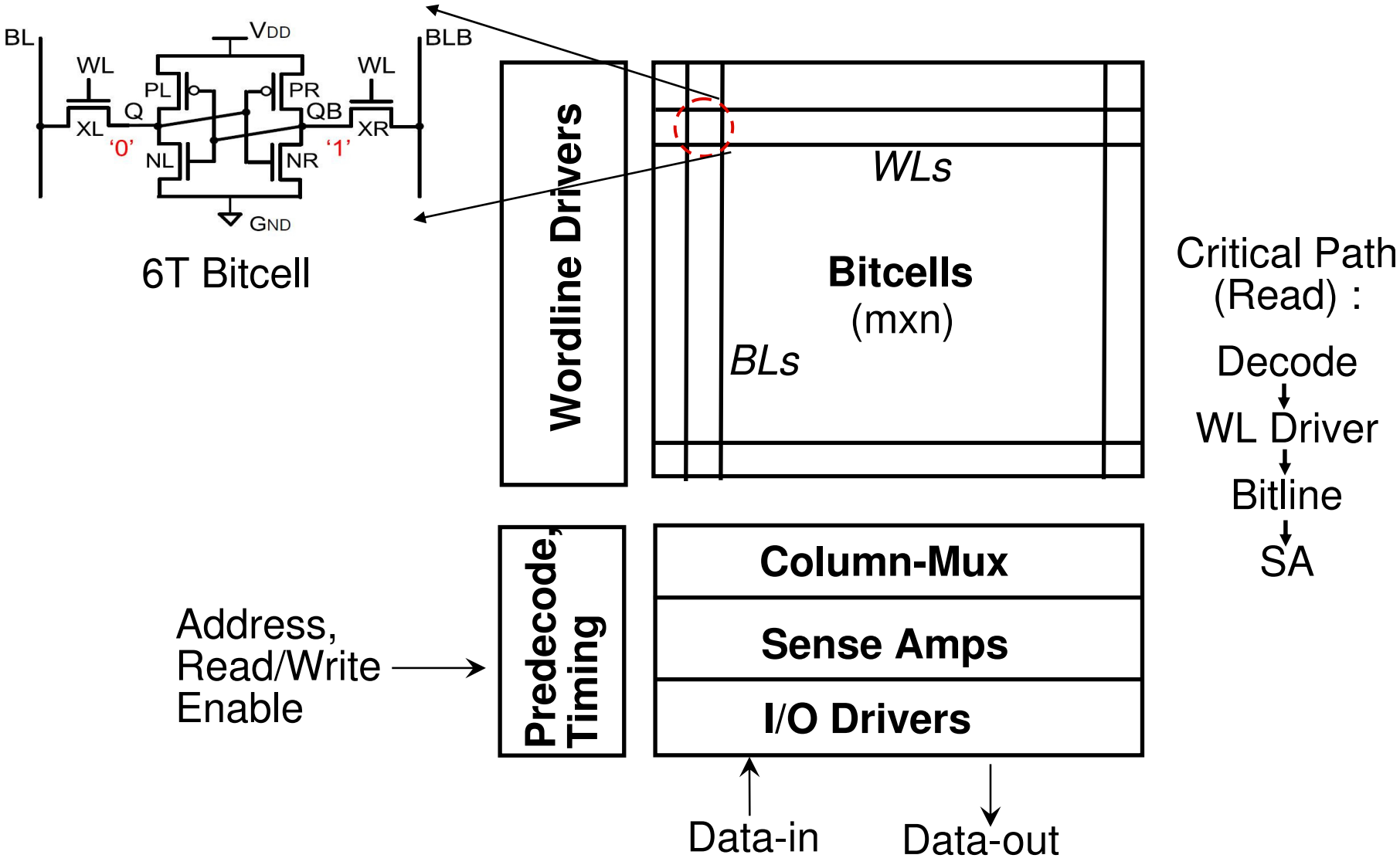
Virtual Prototyper (ViPro): An Early Design Space Exploration and Optimization Tool for SRAM Designers

