`interact`: An Interactive Design Environment for Asynchronous Logic

Jiayuan He, Wenmian Hua, Yi-Shan Lu, Sepideh Maleki, Yihang Yang, Keshav Pingali, Rajit Manohar

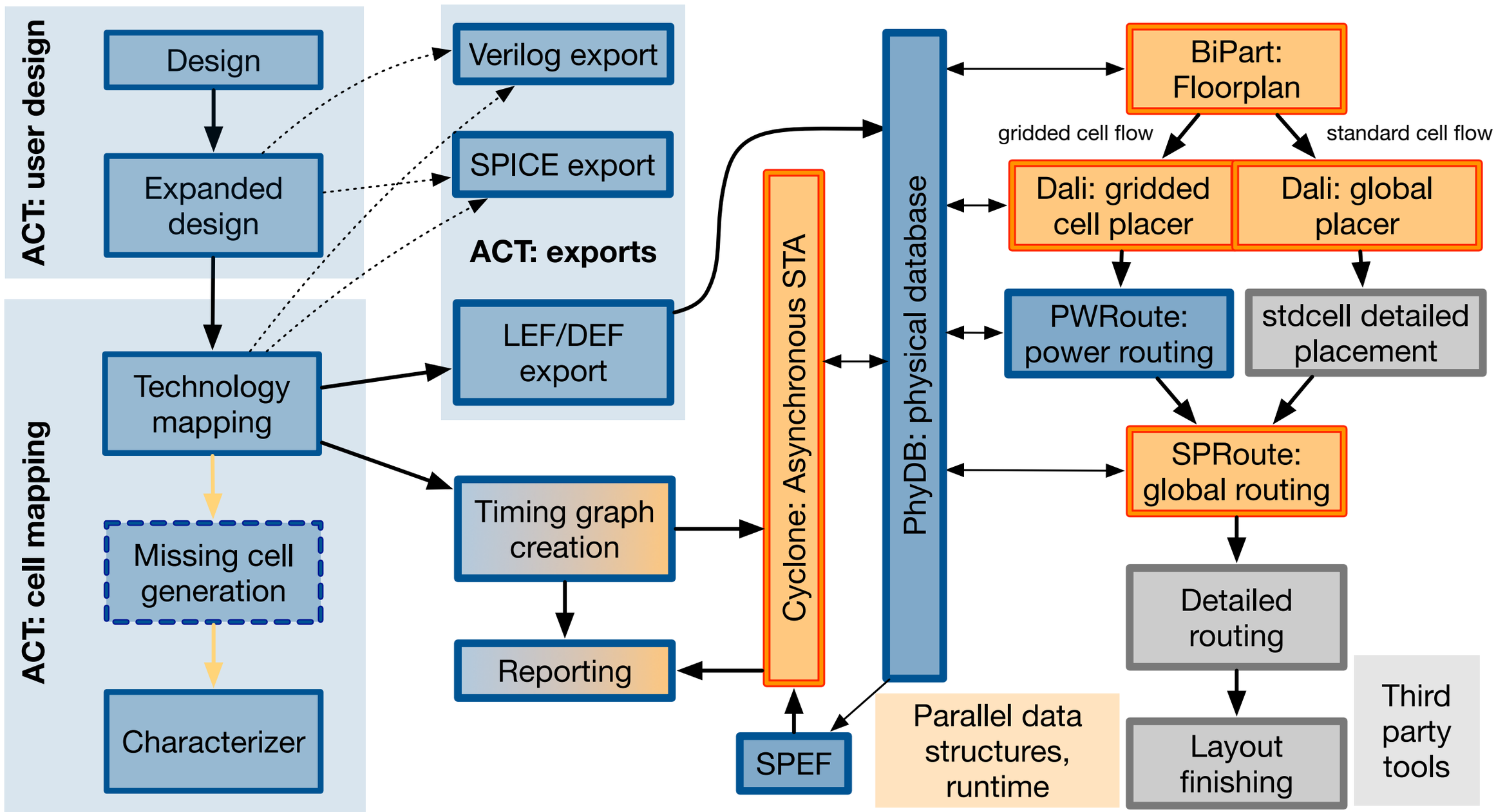
University of Texas at Austin & Yale University

