Introduction to RTL – Test & Debugging

Debugging

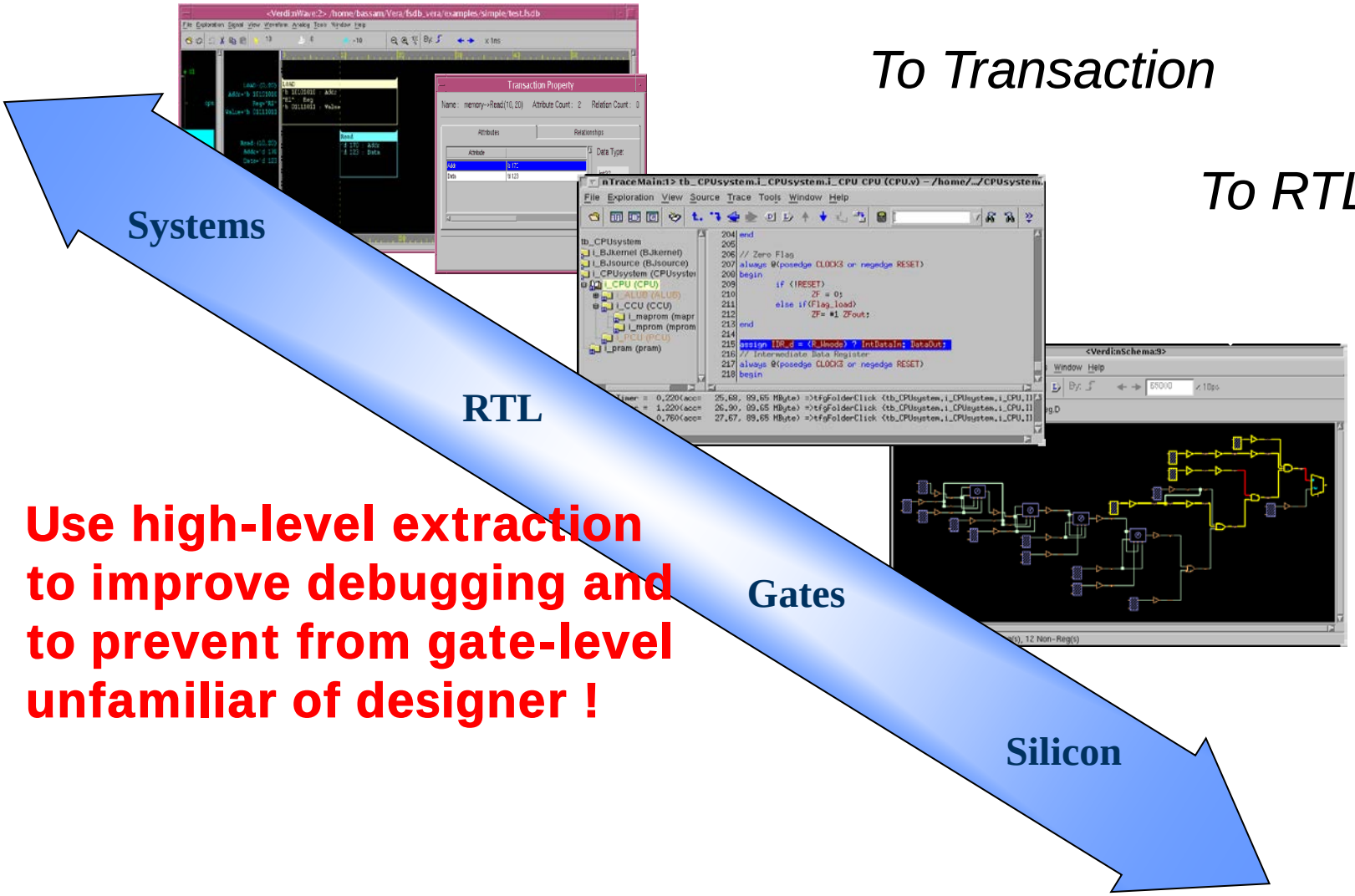
- Why do we need RTL Debugging ?
  - Designers have to build a mental image of how data is propagated and used over the simulation run.
  - As designs get more and more complex, there is a need to facilitate this reasoning process, and automate the debugging.

