IDEAL: An Intelligent Design Environment for Asynchronous Logic

PI: Rajit Manohar (Yale)
Co-PIs: Keshav Pingali (UT Austin), Martin Burtscher (Texas State)

Overview and goals

- A ***parallel*** design flow for ***asynchronous logic***
  - Modularity
    - permitting re-use (similar to software)
    - “re-compile” only for technology-specific optimization
  - Minimizes expectations from the implementation technology

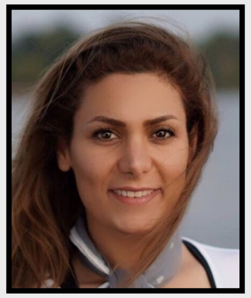
- Built more like a software compiler than current EDA tools
  - Operate on a core database
  - “Passes” for optimization
  - Share analysis phases at multiple levels of abstraction
    - High-level design
    - Circuit
    - Physical

Team and roles

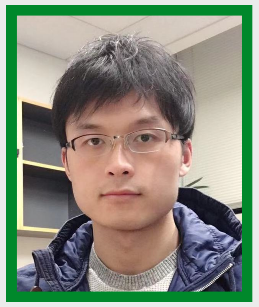
Faculty

Asynchronous design & tools

Rajit Manohar
(Yale)


Dr. Samira Ataei
(memory)

Wenmian Hua
(timing analysis)

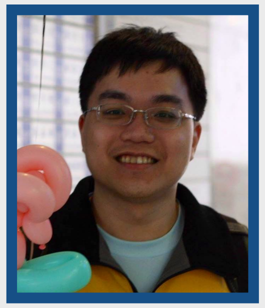

Yihang Yang
(placement)

Yale

Parallel data-structures & algorithms

Keshav Pingali
(UT Austin)

Michael He
(routing)


Yi-Shan Lu
(timing analysis)

Sepideh Maleki
(placement)

TEXAS
The University of Texas at Austin

GPU-based parallelization

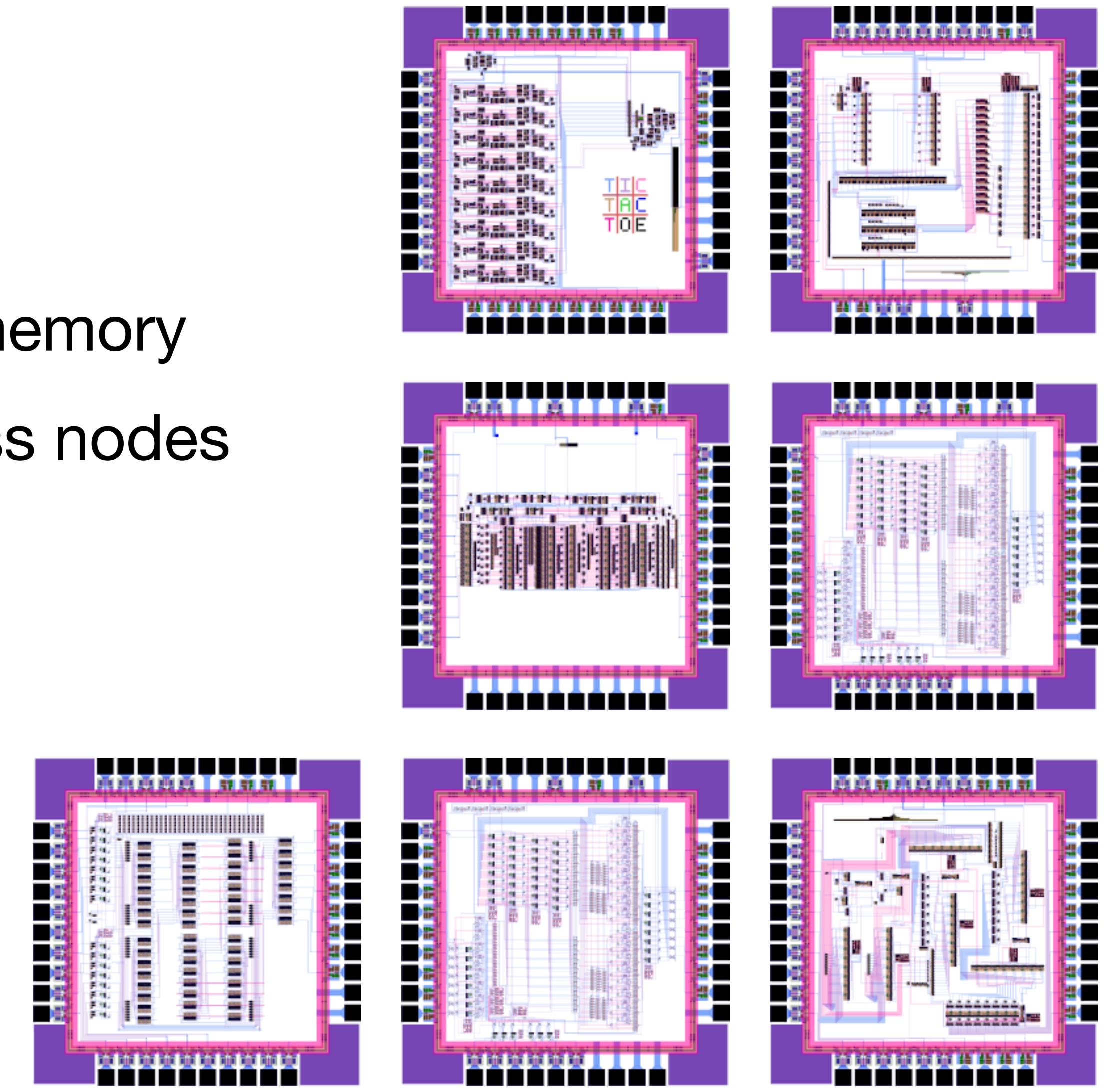
Martin Burtscher
(Texas State)