`http://github.com/asyncvlsi/`

Yale

AVLSI

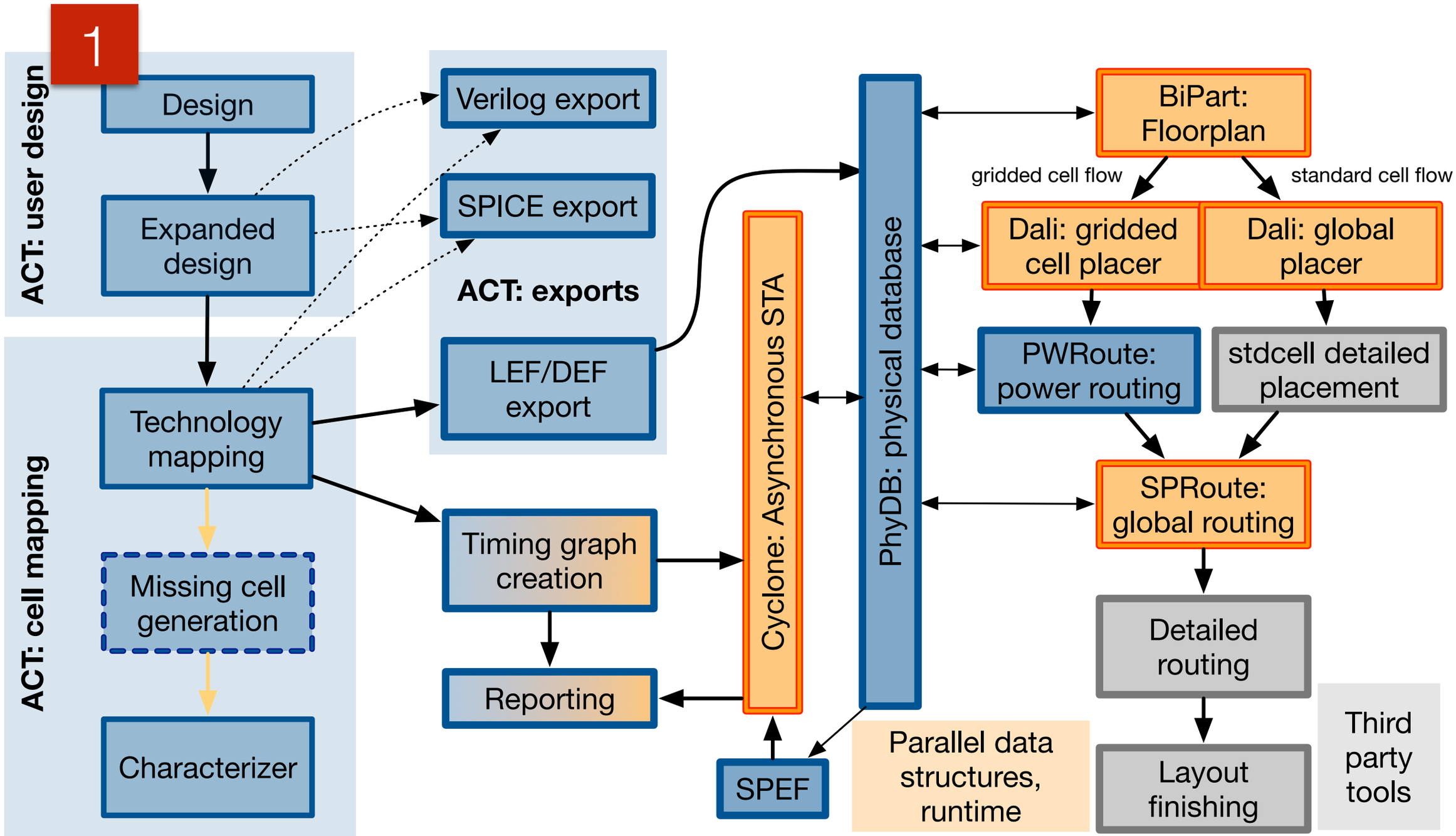
An ASIC flow for asynchronous logic



Yale

AVLSI

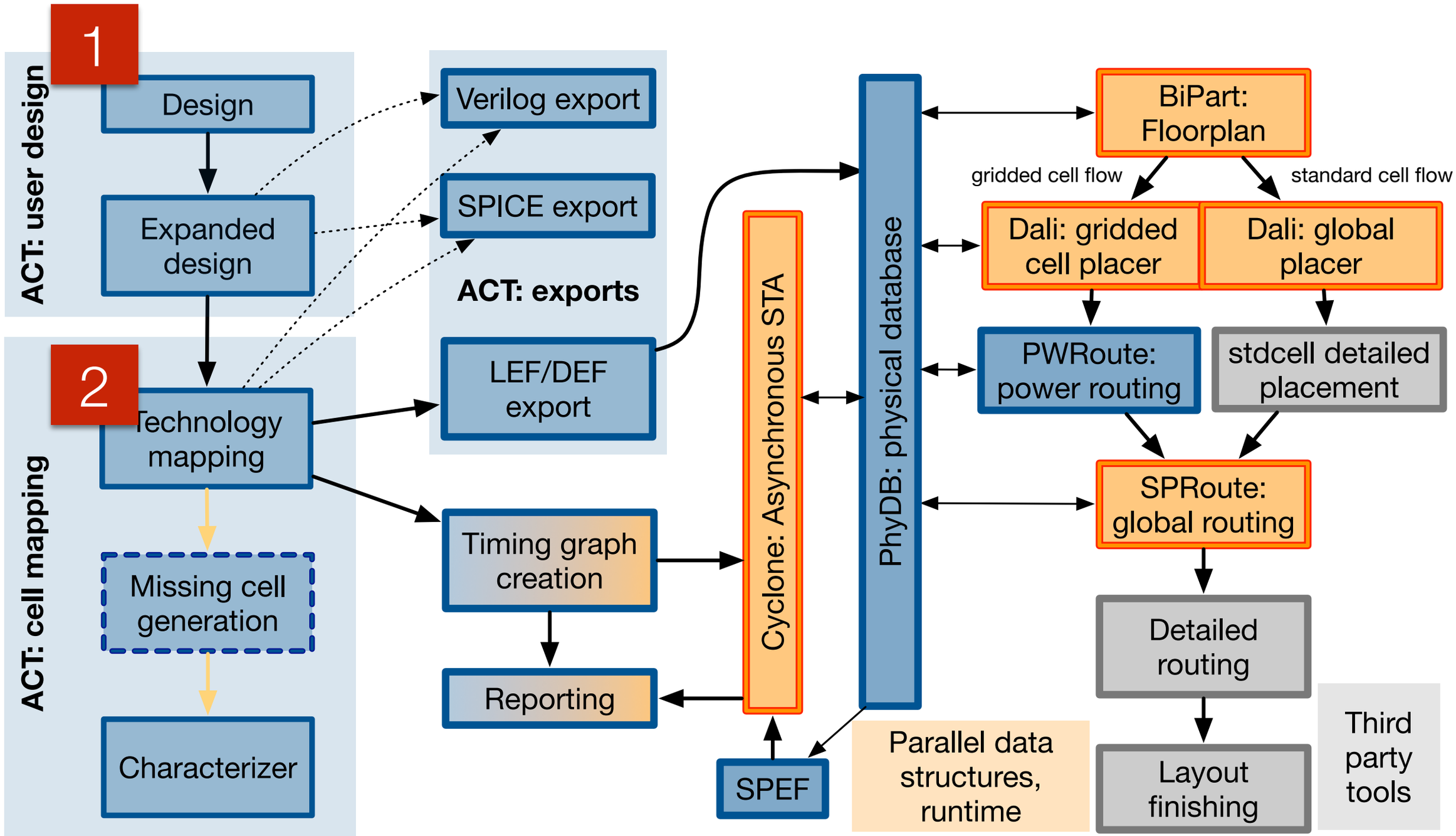
An ASIC flow for asynchronous logic



