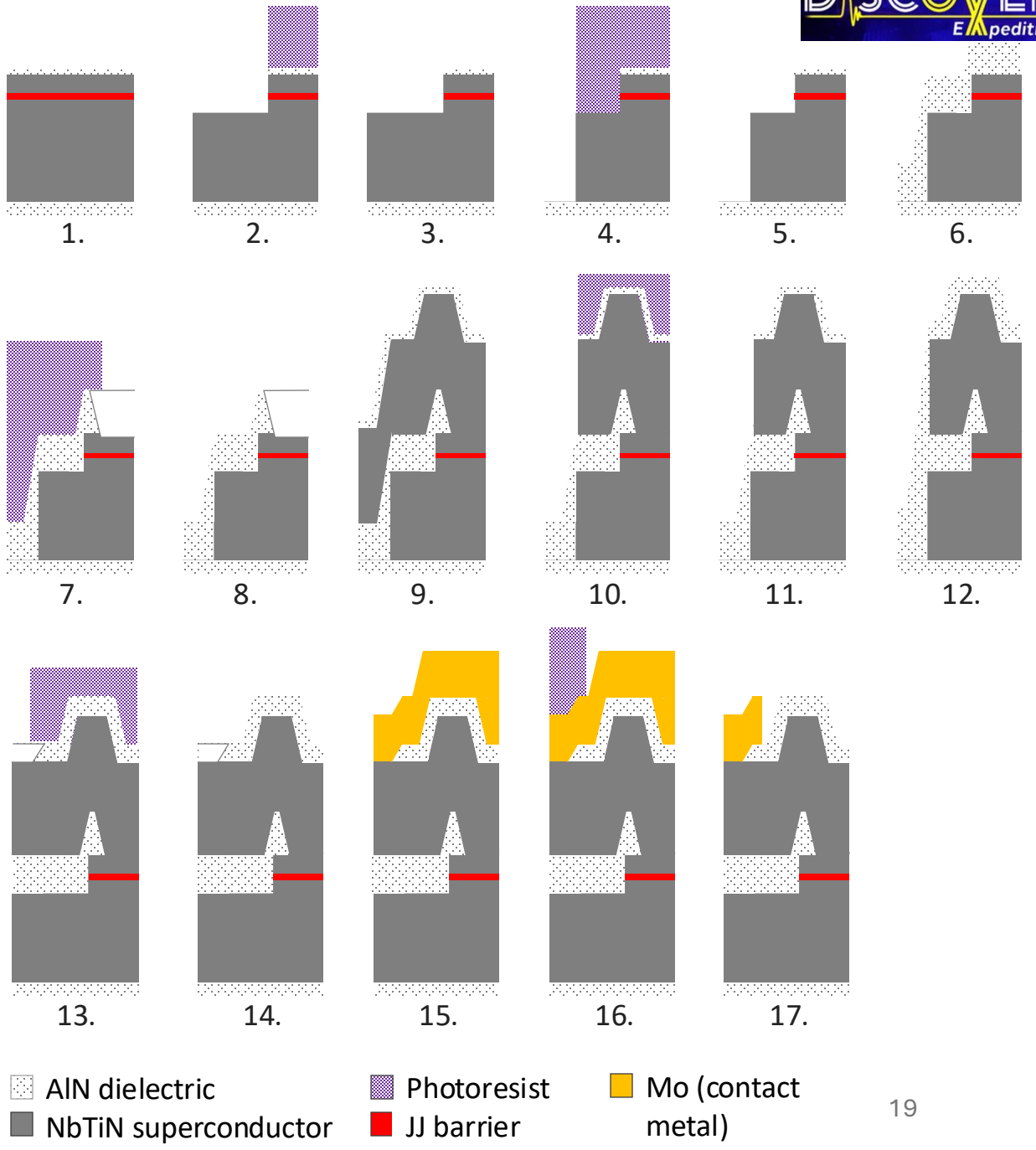
For initial junction testing (cross sections)

Process steps:

1. Junction stack deposition: AlN (30 nm), NbTiN (160 nm), TaNx (~15 nm), NbTiN (25 nm to allow 5 nm for CMP loss), AlN (10 nm)
2. Litho+Etch 1 to cut through AlN, NbTiN + barrier layers (timed stop after barrier material clears) [c1]
3. Remove photoresist
4. Litho+Etch 2 to cut through NbTiN base layer (stop on lower AlN)
5. Remove photoresist [c2]
6. AlN (60 nm) dielectric protection layer [c3]
7. Litho+Etch 3 to open sloped contact holes in AlN [c4], [c5]
8. Remove photoresist [c2]
9. NbTiN (150 nm) wiring layer deposition + AlN (10 nm) protection
10. Litho+Etch 4 to pattern wiring layer
11. Remove photoresist [c2]
12. AlN (20 nm) deposition
13. Litho+Etch 5 to open sloped contact holes in AlN [c4], [c5]
14. Remove photoresist [c2]
15. Mo (75 nm) deposition for contact pads
16. Litho+Etch 6 to pattern pads
17. Remove photoresist

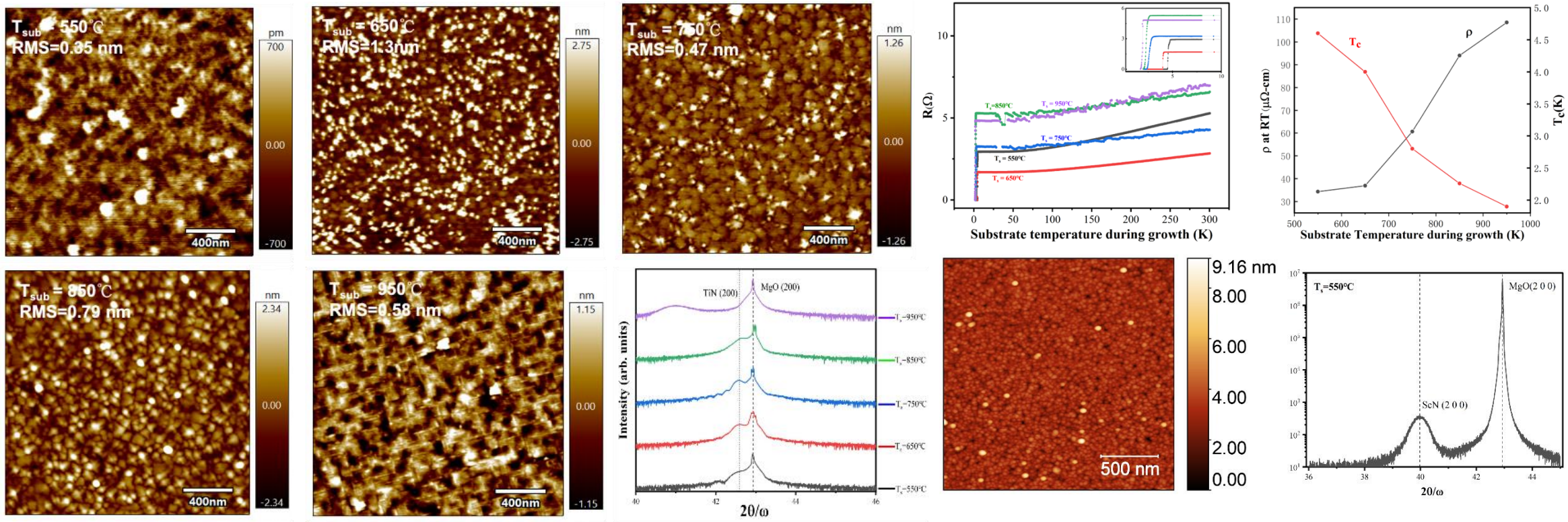
Challenges:

[c1] Trilayer etch with vertical walls and without shorting barrier
[c2] Damage to exposed NbTiN before the next step
[c3] AlN fill around the junction
[c4] Mask alignment for contact holes through AlN (use UV?)
[c5] Etching AlN contact holes with sloped sides and stop on NbTiN



JJ materials exploration

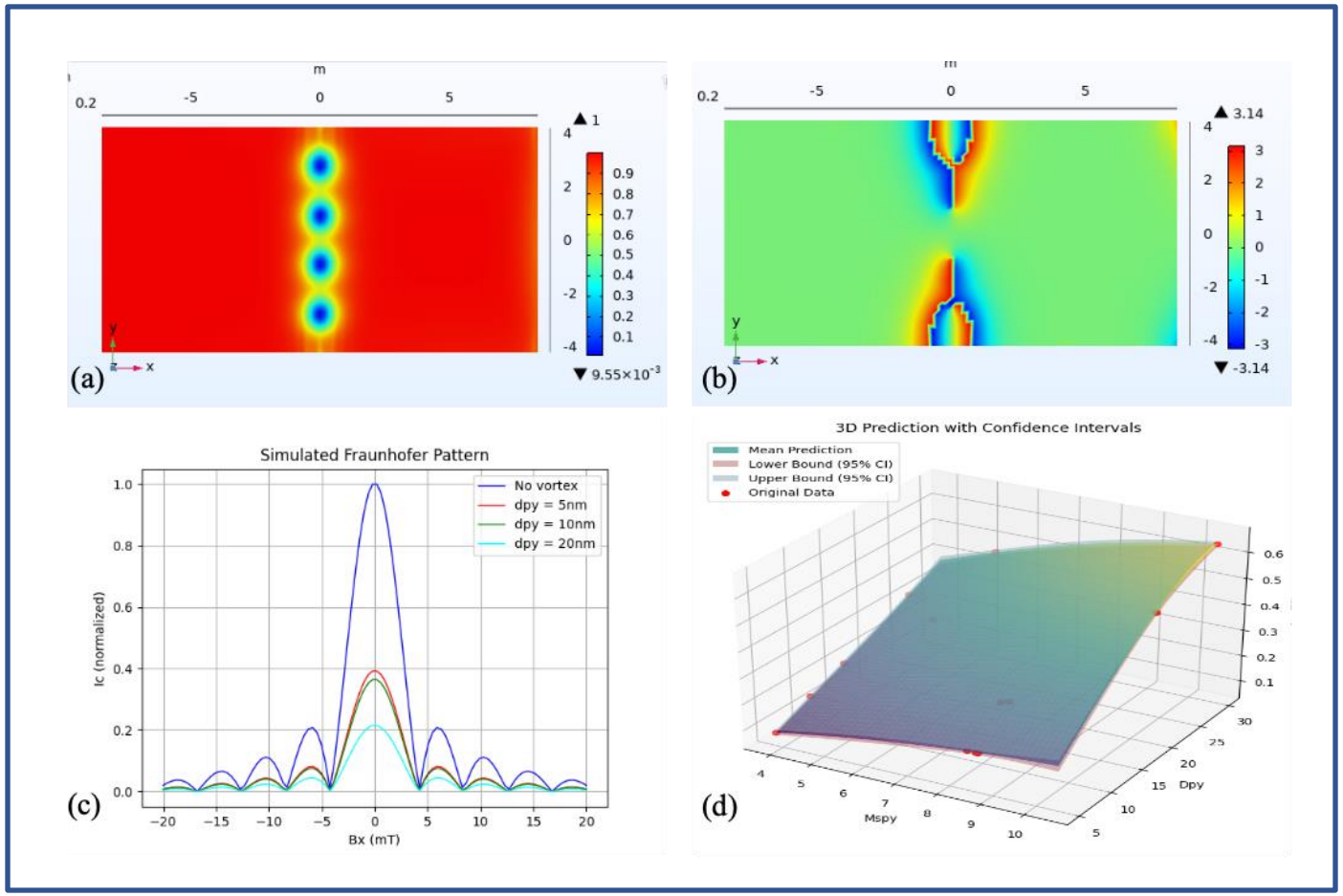
0-JJ superconductor electrode and barrier layer growth by molecular beam epitaxy (MBE)



- Achieved high-quality epitaxial TiN superconducting electrode deposition on MgO (001)
- Explored rock-salt ScN growth on MgO for 0-JJ barrier

Magnetic JJ materials and memory devices

- Magnetic junctions
  - Nb/Ni/Nb
  - NbN/PdNi/NbN
- Device simulation
  - Realized accurate simulation of Gen-3 S1IS2FS3 device phase modulation and magnetic coupling using COMSOL
  - Constructed a Gaussian Process Regression model to optimize the design of JJs



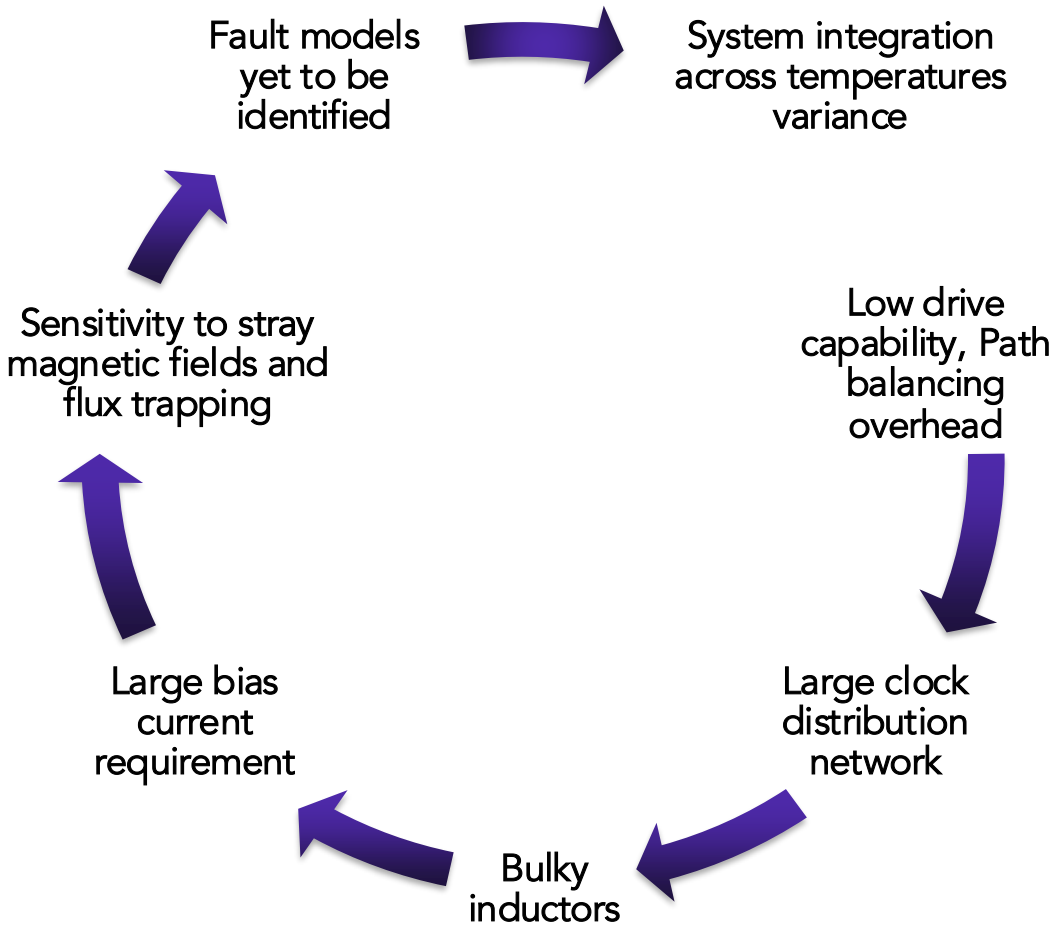
4. Integration and Interfaces

5. Environmental and Economic Impacts

2. Circuits and Architectures (including EDA tools)

EDA tool development

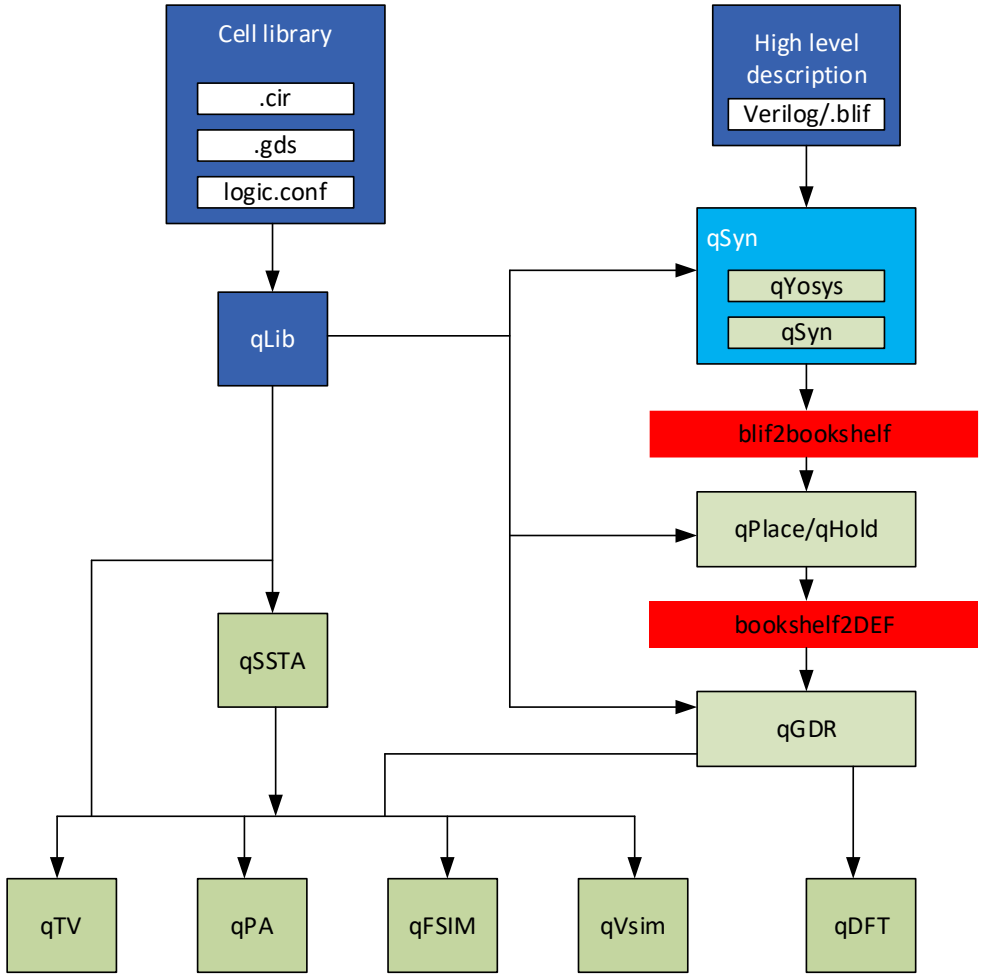
Challenges of SFQ Logic



- Two-terminal JJ's and inductors vs. 3-terminal transistors and capacitors in CMOS
- Josephson transmission lines (JTLs) and passive transmission lines with small series resistors to prevent flux storage (PTLs) vs. lossy interconnect in CMOS
- Clocked logic cells (NOT, AND, OR, XOR, DFF, NDRO) and some clockless cells (delay cells, splitters)
- Low fanout drive capability
- Gate-level pipelined circuits with full path balancing requirement (due to destructive readout)
- Small number of PTL routing layers
- Bulky inductors and need for shunt resistances
- Large bias current requirement
- Sensitivity to stray electromagnetic fields and flux trapping

qPalace v1

qLib	Generating the required formats from the input technology library provided by manufacturer through translation and simulation
qSyn / qYosys	Parsing high level Verilog/BLIF descriptions and behavioral synthesis
qSyn / qABC	SFQ specific logic synthesis, mapping and verification
converters	blif2bookshelf: converting the BLIF format to the bookshelf, and bookshelf2def: converting the bookshelf format to the DEF format.
qPlace	Placement and clock-tree synthesis
qGDR	Global and detailed routing
qSSTA	Statistical static timing analysis
qVSim	Post-routing simulation
qHold	Fixing hold time violations considering process variation
qTV	Validation for post routing netlists
qPA	Power analysis
qFSIM	Fault simulation and test pattern generation
qDFT	Fast fault simulation for BIST performance evaluation



https://coldflux.usc.edu/

Input: high-level design (Verilog/BLIF), cell library (cell layout (.GDS), cell SPICE netlist (.CIR), cell configuration (.conf))
Output: SFQ layout description file **(.DEF)**, circuit analysis report (timing, power, test pattern and fault coverage).