Aarti Kothari
(routing)

TEXAS STATE UNIVERSITY

First integration meeting summary

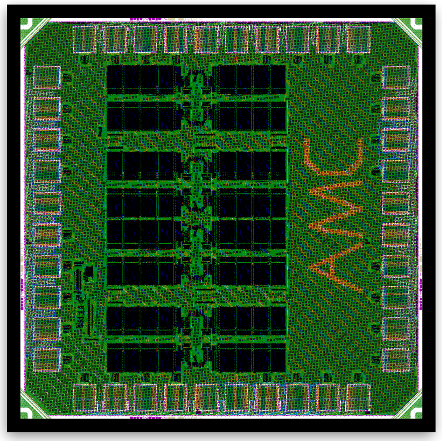
- Parallel tools
  - ❖ Synchronous static timing analysis (STA) + asynchronous STA
    - standalone tools
    - no parasitics
- Asynchronous logic tools
  - ❖ Compiler for pipelined asynchronous memory
    - Preliminary 65nm and earlier process nodes
  - ❖ ACT language and database
  - ❖ Custom circuit design flow
    - Simulation with hazard finding
    - Netlist generation
    - Layout versus schematic
    - Xyce for analog simulation



E.g: dot product engine, simple µcontroller

Second integration meeting summary

- Parallel tools
  - ❖ SPRoute: a parallel global router (*to appear, ICCAD 2019*)
  - ❖ Timer
    - ‣ Inner loop of buffer insertion + gate sizing (*3rd place, TAU contest 2019*)
    - ‣ Re-architected: netlist, delay calculator, parasitics abstracted out
    - ‣ Integrated with async database (ACT)
- Asynchronous logic tools
  - ❖ AMC: async memory compiler (*ASYNC 2019, Best paper finalist*)
    - ‣ Built-in self-test (BIST) support
    - ‣ 28nm support and 65nm test chip
  - ❖ Preliminary place and route flow
  - ❖ Community support
- GitHub



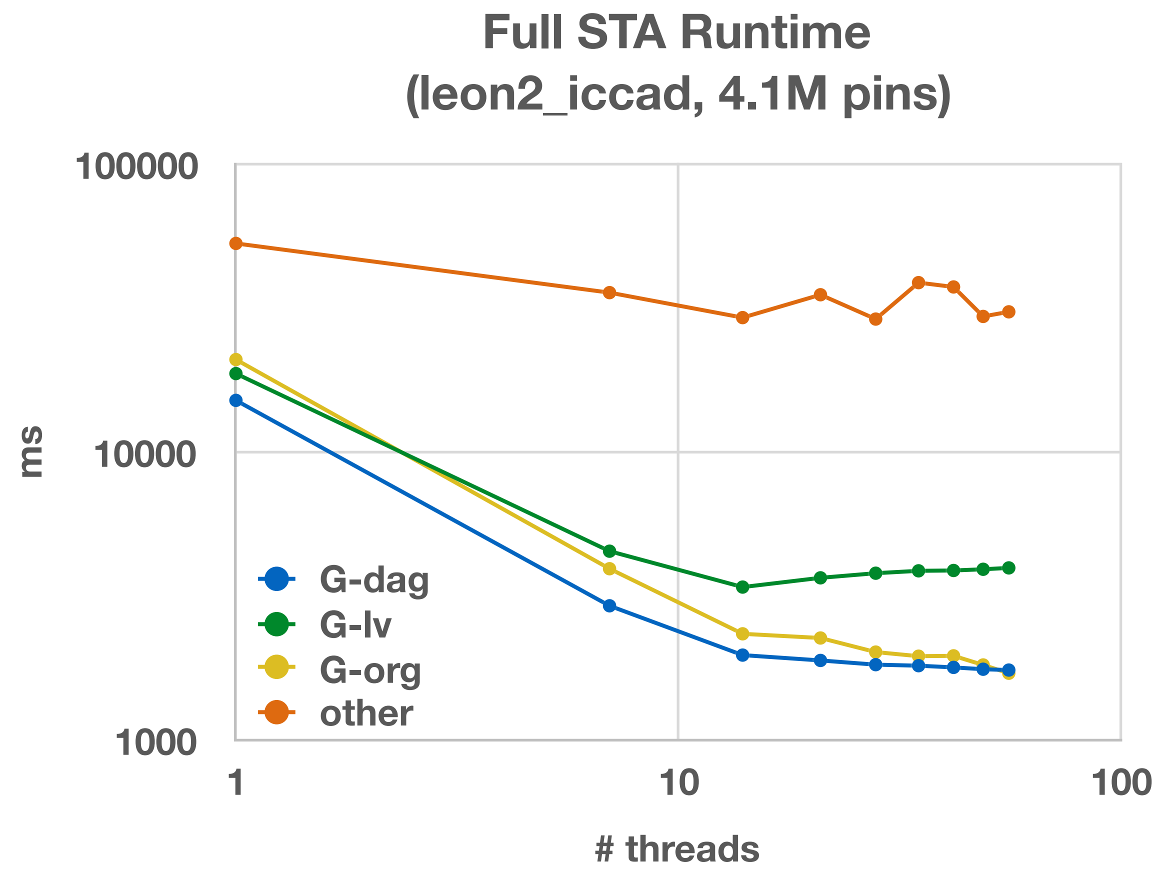
http://github.com/asyncvlsi/{act, AMC}
http://avlsi.csl.yale.edu/act/
http://github.com/IntelligentSoftwareSystems/Galois