Satyanand Nalam, Mudit Bhargava, Ken Mai,
Benton H. Calhoun

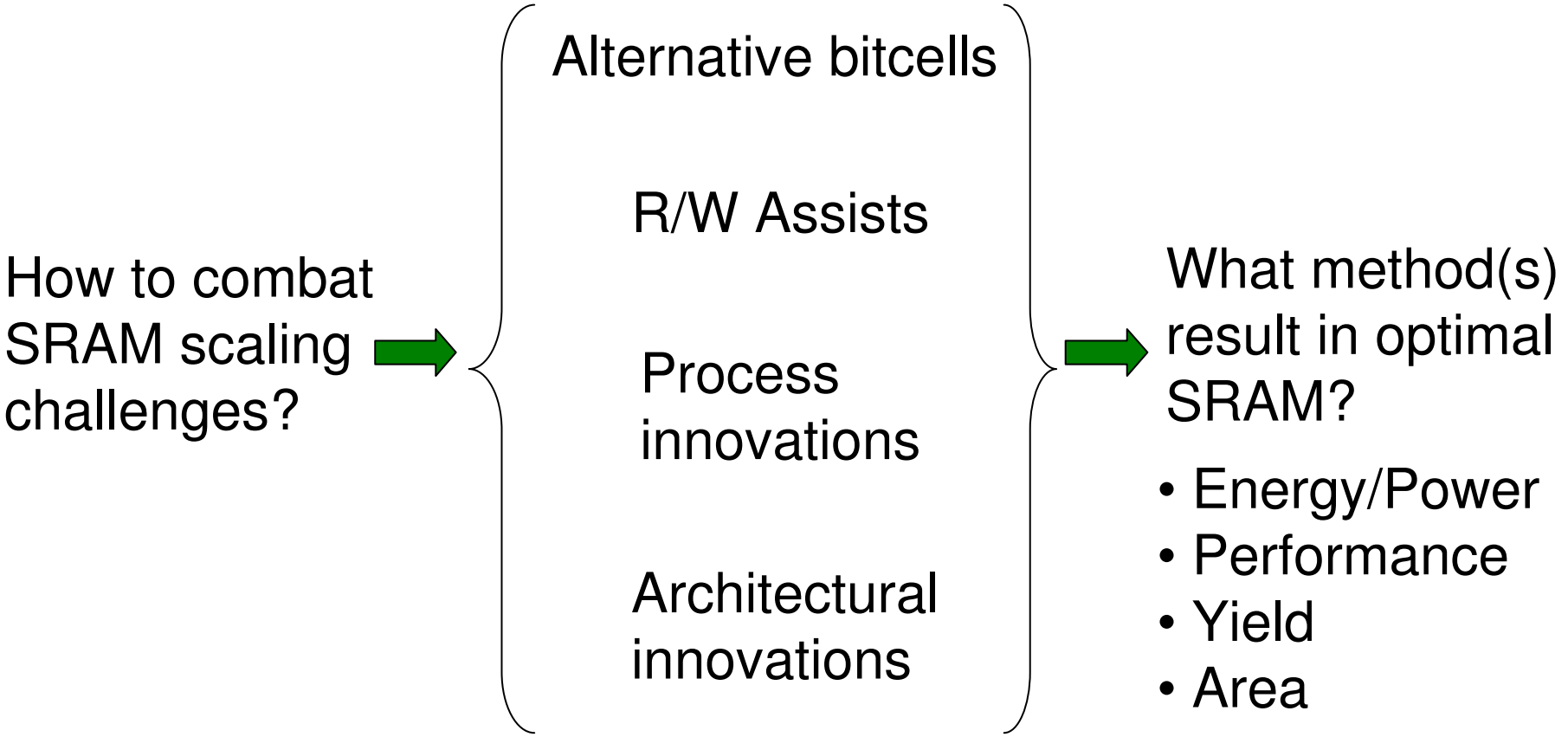
Overview

- Introduction and motivation
- Flow/Design methodology
- Tool components
  - Characterizer
  - Hierarchical SRAM model and “Meta” Compiler
- Usage Example
- Conclusions