Some practical design issues

- Post place & route clock frequency is much lower than the post-synthesis clock frequency; design may not fit into a single chip
- Using simple logic cells increases the circuit depth hence increases the overall latency and path balancing overhead; 1:2 splitters add to stage delay in situations where the fanout count is high
- Design verification is costly and limited
- Row-based placement wastes large amount of area
- PTLs in certain length ranges suffer from severe reflection at high clock frequencies.

Solution: Integrated circuit partitioning and floor planning

Solution: Multi-chip design integration

Solution: Complex cell library containing multi-stage cells and high fanout splitters

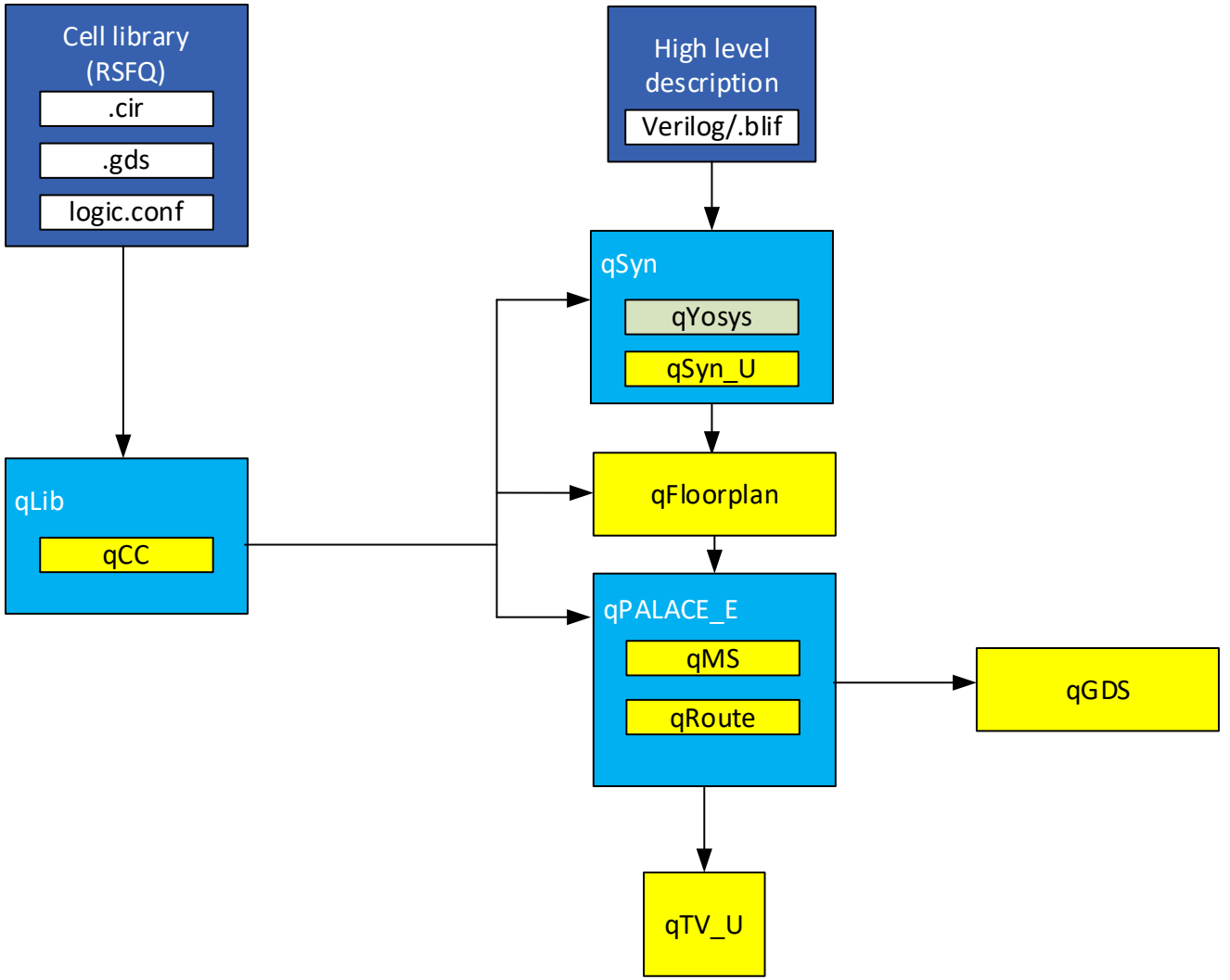
Solution: Slow/fast dual clocks and dual-phase clocks

Solution: Mixed-mode simulation and hierarchical full design verification

Solution: Design flow and tools supporting an elastic (on-the-fly-generated) cells and macrocells

EDA Tools: qPALACE v2

qTool	Short Description
qSYN_U	Circuit partitioning and synthesize each part with **support for dual phase clocking**
qFloorplan	Floor planning and levelized placement followed by **hold violation fixes**
qMS	Macrocell layout synthesis
qPALACE_E	**Hierarchical hybrid routing** (JTL & PTL)
qTV_U	Full-design timing analysis and verification on hierarchical designs



Input: high-level design (Verilog/BLIF), cell library (cell layout (.GDS), cell SPICE netlist (.CIR), cell configuration (.conf))
Output: SFQ layout description file (.GDS), circuit analysis report (timing, power, test pattern and fault coverage).

RSFQ Cell Library

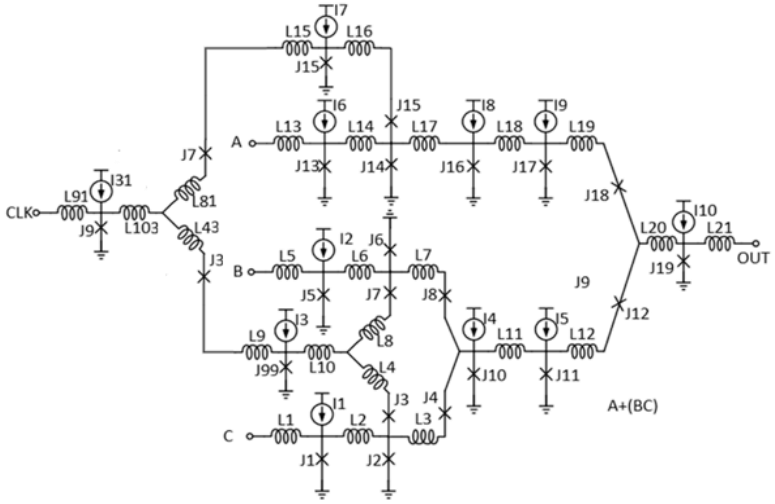
Developed a new cell library, named SportLib. The library includes

- Synchronous and asynchronous-reset logic cells
- Fixed, pre-characterized cells, and on-the-fly synthesized and characterized cells
- Input-synchronous compound cells, comprising multiple logic stages with internal asynchronous data flow
- AC and DC biasing

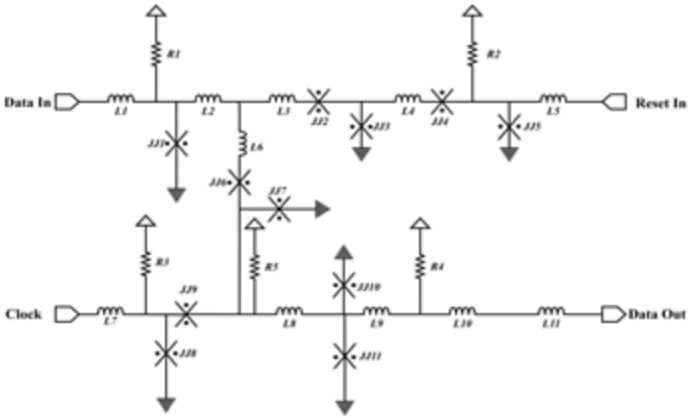
Each cell is validated with post-tapeout measurements

- High critical margins, typically more than +/- 20%
- Cell variants with and without integrated PTL receivers and drivers (to enable both PTL and JTL routing)
- Cell variants with delayed output signal or clock signal (for hold violation fix)

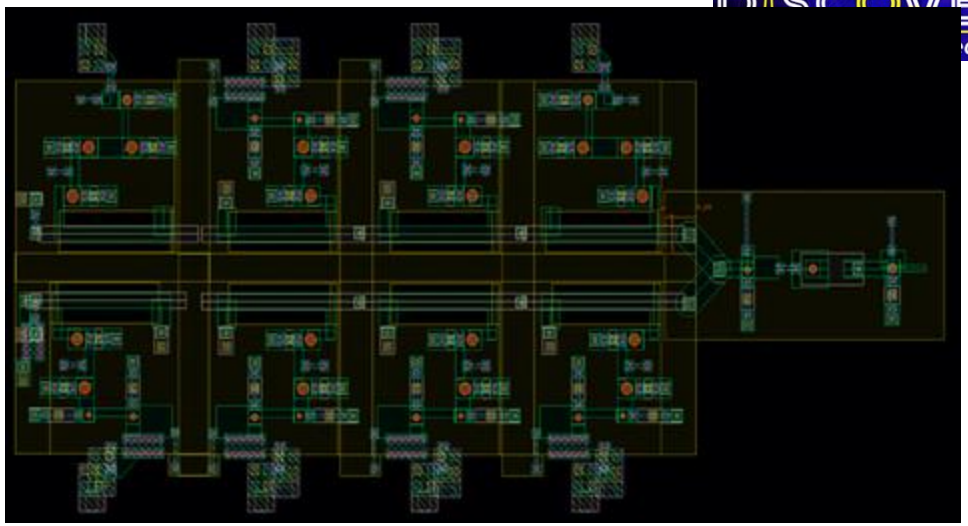
Compound Cells



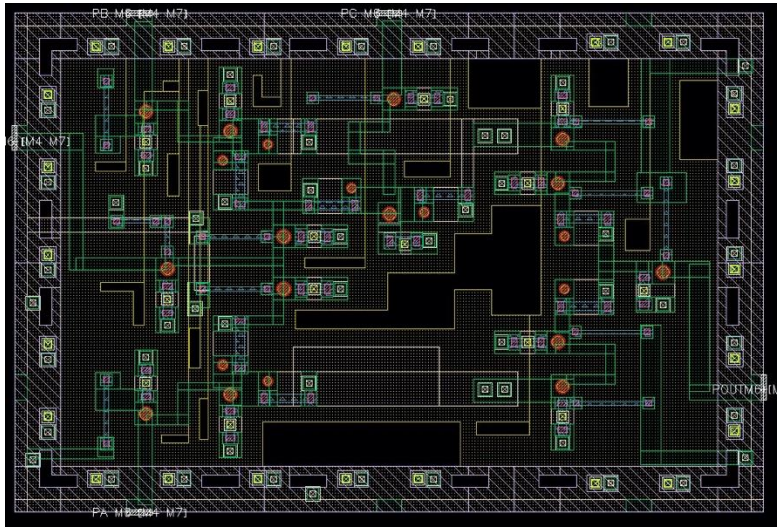
The A+BC logic cell schematic



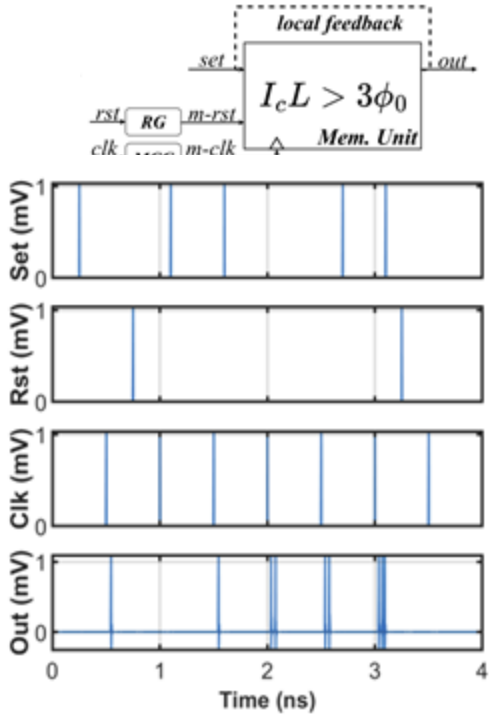
Multi-fluxon NDRO



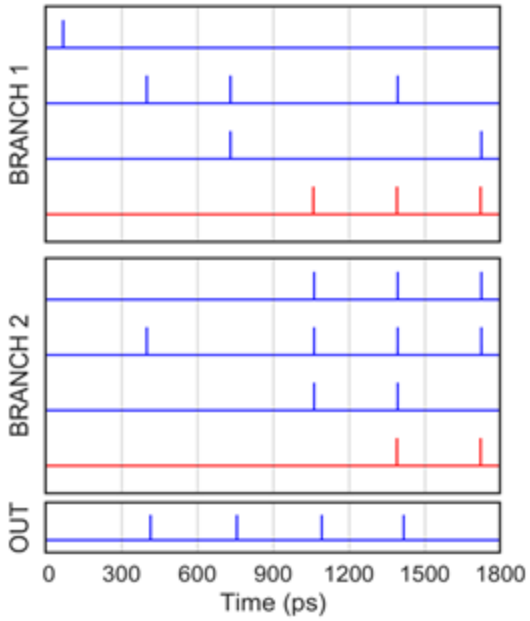
Layout of Neuron w/ 6 positive & 2 negative inputs



Layout of the A+BC logic cell



Simulation of the MF-NDRO

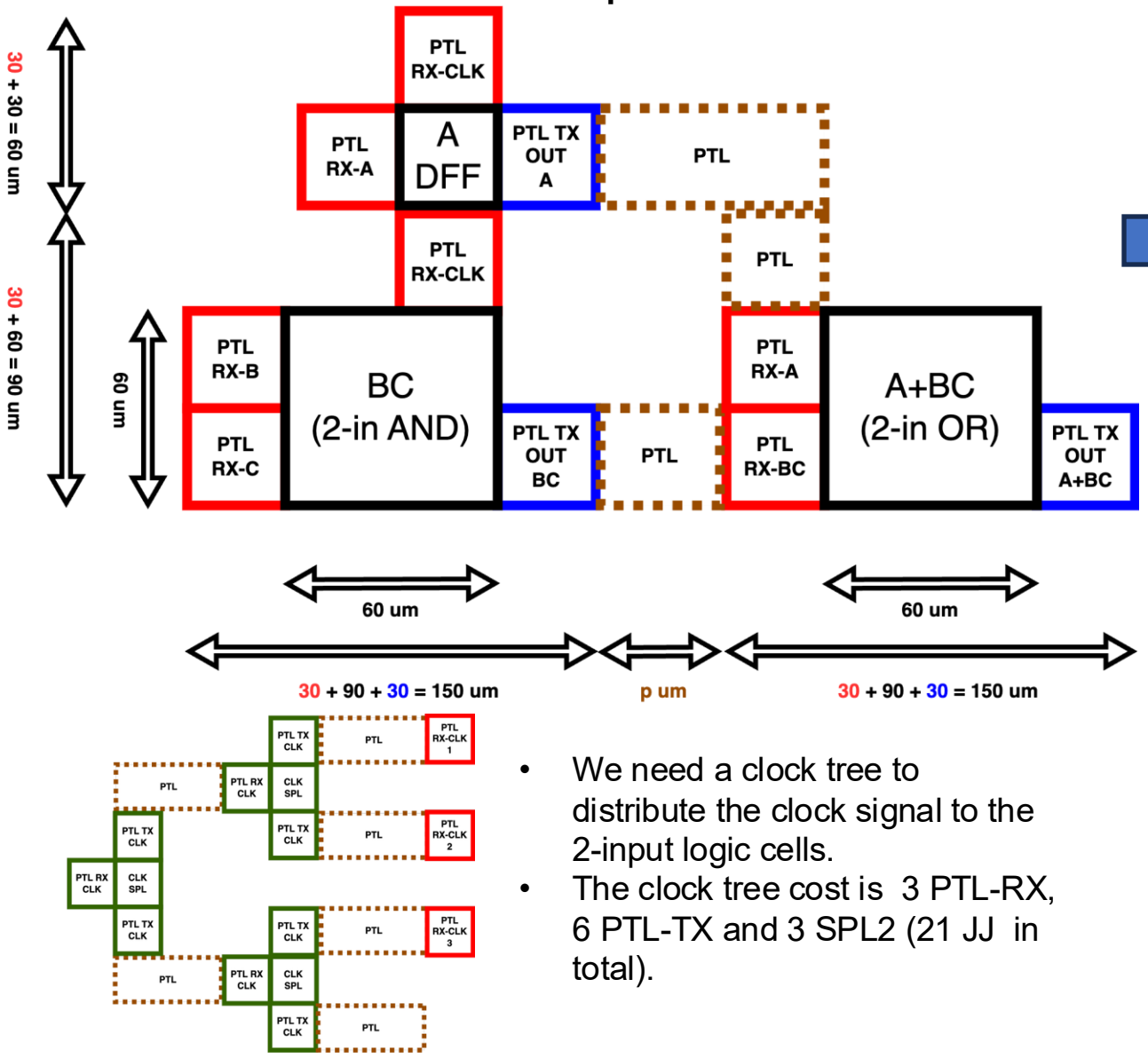


Simulation of the neuron circuit

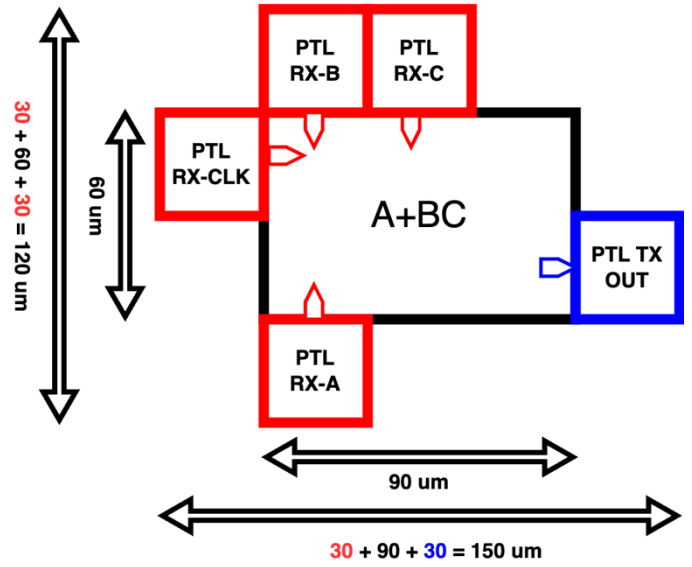
Cell Type	Dela(ps)	Margin	JJ Count
(A+B)(C+D)	7	-	22
AB+CD	12.57	-	26
A+BC	9.9	-	21
(A+B)C	6.9	-	15
7:3 Compress	22	-	36
NEURON	10	-5,+7	1
MF-NDRO	7.1	-11,+10	23