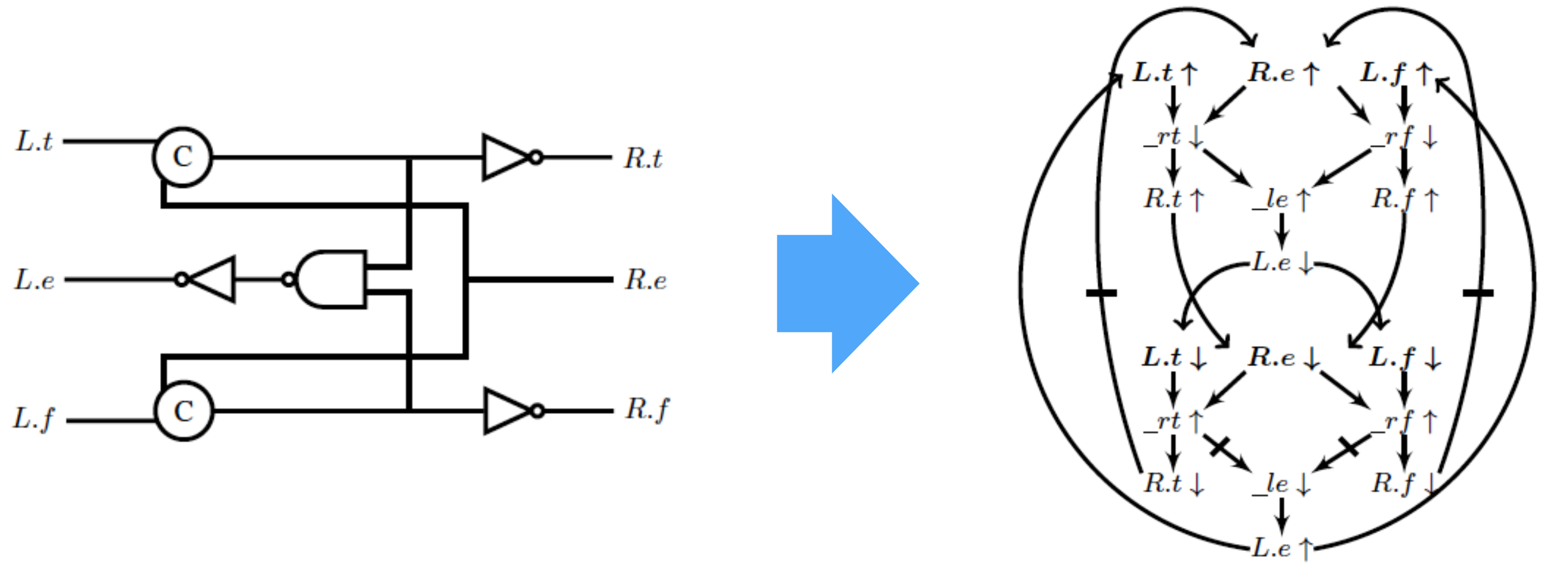
Parallel static timing analysis

update

- Parallel STA for synchronous logic
  - ❖ Supports parasitics (.spef)
  - ❖ Used as core library in gate sizing
  - ❖ Refactored to be integration-ready

- Parallel STA for asynchronous logic
  - ❖ Inherits features of new STA core
  - ❖ Integrated with ACT database (was: Verilog + timing arcs file)