SRAM – single bank macro



Optimal SRAM design



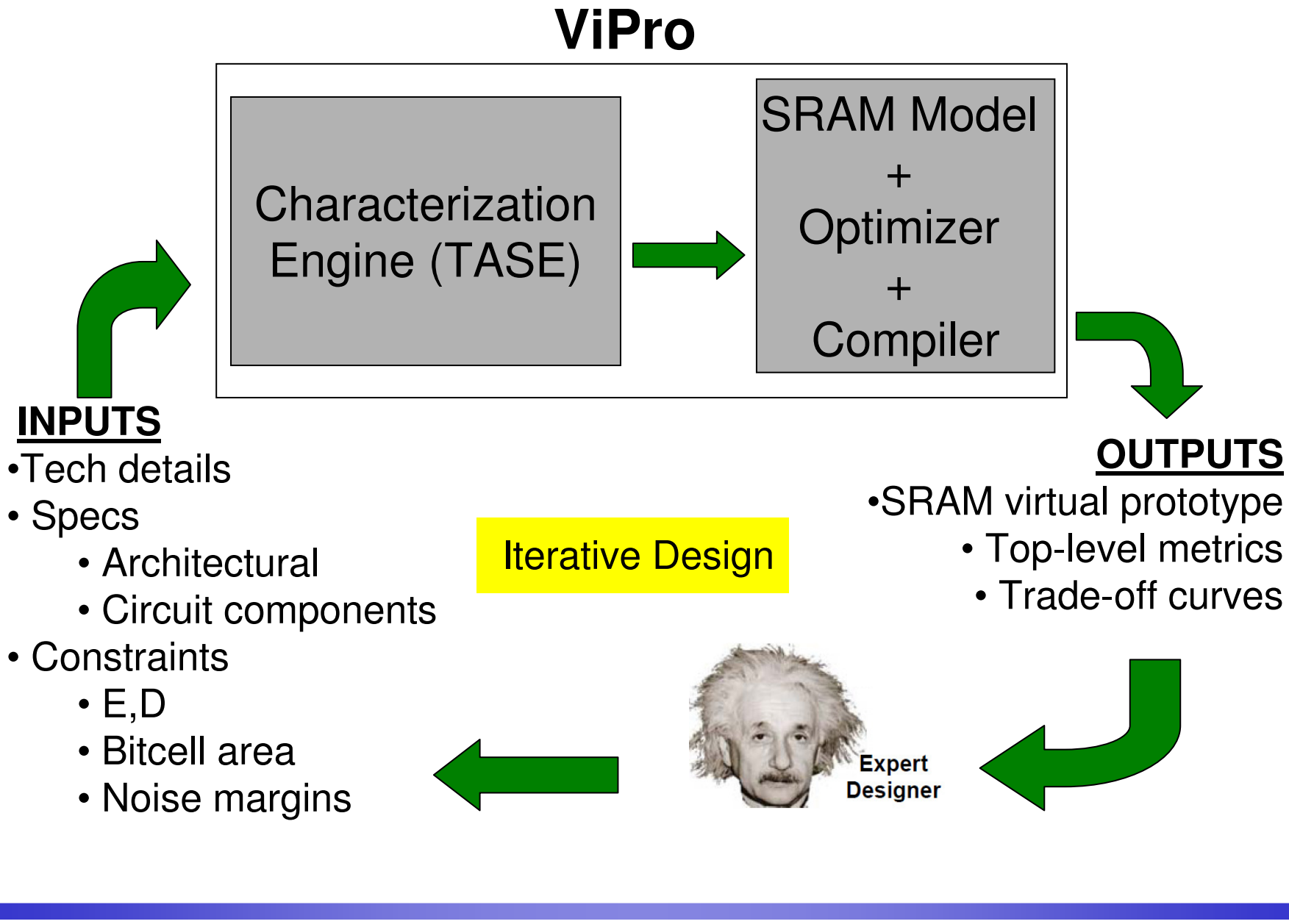
Virtual Prototyping tool (ViPro)

1. Base-case SRAM

2. Quick re-optimization for rapid design space exploration – by using variable levels of detail

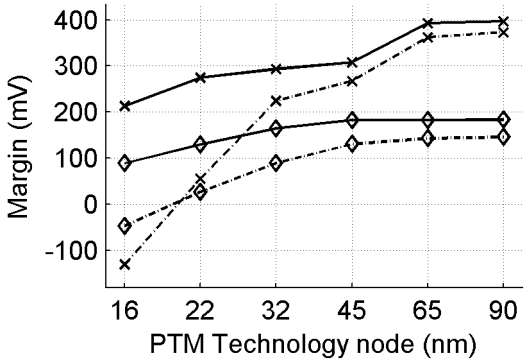
Key Idea: Iterative optimization through designer intervention

Design Methodology using ViPro



Technology Agnostic Simulation Environment (TASE)

Observe trends, “port” circuit analysis



[Nalam et al, ICCD ’09]