Compound Cell Benefits

A+BC realized with 2-input cells



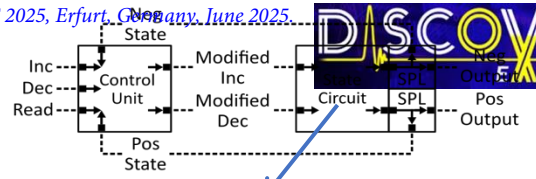
A+BC realized as a compound cell



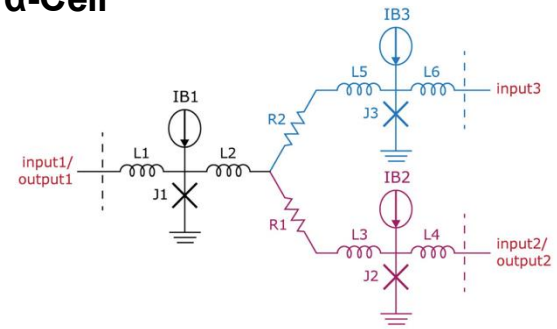
- We need a clock tree to distribute the clock signal to the 2-input logic cells.
- The clock tree cost is 3 PTL-RX, 6 PTL-TX and 3 SPL2 (21 JJ in total).

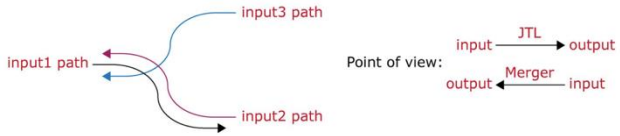
	A+BC Compound Cell	A+BC (Design with 2-in Gates)
JJ Count (Functional)	21	23
JJ Count (Driver/Receiver)	11	17
Area (Functional)	90 µm x 60 µm	30 µm x 30 µm + 60 µm x 60 µm + 60 µm x 60 µm
Area (Total)	120 µm x 150 µm	150 µm x 300 µm*
Delay (ps)	9.9	15.8

Cells with Asynchronous Reset

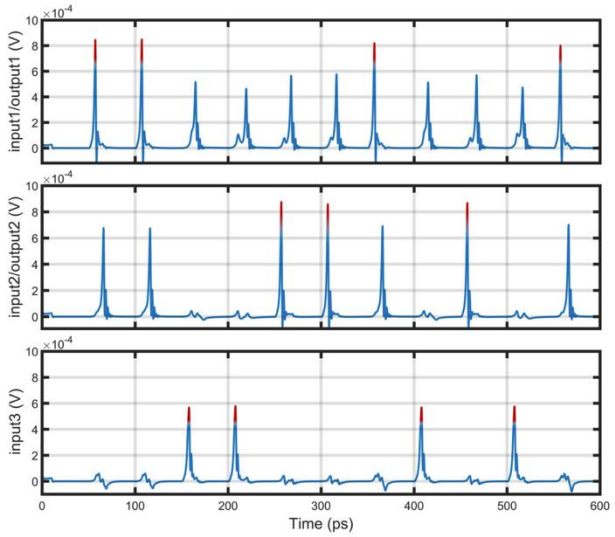


α-Cell

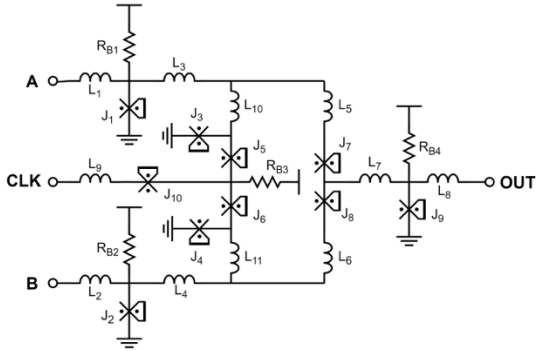




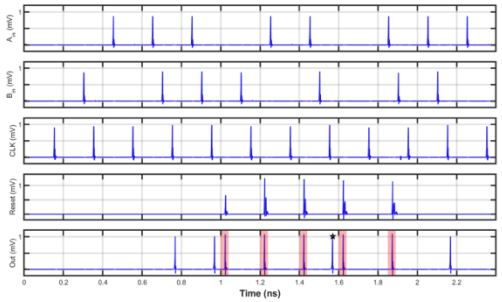
α-cell schematic



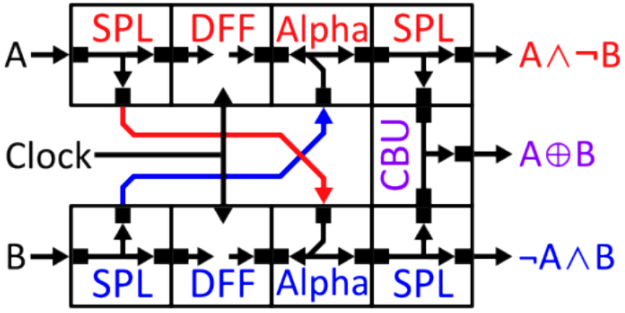
α-cell simulation results



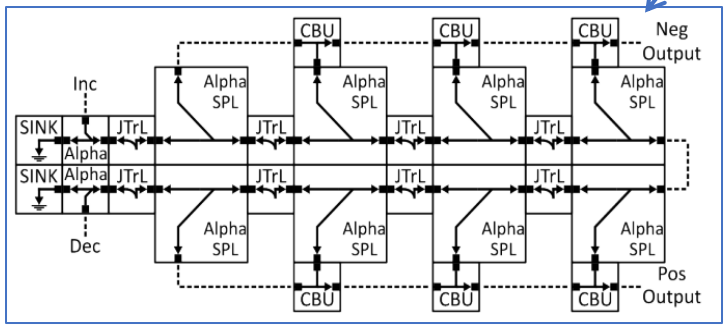
Alpha compatible AND gate schematic



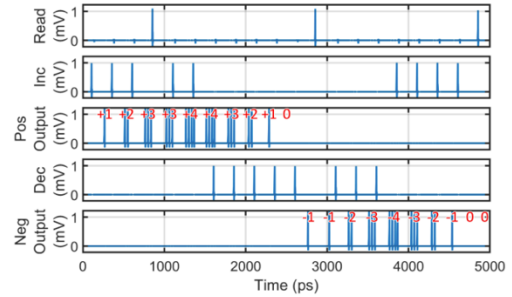
Alpha compatible AND gate simulation



Compound Cell Design with alpha cells



Up-Down Counter design with α-cells

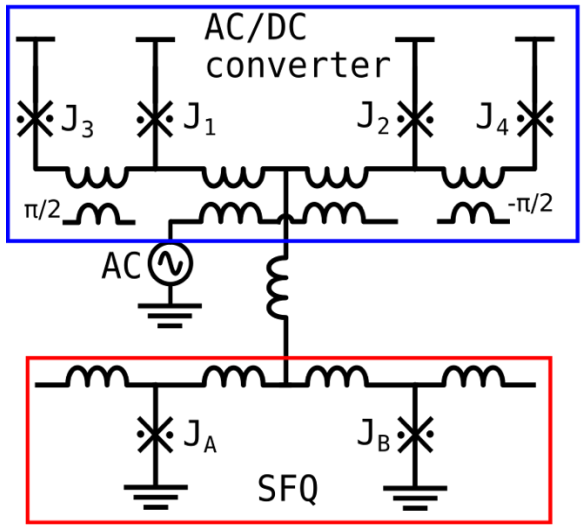


Up down counter simulation

AR cells	Margin	JJ count
AND_R	-26, +22	9
OR_R	-29, +50	9
XOR_R	-26, +35	9
DFF_R	-37, +37	4
M-DRO_R	-11, +10	11
TFFR_R	-37, +37	11

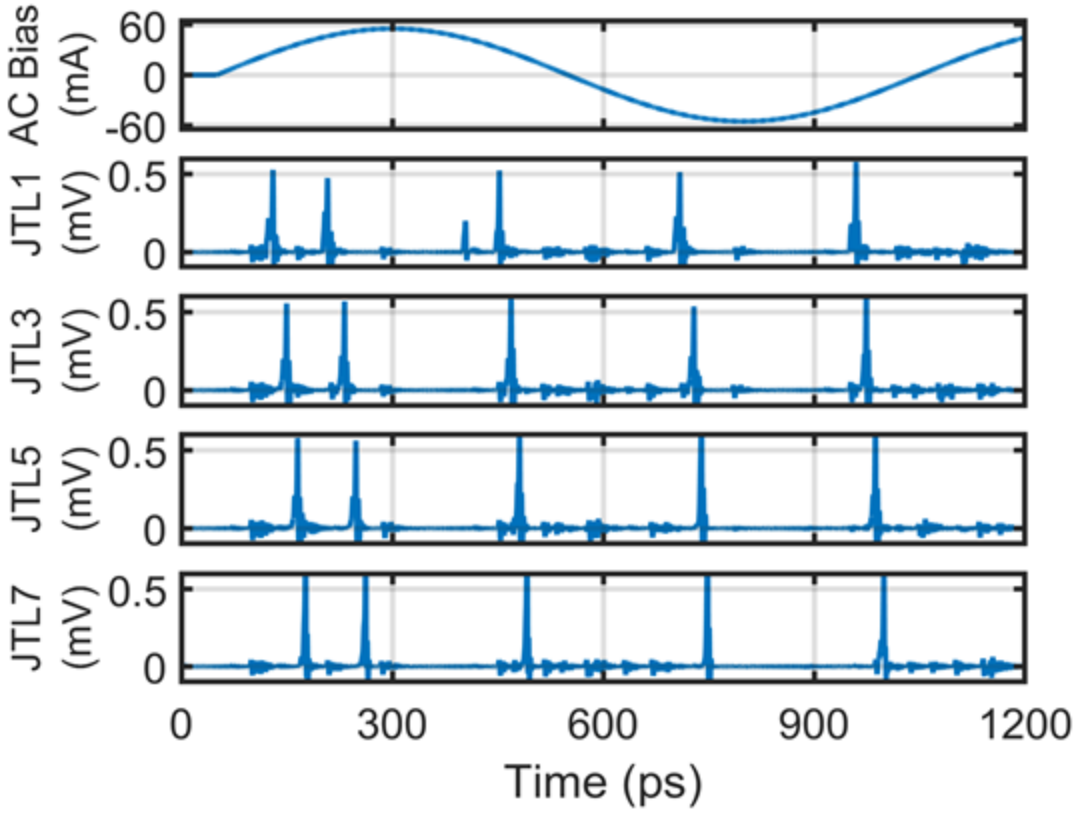
AC power

- Eliminates static power consumption
- Simplifies bias delivery to the cold stage logic
- Requires little change to the cell library design by replacing the resistive bias network with an AC/DC converter



AC Biased JTL Schematic using AC/DC Converter

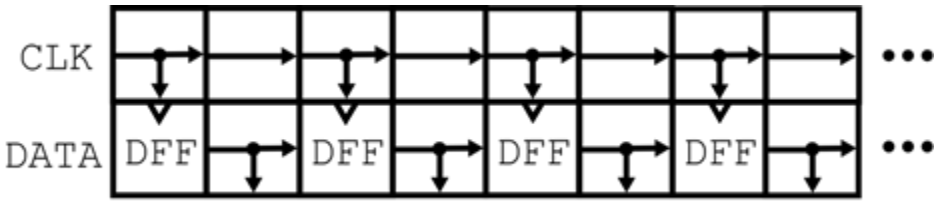
Simulation Result for 8-stage JTL Chain in SportLib with AC power.



Semenov+, “SFQ bias for SFQ digital circuits,” *IEEE Trans. Appl. Supercond.*, 2021, doi: 10.1109/TASC.2021.3067231.

Cost-Benefit of AC vs DC Power

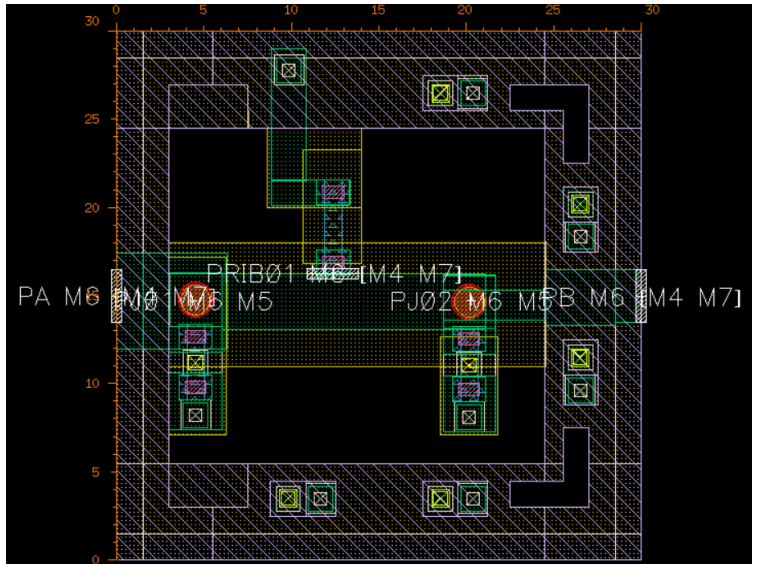
64-bit Shift Register Memory



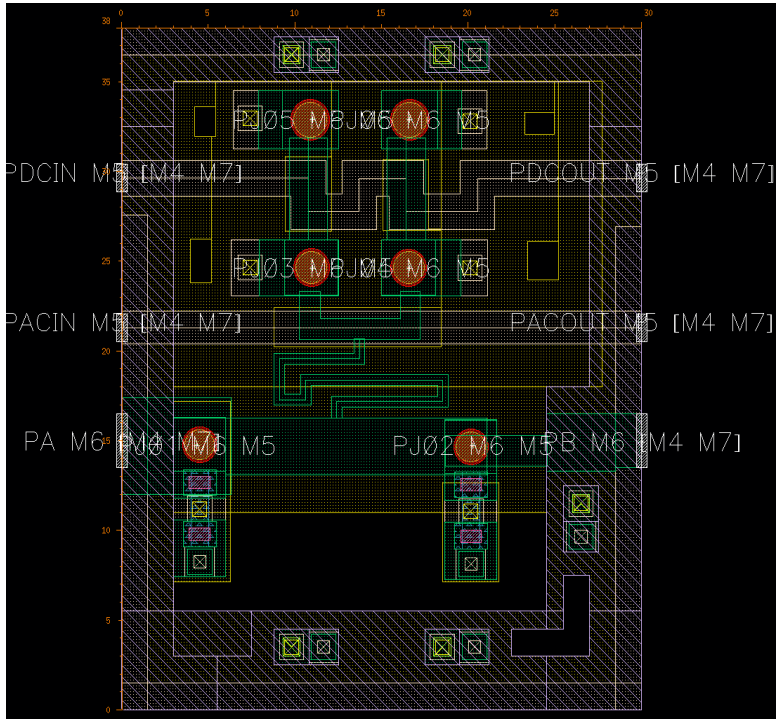
Comparison between AC and DC biasing methods

Bias	Static Power	Dynamic Power (@ 20 GHz)	JJ Count	Area	Maximum Frequency
DC	358.4 μW	1.02 μW	768	0.23 mm^2	60 GHz
AC	0	2.05 μW	1792	0.29 mm^2	20 GHz

DC resistively-powered cell (30 x 30 μm^2)



AC-powered cell (30 x 38 μm^2)