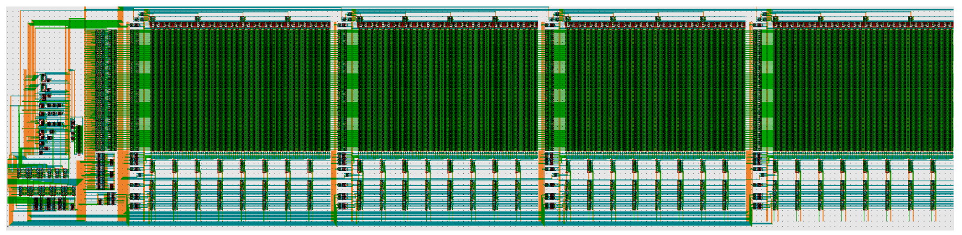
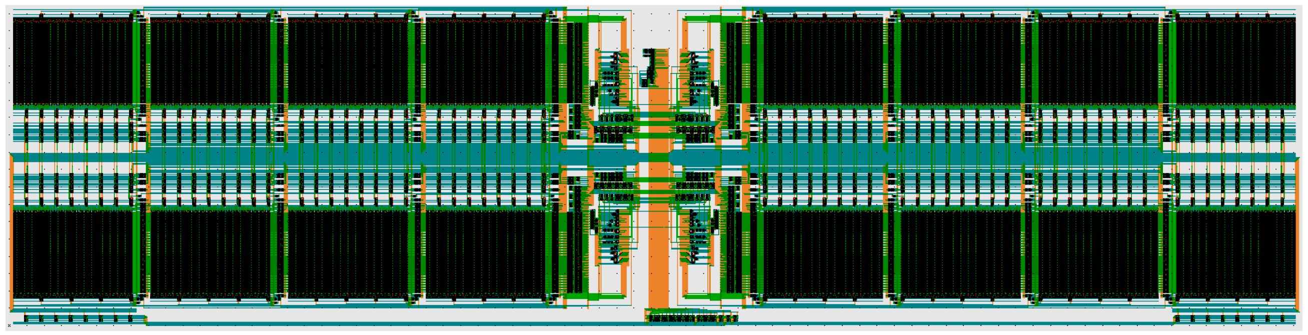
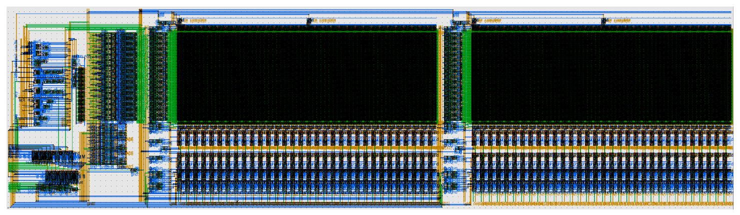
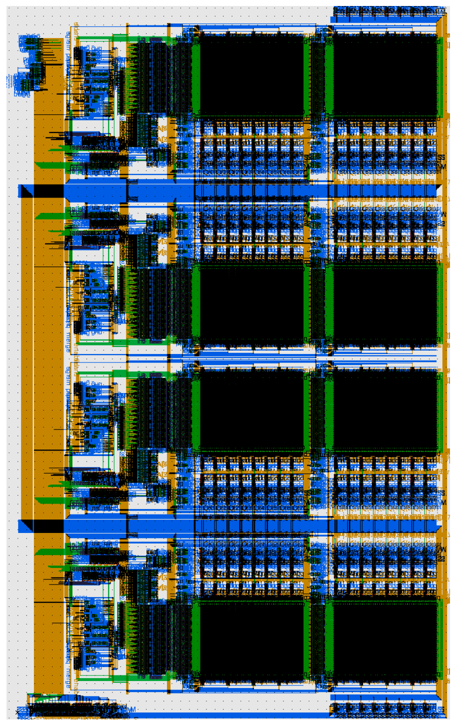


TEXAS The University of Texas at Austin Yale

AMC: Asynchronous Memory Compiler

update

- Features
  - ❖ New command: read-modify-write
  - ❖ Banking
    - Sub-banking for better power/performance at area cost
    - Flexible orientation
- Updates
  - ❖ 1st version on Github (May 2019)
  - ❖ Built-in self-test (BIST) engine
  - ❖ 65nm tape-out
    - Thin-cell layout, min area, asymmetric vias
  - ❖ 28nm generator with foundry cells