Yale

AVLSI

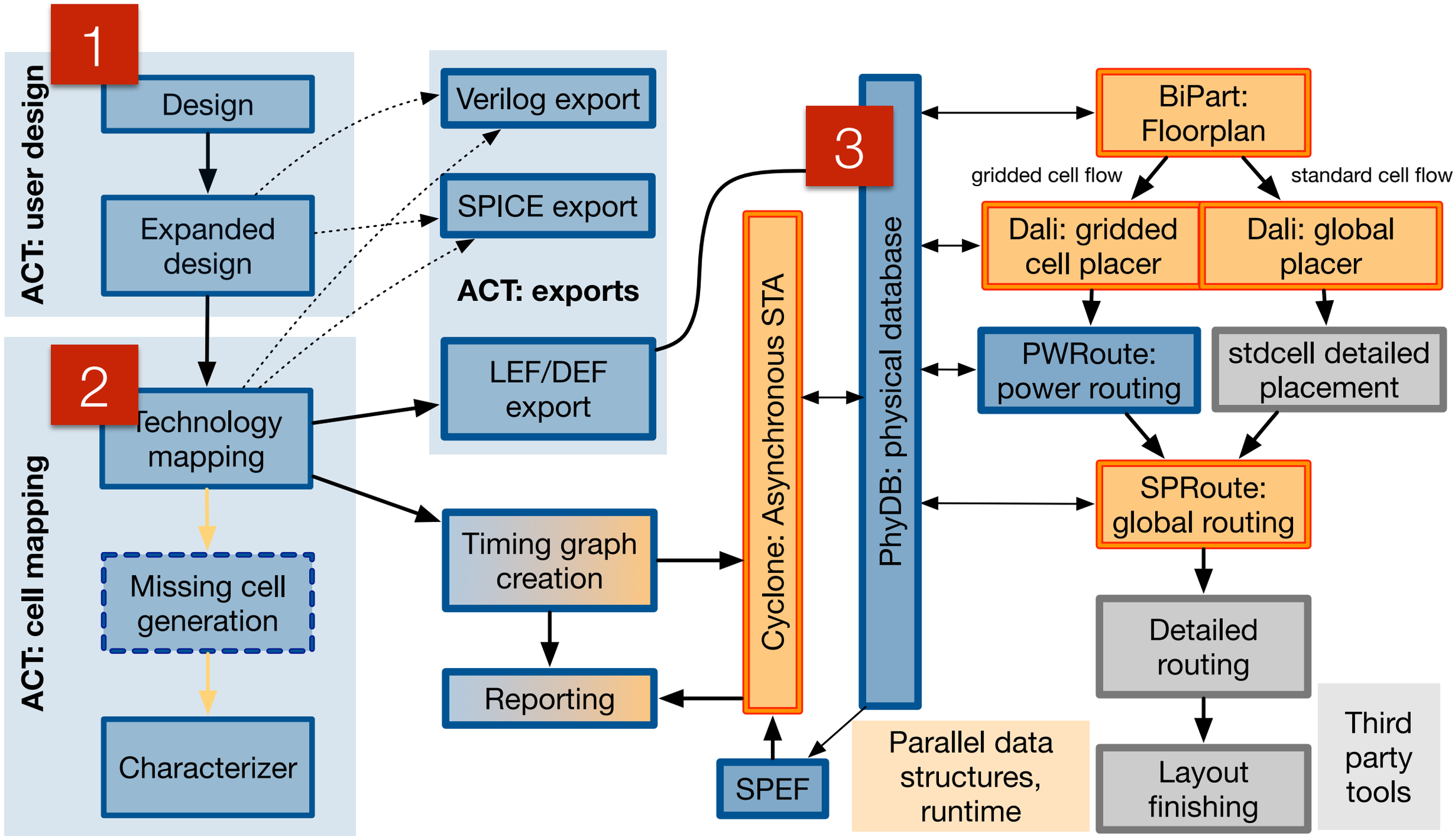
An ASIC flow for asynchronous logic



Yale

AVLSI

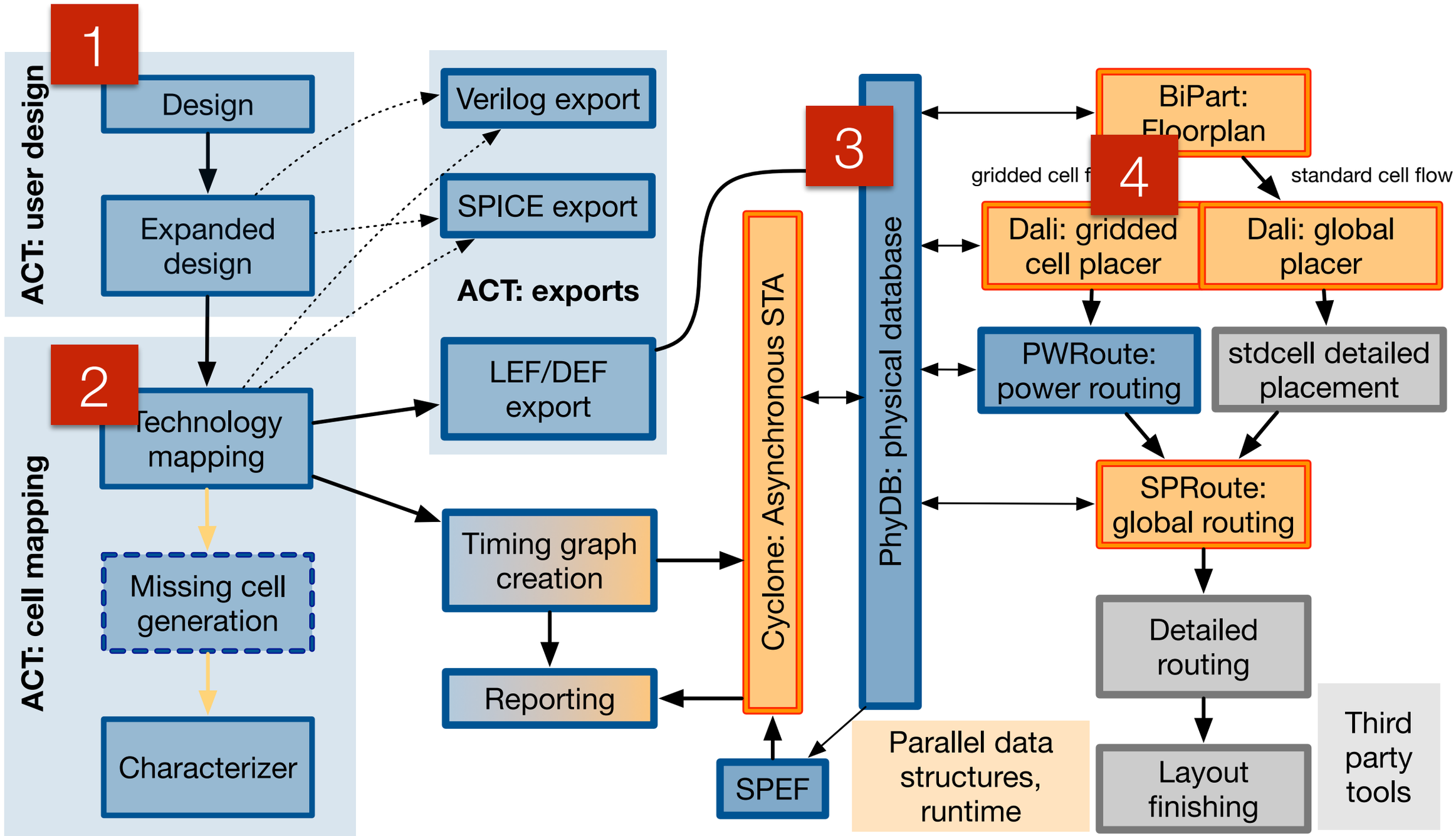
An ASIC flow for asynchronous logic



Yale

AVLSI