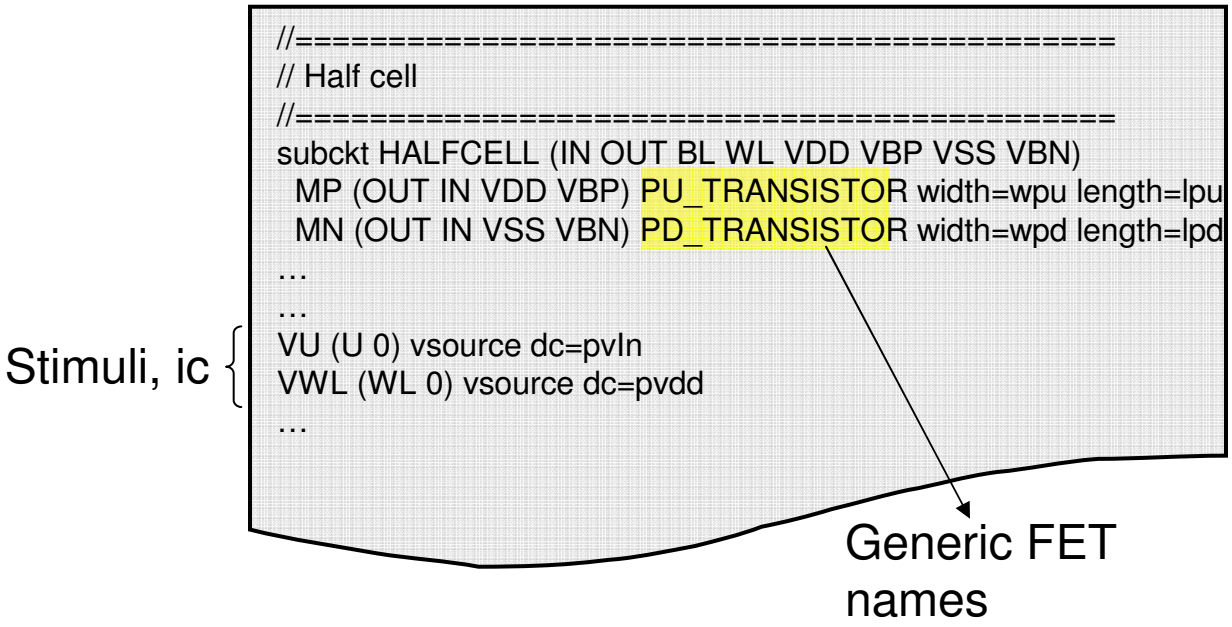
Use 1: Stand-alone

TASE

Use 2: Part of ViPro

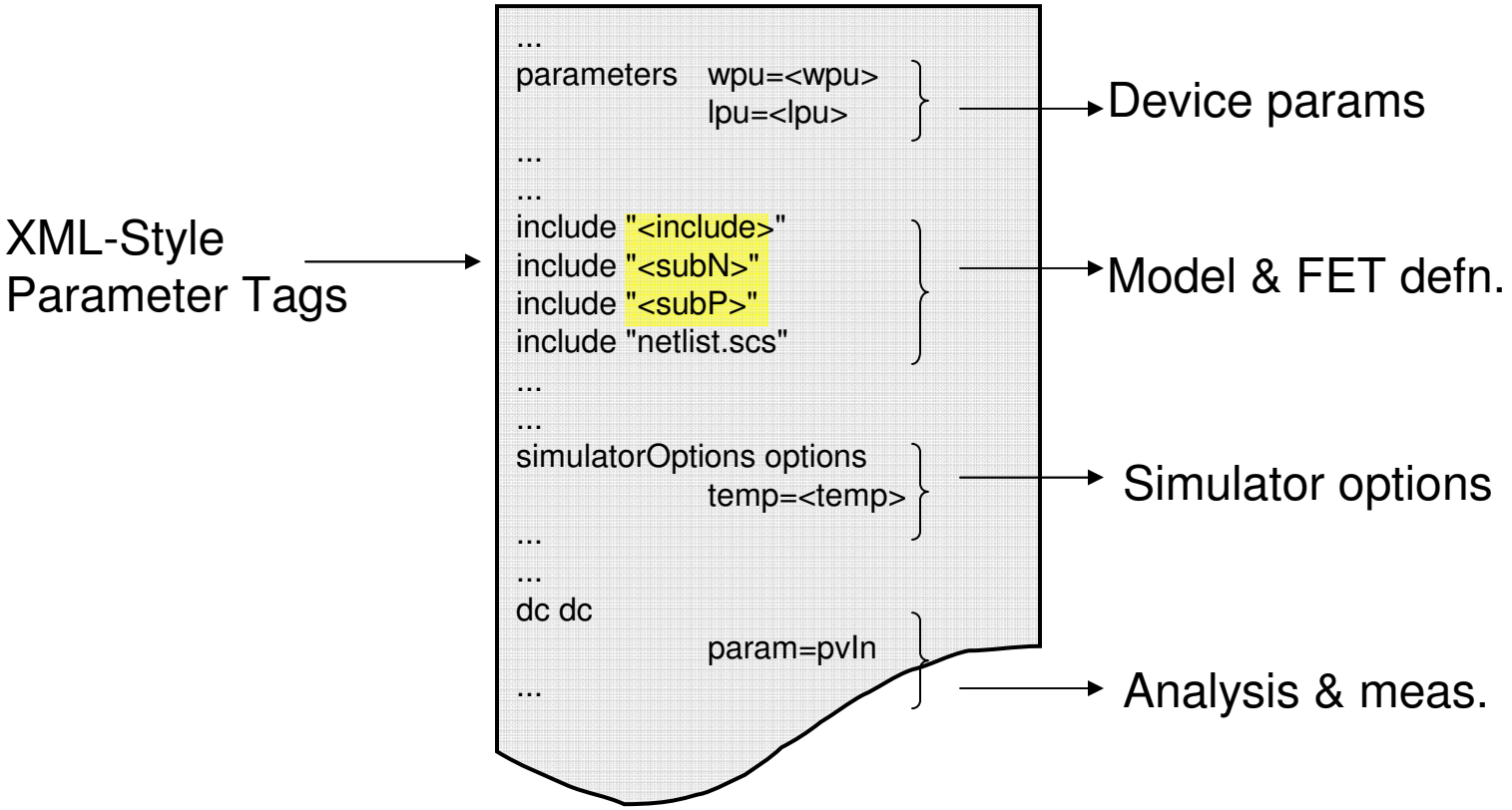
Characterize SRAM components in terms of Power and performance