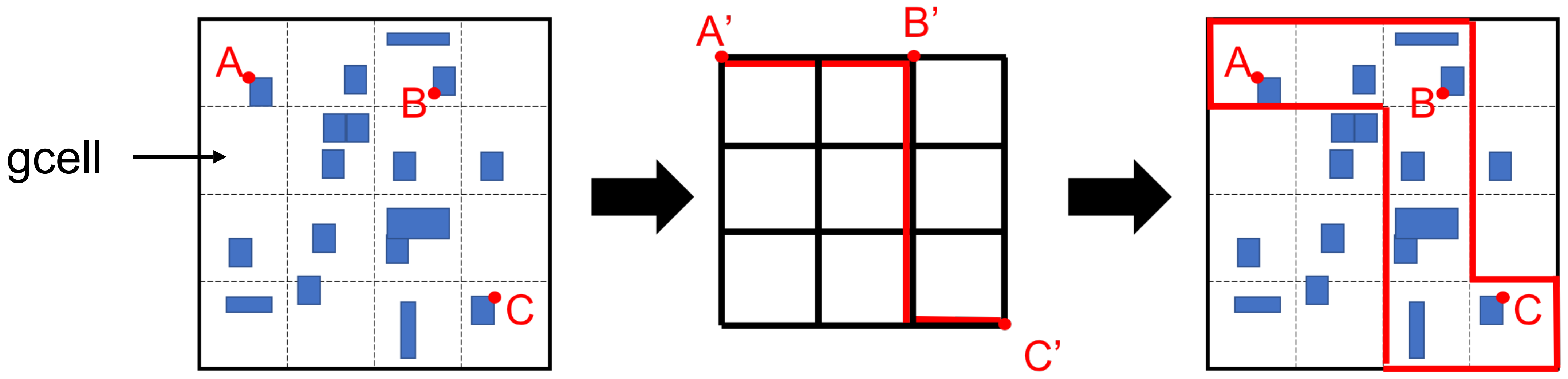
Words	Bits	Bank	Area (um^2)	Perf. (GHz)	Power (uW/MHz)	Leakage (uW)
256	8	1	2744	2.4	1.02	2.45
512	8	1	3145	2.1	1.22	2.70
1024	8	2	7410	1.8	2.36	3.95
8192	8	4	49010	2.2	3.60	13.80

Yale

Parallel global router

new

- The global routing partitions the chip area and forms a grid graph



- The **capacity** of an edge: the number of routing tracks available between two adjacent gcells.
- The **overflow** of an edge: the number of wires that exceeds the capacity.

The global routing problem: For every input net, to find a route that connects all the pins of the net on the grid graph.

❖ Constraint: No overflow (routability).

❖ Objectives: Minimize the total wire length and the number of vias.