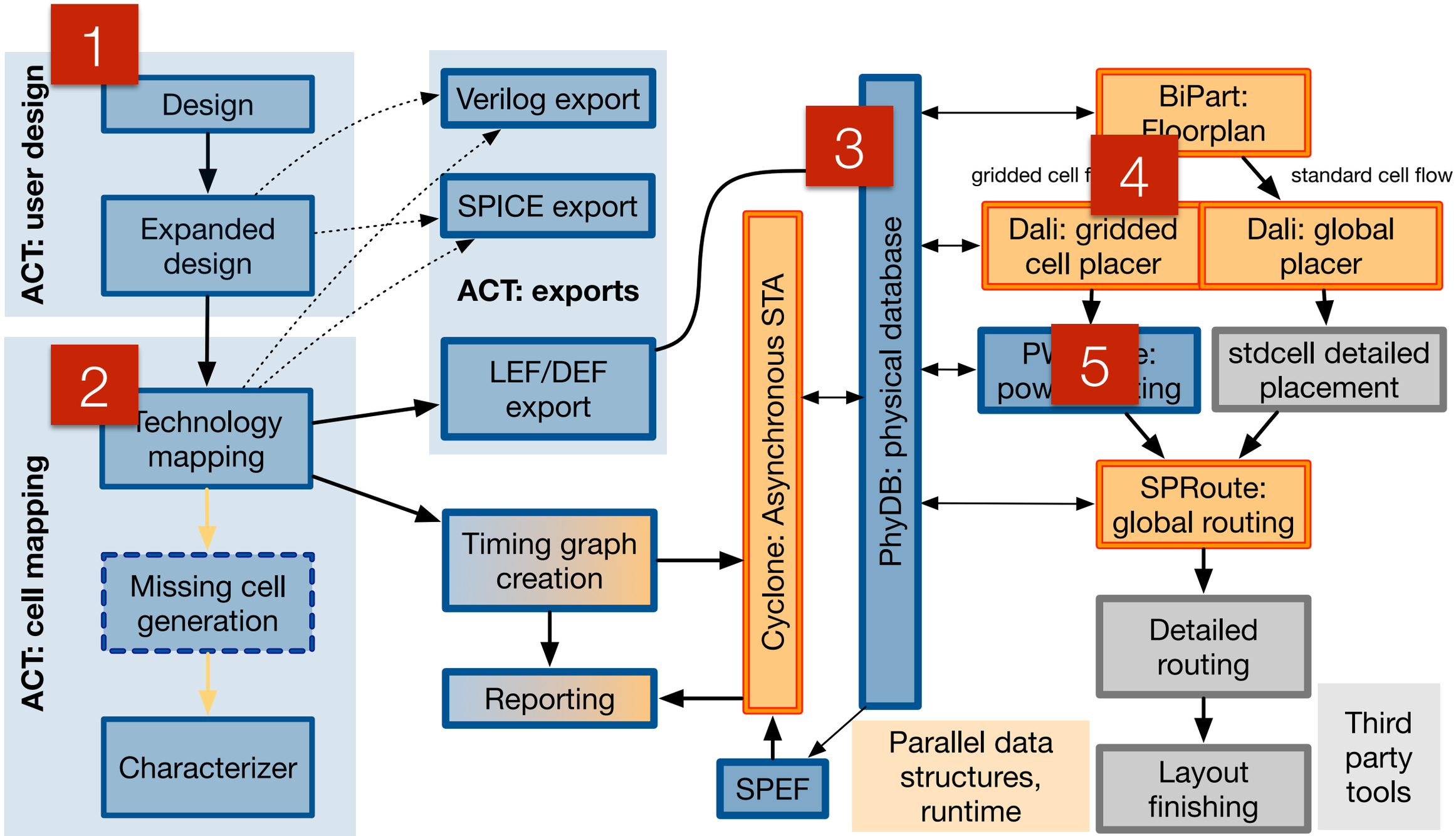
An ASIC flow for asynchronous logic



Yale

AVLSI

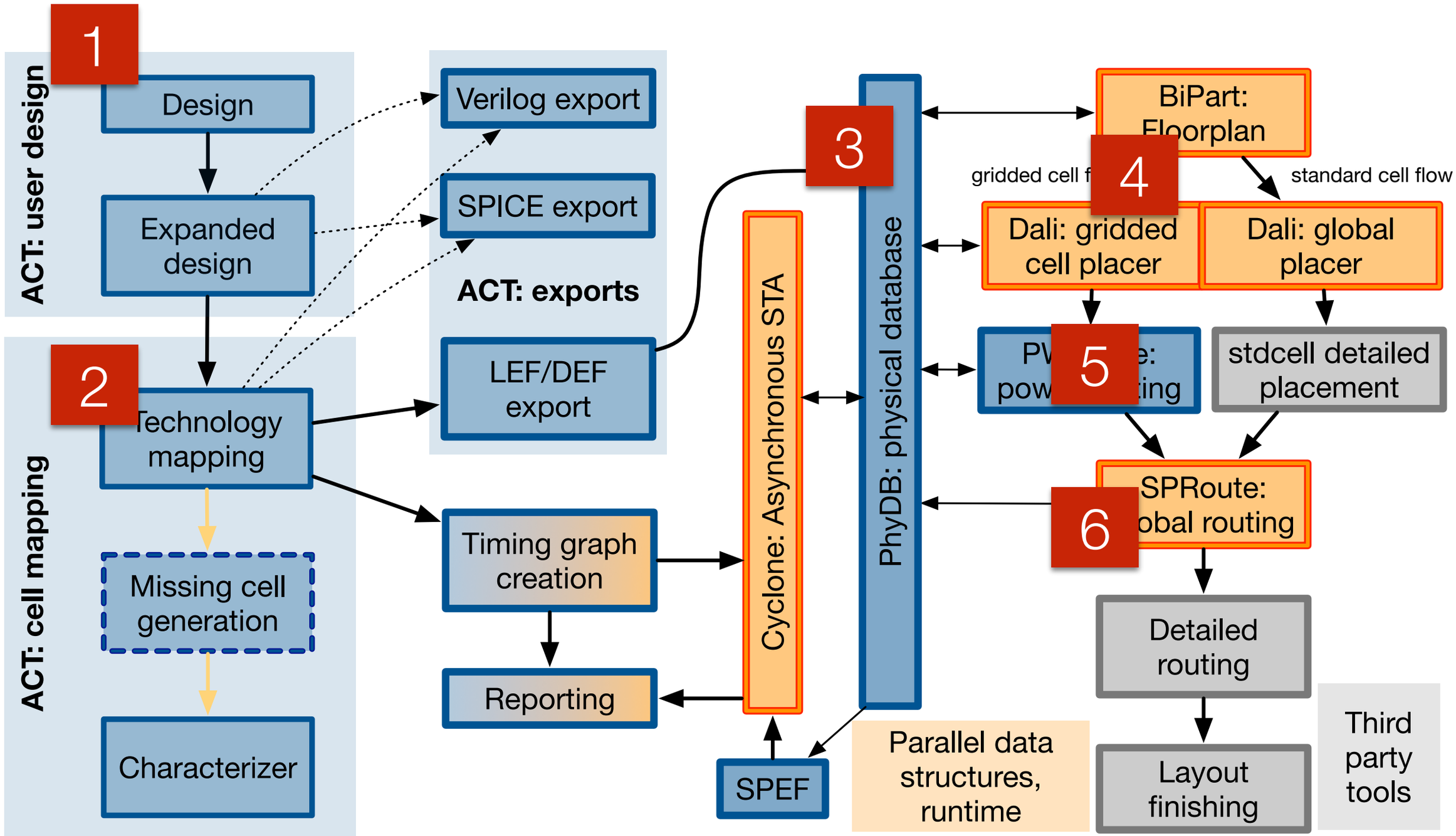
An ASIC flow for asynchronous logic



Yale

AVLSI

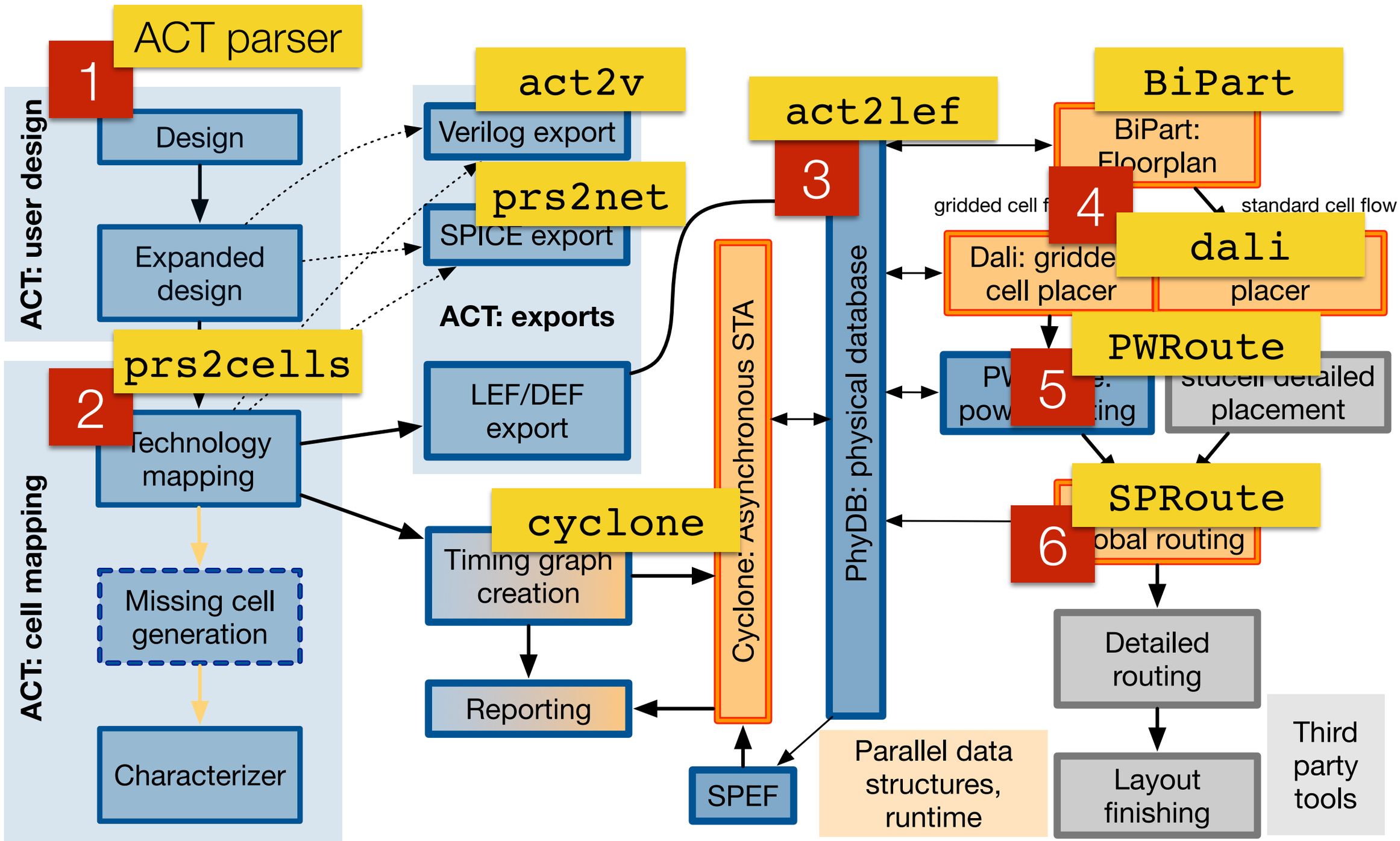