TASE: Example template (Spectre)



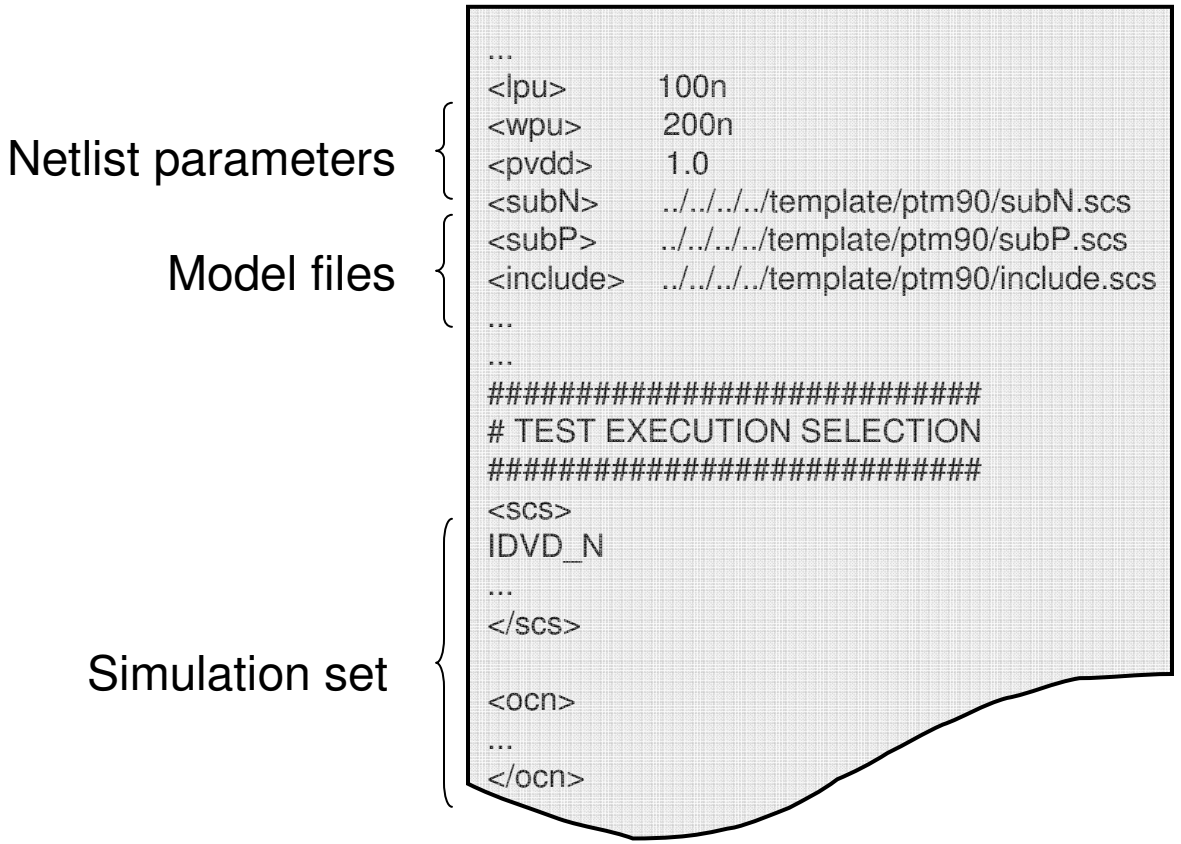
Sub-component template netlist

TASE: Example template (Spectre)