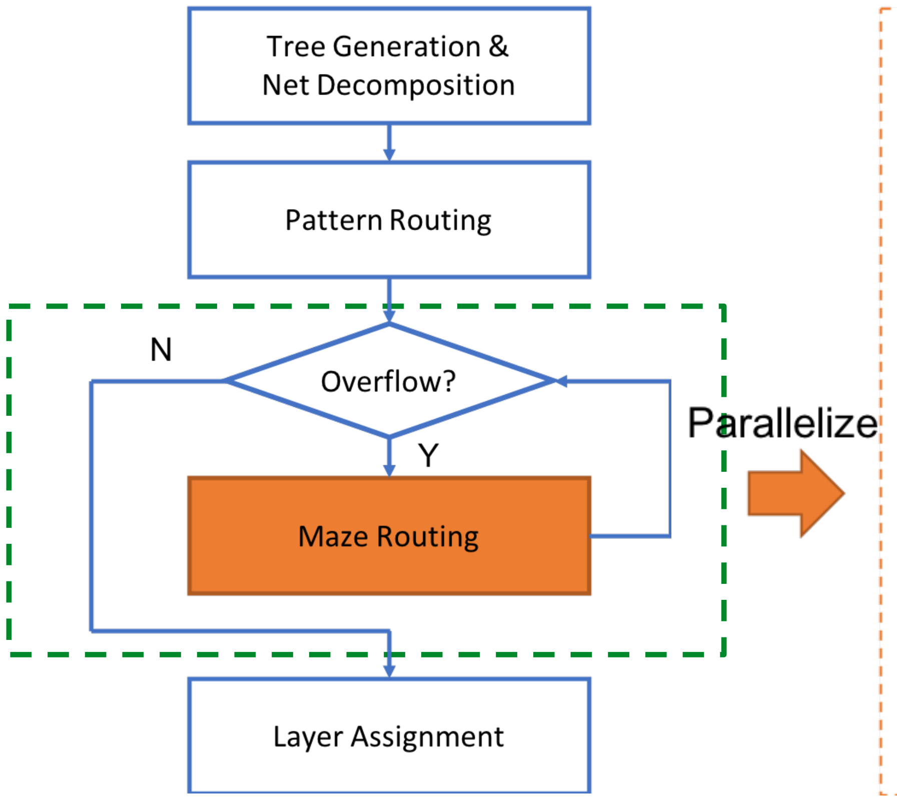
TEXAS The University of Texas at Austin

TEXAS STATE UNIVERSITY The rising STAR of Texas

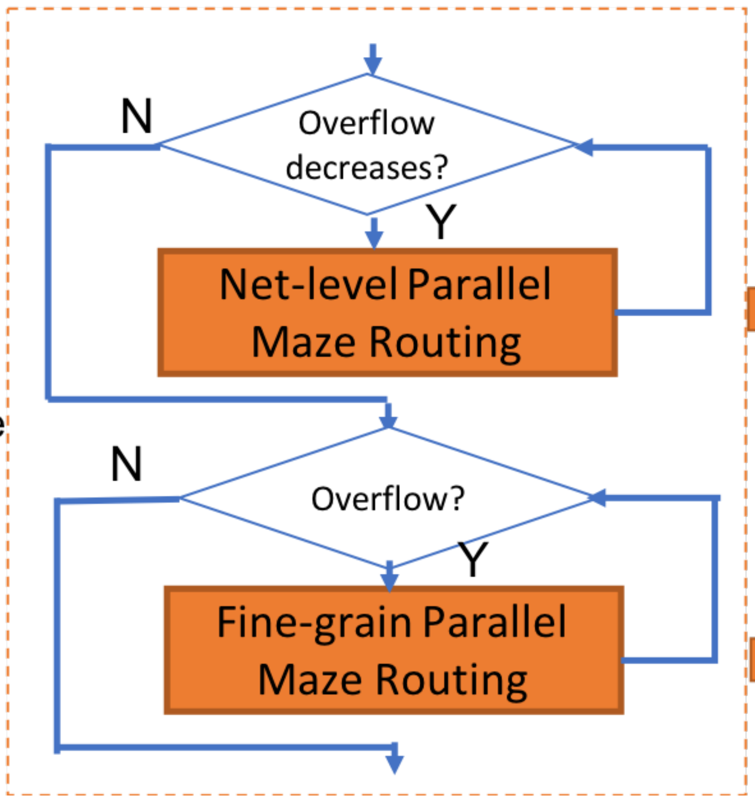
Parallel global router: SPRoute

new

- SPRoute: A Scalable Parallel Negotiation-based Global Router
  - *To appear, ICCAD 2019*
- Exploit nested parallelism and solve the live-lock issue by a novel hybrid parallelization scheme.

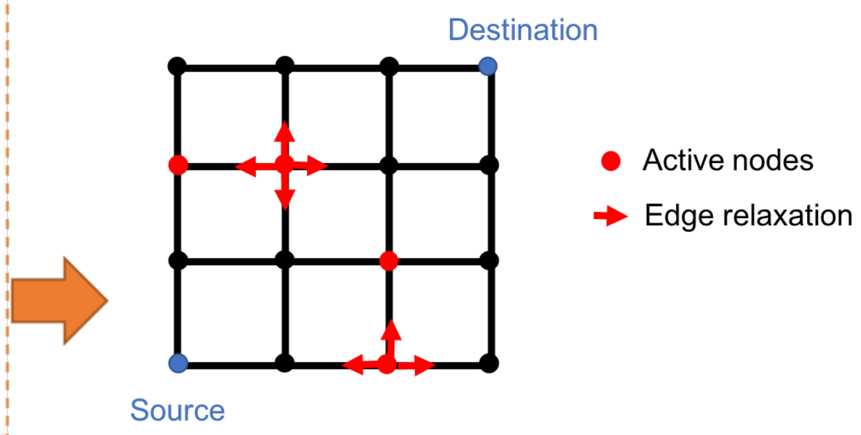


Design flow of SPRoute


N
Overflow decreases?
Y
Net-level Parallel Maze Routing
N
Overflow?
Y
Fine-grain Parallel Maze Routing

Two phase parallel scheme

Each thread works on a net.



All threads work on the same net and each net works on a work item of the frontier

TEXAS The University of Texas at Austin

TEXAS STATE UNIVERSITY The rising STAR of Texas

Parallel global router: SPRoute

new

- SPRoute is implemented in C++ on Galois 4.0; builds on FastRoute
- We evaluate SPRoute on
  - ❖ ISPD 2008 benchmark suite
  - ❖ A 28-core Intel Xeon Gold machine
- Compare with FastRoute 4.1 and NCTU-gr 2.0. (Normalized to SPRoute)

Overflow-free Cases (11 benchmarks)				
	SPRoute	FastRoute 4.1	NCTU-gr 2.0 Fast	NCTU-gr 2.0 Regular
Wirelength	1.0	0.994	1.024	0.999
#Vias	1.0	0.984	1.085	1.022
Time	**1.0**	**11.0**	8.4	10.0