An ASIC flow for asynchronous logic



Yale

AVLSI

Previous setup: separate tools per step + Makefile



Yale

AVLSI

Core components of the flow

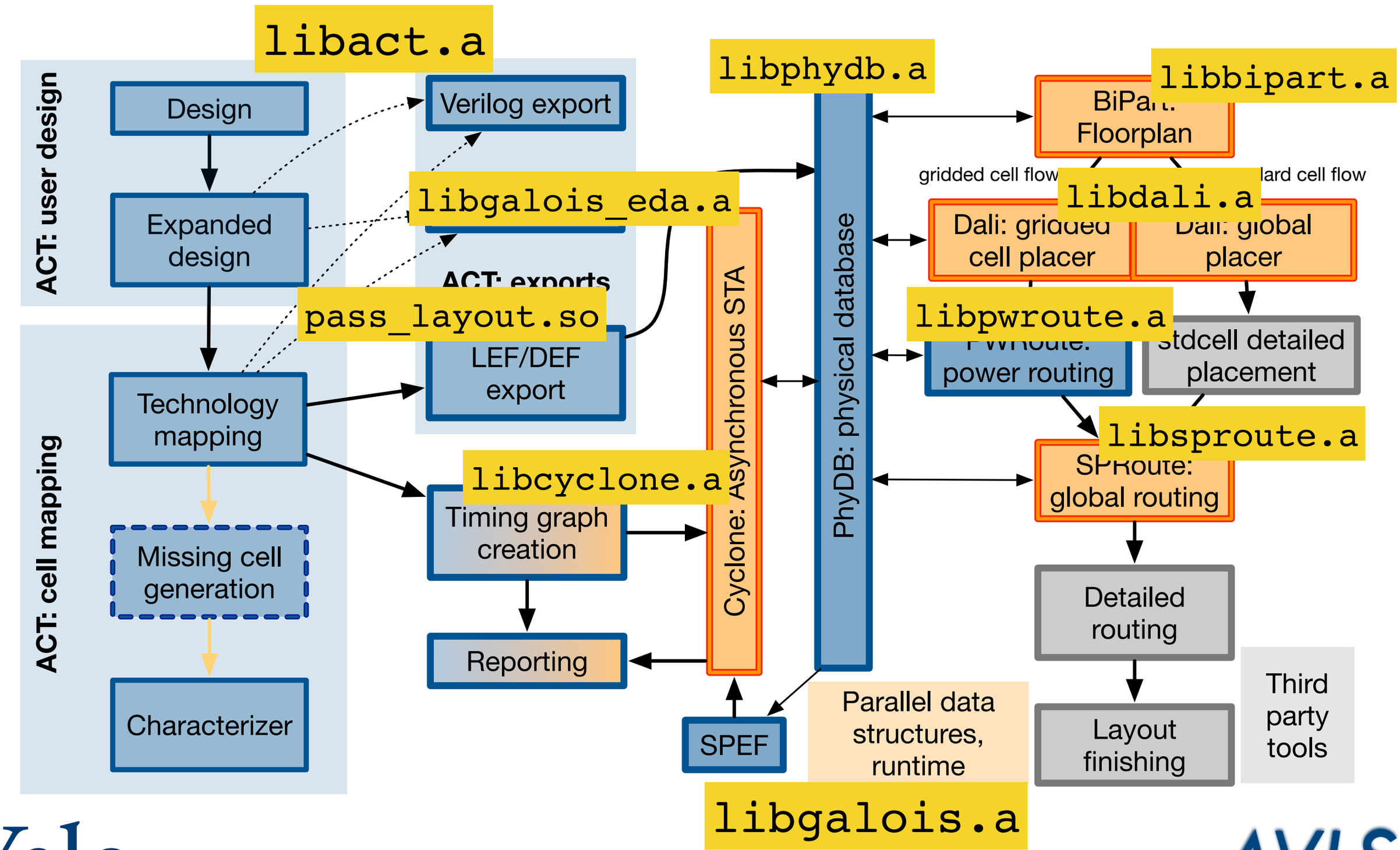
- ACT library: front-end tools
  - ❖ Design and netlist management
  - ❖ User design at the behavioral, gate, and circuit abstraction
- Timer library: both front-end and back-end tools
  - ❖ Timing graph creation
  - ❖ Cyclic analysis, critical cycle, and incremental update
  - ❖ Timing constraints and violating paths
  - ❖ Parasitic manager
- PhyDB library: back-end tools
  - ❖ Physical database
  - ❖ LEF/DEF import/export
  - ❖ Parasitics interface to timer