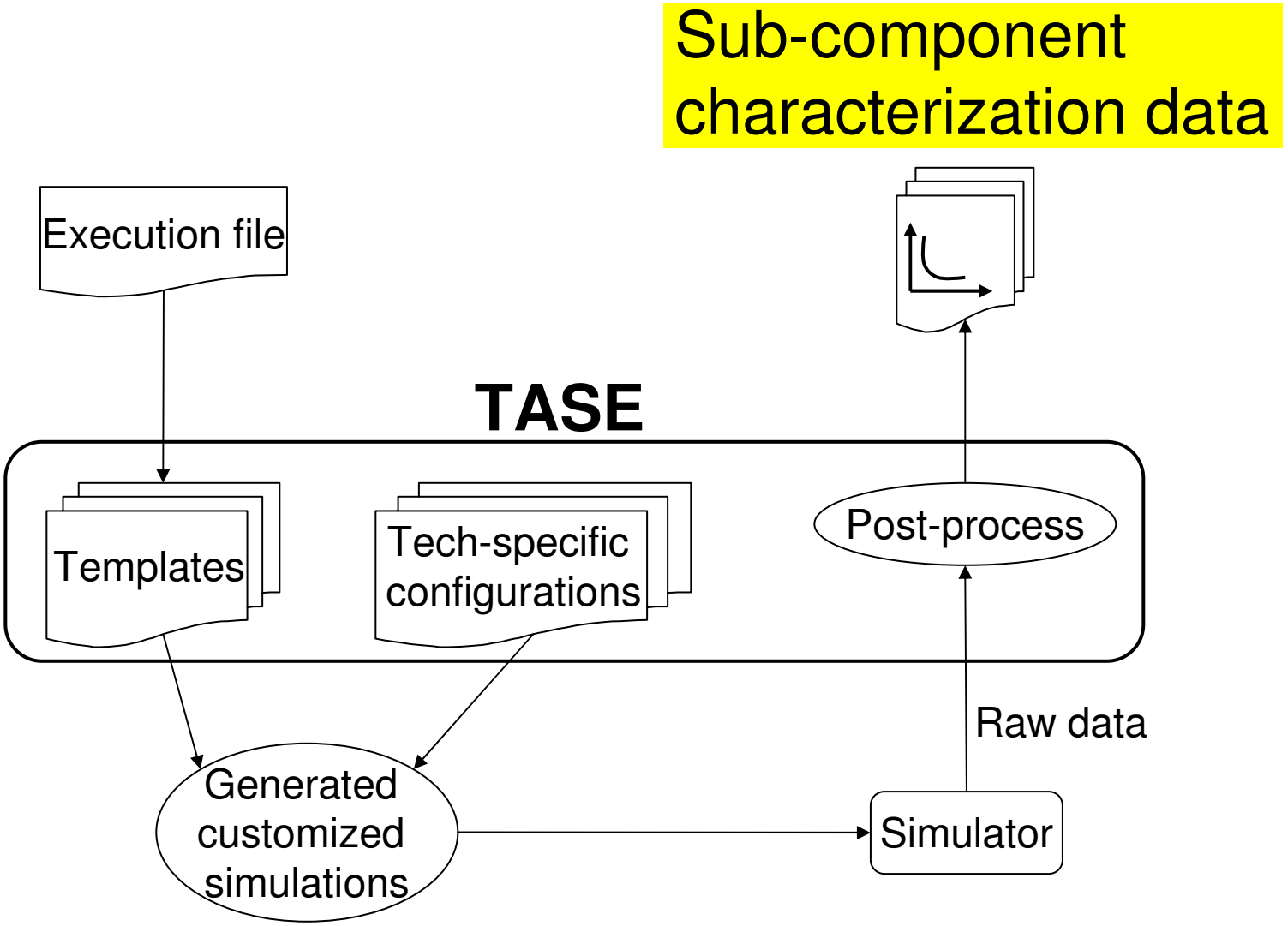
Technology-agnostic characterization sweeps