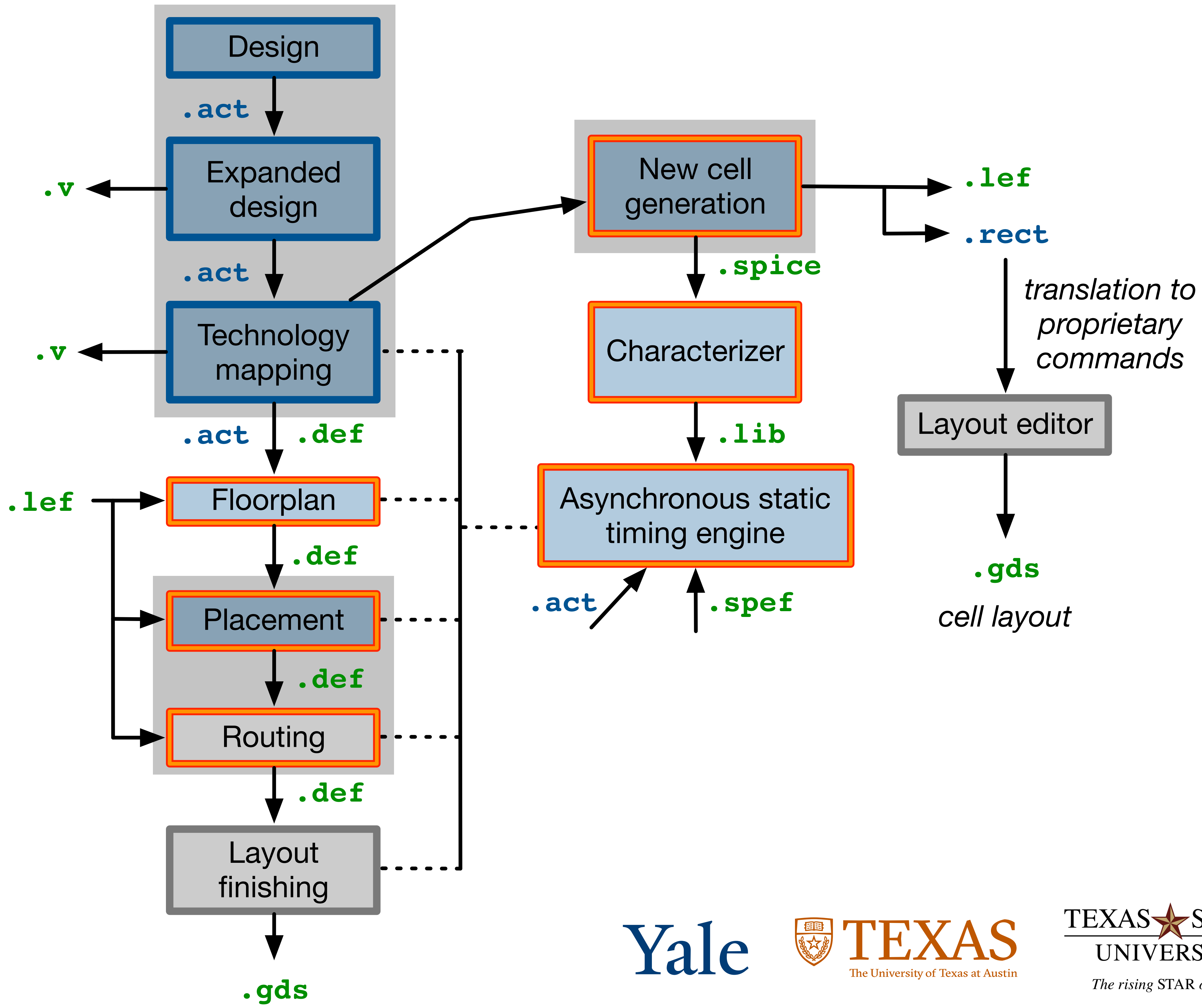
Hard-to-route Cases (4 benchmarks)				
	SPRoute	FastRoute 4.1	NCTU-gr 2.0 Fast	NCTU-gr 2.0 Regular
Total overflow	1.0	0.930	5.17	2.477
Wirelength	1.0	0.994	1.03	0.997
#Vias	1.0	0.989	1.03	0.998
Time	**1.0**	**3.1**	0.37	80.1