Modular approach to development

- Our project: an academic environment with students
  - ❖ Difficult to start the project by using a unified data structure (… no previous integrated async flow to mimic)
  - ❖ Need to enable parallel development by different students

- The solution
  - ❖ Each tool was converted to a **library**
  - ❖ Primary input/output from tool must read/write from core databases
    - ‣ ACT for design/netlist
    - ‣ PhyDB for physical design
  - ❖ **Netlist adaptor** : an abstract API that simplifies tool development
    - ‣ Provided by the ACT library
    - ‣ Used to implement basic netlist functions
    - ‣ Tools can map ACT netlist objects to/from tool-specific representation

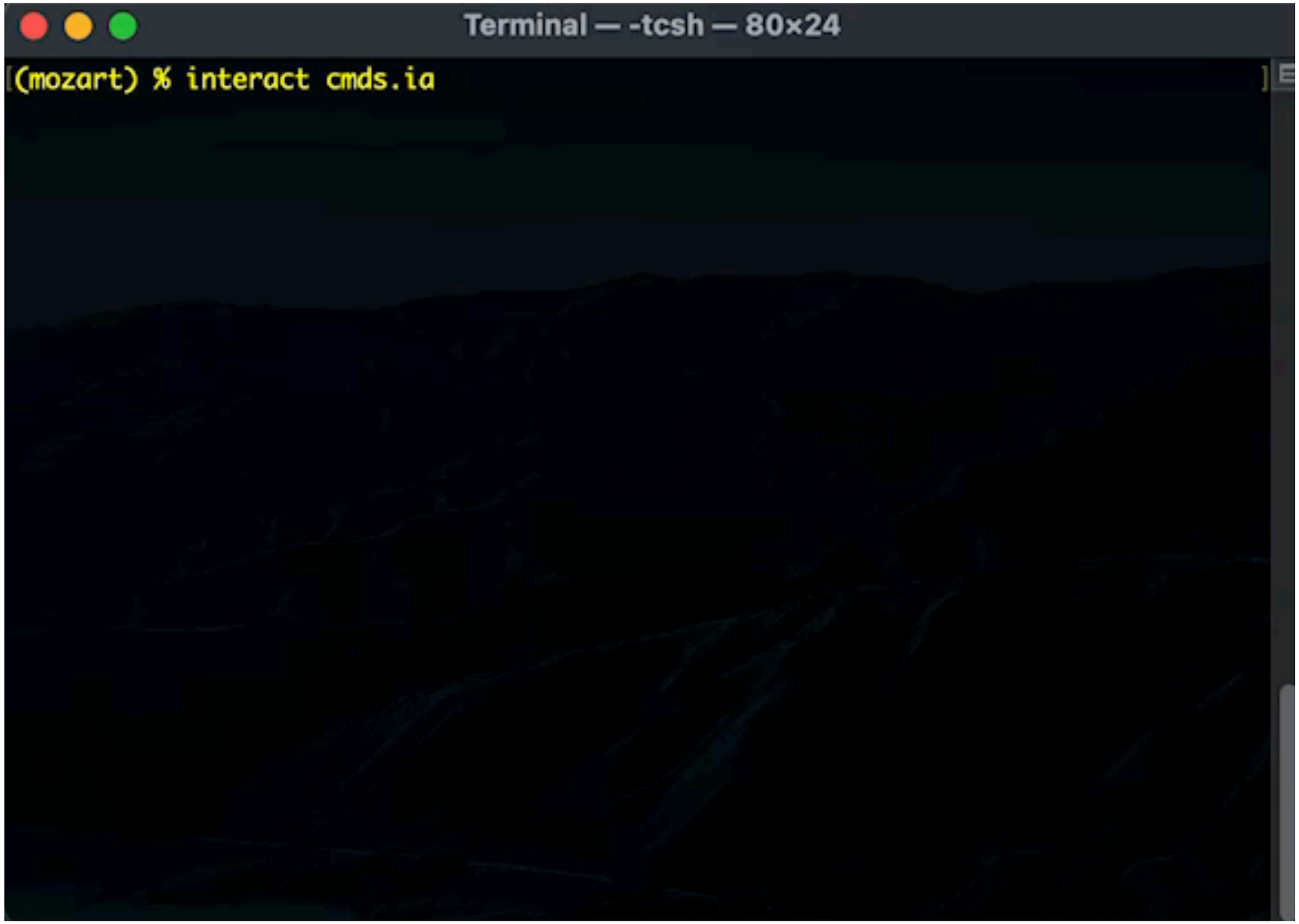
Libraries and our design flow



Yale

AVLSI

Example



Summary

- `interact` is an integrated tool to implement asynchronous circuits
  - ❖ Design management, cell mapping
  - ❖ Placement, global routing
  - ❖ Timing analysis

- Core databases: ACT (design) and PhyDB (geometry)

- Decoupled software architecture to simplify integration
  - ❖ Netlist adaptor permits language-agnostic tool development