Partitioning and Floorplaning

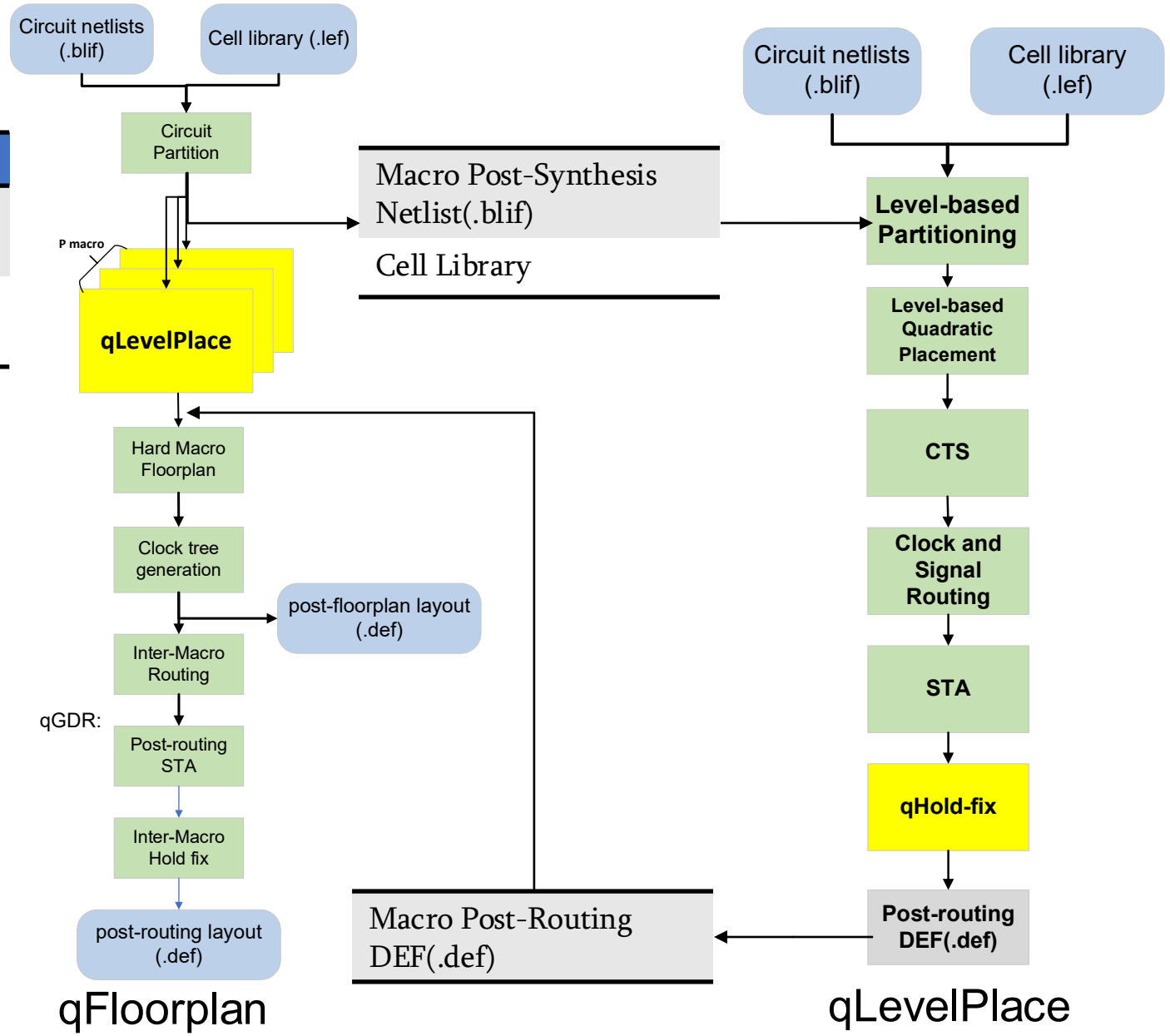
- Motivation:
  - Placement & routing of a large circuit can severely reduce the post-synthesis clock frequency due to the delay of long interconnects
    - Pipelining long interconnects can be very expensive due to the need to rebalance the circuit after such an operation
    - Partitioning the circuit into smaller modules, floorplaning the module set, doing pseudo-pin assignment on module boundaries, and pipelining inter-module interconnects is a solution
- Problem statement:
  - For a large circuit, generate a circuit partitioning and floorplaning solution that meets a target post-layout clock frequency (close to the post-synthesis result)

qFloorplan and qLevelPlace

qTool	Short Description
qLevelPlace	Perform Level-based placement
qHold-fix	Fix hold by inserting JTL delays after routing

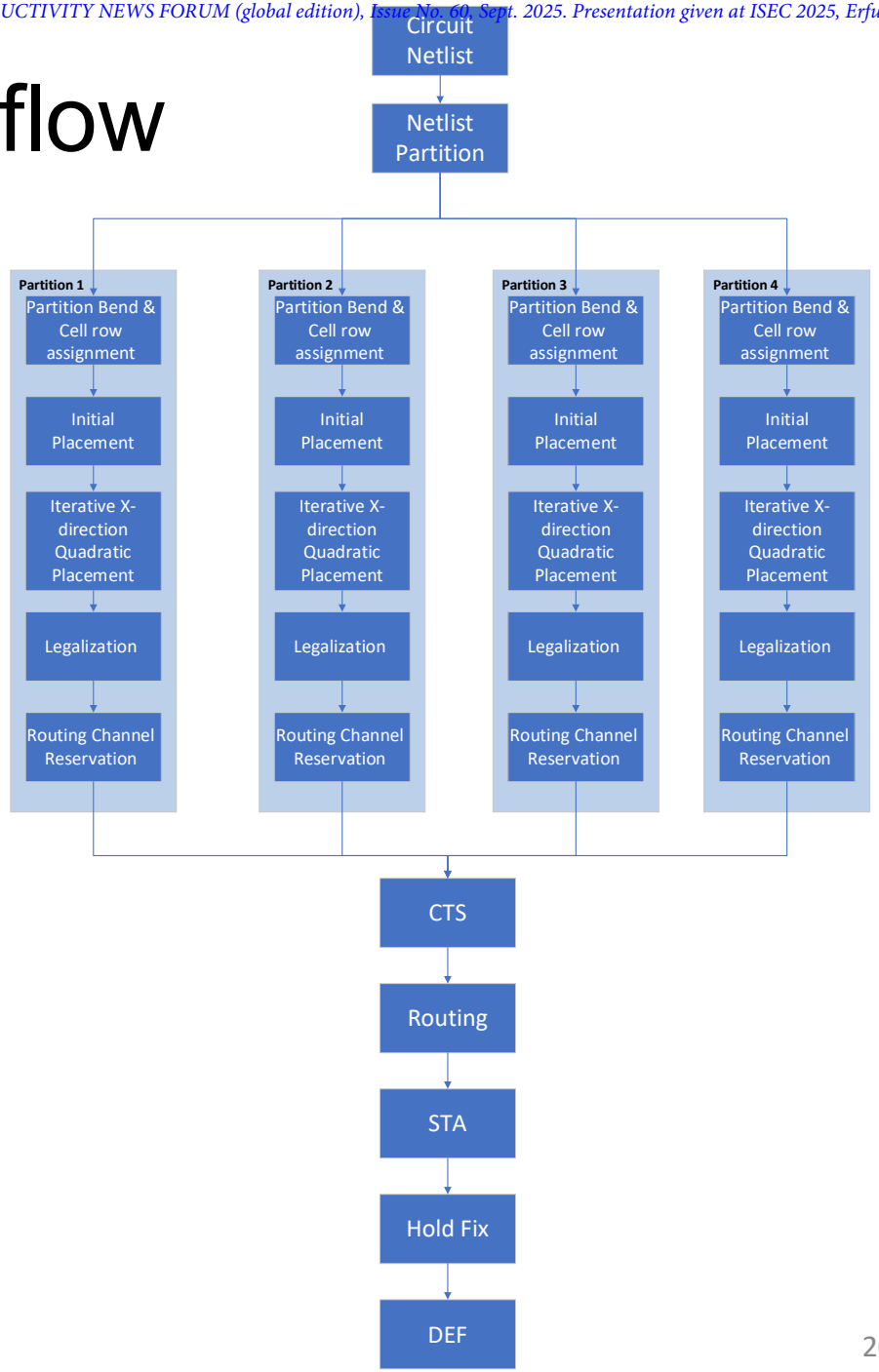
qFloorplan:
Input: Post-synthesis netlists(.blif), Cell library(.lef)
Output: Post-routing DEF(.def), timing report

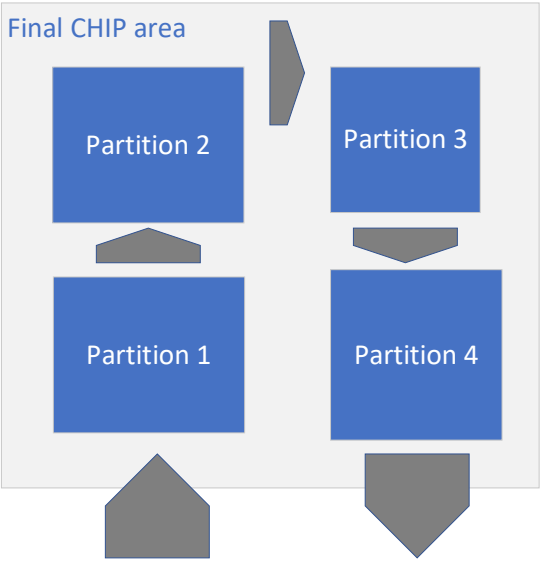
qLevelPlace:
Input: Post-synthesis netlist(.blif), Cell library(.lef)
Output: Post-routing DEF(.def), timing report



Floorplan automated flow

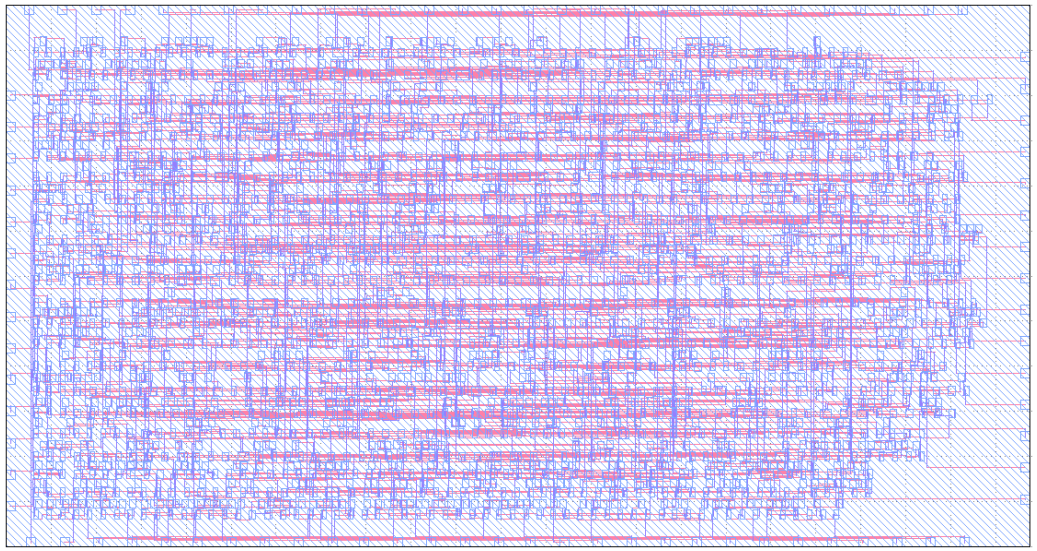
1. Partition the large netlist into 4 parts.
2. Perform placement independently for each partition.
3. After completing all partitions, arrange their locations based on a fixed partition arrangement.
4. Take the final placement and perform Clock Tree Synthesis (H-Tree Generation) to form a global clock tree.
    1. Generate the H-tree topology via the MMM (Method of mean & median) algorithm.
    2. Construct an H-tree position using the BST-DME (Bounded-Skew Clock Tree Router).
5. Perform routing in a single step.
6. After routing, run post-layout STA.
7. Hold time fix by JTL delay insertion

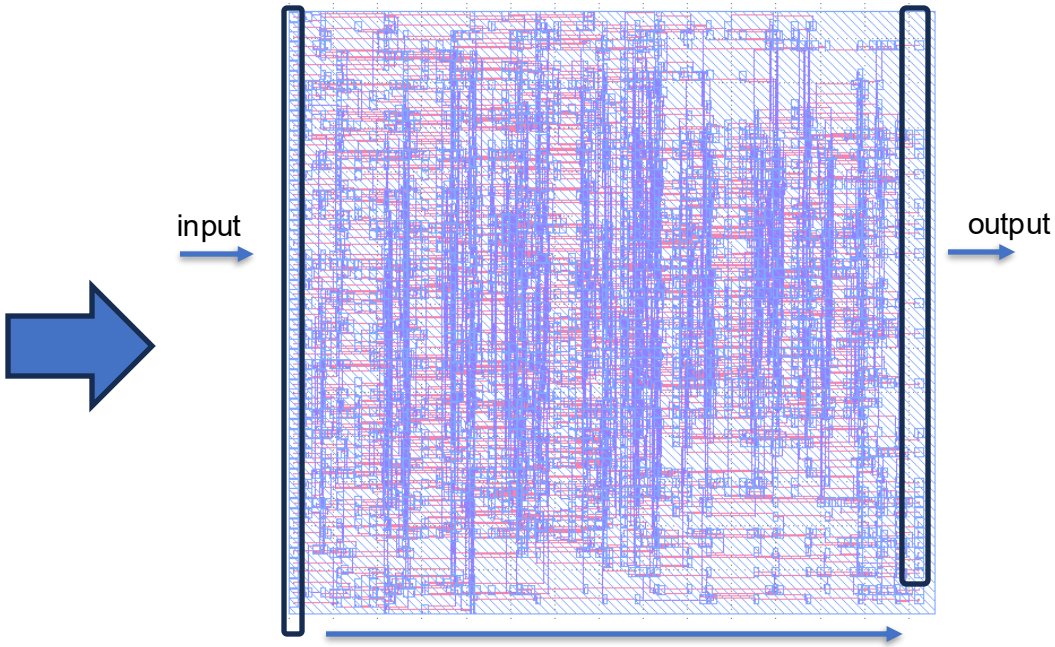




qLevelPlace (Levelized Placement) Tool

C880 – 8-bit ALU Place and Route example





qPALACE v1 result (qPlace)

Frequency (GHz)	14.5
Clock skew (ps)	8.45
Size (μm²)	4700 * 2490

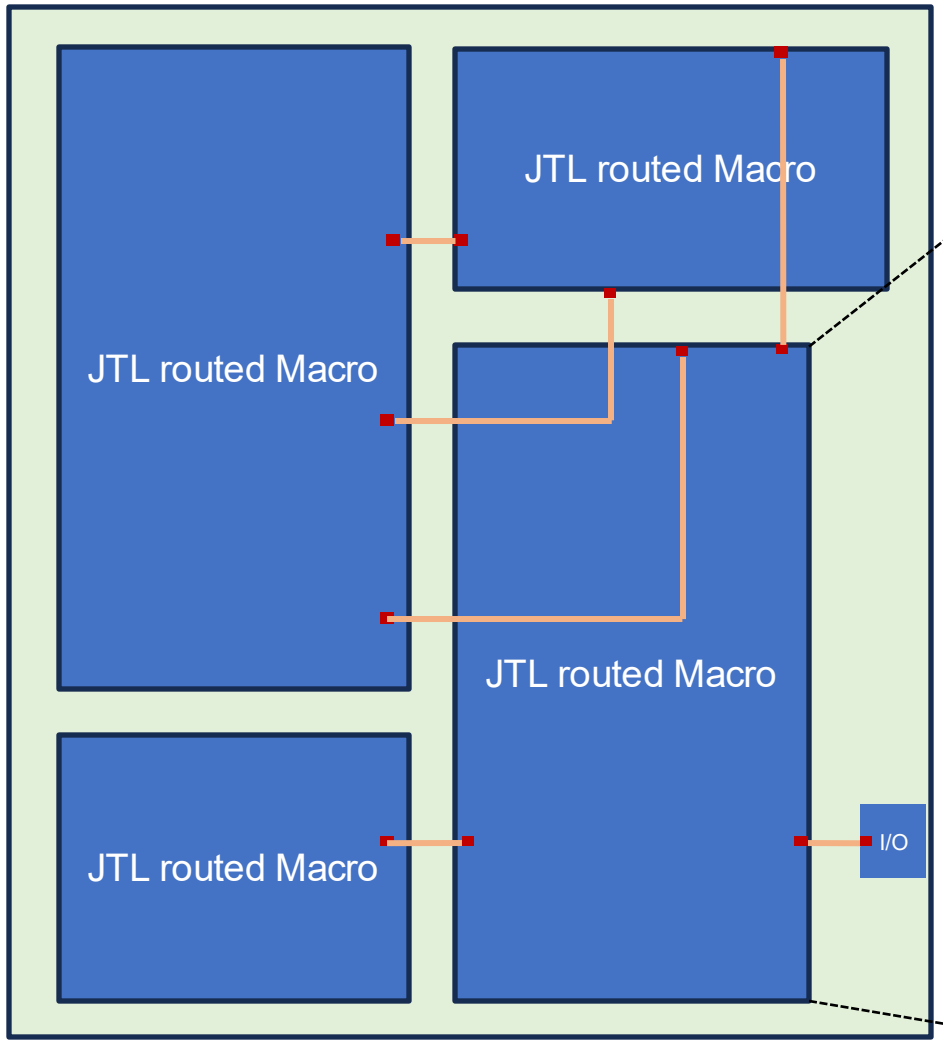
qPALACE v2 (qLevelPlace)

Frequency(GHz)	15.8
Hold time violation	0
Size (μm²)	2920 * 2725

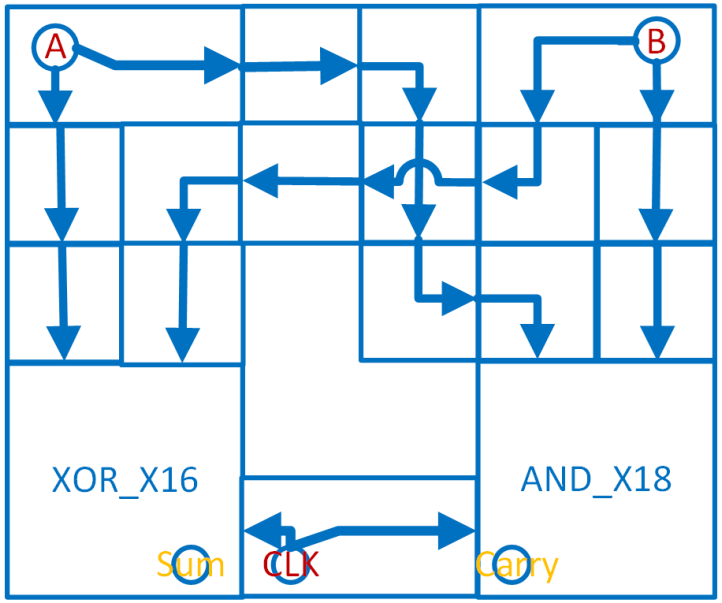
Hierarchical JTL/PTL Routing

PTL:

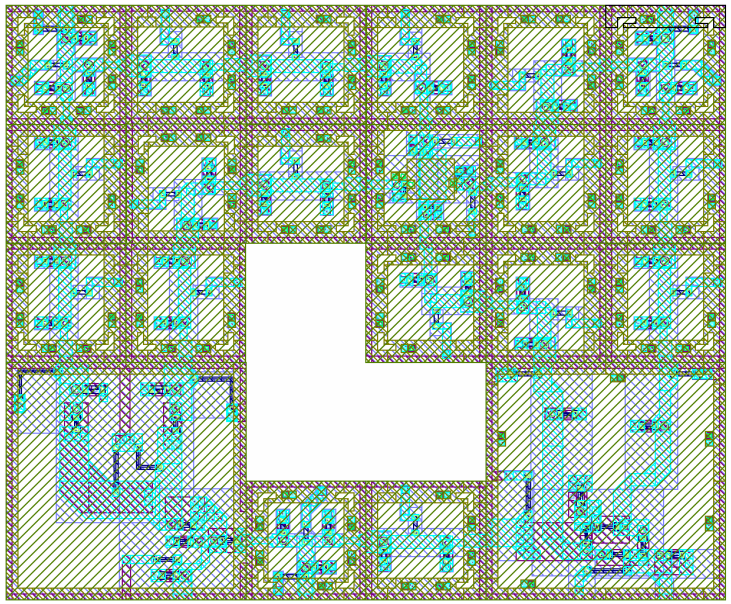
JTL (different types): …



JTL Routed Macro: 1-bit half adder



Schematic view



Layout view

Results of Level-based Quadratic Placement

Testbench circuit	Circuit size cells w/ clock splitters	qPALACE v2 level-based placer			qPALACE v1		
		Post-placement Area (μm * μm)	HPWL estimation (Routing Channel Added)	**Connection, longest (μm)**	Post-placement Area (μm * μm)	HPWL estimation (w/ Routing Channel)	**Connection, longest (μm)**
KSA32	959	2950 * 2400	490,541	**1299**	4920 * 2620	1,129,326	**3310**
c1355	978	1870 * 2375	362,032	**1086**	3920 * 1940	484,846	**3470**
c499	990	1865 * 2475	369,649	**971**	4080 * 2000	542,728	**3170**
8-bit ALU	1399	2725 * 4055	609,147	**1312**	4550 * 2400	744,036	**3540**
c432	1506	1770 * 4135	679,183	**1158**	4700 * 2490	821,303	**3670**
c3540	3599	5405 * 5275	2,317,652	**3632**	7570 * 4170	2,041,975	**6140**
c7552	5561	9340 * 6730	4,273,669	**3773**	8880 * 4770	5,118,595	**7850**
16 bit Multiplier	6180	6715 * 5910	2,853,894	**2425**	9090 * 4970	3,784,455	**9030**
c5135	8447	10840 * 8665	9,804,521	**5205**	10720 * 5570	24,743,000	**11211**
c6288	149083	91810 * 32475	2.4848 * 1e8	**27,283**	37730 * 20640	2.3424 * 1e8	**38,860**
64-bit BU	241116	99575 * 93320	8.13 * 1e8	**35,000**	57890 * 31440	3.7871 * 1e8	**54,410**

Internal flow-based clocking scheme

- Advantages:
  - Lower circuit depth and fewer path balancing DFFs
  - Does not require a large clock tree in the circuit

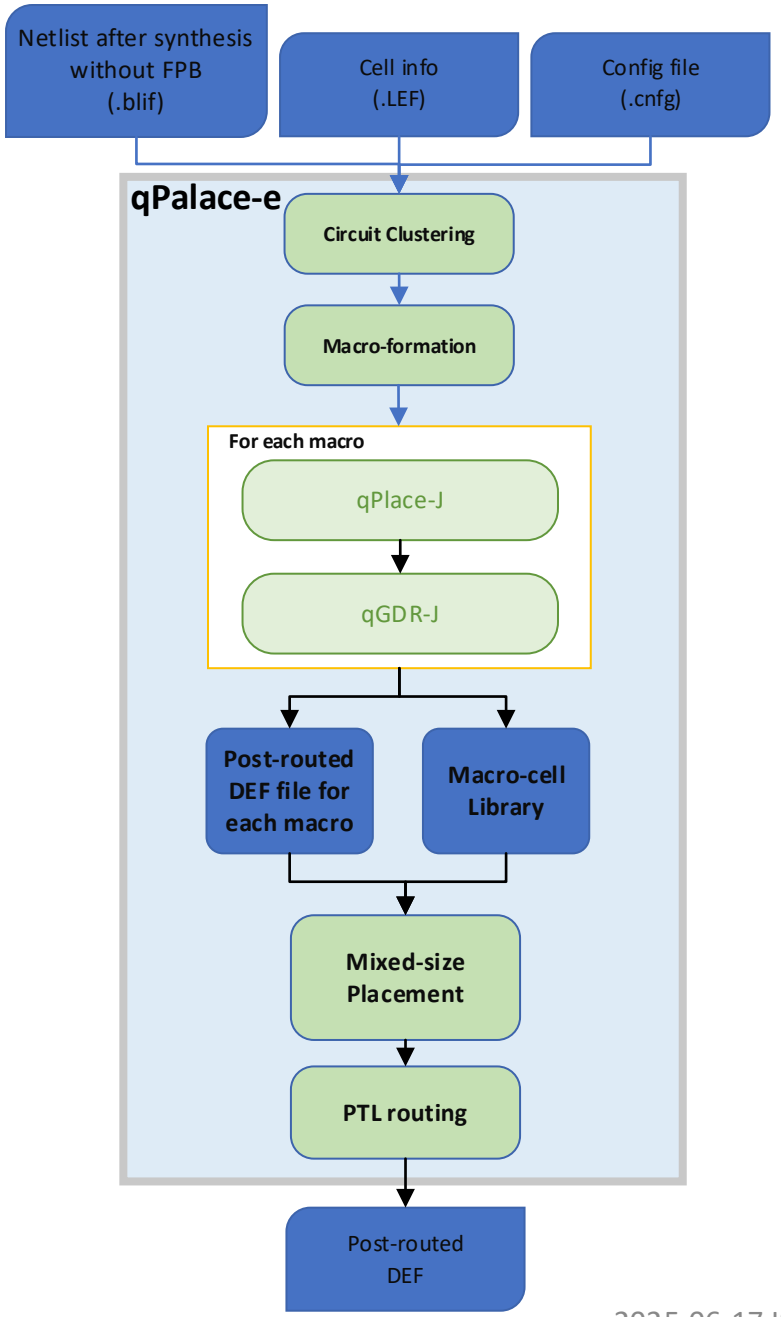
- Challenge:
  - If we only use PTL to route:
    - We cannot meet the timing requirement due to the low controllability of the PTL length.
    - The reflection is quite severe when the circuit works under high clock frequency and the PTL length is in some range.
  - If we only use JTL to route:
    - JTL routing has delay and area penalty compared with PTL routing.
    - JTL routing requires various JTL routing pattern cells to be included in the library.

Macro-based hierarchical design

- Problem statement:
  - For a given circuit, develop a methodology that partitions the target design into a netlist of macrocells and then places and routes the macrocells
- Key idea:
  - Use a size-constrained, fanin-cone based circuit partitioning method to partition the circuit into small macrocells such that each macrocell is a signal flow-aware strongly-connected set of standard cells and the inter-macrocell connection count is minimized
  - This is followed by primitive cell placement and JTL routing within the macrocells, and finally, macrocell placement along with PTL routing to connect the macrocells

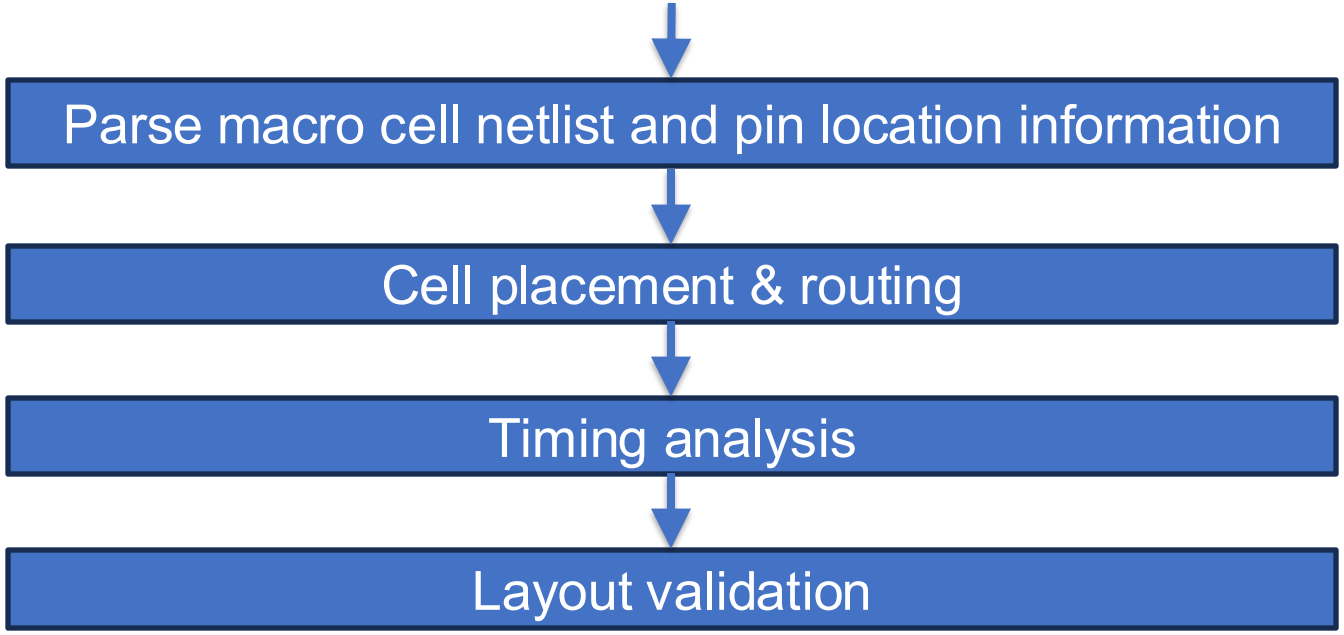
qPalace v2.b flow

- Transform the circuit netlist into a netlist of macrocells through clustering
- Selectively insert partial path-balancing DFFs to enable efficient macrocell placement and routing
- Use qPlace-J (JJ placer) and qGDR-J (JTL based router) to complete the layout of each microcell thus constructed
- Perform macrocell placement & routing (PTL based) to complete the full-chip integration



qMS (Macrocell synthesis)

- Motivation: To minimize the overall routing distance between macro cells, pin locations should be predetermined during macro cell construction based on the floorplan results. The design tool should also support constraints to enforce these predefined pin positions



Parametrizable JTL cell

- Each JTL can be tuned by adjusting: L_loop2, L_loop3, JJ_Area, RBJTL1, RBJTL2, RJTL1
- JTL cell size is fixed 30 μm × 30 μm

Table 2. Parameter JTL delay

Delay [ps]	Lloop2 [pH]	Lloop3 [pH]	RBJTL2 [ohm]	RBJTL1 [ohm]	RJTL1 [ohm]	JJ_area [μm²]
2	0.3	2.4	4.06	4.06	5.01	2.71
2.5	0.3	3.76	3.98	3.98	5.12	2.77
3	0.3	4.34	3.91	3.91	5.21	2.28
3.5	0.3	4.61	3.87	3.87	5.25	2.84
4	0.3	4.43	3.76	3.76	5.41	2.92
4.5	0.3	4.03	3.58	3.58	5.68	3.07
5	0.3	3.83	3.47	3.47	5.86	3.17
5.5	0.3	3.64	3.38	3.38	6.02	3.25
6.01	0.3	3.88	3.44	3.44	5.91	3.2
6.5	0.3	3.79	3.39	3.39	6	3.25
7	0.3	4.25	3.61	3.61	5.63	3.05

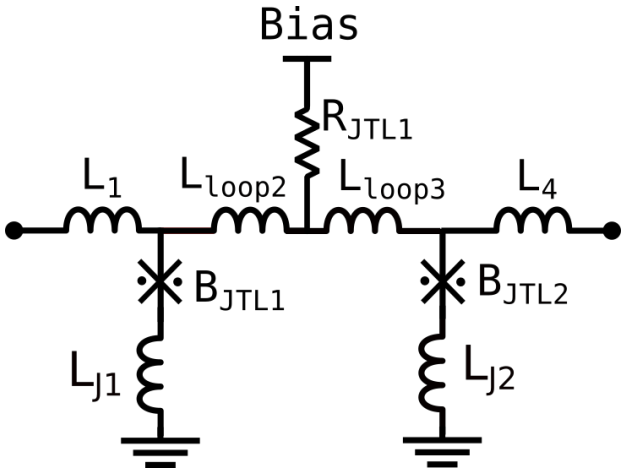
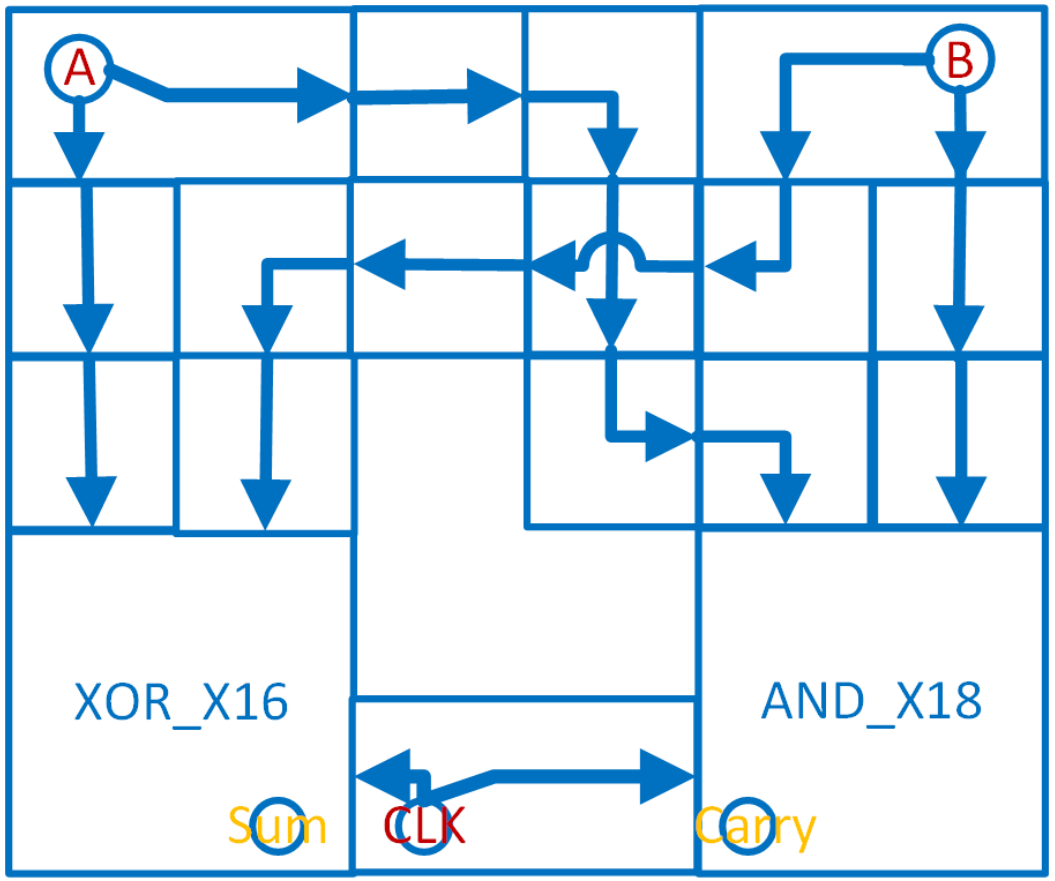
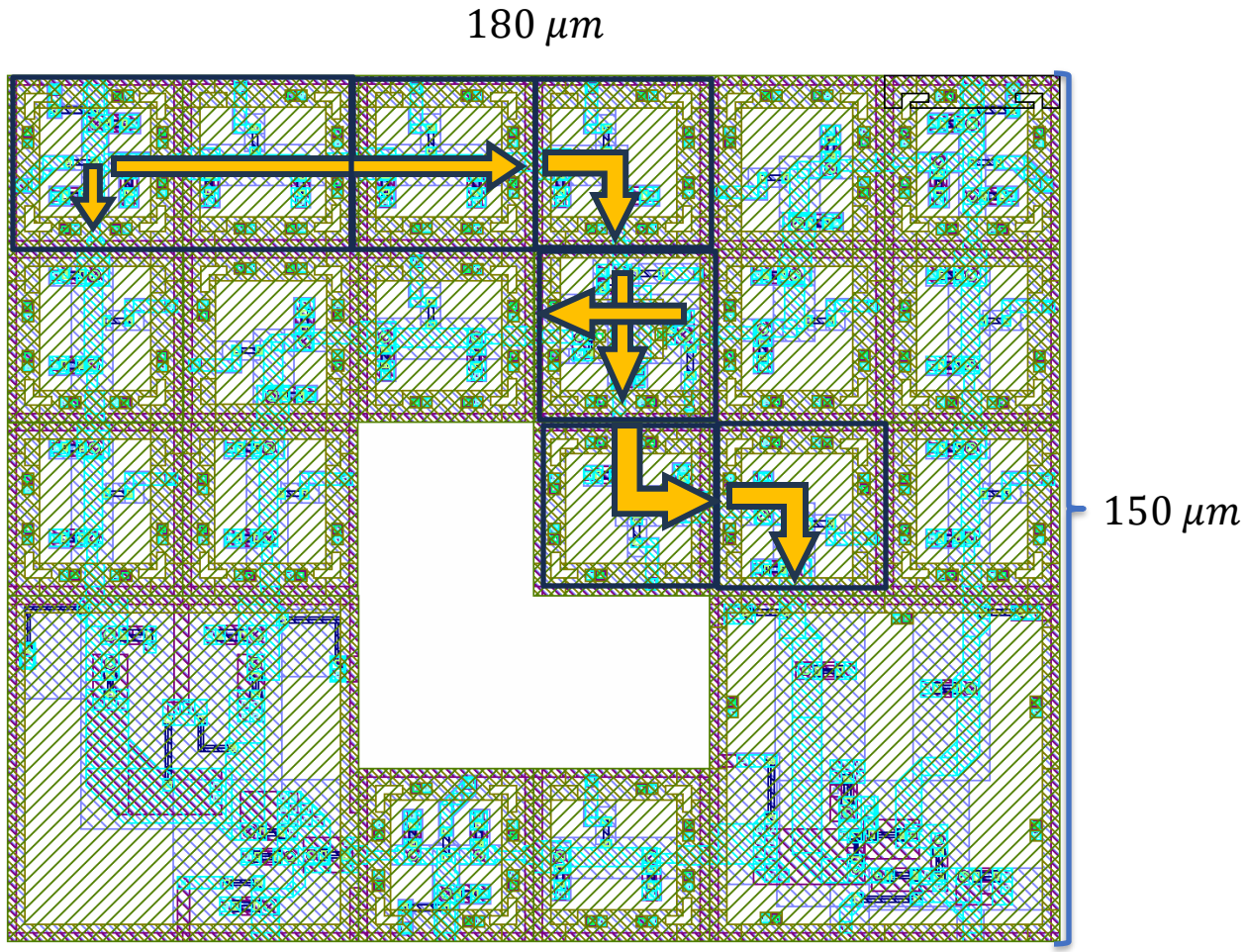


Fig 1. JTL parameters

qMS example – One-bit half adder



Schematic view



Layout view

*All pins are predefined: red pins are inputs, and yellow pins are outputs.