TEXAS
The University of Texas at Austin

TEXAS STATE UNIVERSITY®
The rising STAR *of Texas*

Digital flow

new

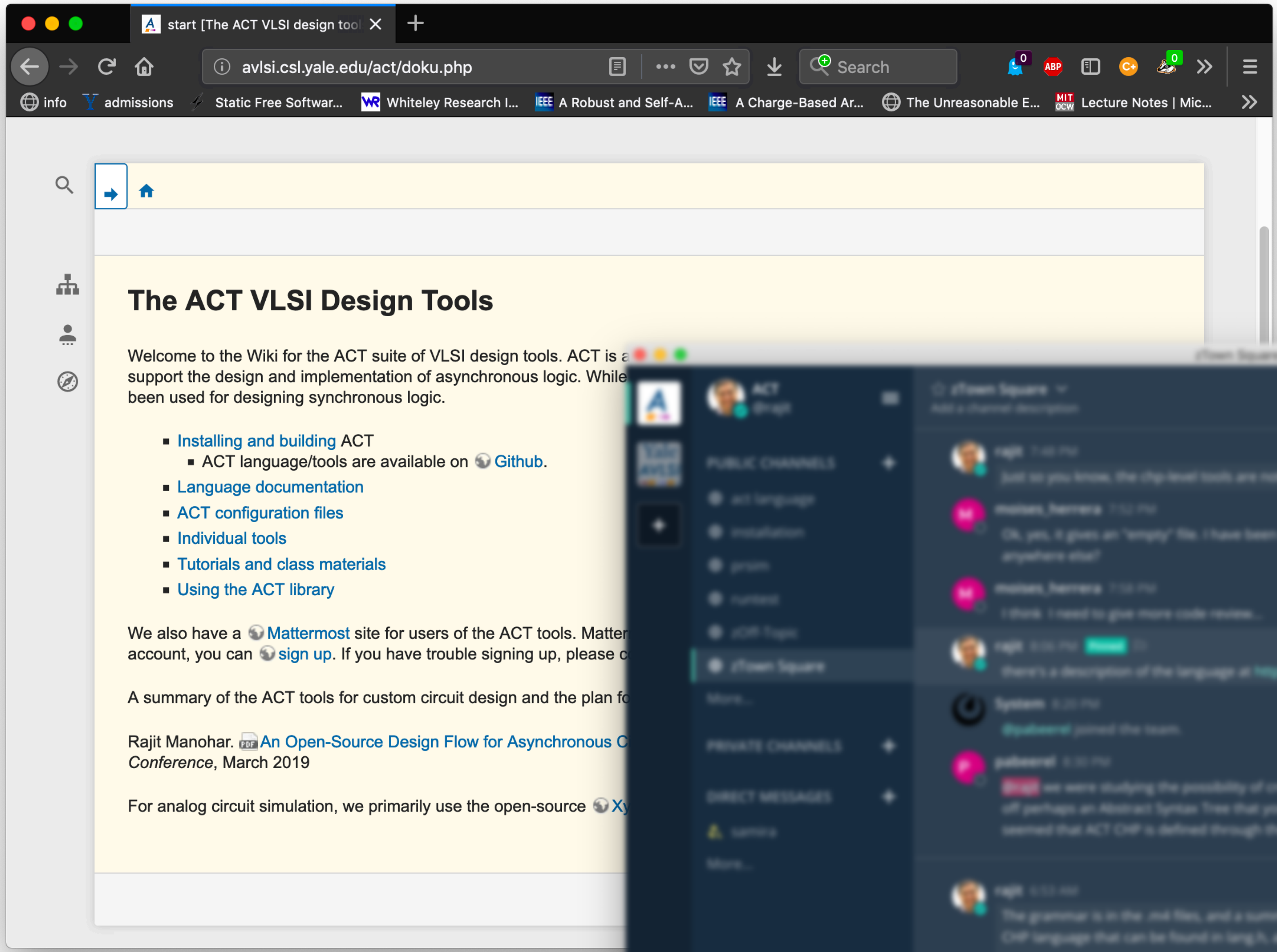


Yale TEXAS The University of Texas at Austin TEXAS STATE UNIVERSITY The rising STAR of Texas

Community

new

http://avlsi.csl.yale.edu/act



The ACT VLSI Design Tools

Welcome to the Wiki for the ACT suite of VLSI design tools. ACT is a
support the design and implementation of asynchronous logic. While
been used for designing synchronous logic.

- Installing and building ACT
  - ACT language/tools are available on Github.
- Language documentation
- ACT configuration files
- Individual tools
- Tutorials and class materials
- Using the ACT library

We also have a Mattermost site for users of the ACT tools. Matter
account, you can sign up. If you have trouble signing up, please c

A summary of the ACT tools for custom circuit design and the plan fo

Rajit Manohar. An Open-Source Design Flow for Asynchronous C
Conference, March 2019

For analog circuit simulation, we primarily use the open-source Xy

NII Shonan Workshop 2019

async research community to use ACT

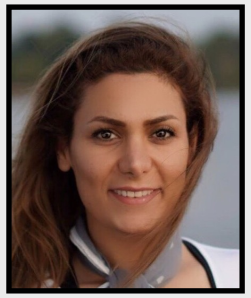
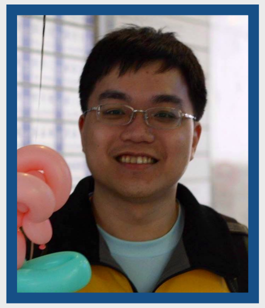
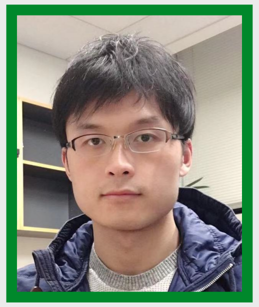
First external contributor: control logic synthesis

Known users: 9 academic groups; 4 companies

Yale

Come talk to us!

Faculty

Asynchronous design & tools	*Parallel data-structures & algorithms*	*GPU-based parallelization*
Rajit Manohar (Yale)	Keshav Pingali (UT Austin)	Martin Burtscher (Texas State)
 Dr. Samira Ataei (memory)	Michael He (routing)	Aarti Kothari (routing)
Wenmian Hua (timing analysis)	 Yi-Shan Lu (timing analysis)	
 Yihang Yang (placement)	Sepideh Maleki (placement)	