- Rely entirely on the engineer's ability to deduce the design's behavior from its structure.
  - Fight to hold how the design is supposed to work or why it doesn't.
  - Leads to long debug cycles.
- Main goal is to improve debug productivity by automating the process.

Bugs' Types

- Five kinds of bugs in RTL coding and Synthesis
  - Unknown
  - Timing Error
  - Function Error
  - Transition Error
  - Value Error

Debug at Higher Level of Abstraction



- **Use high-level extraction to improve debugging and to prevent from gate-level unfamiliar of designer !**

Check Rules

- The primary rules are as follows :
  - Latch Inference
  - Register Inference
  - Incomplete Sensitivity List
  - 2-D memory array
  - MUX/priority-encoder Inference
  - The non-inferable RTL

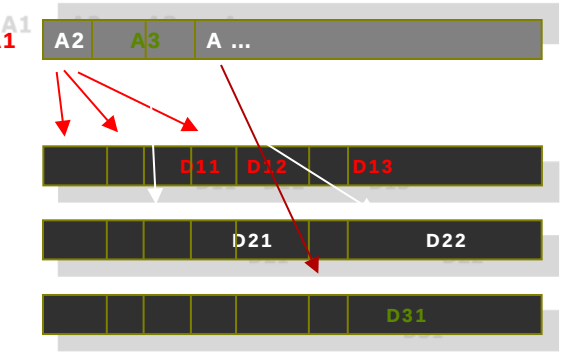
Logic Model

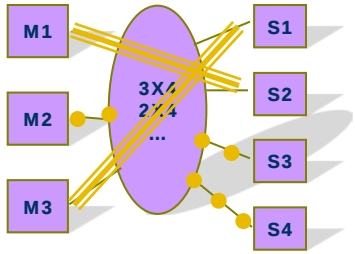
- An inference step converts an HDL description into a logic behavioral model using a “rules-based” approach.
  - Each statement is represented as a component block.
  - No optimization is performed on the logic model.
  - The input are classified into data-path and control inputs using a pre-defined set of rules.

Timing Model and Activity Analysis

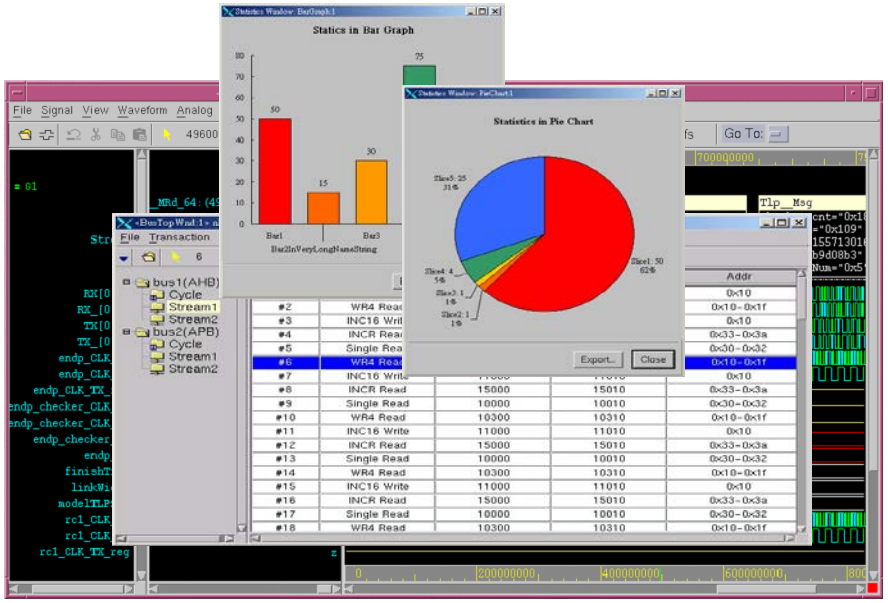
- Simulation result is used to extract the temporal behavior for an identified (problem) signal, starting with the problem signal.
  - Fan-in logic is traversed until FLIP-FLOPs or inputs are hit.
  - The active clock transition time is determined.
  - Fan-in cone logic is evaluated to determine which signal are (in) active.
- Activity analysis serves as the basis for enabling automation of debug in the time domain.