Preliminary Results for Hybrid PTL/JTL Routing

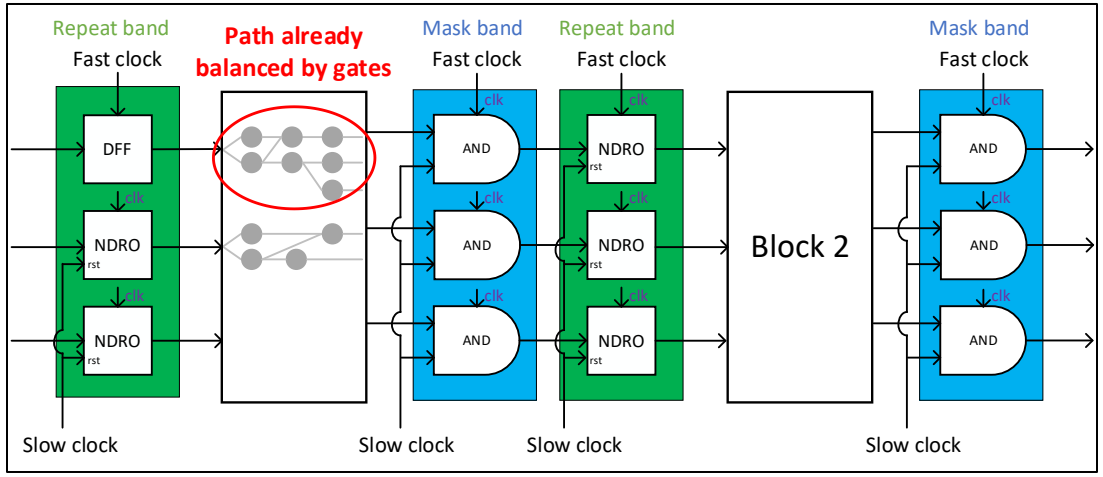
Benchmark circuit	Hybrid-routing DFF count	qPlace DFF count	% DFF reduction	Hybrid-routing performance (GHz)	qGDR performance (GHz)	% clock-frequency Improvement
interrupt controller	289	379	23.75%	15.1	14.4	5%
16-bit error detector	348	627	44.50%	15.2	14.5	4%
8-bit ALU	916	1402	34.66%	12.3	10.4	18%
9-bit ALU	2362	3880	39.12%	7.3	6.2	17%
16-bit Multiplier	25259	33494	24.59%	9.3	7.7	20%
34-bit adder	1236	2633	53.06%	7.0	5.6	25%

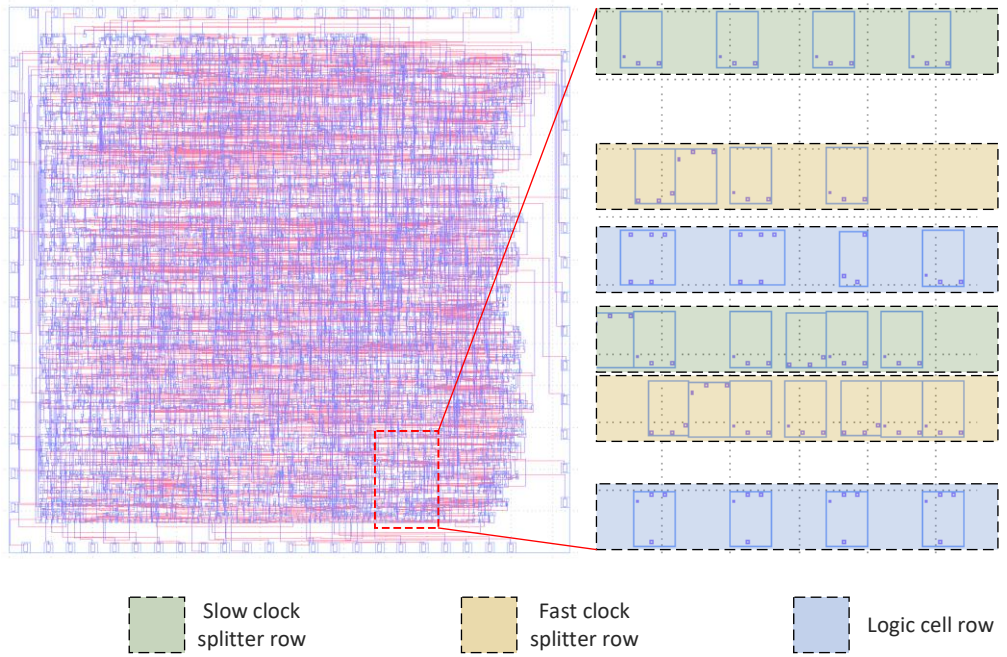
Lowering the path balancing overhead

- Optimize the clocking scheme
  - Dual clock scheme and/or dual-phase clocking
    - Difficulty: Hard to perform placement & routing for multiple clock domains given the required splitter trees and limited routing layers

- Develop a standard library containing complex cells
  - The Coldflux v1 cell library contains only basic logic cells (AND, OR, XOR, DFF, INV), which causes a deep pipeline (long latency) and large DFF path balancing overhead
  - Adding various multi-stage complex cells, which receive a clock signal only at their input and output stages while using clock-follow-data scheme in other stages is our solution
    - Difficulty: logic depth of each complex cell must be limited to 4 or 5 depending on the target clock frequency

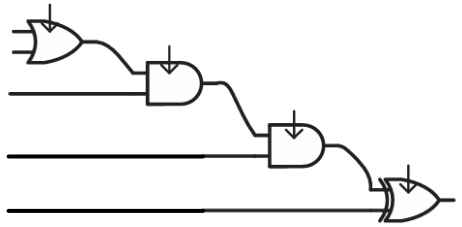
Dual clocking scheme



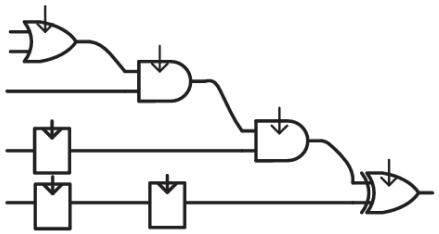


qPlace: Clock tree for dual clocking scheme

No path balancing



Partial path balancing



Circuit	JJ count [1]	JJ count (ours)	Gate count	Gate count (ours)	Zero path balancing	Partial path balancing
c499	6641	5020	493	273	9.7GHz	15.8GHz
c880	11525	7049	793	330	8.1GHz	14.5GHz
c1908	6217	6606	450	326	7.2GHz	14.5GHz
c3540	23460	25984	1580	1144	4.7GHz	10.4GHz
c5315	39464	28462	2831	1337	3.7GHz	6.2GHz
c7552	38998	26123	2834	1368	4.0GHz	5.6GHz

Dual-phase clocking

- 24% average area savings
- 31% average latency reduction

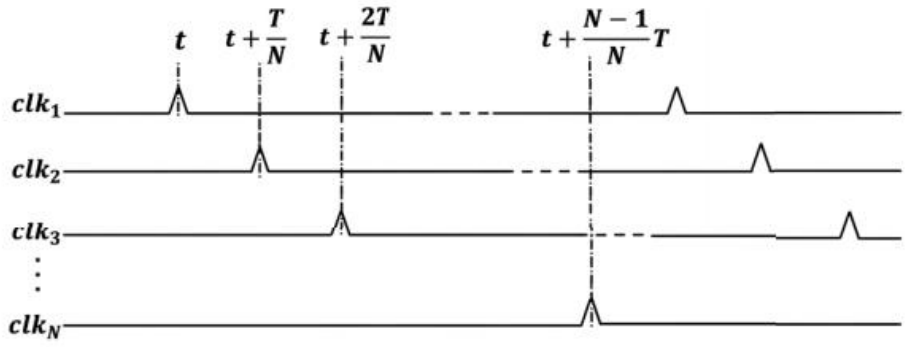
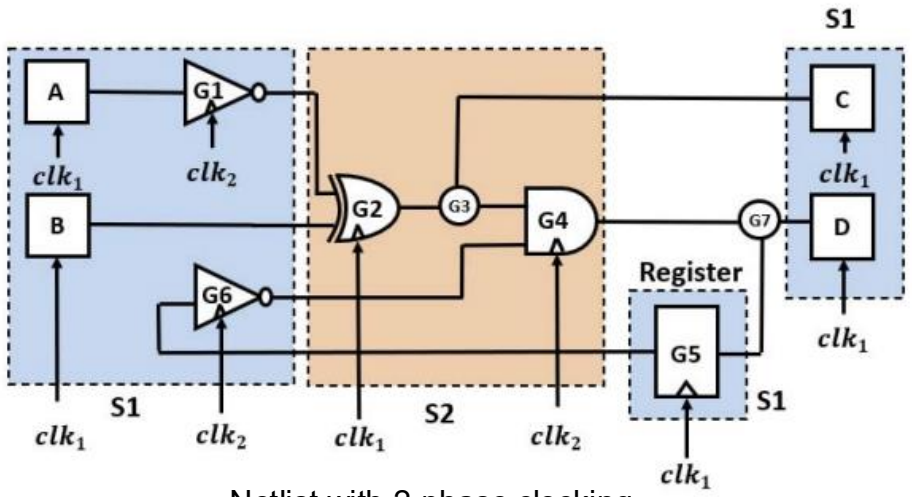
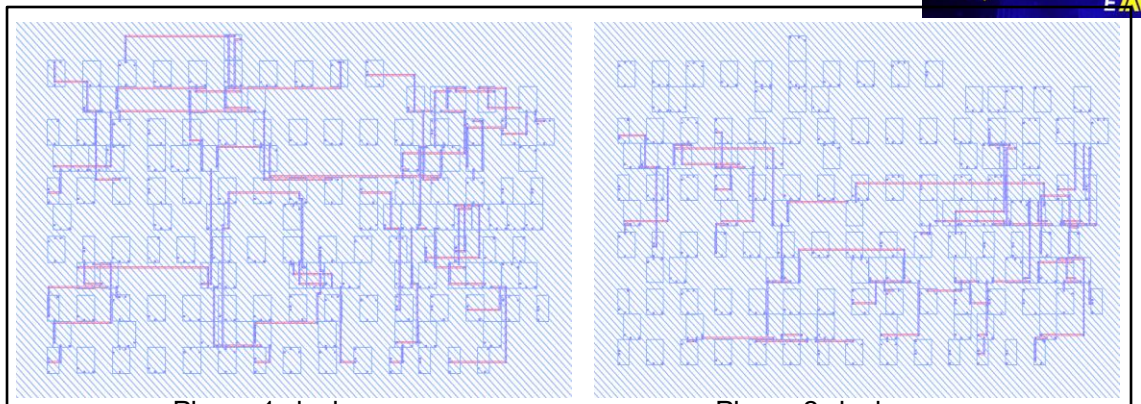


Fig. 2. Multi-Phase Clocks



Netlist with 2-phase clocking



Phase 1 clock Phase 2 clock

Clock trees for 2-phase clocking

Circuit	Full Path Balancing (T=30ps)		Timing Constrained Clocking 2-Phase T=30ps	
	Cycles	Area	Cycles	Area
int2float	15	1.39	9	1.23
cavlc	18	4.07	12	3.74
i2c	17	9.55	10	8.02
c5315	24	10.69	17	9.28
c7552	25	10.25	22	10.04
Sin	146	43.13	109	33.53
square	143	131.3	129	98.5
Max	140	130.42	77	71.4
arbiter	89	195.5	46	77.32
log2	325	395.51	235	207.06
BU_64	175	386	139	343
Average Ratio			**0.69**	**0.76**

Area and Cycles comparison for FPB and 2-Phase clocking

Courtesy: Peter Beerel

qPALACE v2 vs. v1 Results

qPALACE v1

	Logic cells	DFF/NDRO	Logic splitter	Splitters	Total Cells	Total JJ	Area
KSA-32bit	526	455	489	980	2450	18278	9.56*3.93
Integer_divider8	3679	4137	3221	7815	18852	136923	24.25*9.78
EPFL_Mult	78827	136587	66256	215413	497083	3496507	Timed out
EPFL_Divider	148230	671999	125276	820228	1765733	12027387	Timed out
C6288	28993	32981	25136	87109	149083	1079909	66.64*26.3
C7552	1822	3179	1433	5000	11444	80992	18.94*9.08
Mult-16bit	2936	4556	2445	7491	17428	122811	89.68*35.26

qPALACE v2 with dual clocking (slow-fast)

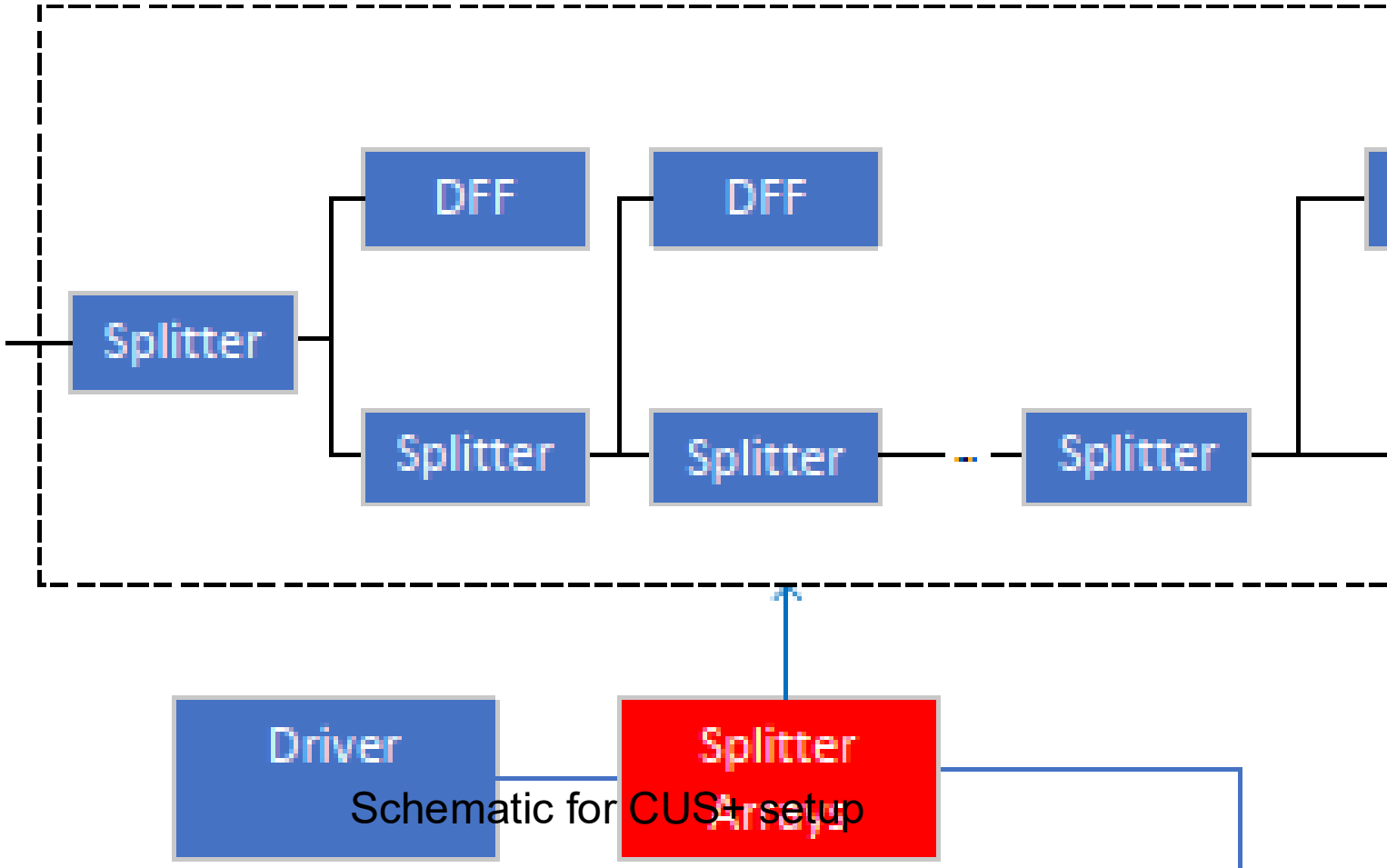
	Logic cells	DFF/NDRO	Logic splitter	Splitters	Total Cells	Total JJ	Area
KSA-32bit	410	65	401	810	1220	11508	3.33*4.06
Integer_divider8	415	16	432	846	1261	11237	3.39*4.11
EPFL_Mult	19839	128	21178	41016	60855	509595	23.22*23.66
EPFL_Divider	45568	128	47554	93121	138689	1237674	34.35*36.88
C6288	1550	32	1655	3204	4754	39598	6.95*7.02
C7552	1475	206	1054	2528	4003	33838	14.47*6.3
Mult-16bit	2406	32	2844	5249	7655	67543	8.67*9.02