TASE Example Execution file

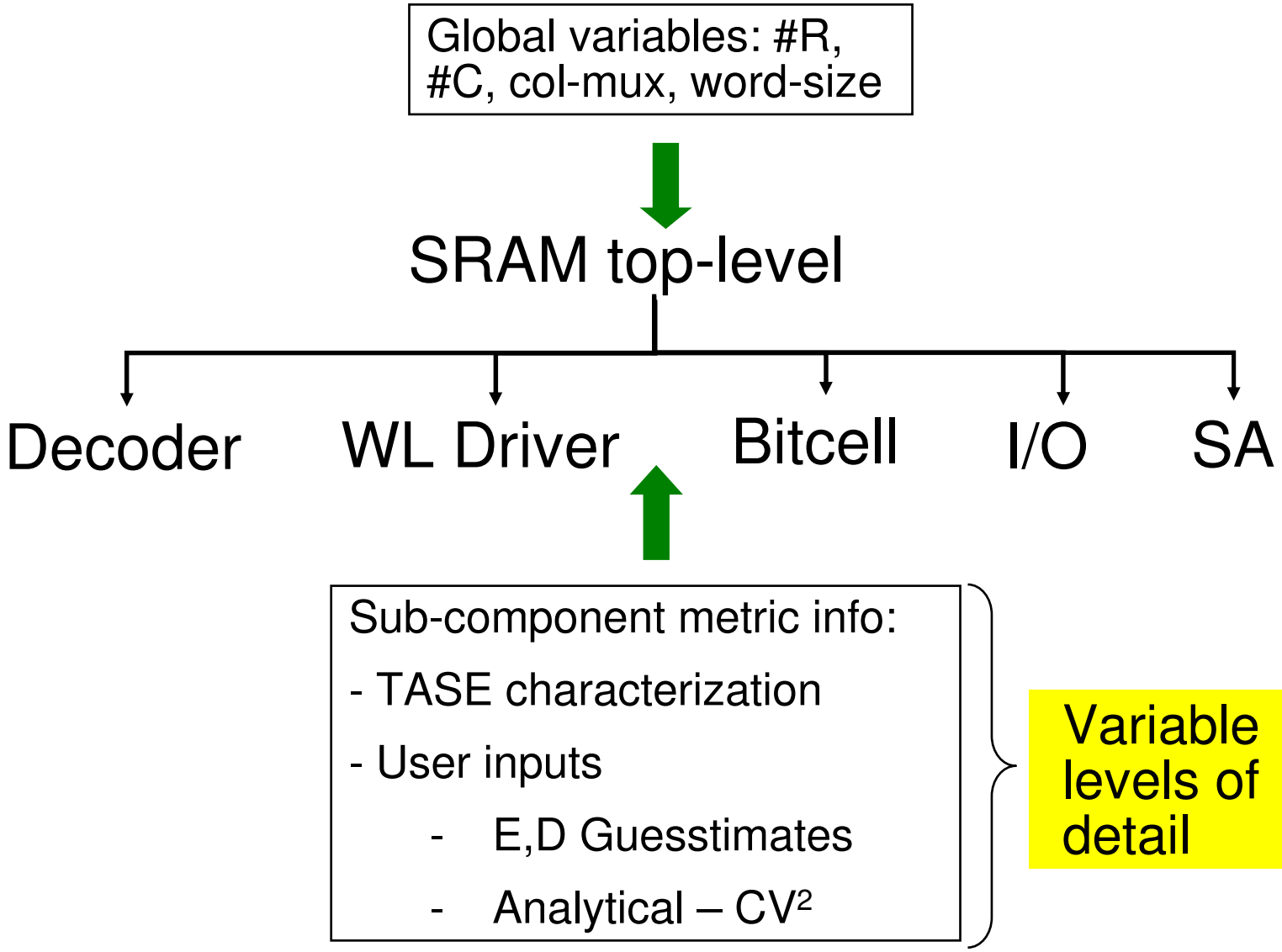


Tech-specific parameter tag defns. in exec. file

TASE: Characterization Flow



Hierarchical SRAM model



Hierarchical SRAM model

- MATLAB implementation
- Component objects
  - Properties
    - Local – E.g. SA offset
    - Global/Inherited – E.g. number of rows
  - Methods
    - E, D wrapper functions
      - TASE characterized output
      - User provided guesstimates/ analytical expressions
- Co-optimization using global properties inherited from a parent class.
- Example in next slide.

Co-optimization example

Bitcell object:

Local property – device width

Inherited property - #rows

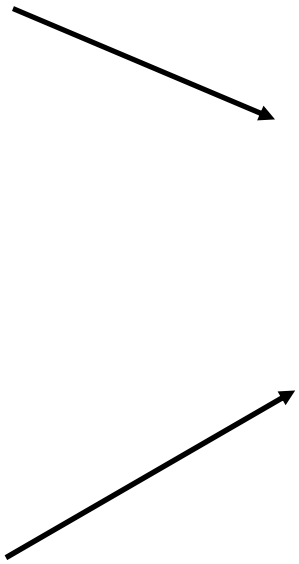
E/D = f(local & inherited)

SA object:

Local property – input device width, offset

Inherited property - #rows

E/D = f(local & inherited)



- Capture bitcell, SA dependencies
- Co-optimize #rows
  - Change SA input device width to adjust offset
  - Increase bitcell pass-gate width to improve drive

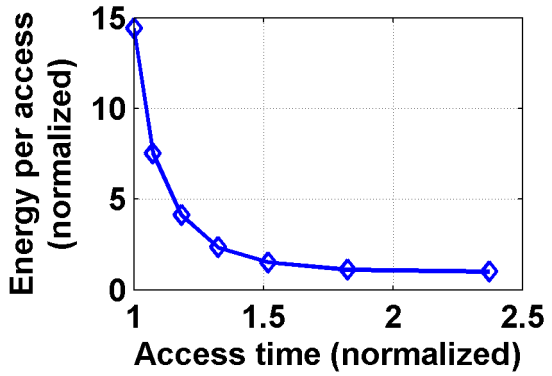
Optimization details

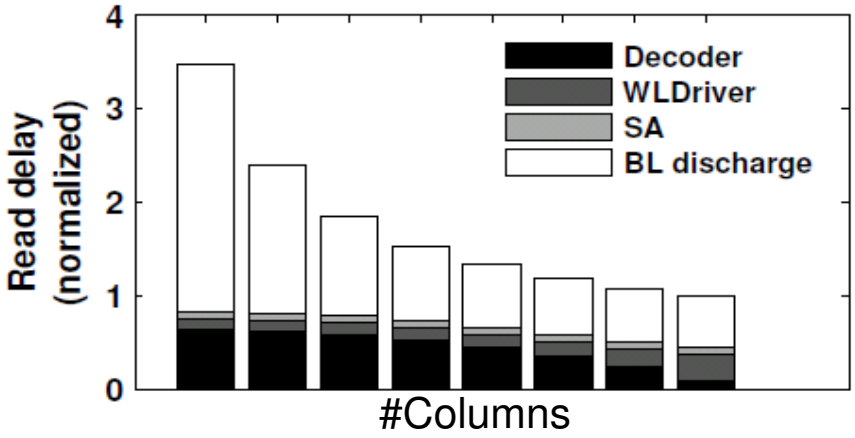
- $E_{SRAM} = \Sigma E_{components}$
- $D_{SRAM} = \Sigma D_{CPath}$
- Yield model (J. Wang, ESSCIRC '07) and bitcell area model incorporated
- Optimization objectives and constraints – SRAM Energy per access and access time (E, D)
- Optimization knobs - #rows, #columns of bitcell array
- Brute force search for optimal solution
- Framework extensible:
  - Introduce more knobs – local and global
  - Introduce more objectives – area and yield
  - Use more powerful optimization tools

Usage Example

Steps 1,2:

1. Process and Sub-component characterization (TASE)
2. Generate Base-case prototype





Outputs:

1. Trade-off (e.g E-D) curves
2. Cross-component metric breakdown

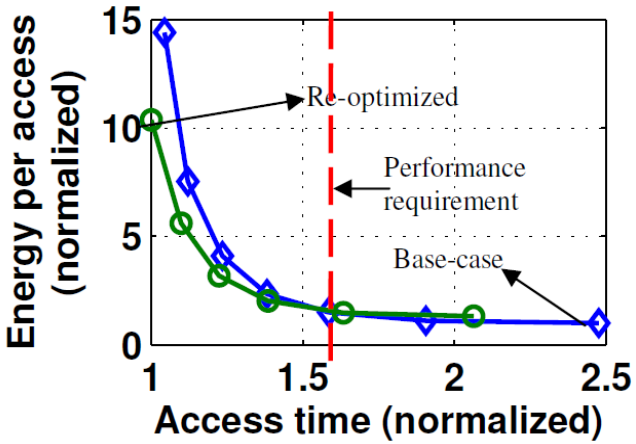
Usage Example

Step 3: Design space Exploration

- Change bitcell
  - Larger PD for more I_{READ}
- Use faster SA – still in early stages of design
  - 24% lower delay
  - 15% higher offset
- Inputs provided with variable levels of detail:
  - Modified bitcell template added to TASE library
  - Guesstimates of SA delay and offset input to SRAM model

Usage Example

Step 4: Re-run tool for re-optimization



Access time < 1.6
⇒ New design optimal

Step 5: Continue exploration and re-optimization or stop

Conclusions

- Presented an early SRAM design space exploration tool
  - Technology-agnostic
  - Base-case generation
  - Variable levels of detail
  - Re-optimization
- Virtual prototyping methodology extensible to any design space exploration