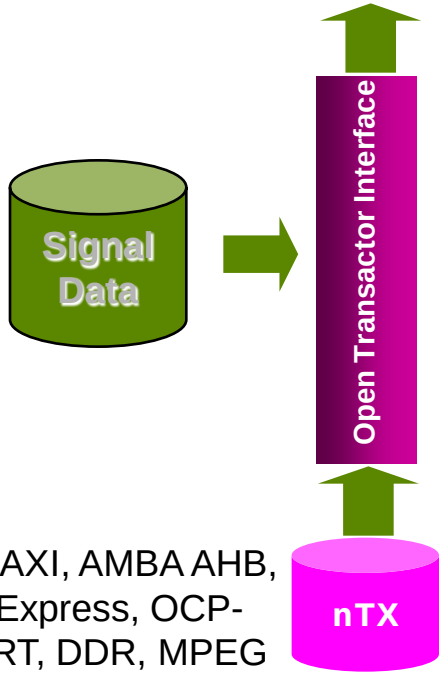
Abstracting Signal Data to Transactions





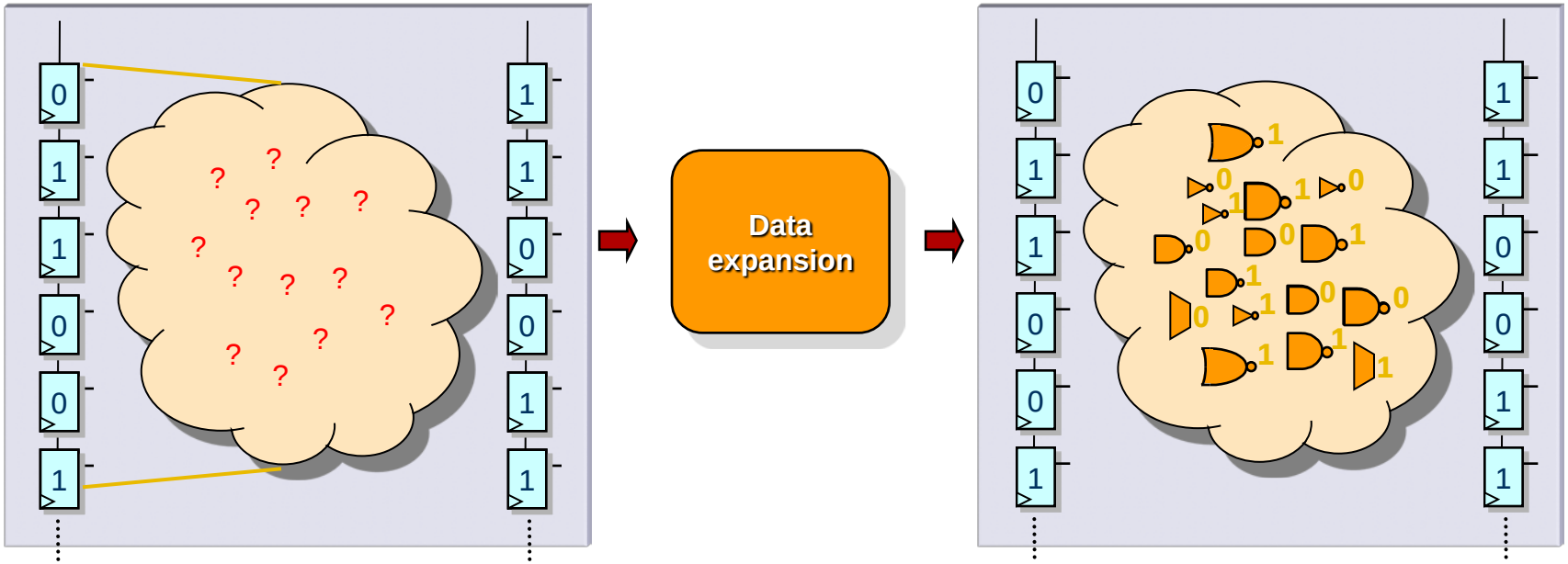
Transaction visualization eases understanding of communication protocols





AMBA AXI, AMBA AHB, PCI Express, OCP-IP,UART, DDR, MPEG

Data Expansion



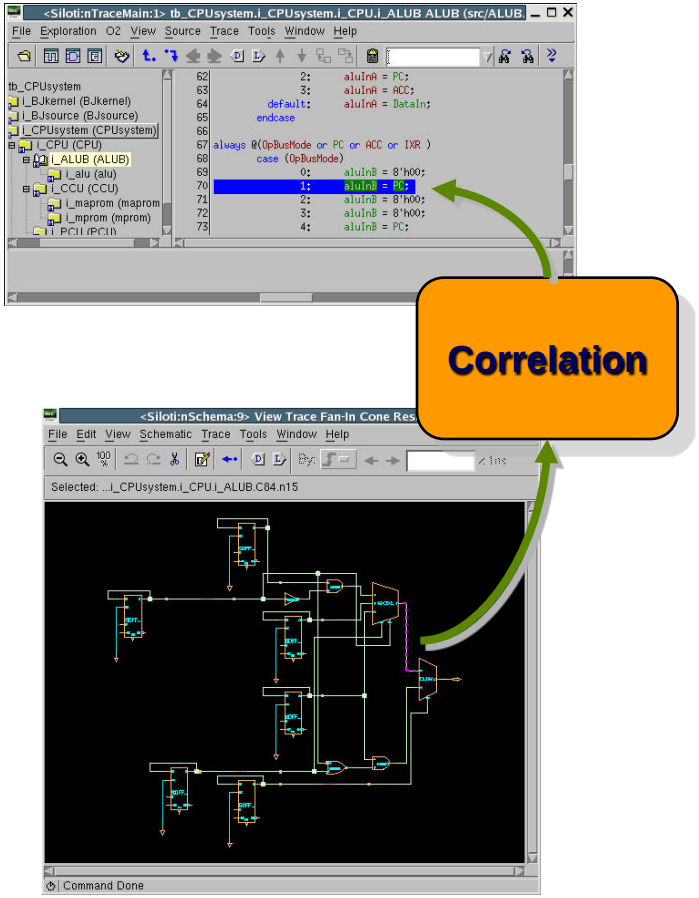
- Data expansion (DE) makes limited data useful – more visibility
- Boolean calculation of unobserved combinational network values
- Maximize value computation to use of “Don’t Care” truth table results
- DE metric: signals-with-values/total-number-of-signals
- Performance optimized by expanding only the logic under investigation and no simulation-like timing wheel

Behavior-Based Debug Infrastructure

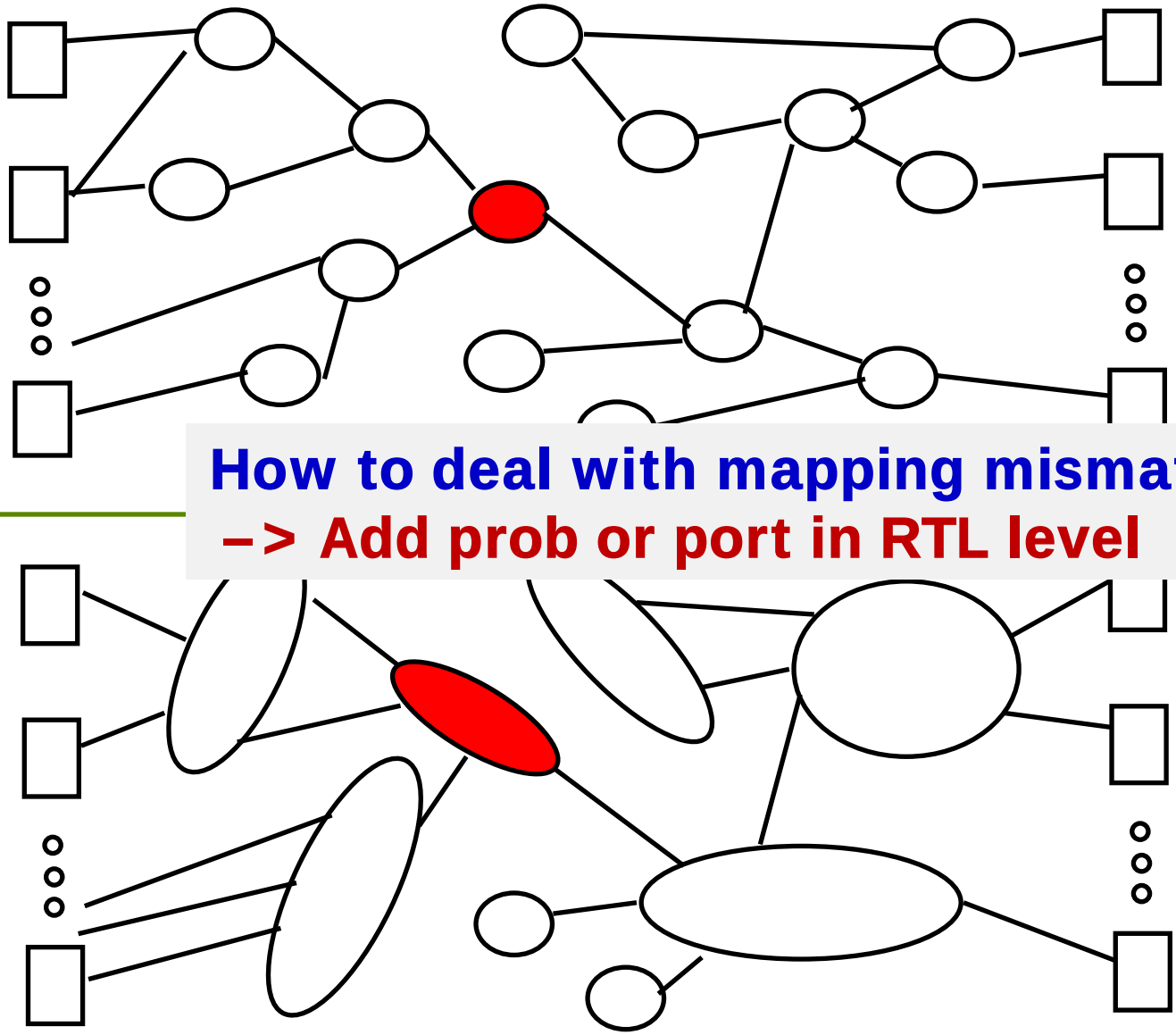
- Starting from an error source, debug traces back towards the cause of this error.
  - It does so by marrying the logical model and the timing activity models.
  - Builds a behavior trace of the signal in question.
  - This expansion can be performed interactively again and again moving backwards in time.

Abstraction Correlation

- Silicon signal data is usually easy to assign to gate-level signals
- Gate-level netlist and values are difficult for RTL designers to understand and debug
- Data must be mapped to the “designers world” for efficient debug and collaboration
- One-to-one correspondence is not always the case after synthesis transformations



Abstraction Correlation - Mapping



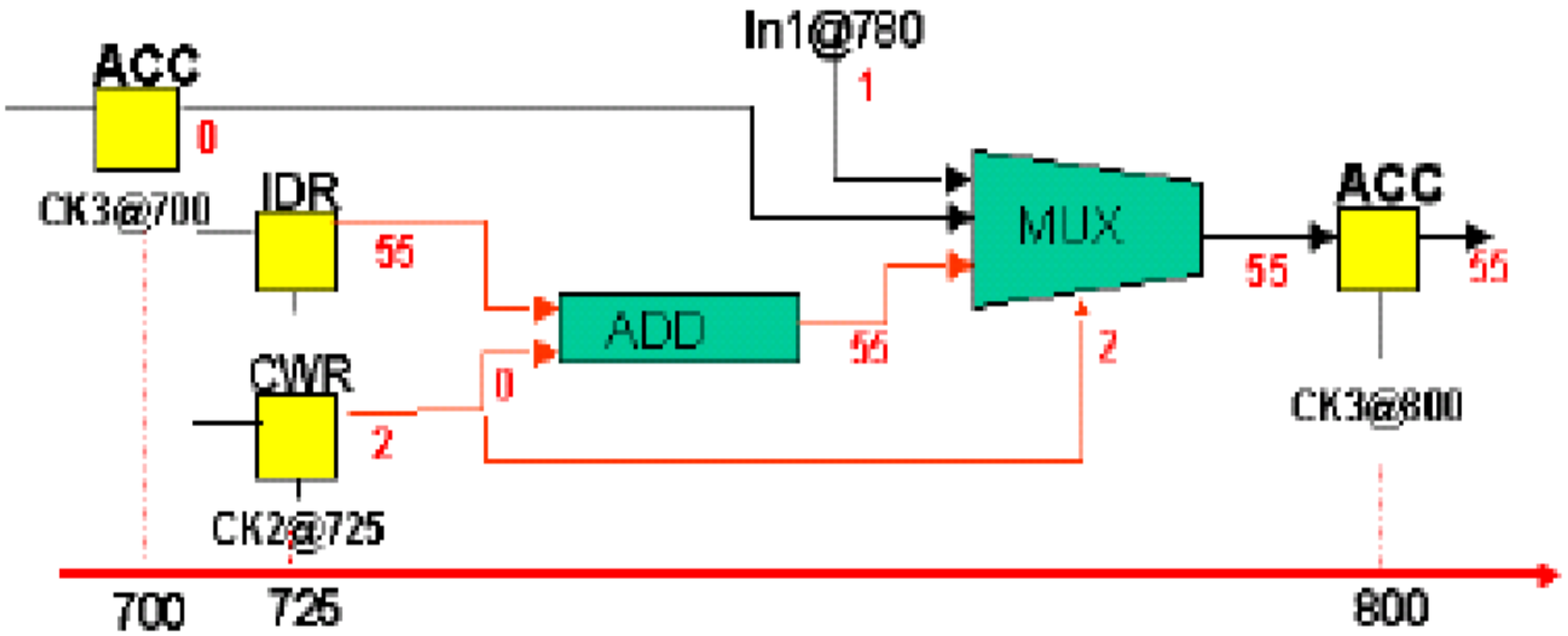
How to deal with mapping mismatch ?

–> Add prob or port in RTL level

Unknown Tracing

- Get the first unknown after reset
- Trace the distributed tree of unknown
- Clear initial unknown
- Remove the first unknown

Tracing and Clocking



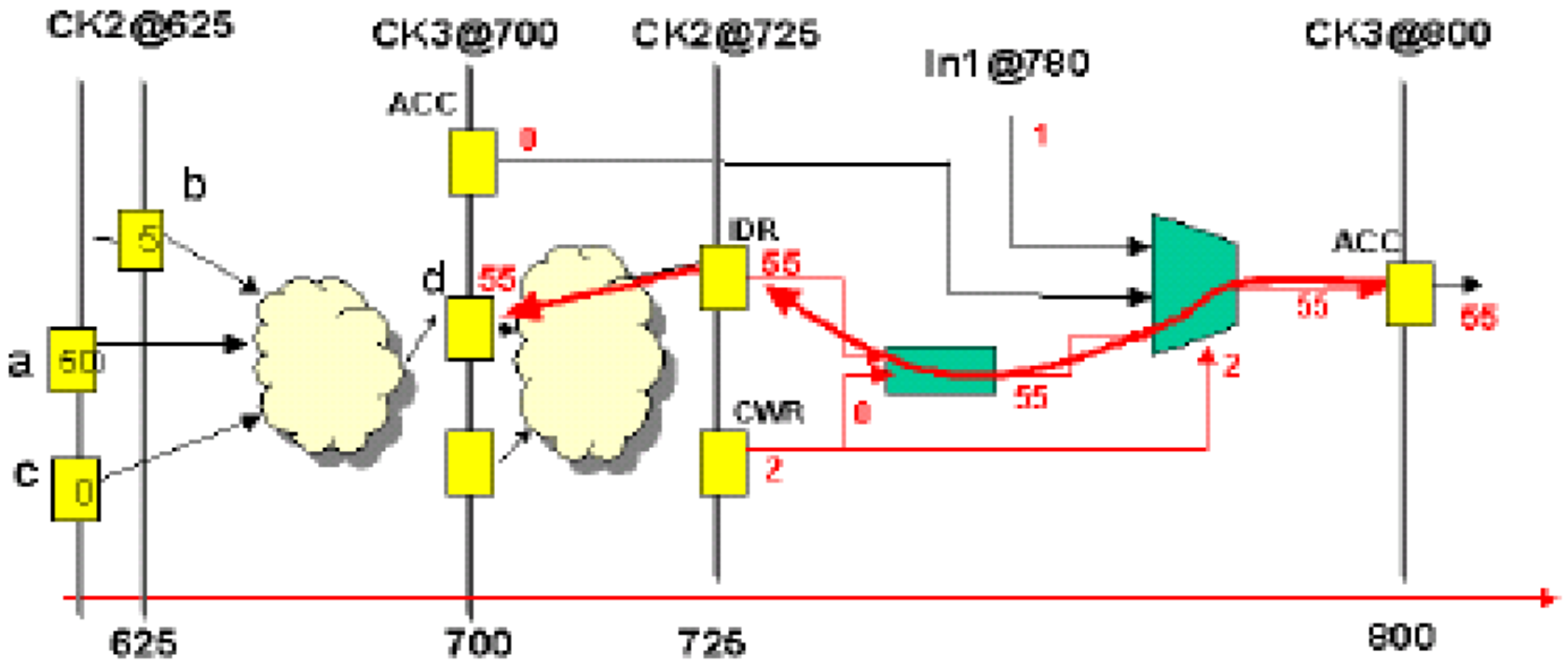
Tracing and Clocking

Value Tracing

- Can be specialized to search for the first unknown (X) that propagates to the output.
- Can determine when the content of a 2-D array element has been written and with what value.
- This debug infrastructure is used to build advanced debug approaches :
  - Behavior exploration.
  - Behavior query.

Trace a Suspicious Value

- Automatically trace value back across multiple cycles

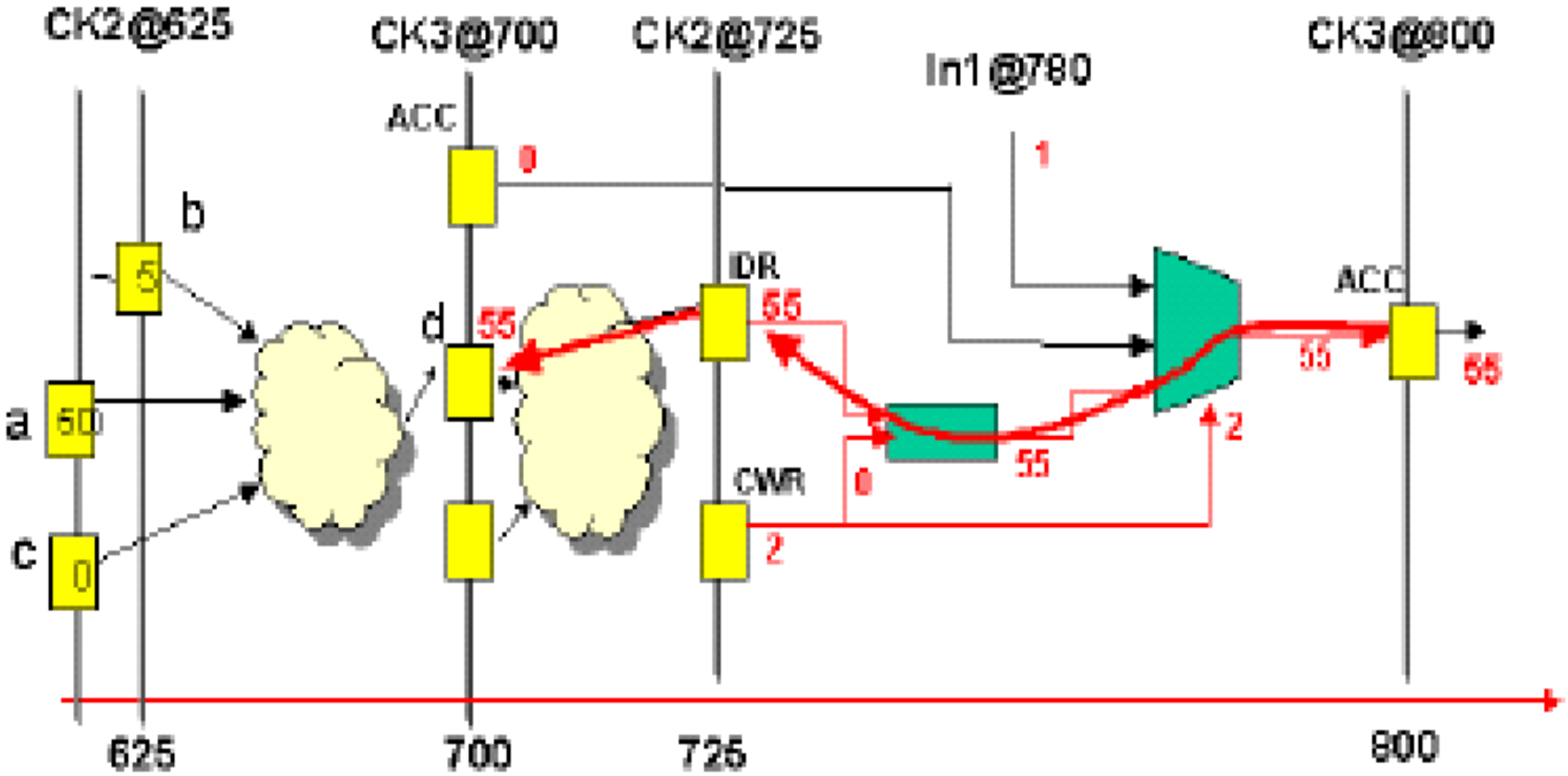


Tracing a Suspicious Value

“What if” analysis

- Provides the ability to quickly evaluate a potential bug fix by changing the current values in the design with new values.
  - First performs fan-out marking.
  - Backward timing expansion start from the target signals or time to the starting signals.
- Performance depends on the size of the expanded model.

“What if” analysis - Example



“How can” analysis

- Provides the inverse capability to “what if”.
  - Give a way to find all possible combinations of a set of signals that achieve a specific value for a target signal at a specific time.
- Performance depends on :
  - Size of the expanded model as in “what if” analysis.
  - Numbers of set symbolic values.

Conclusions

- Behavior analysis and debug technique can be used for :
  - Quickly locate and diagnose errors with behavior query.
  - Evaluate potential corrections with behavior exploration.
  - Quickly trace back to the root causes.
  - **Create probs. in RTL to prevent from naming mismatch.**