Design verification problem

- As circuit size increases, SPICE level simulation cannot be used to verify the circuit behavior and timing due to the excessively long simulation times
  - Consider simulating 100k patterns for a circuit with 10k cells and 100 logic levels deep operating at 10 GHz at a 1ps time step → Total simulation point count is 10,000 + (100 * 100,000) and for each simulation point, we need to store & calculate information more than ten million devices.
  - Even iVerilog simulator struggles when simulating such a circuit.
- Existing cell delay models (which are typically load-independent) tend to result in overestimating the circuit performance.
  - The challenge is to develop an efficient, yet accurate, timing model for each library cell, which has been validated and can be stored in the library and used during cell-level simulations

Verifying a cell's function under different loads

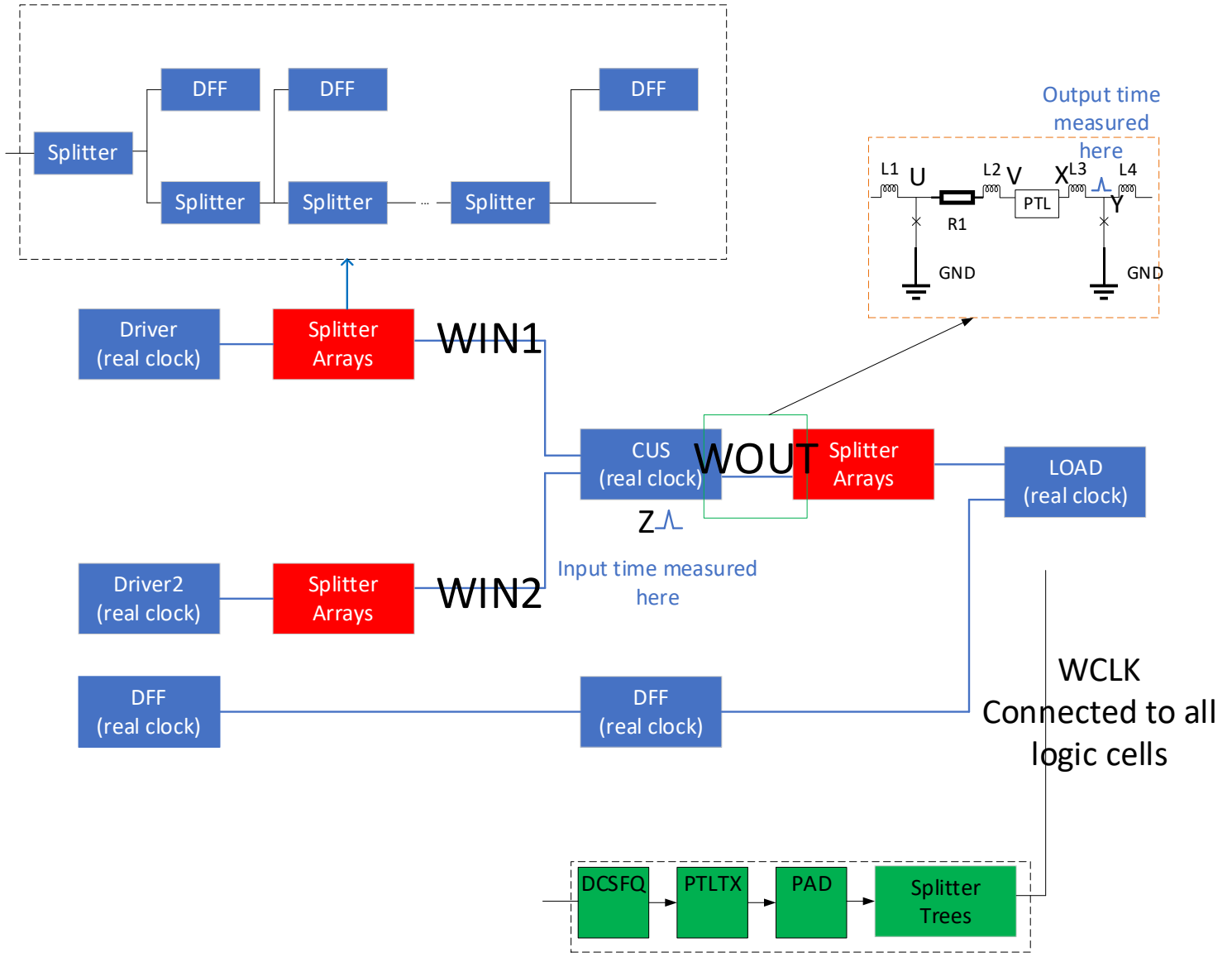
- CUS (cell under study): Each logic cell in the given cell library is considered as a CUS
- CUS+: Netlists created by combining each CUS with every possible cell as driver and every possible cell as the load
- For each CUS+ configuration, create other versions where a chain of splitters is inserted between the driver and the load
- Exhaustive input pattern simulations using JoSIM are performed on each CUS+ to verify functional correctness



Schematic for CUS+ setup

Timing characterization for each cell

- Use a **realistic clock tree** (instead of an ideal clock tree)
  - Enumerate the clock tree for the CUS with 1,2,3 splitters
- Consider the influence of **PTL reflections**
  - Sweep input, output and clock (WIN1, WIN2, WOUT, WCLK) wire delays between 1 – 20 ps
- Vary the interval between clock and input signal for **setup and hold characterization**
  - Sweep input clock interval between −5 and +20 ps



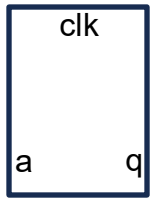
Hold Time Violation Fix

Method: Uses multiple versions of the SFQ cell, each with exact same footprint and pin locations

1. After STA, identify all hold-time violations
2. For each path, add a maximum of 4 ps of JTL delay to each input/output pin without changing the cell size
3. The delay insertion fixed most of the hold time violations (without creating any setup-time violations)
4. If there are still hold violations, add JTL delays to the source driver's clock pin or to the clock tree

2~4 ps JTL

circuit	Original hold violations	Stage1: insert data path JTL	Stage2: insert clock path JTL	Critical path
C3540(qLevelPl)	168	(after 207 JTLs) 0	0	75.69 ps
C7552(qLevelPl)	167	(after 196 JTLs) 1	(after 1 JTL) 0	113.48 ps
c3540 (qQuadPl)	708	(after 757 JTLs) 0	0	110.45 ps
c7552 (qQuadPl)	731	(after 824 JTLs) 0	0	124.56 ps



DFFT (each pin has a JTL)

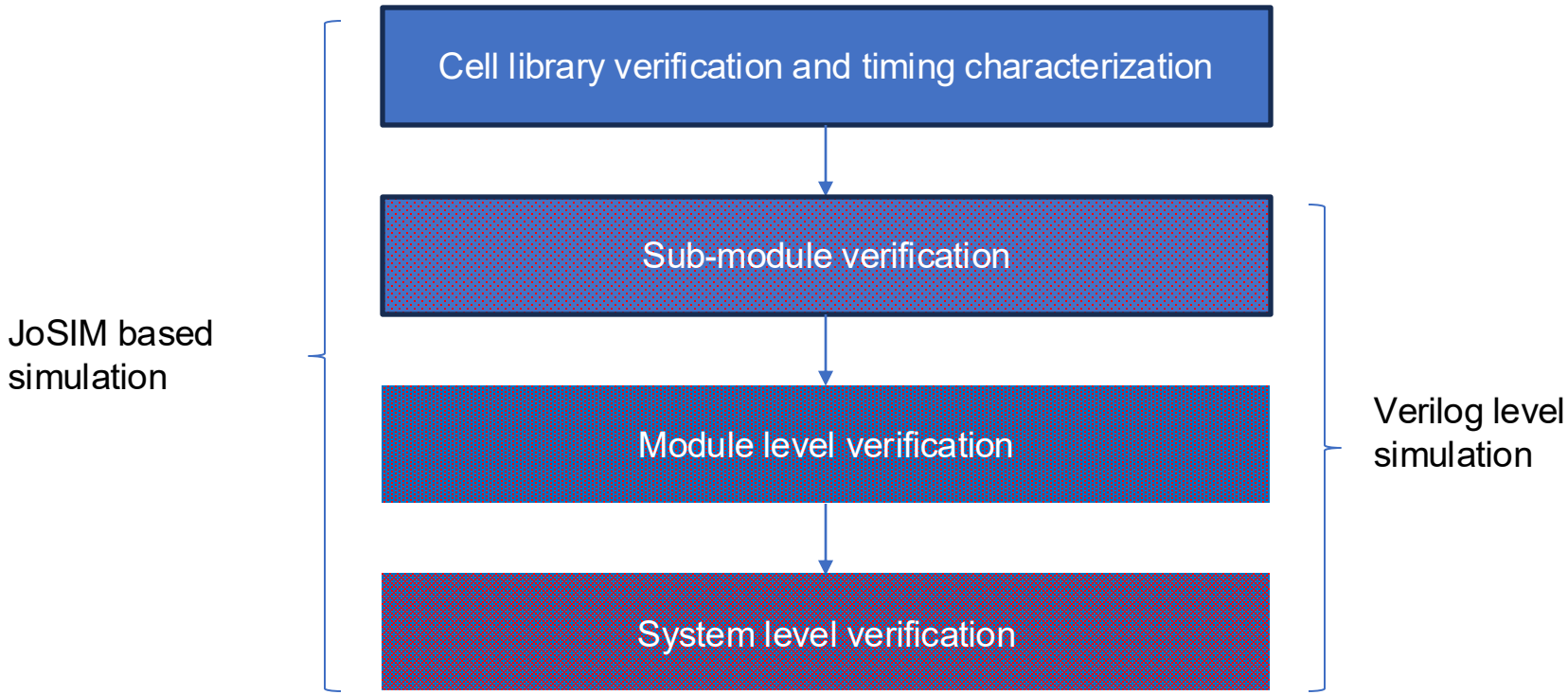
[2,3,4] ps JTL

circuit	Original hold violations	Stage1: insert data path JTL	Stage2: insert clock path JTL	Critical path
c3540(qLevelPl)	168	(after 195 JTLs) 0	0	75.69 ps
c7552(qLevelPl)	167	(after 192 JTLs) 1	(after 1 JTL) 0	113.48 ps
c3540 (qQuadPl)	708	(after 749 JTLs) 0	0	110.45 ps
c7552 (qQuadPl)	731	(after 820 JTLs) 0	0	124.56 ps

Hierarchical design verification methodology

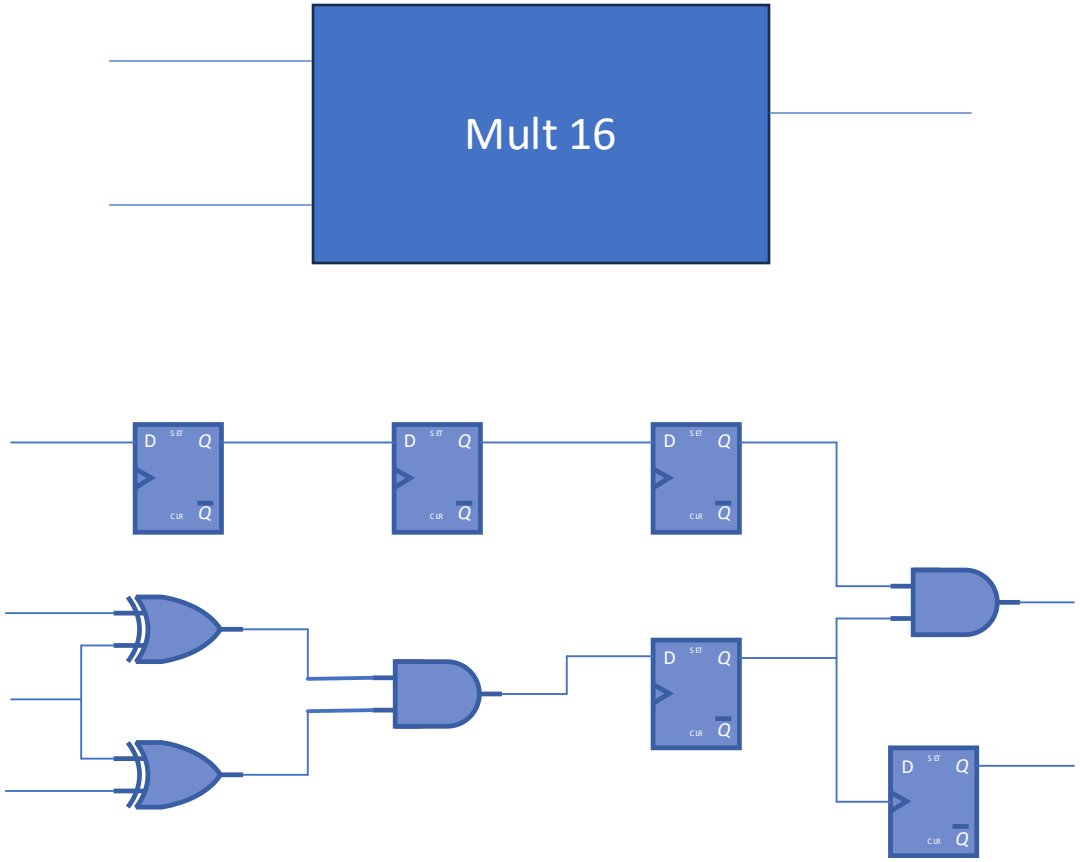
Key ideas

- **Timing model**: Use decision tree regression to generate characterization data for each library CUS
- **Event-driven simulation engine**: Each RSFQ cells is reset in every clock cycle, so we only need to simulate the time steps during which there are input or intermediate pulse events rather than simulating all time steps



Sub-module verification

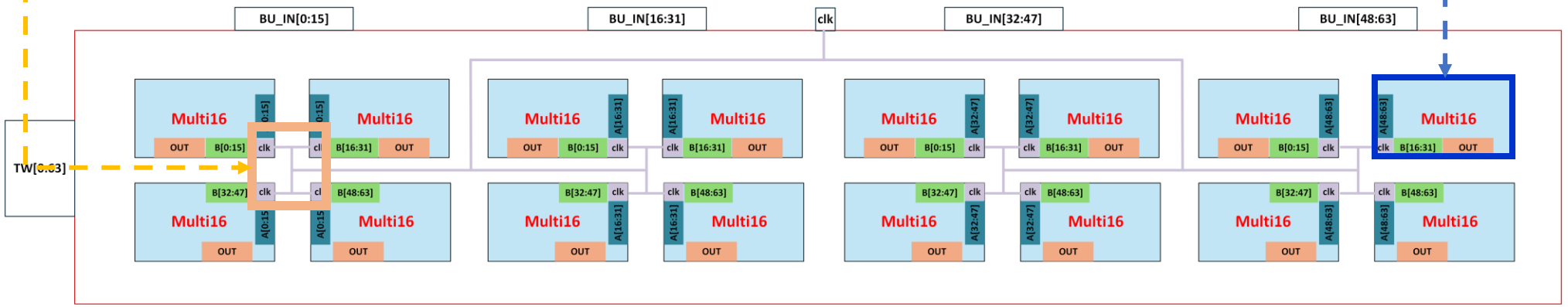
- Challenges:
  - JoSIM based timing validation is not scalable for circuits with > 10,000 logic cells.
  - Simply applying the Verilog level simulation fails to verify the electric properties like reflection.
- Main idea:
  - Exploit a mixed-simulation approach
  - Instead of JoSIM simulation of the whole circuit, we use JoSIM for each 3-level subcircuit and report the logic values and delays.
- For each sub-module:
  - Divide the circuit into an exhaustive set of three level subcircuits.
  - Calculate the clock arrival time for each cell.
  - Apply exhaustive patterns/sequences to each subcircuit.
  - Perform JoSIM simulation for each subcircuit and compare the results with the results of timing simulation.
  - Note that we also apply random pattern and perform logic simulation for the entire sub-module.



One submodule sample

Module verification

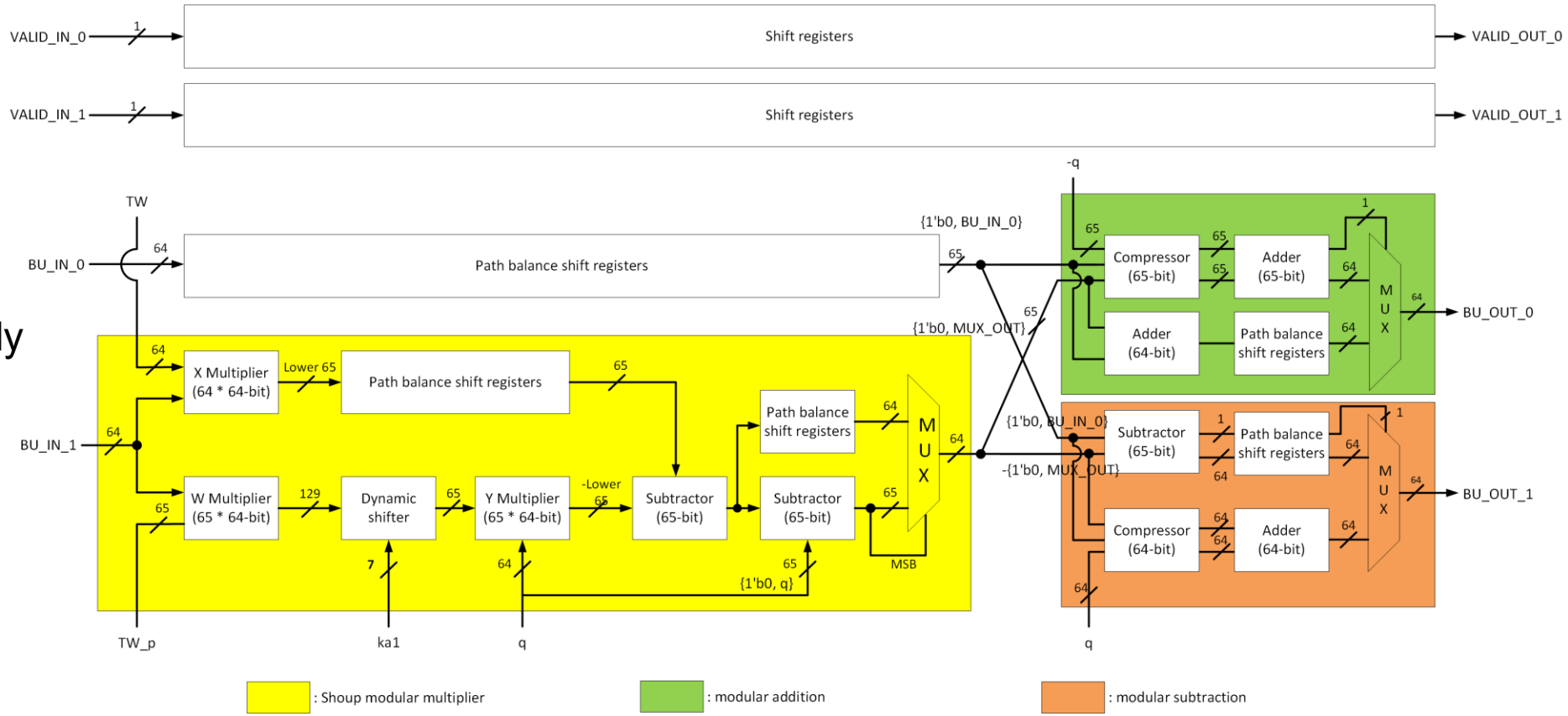
- Electrical check:
  - For each interconnection between two sub-modules:
    - Extract driver cell, load cell, interconnection and all the path-balancing DFFs along the path.
    - Perform JoSIM simulation to verify each interconnection.
  - Module verification
    - Logic simulation is performed for submodule logic verification.
    - For module logic verification:
      - Random pattern simulations are performed.
      - Simulations are performed for patterns derived from functional system simulation.



Full design verification

- We perform JoSIM simulations to do electrical check for all inter-module interconnections.
- Functional simulator provides test patterns and corresponding golden results as assertions to check if the system works as expected.

Butterfly Unit



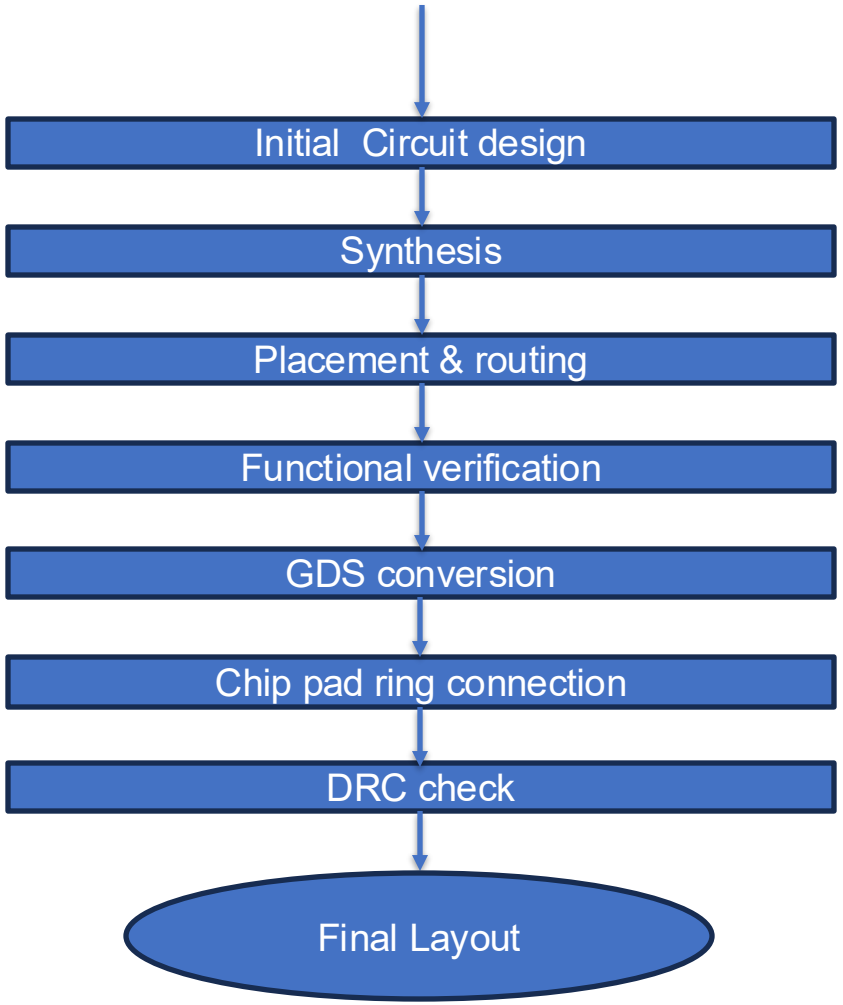
Model validation

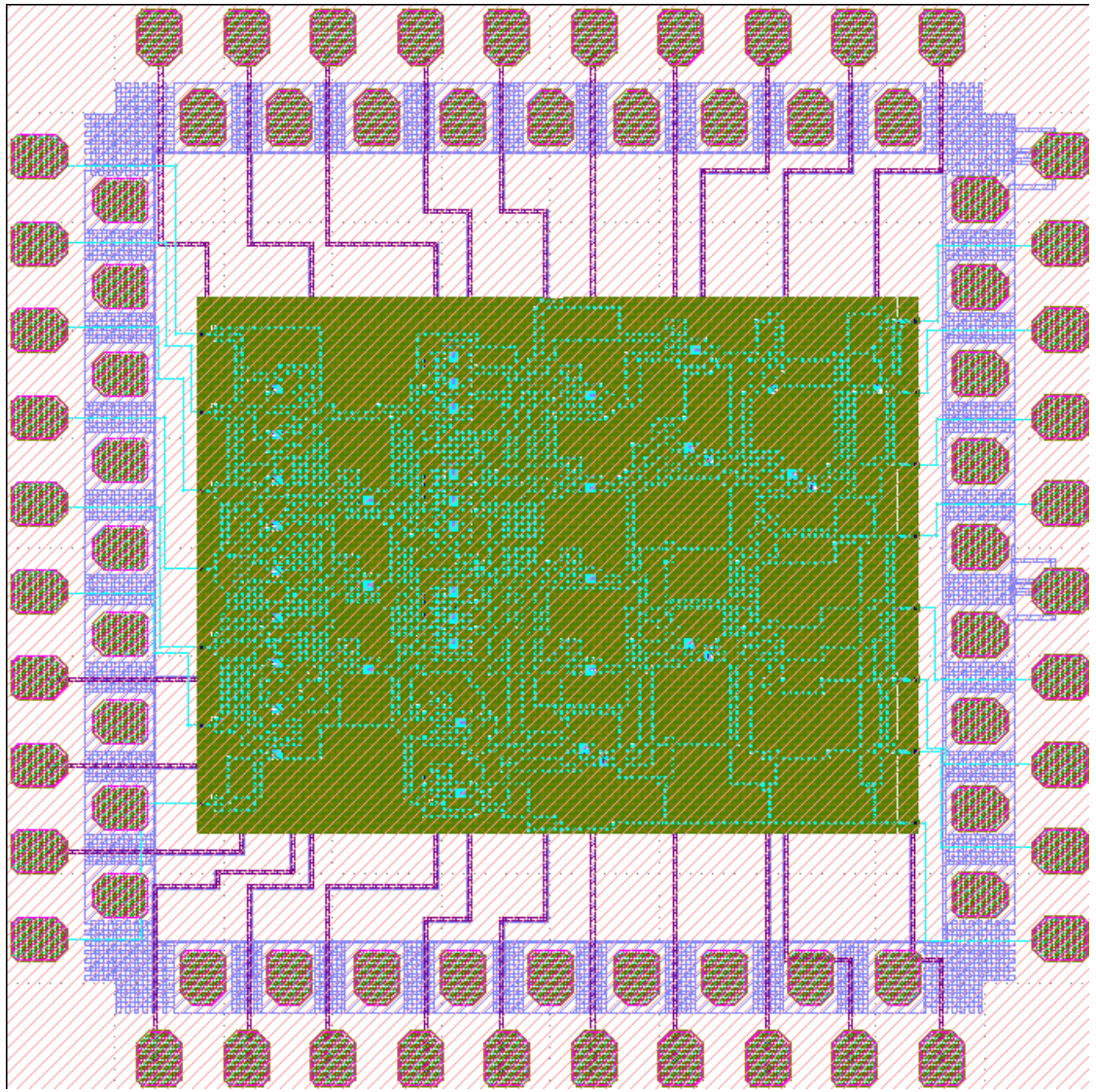
- By simulating multiple circuits, we validate our tool by comparing the timing simulation result with JoSIM simulation result on ISCAS85 circuits (5 patterns are applied) to check the variance.

Circuit name	Maximum estimation error (ps)	JoSIM simulation time (s)	Proposed tool simulation time (s)
c432	1.43	2614.2	24.3
c499	1.61	431.4	7.4
c880	1.80	2996.5	42.0
c1355	1.59	625.7	7.7
c1908	1.74	2118.3	12.6

Multiply-Accumulate (MAC) chip design

USC cell library design submitted for fab



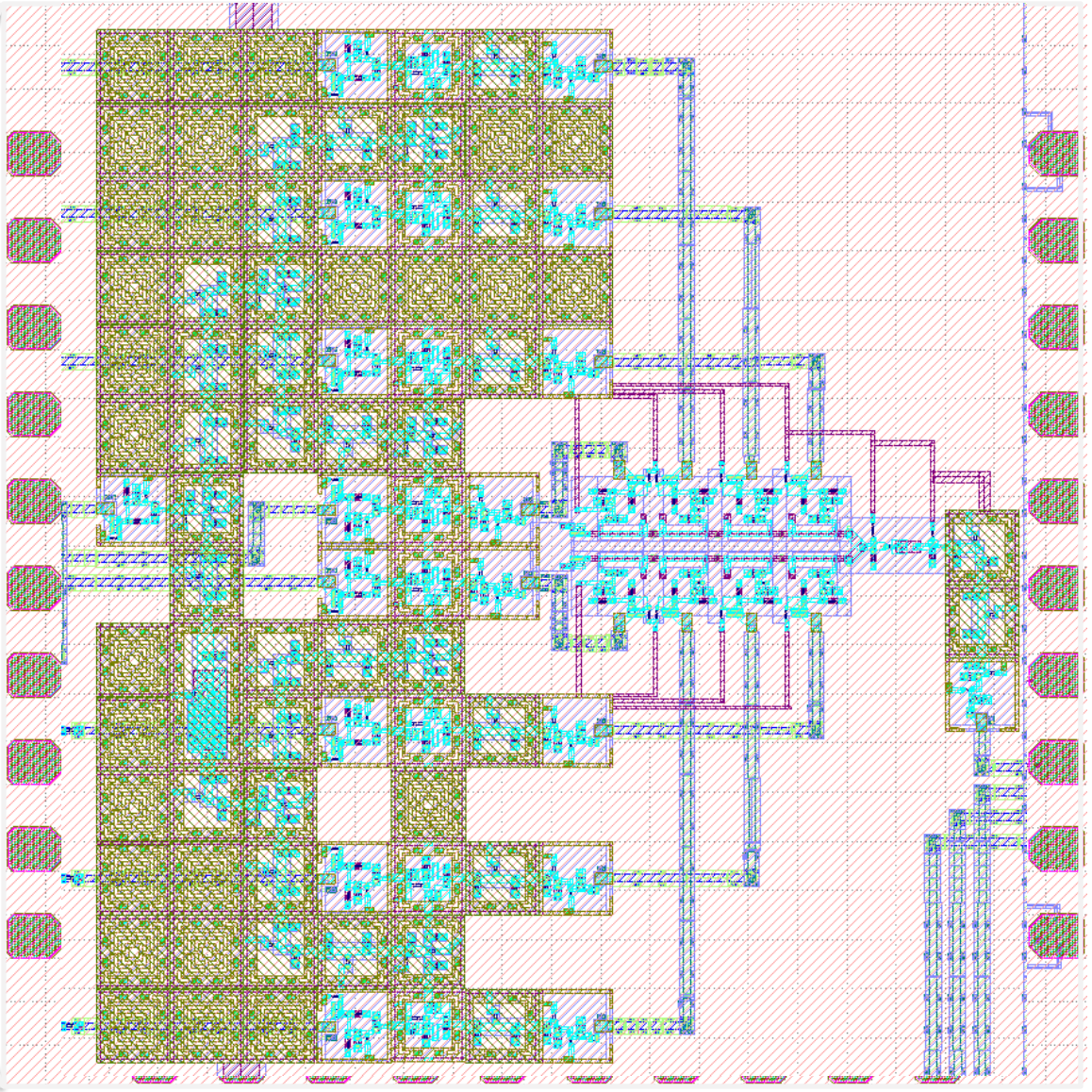


Spiking Neural Network (SNN) chip design

USC cell library design submitted for fab

A small spiking neural network (SNN) chip:

- An inference SNN for classification of a reduced MNIST dataset.
- There are 18 neurons in hidden layer and 3 neurons at the output. The throughput is 10GHz and it consumes about 2.4 mW power.
- Each neuron + synapse is a macro cell with the buffers at the edge, inside of which is routed by JTLs.



SNNChip: Green boxes are JTL-connected components, and lines between them are PTLs

Memory chip design

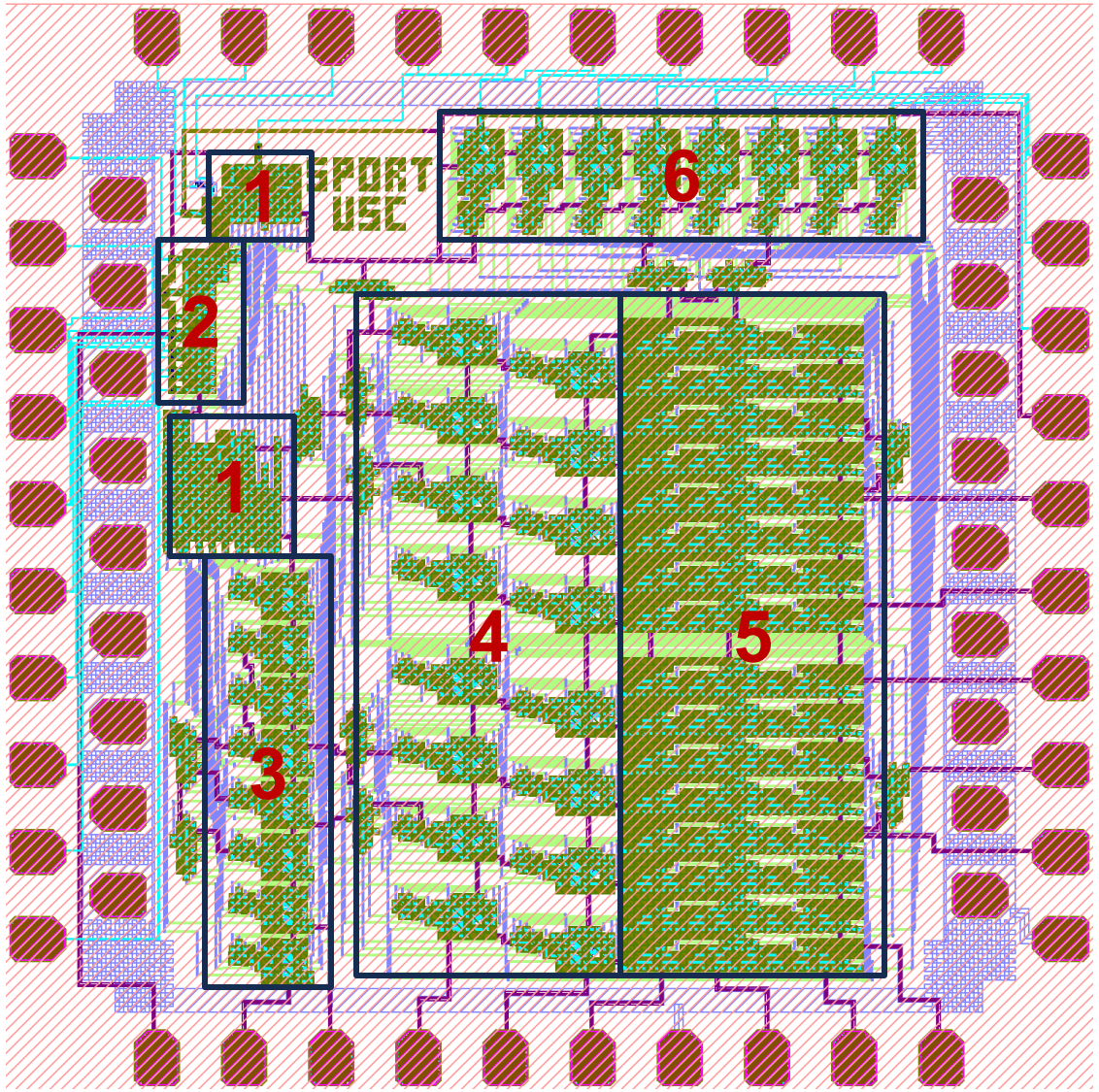
USC cell library design submitted for fab

Basic statistics:

Frequency	20 GHz
JJ count	9,226
Design area (w/o PAD)	3560 μm (w) x 4030 μm (h)
Die area	5000 μm (w) x 5000 μm (h)

Components & Data flow sequence:

Component ID	Component
1	Global clock network
2	Input data DFFs
3 & 4	Data Selection Units
5	Memory cells
6	Output Router Unit



qPALACE v2 is available!

- You may download the latest version of the Coldflux tool suite (includes libraries and qPalace) from: https://coldflux.usc.edu/

- Acknowledgements: Many thanks to our team members and collaborators: Sasan Razmkhah, Peter Beerel, Sandeep Gupta, Murali Annavaram, Scott Holmes, Eby Friedman, Michael Hamilton, Grace Xing, Ivan Nevirkovets, Oleg Mukhanov, Coenrad Fourie, Nobuyuki Yoshikawa, Chris Ayala, and many PhD students in the SPORT Lab

Sponsors: