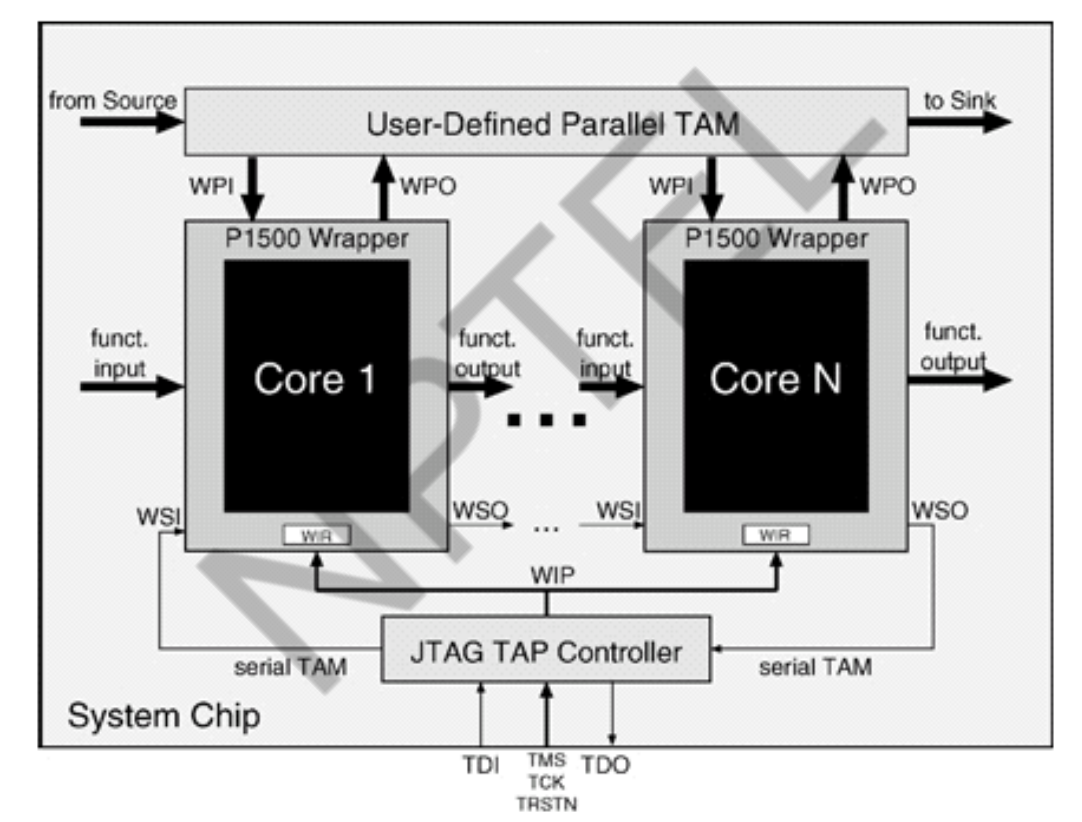
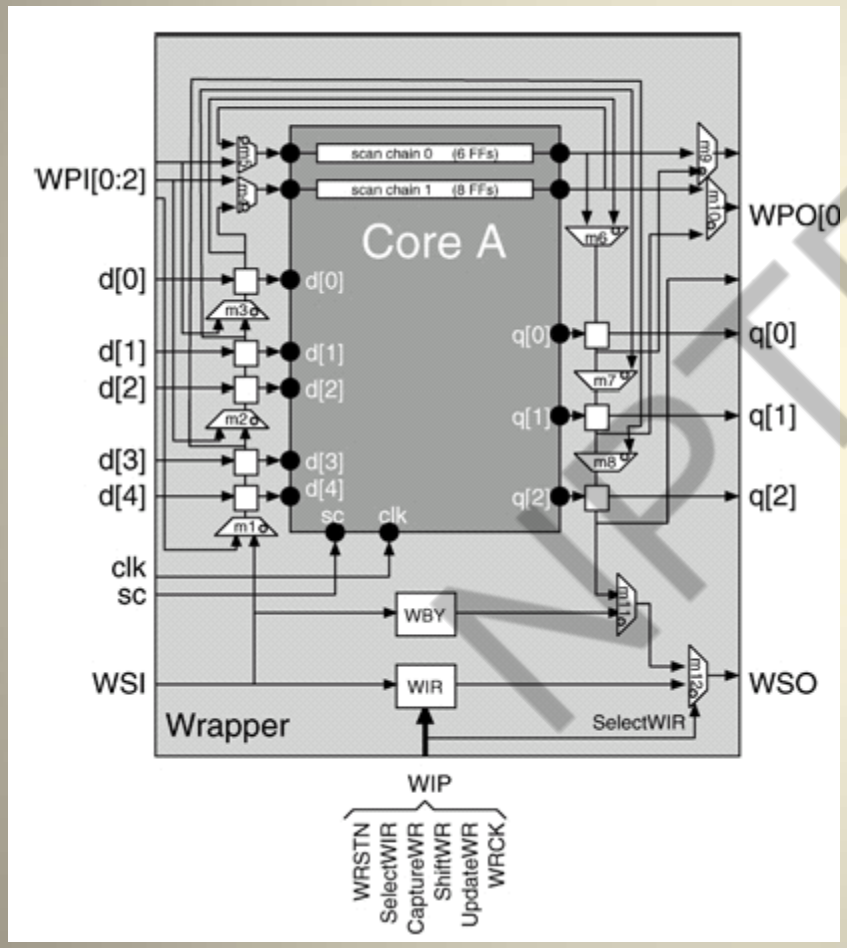
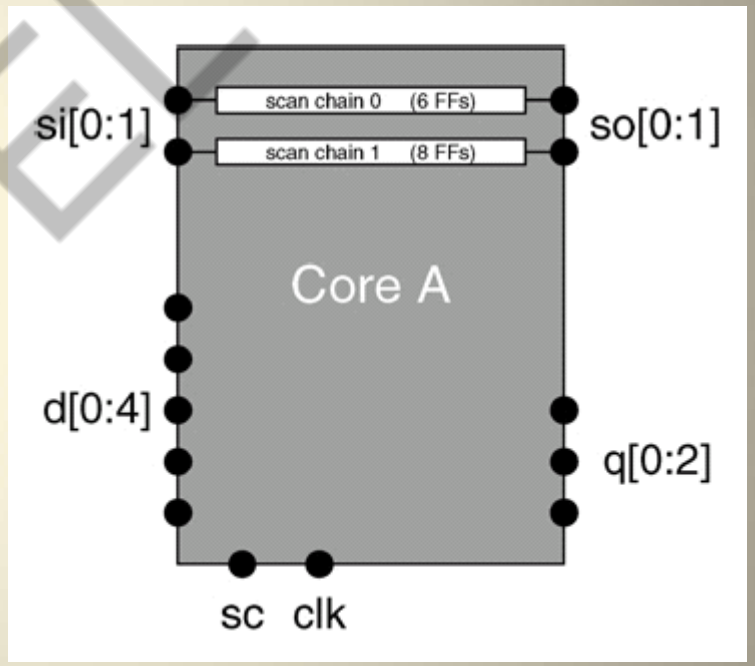
Lecture 47

Wrapper Usage

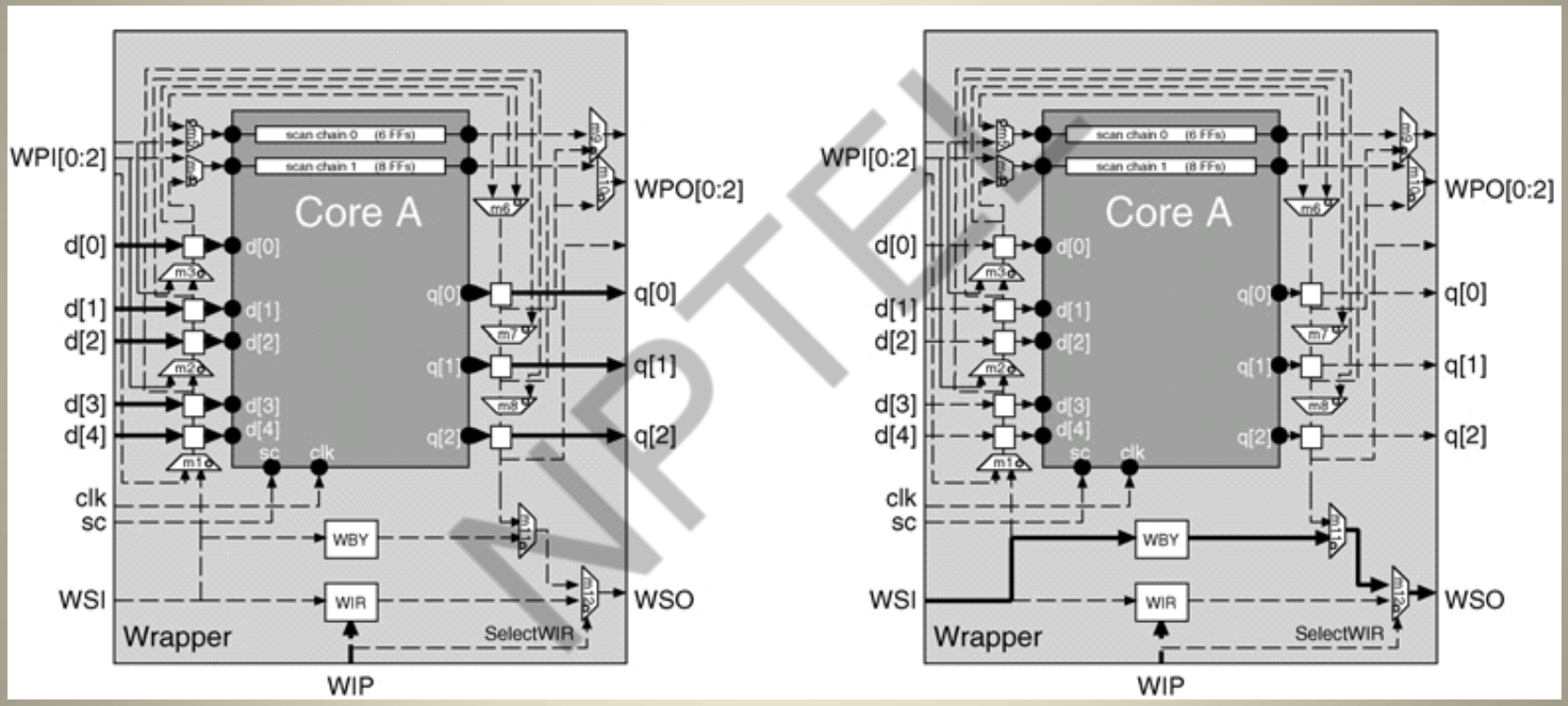


Wrapped Embedded Cores





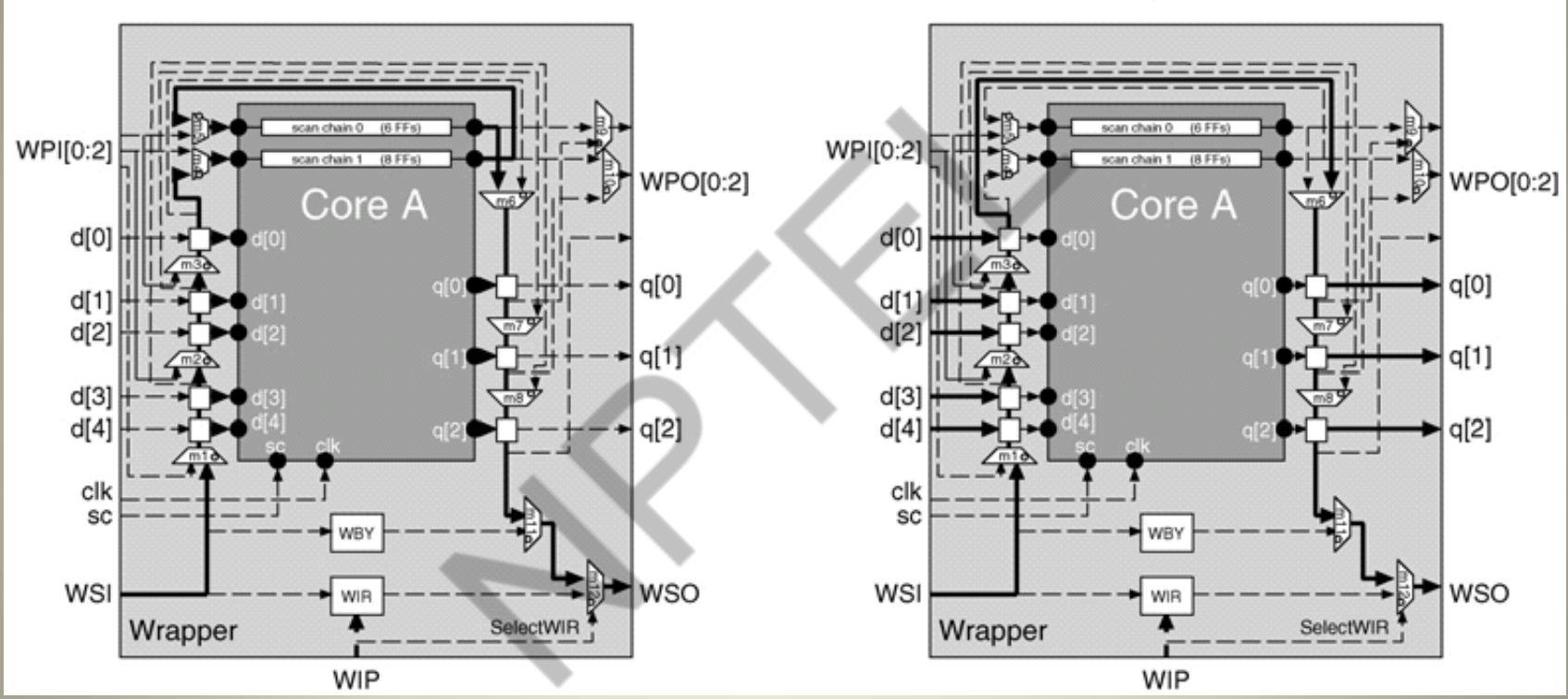
Wrapper Operation Modes (I)



Normal Mode

Serial Bypass Mode

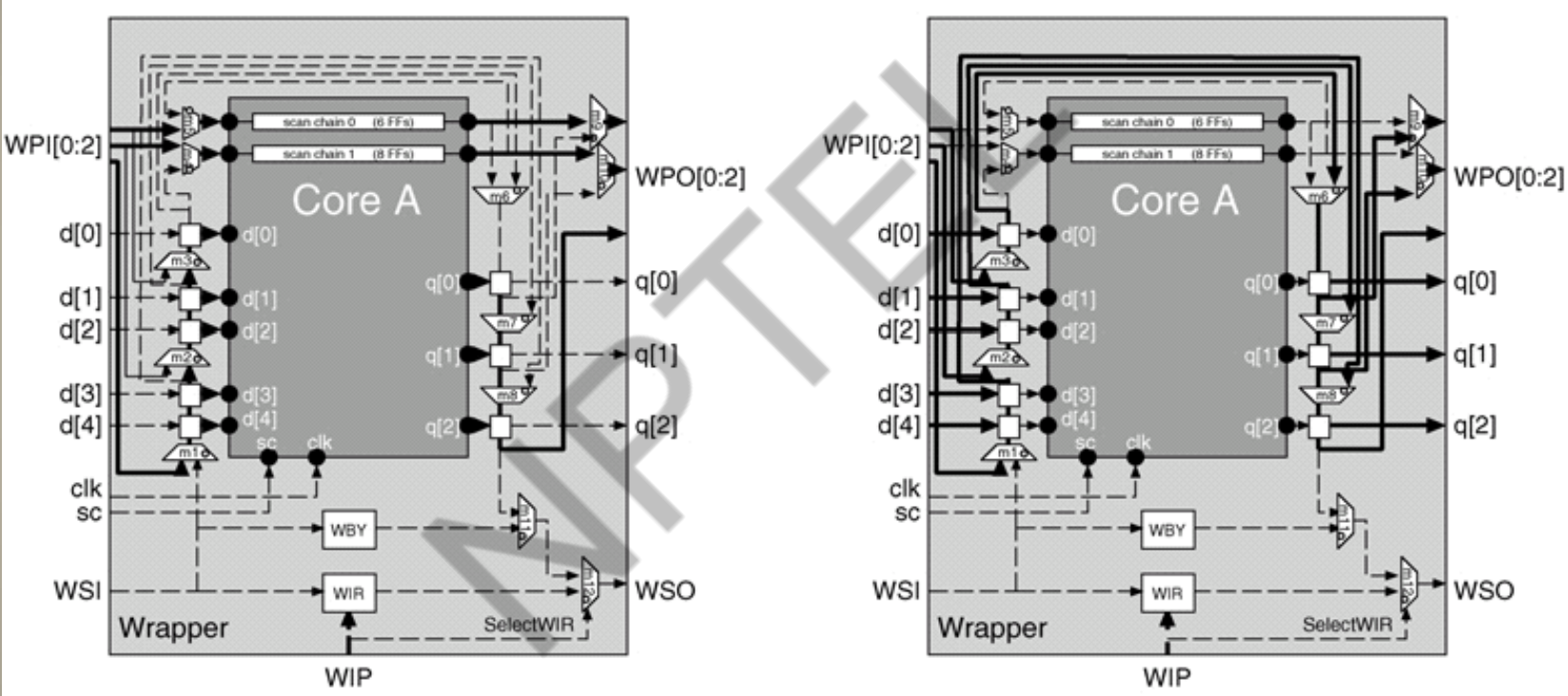
Wrapper Operation Modes (II)



Serial Internal Test Mode

Serial External Test Mode

Wrapper Operation Modes (III)

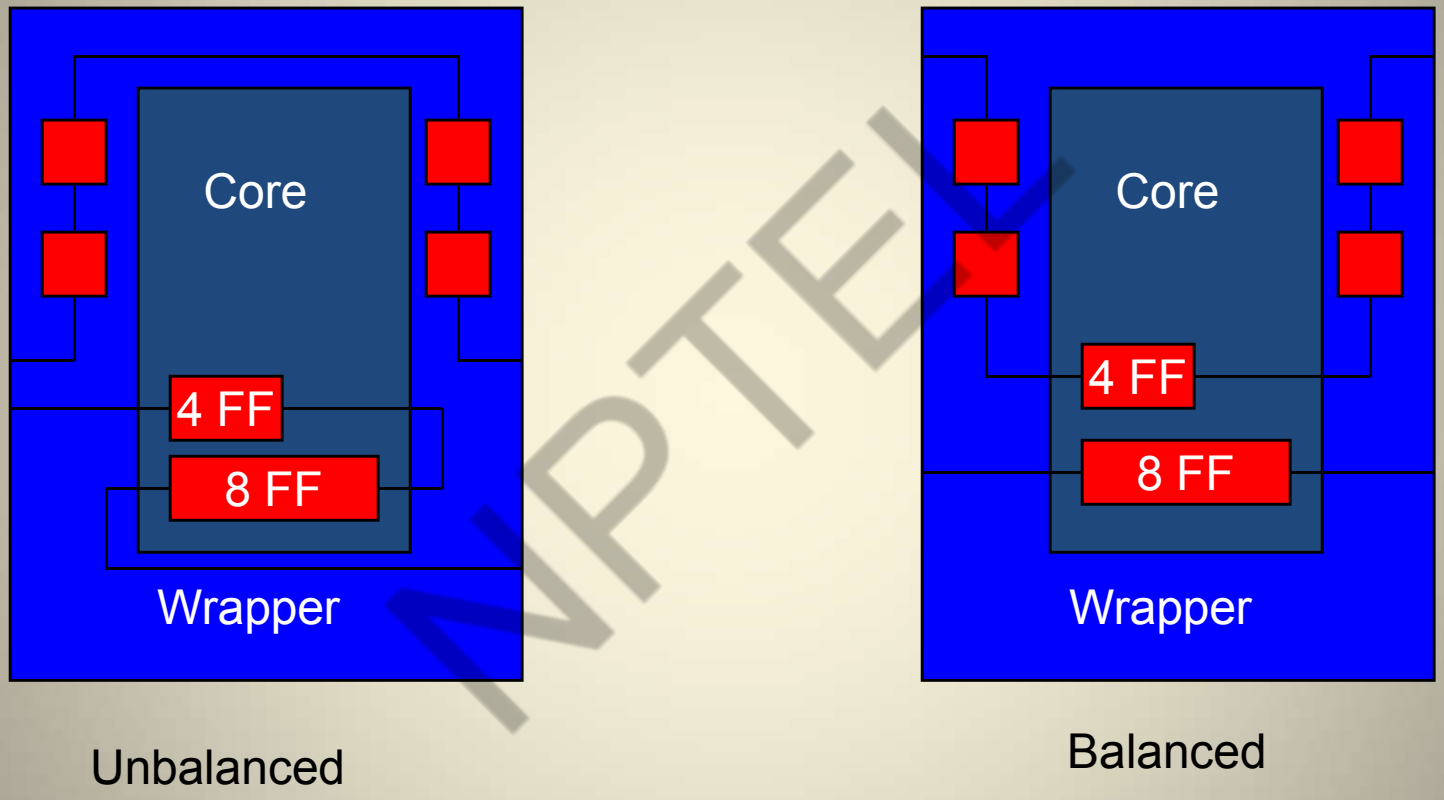


Parallel Internal Test Mode

Parallel External

Test Wrapper Optimization

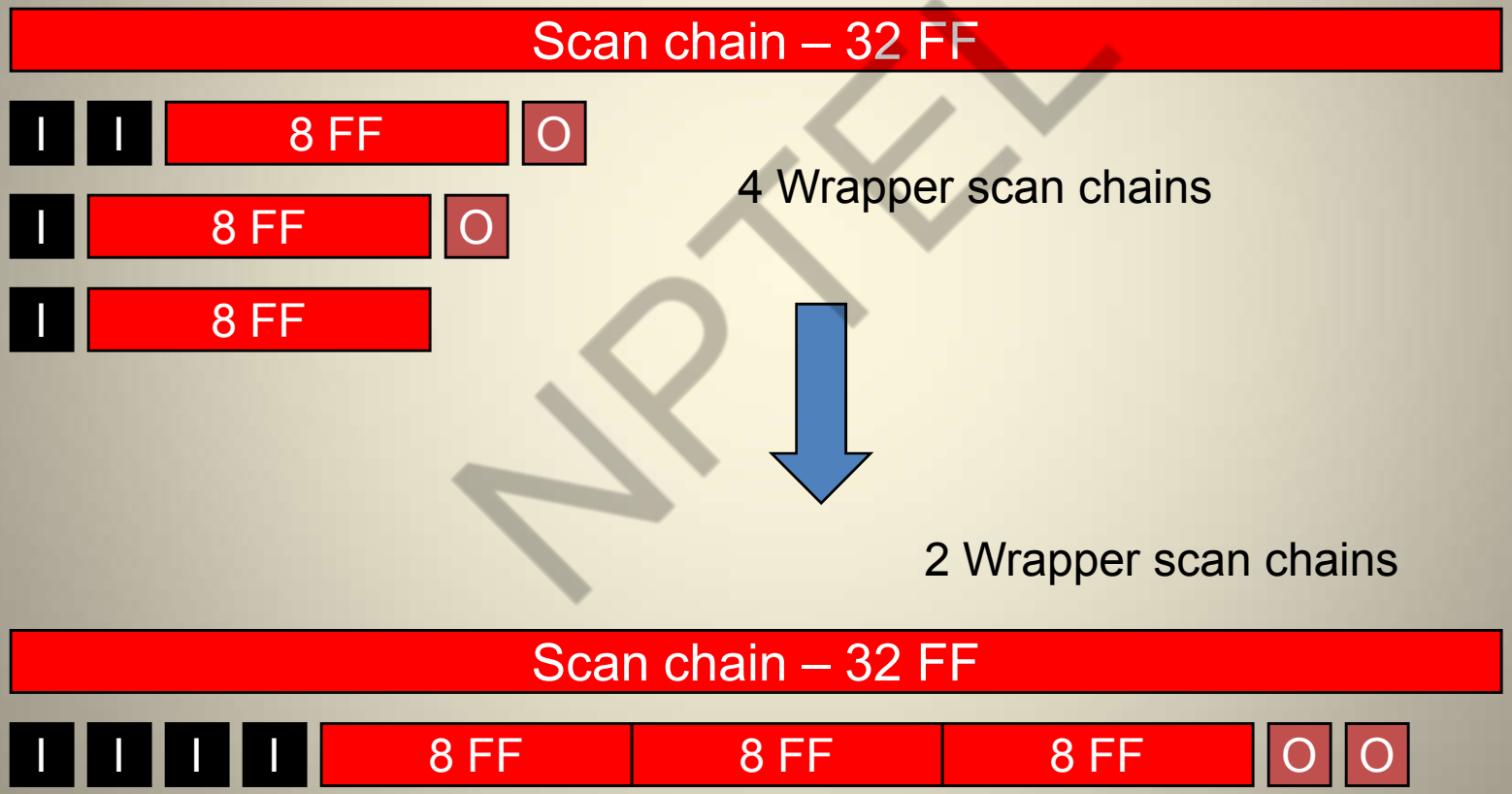
Priority 1: Balanced Wrapper Scan Chains



Minimize length of longest wrapper scan in/out chain

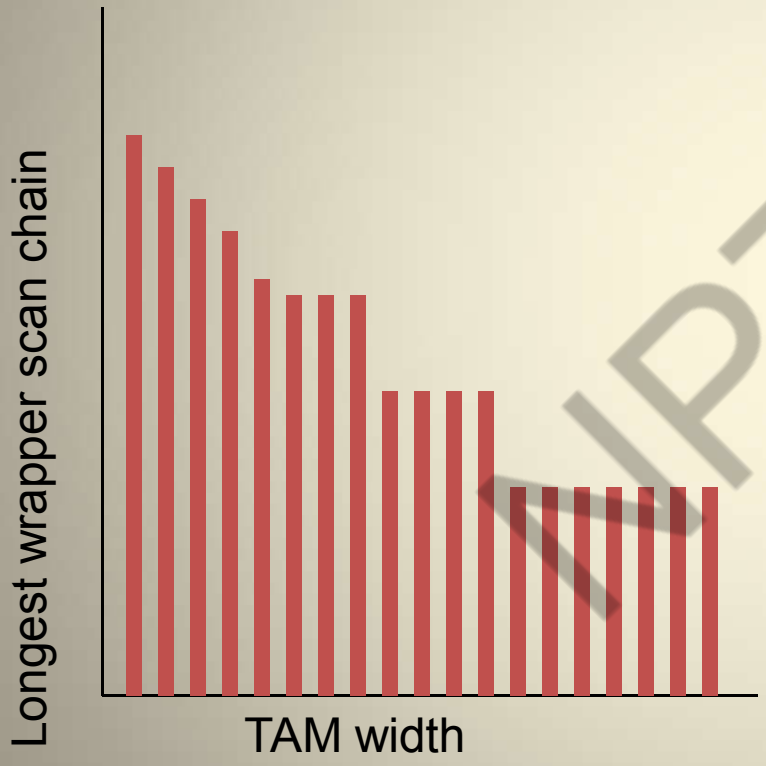
Reducing TAM Width

Priority 2: Minimize wrapper scan chains created



Two-Priority Wrapper Design Algorithm

1. **Minimize length of longest wrapper scan in/out chain**
2. **Minimize number of wrapper scan chains**



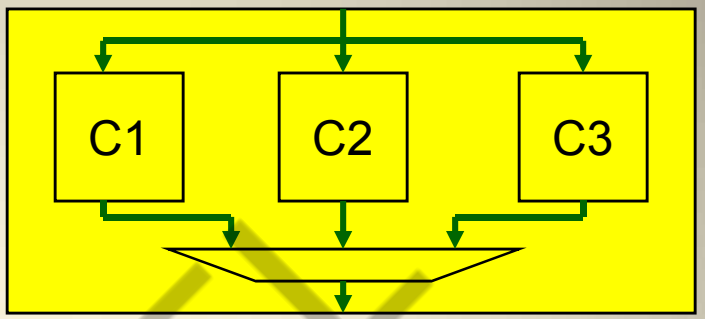
Design_wrapper algorithm uses the BFD heuristic for Bin Design

Lecture 48

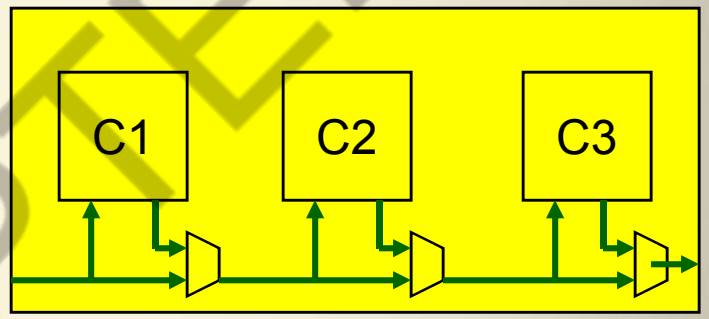
Test Access Mechanisms

Types of TAMs

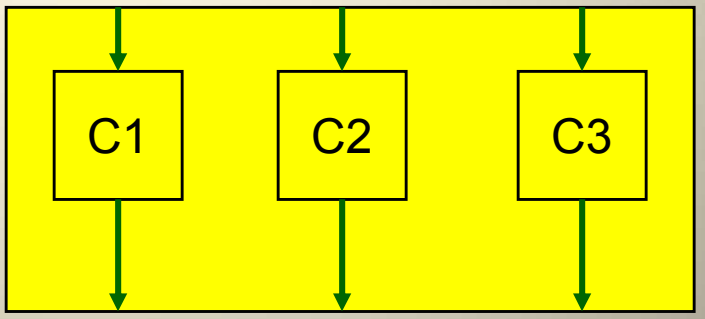
- Multiplexed access [Immaneni, ITC'90]
- Reuse system bus [Harrod, ITC'99]
- Transparent paths [Ghosh, DAC'98]
- Isolation rings [Whetsel, ITC'97]
- Test Bus [Varma, ITC'98]
- Test Rail [Marinissen, ITC'98]



Multi-plexed



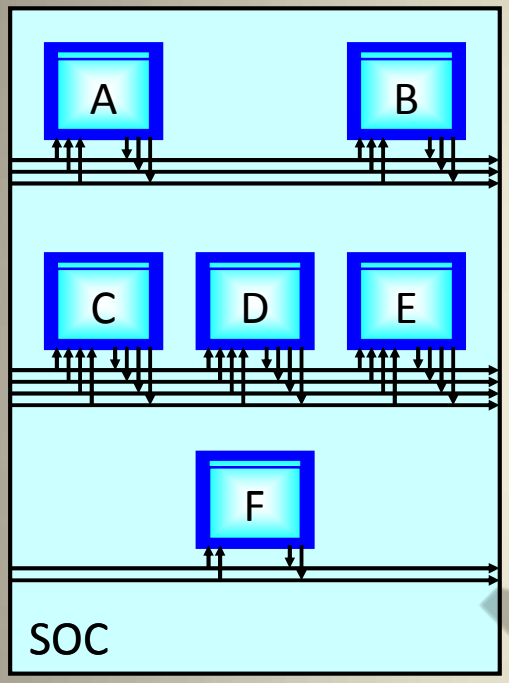
Daisy-chain



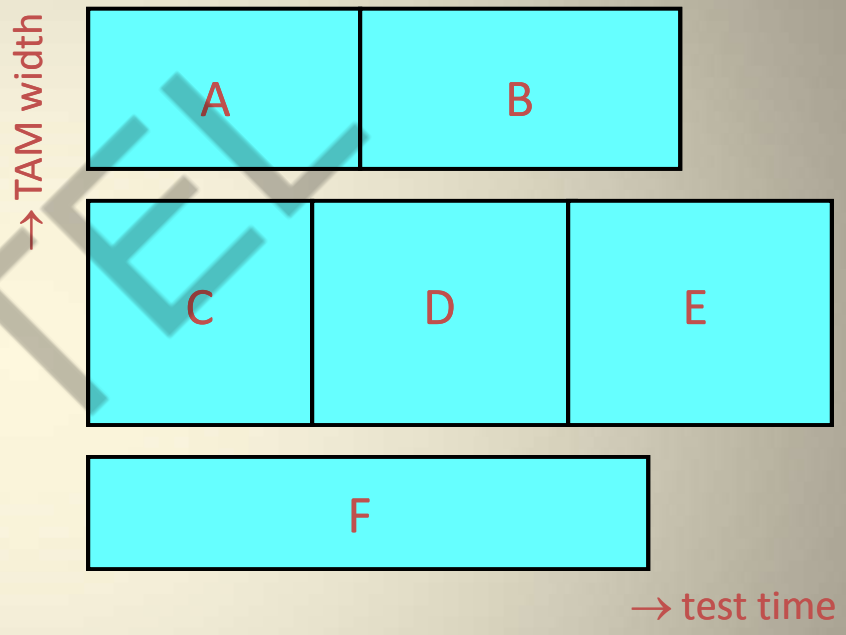
Distri-bution

Test Bus Architecture

Architecture



Schedule: Serial

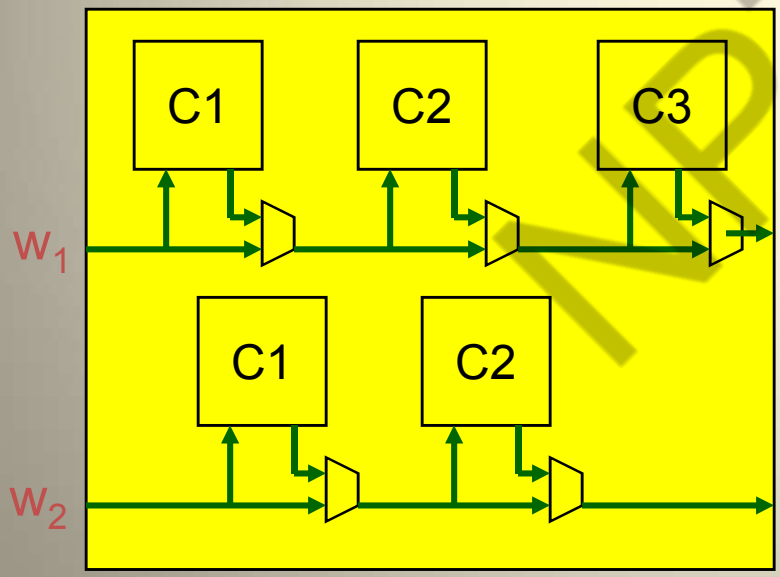


- Combination of multiplexing and distribution
- Supports only serial schedule
- Core-external testing is cumbersome or impossible

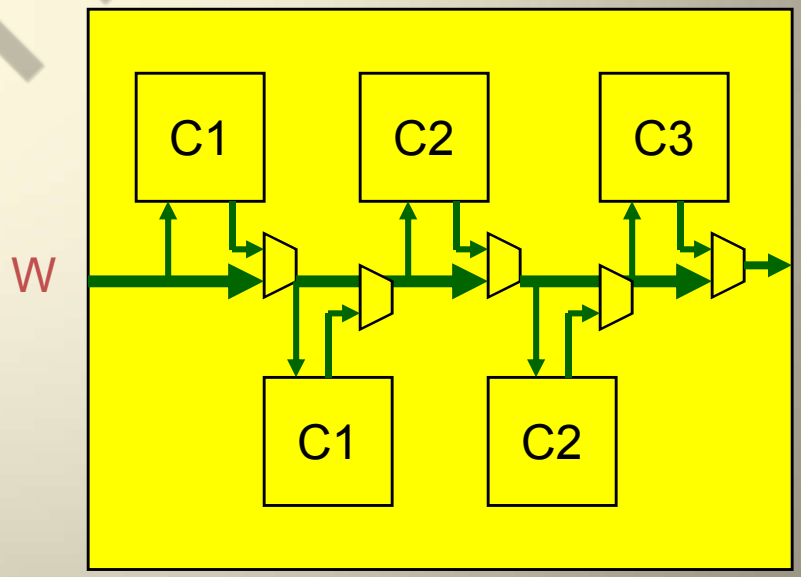
TestRail Architecture [Goel ITC'02]

- Combination of Daisy chain and Distribution architectures
- Cores connected to a TestRail can be tested simultaneously as well as sequentially
- Multiple wrappers can be activated simultaneously for Extest
- TestRails can be either fixed-width or flexible-width

Fixed-width TestRails

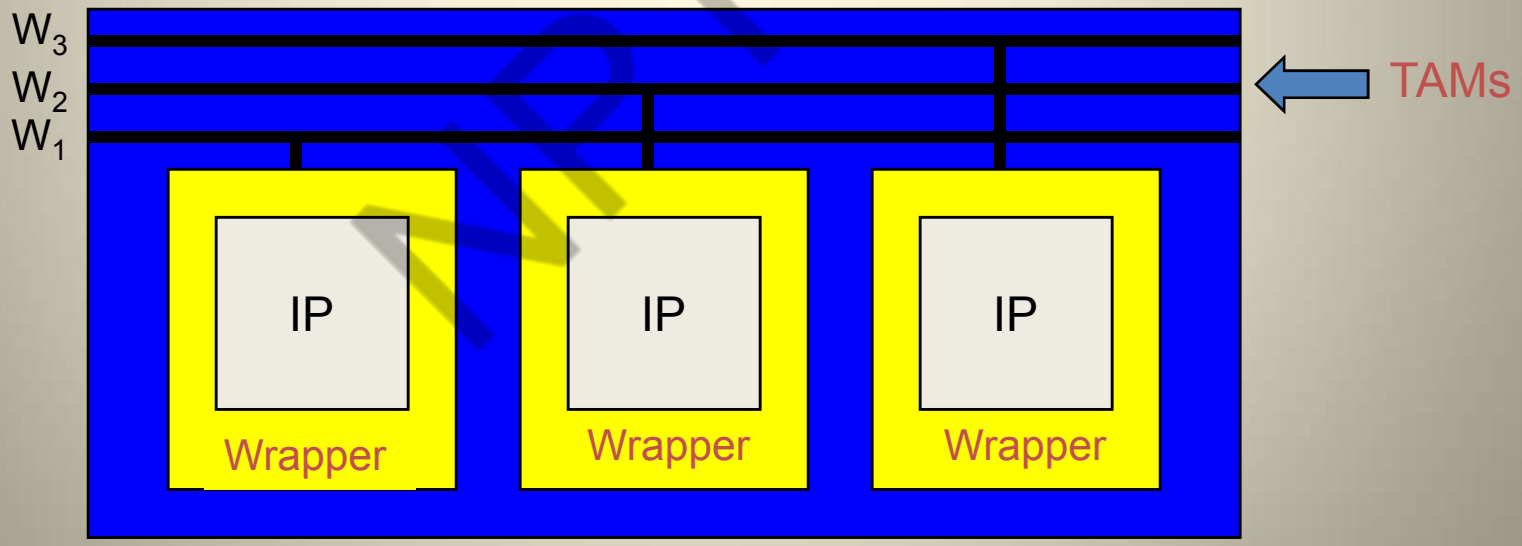


Flexible-width TestRails



Step-by-Step Approach to Wrapper/TAM Co-optimization

1. P_W: Wrapper design
2. P_{AW}: Core assignment + P_W
3. P_{PAW}: TAM width partitioning + P_{AW}
4. P_{NPAW}: Number of TAMs + P_{PAW}



Mathematical Programming Model for TAM Partitioning

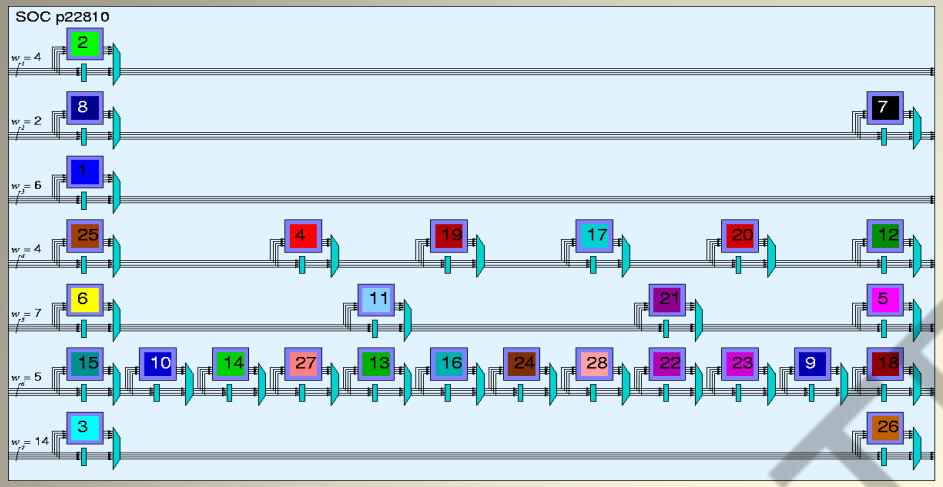
- Variable $x_{ij} = 1$, if core i assigned to TAM j
- Testing time of core i on TAM width $w_j = T_i(w_j)$
- Testing time on TAM $j = \Sigma_i T_i(w_j)\, x_{ij}$
- **Objective:** Minimize $T = max_j \Sigma_i T_i(w_j)\, x_{ij}$
- Constraints
  1. $\Sigma_i x_{ij} = 1$, every core connected to exactly one TAM
  2. $\Sigma_i w_j = W$, total TAM width is W
  3. $w_j \leq w_{max}$, maximum width of any TAM is w_{max}

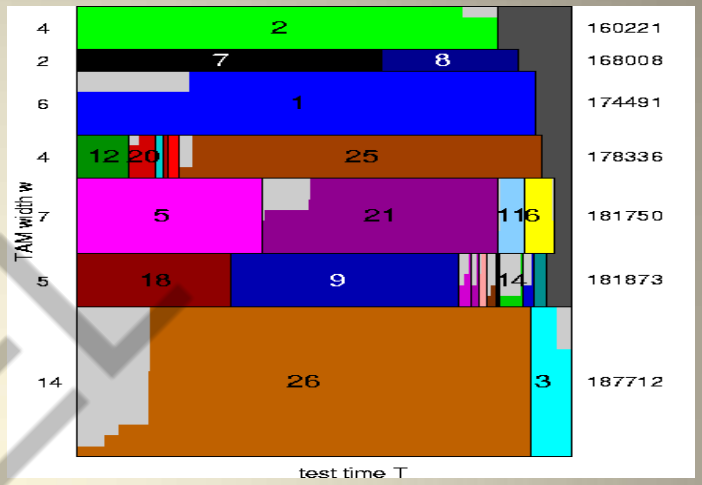
TAM Design and Test Scheduling

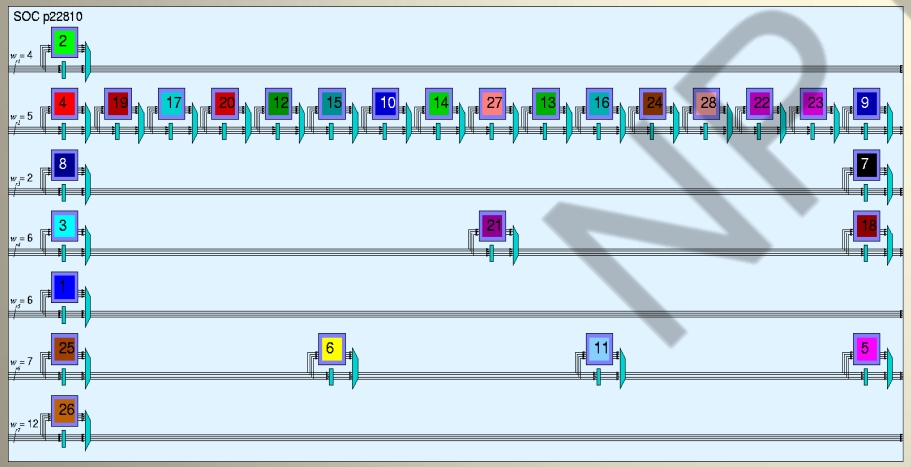
- Given the test set parameters for the cores and the total TAM width *W*
- Assign a part of *W* to each core, design a wrapper for each core, and determine the test schedule,
- Such that
  - *W* is not exceeded at any time and
  - *Testing time is minimized*

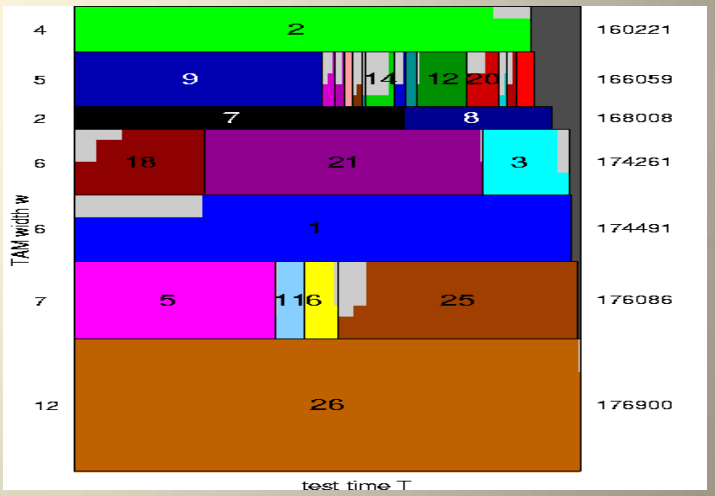
Lecture 49

Architectures Determine Schedules



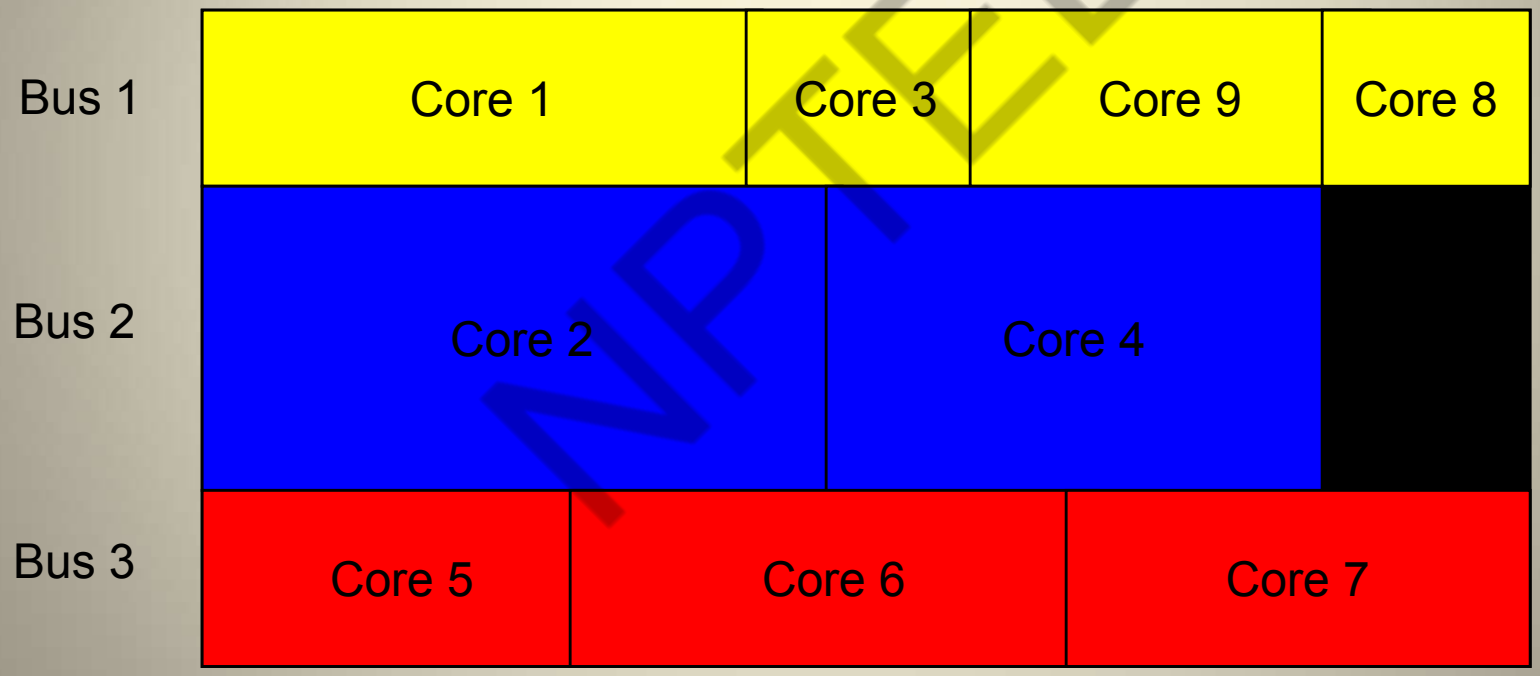






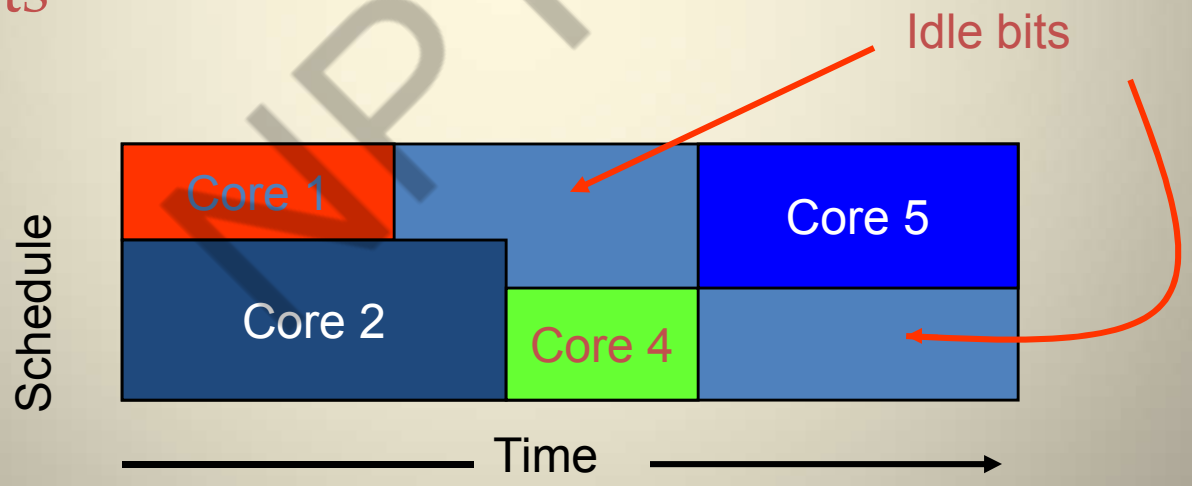
Rectangle Model for Test Buses

Three test buses
Each core on same bus gets equal, fixed TAM width



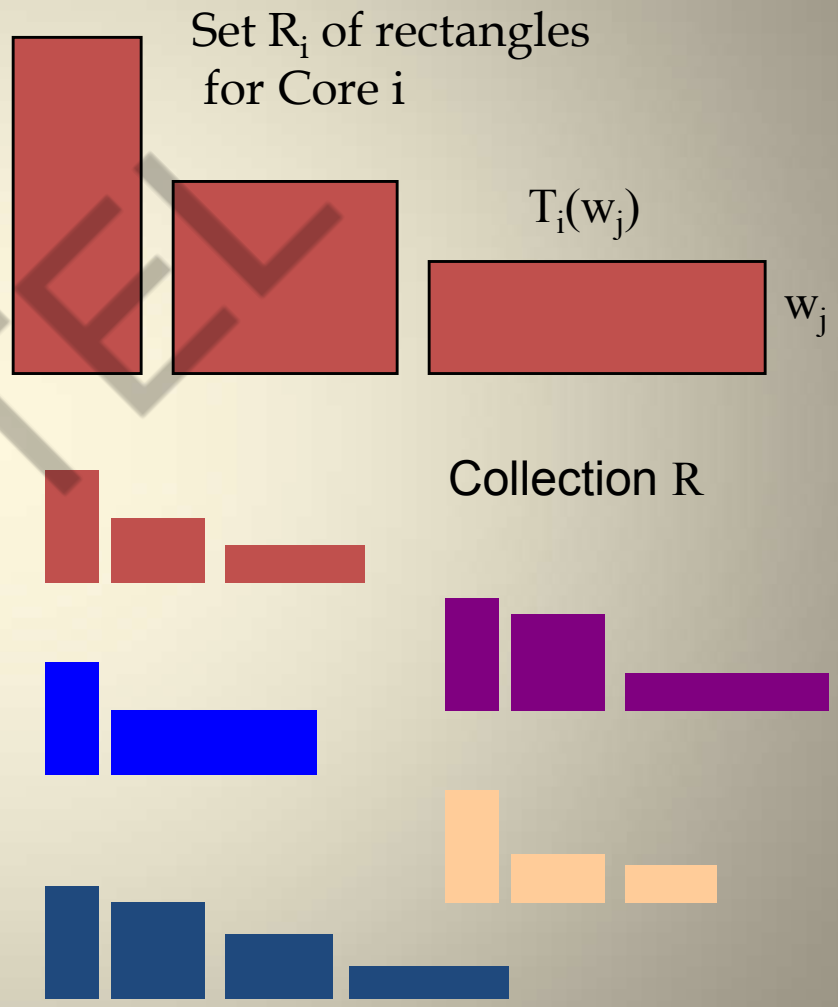
Test Scheduling

- Test scheduling determines sequence of core tests on the TAMs
- Avoid test resource conflicts
- Minimize testing time
- Ineffective scheduling can increase tester data volume: Idle bits



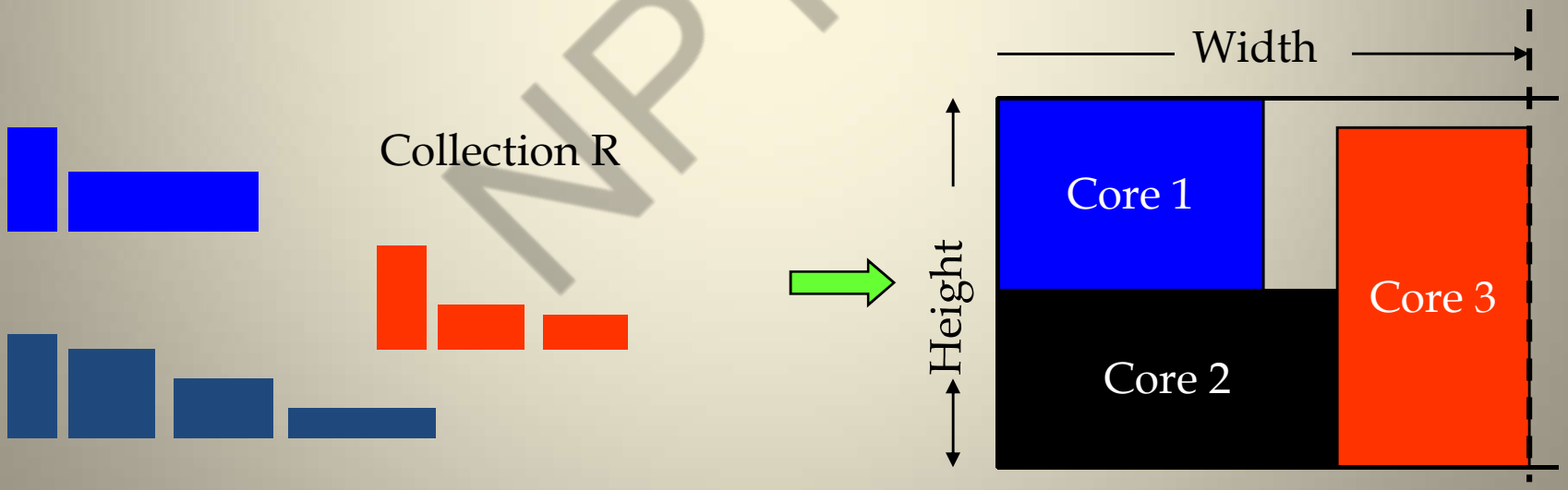
Rectangle Representation

- Testing time $T_i(w_j)$ for Core i and TAM width j
- Rectangle R_{ij}
- Set of rectangles $\mathbf{R_i}$ for each core
- Collection of rectangles **R** for SOC

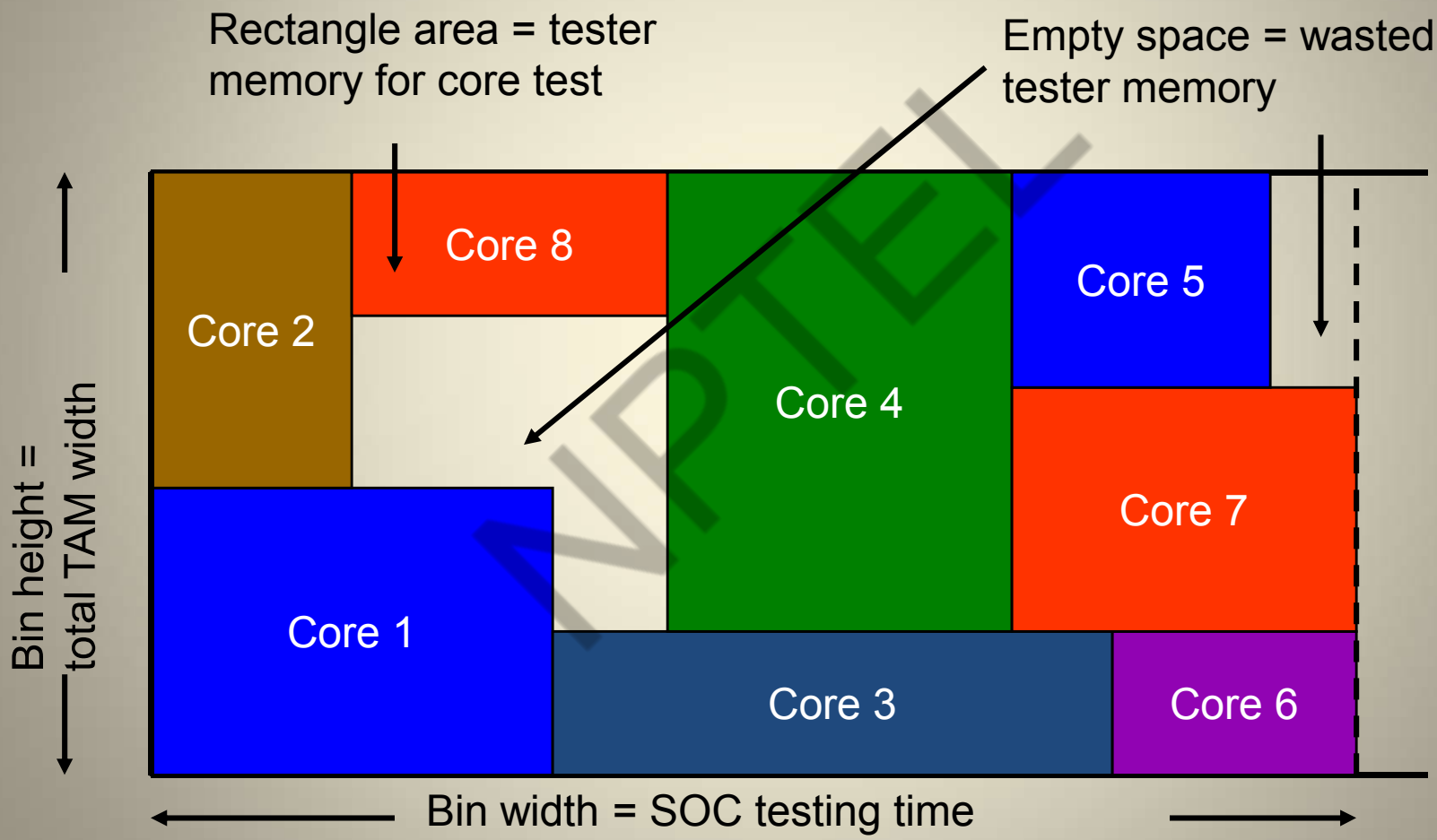


Rectangle Packing Problem

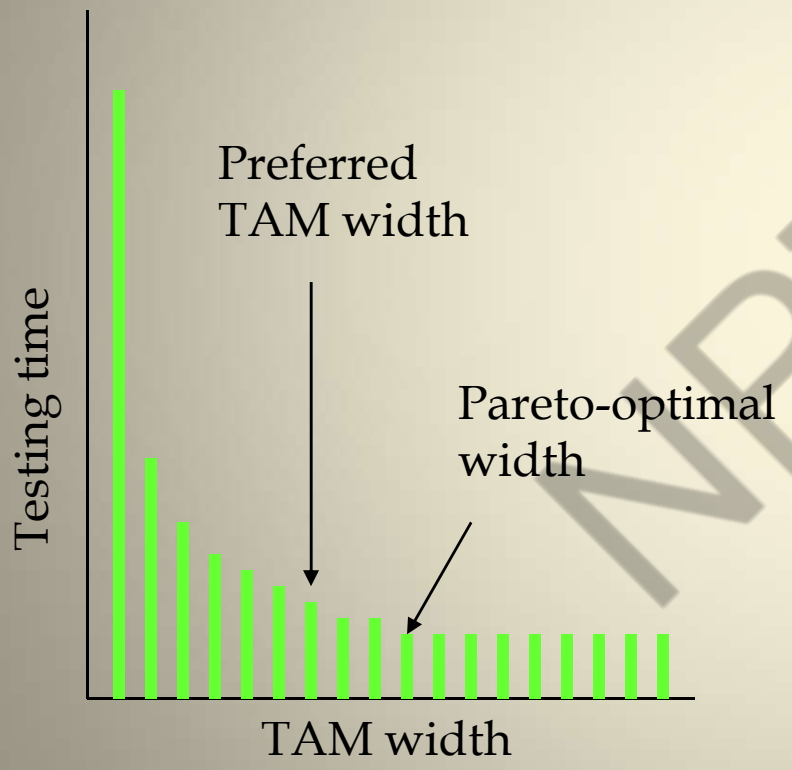
- Given collection **R** of rectangle sets for the SOC cores,
- Select one rectangle R_{ij} for each Core i
- Pack the selected rectangles into a bin of fixed height,
- Such that bin width is minimized



Packed Bin = TAM Design + Test Schedule

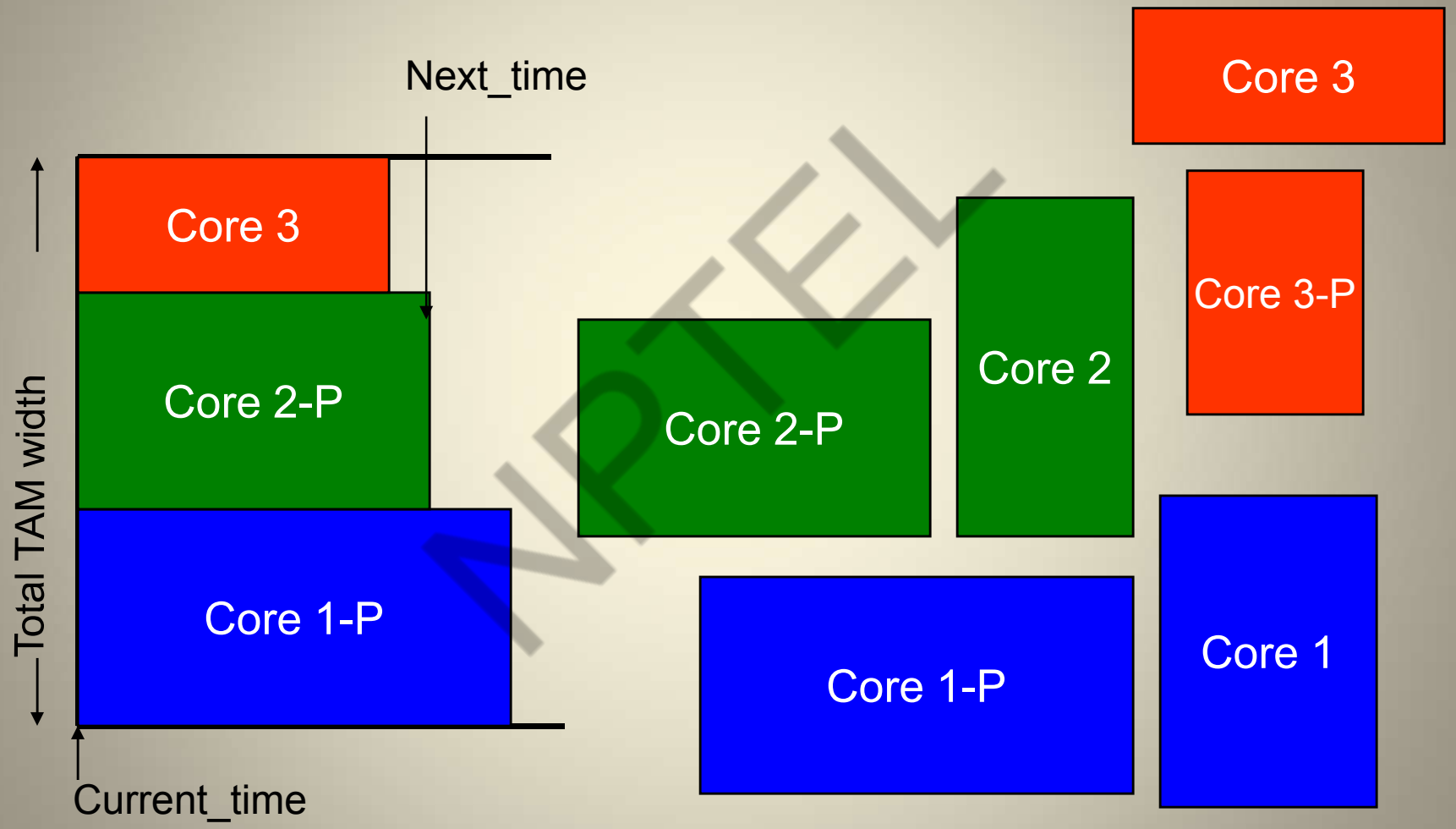


Preferred TAM Widths

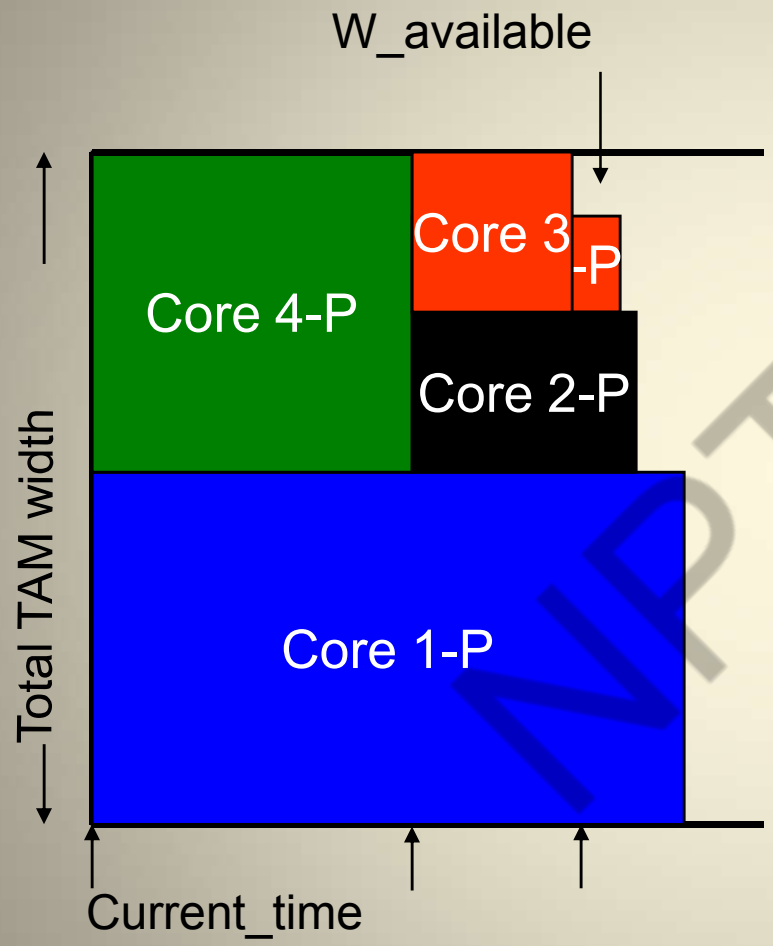


- Only Pareto-optimal TAM widths are considered
- **Procedure:** Tests are scheduled at current time in decreasing order of preferred TAM width until no TAM width remains

Non-Preferred Rectangles: Fill Idle Time



Increasing Current TAM Widths



- Modify current rectangle that will benefit the most from an increase in TAM width

If idle time is inevitable, advance Current_time and repeat procedure from the start

Current-Generation ATEs

- Port scalability features
- Digital speeds of up to 2.5 Gbps
- Application flexibility

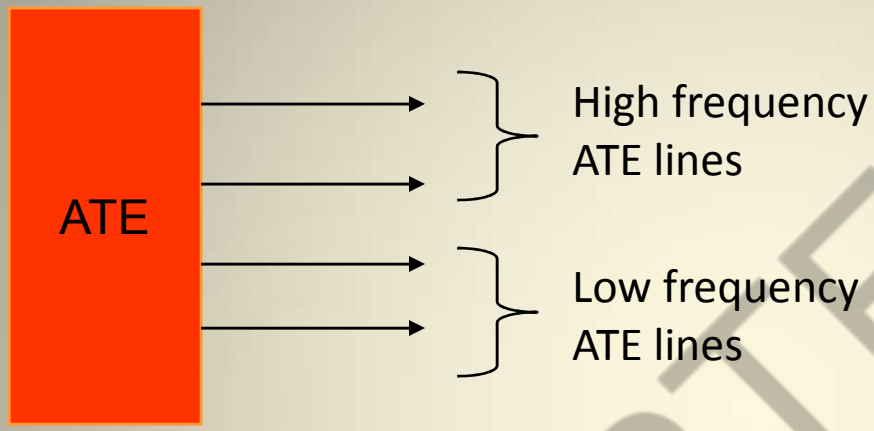
Every port of a tester, consisting of multiple channels, can configured at a desired data rate

Virtual TAMs

- Embedded core test frequency is limited by scan frequency
  - Scan frequencies are low to meet power, routing, and clock skew constraints
- Virtual TAMs allow use of high frequency ATE pins
- How can we match fast ATE data rates to slow scan frequencies?

Lecture 50

Bandwidth Matching



ATE pins : W_{ATE}= 4

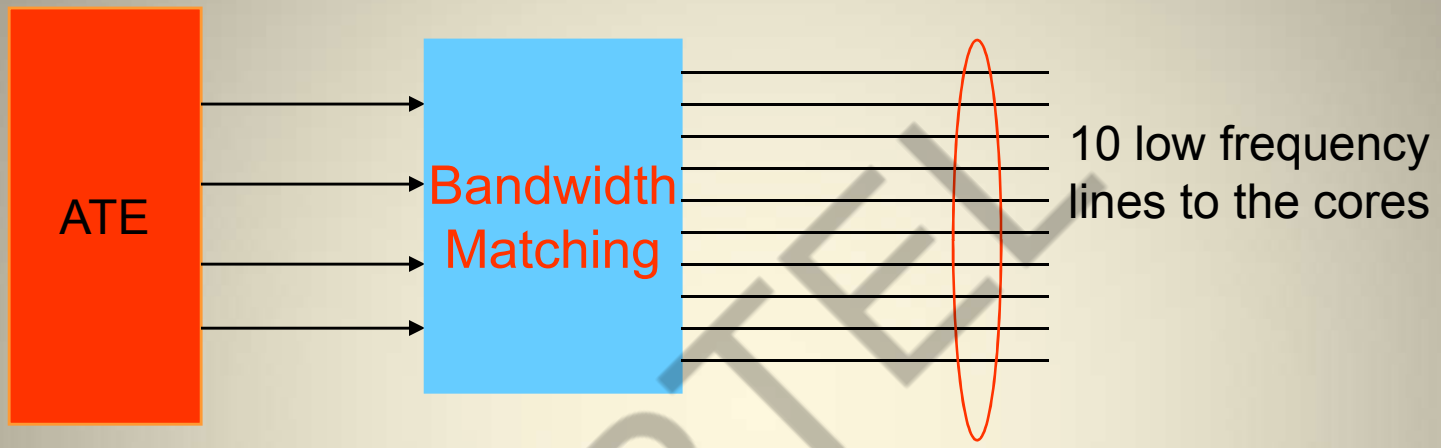
ATE frequency factor : n = 4

High frequency pins U = 2

$$W_{ATE} \times f_{ATE} = W_{TAM} \times f_{TAM}$$

$$U \times f_{ATE} + (W_{TAM} - U) \times f_{TAM} = U \times n \times f_{TAM} + (W_{ATE} - U) \times f_{TAM}$$

Bandwidth Matching



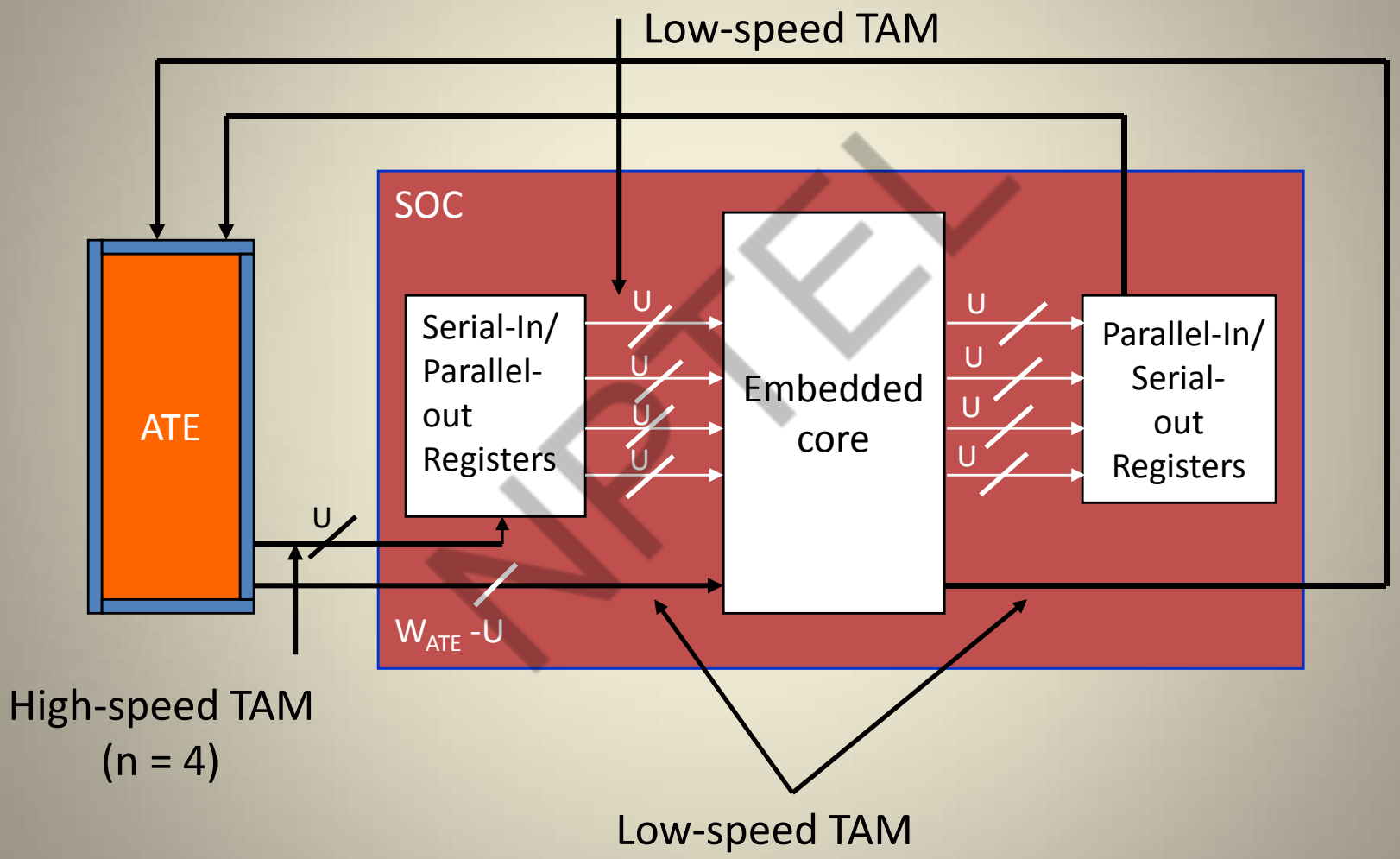
ATE pins : W_{ATE}= 4

ATE frequency factor : n = 4

High frequency pins U = 2

$$W = (n-1) \times U + W_{ATE}$$

Implementation of Bandwidth Matching



Selection of U and n

- Testing of SOC is often dominated by the testing time of bottleneck cores
- Testing time of SOCs containing bottleneck cores does not decrease for TAM widths greater than W^*
- The lower bound on test time in such SOCs is T^* corresponding to TAM width W^*

SOCs with Bottleneck Cores

SOC	W^* (bits)	T^* (clock cycles)
u226	48	5333
d281	48	3926
g1023	40	14794
p34392	36	544579
t512505	36	5228420
h953	16	119357
f2126	16	335334
q12710	16	2222349

Relationship of U, n and W*

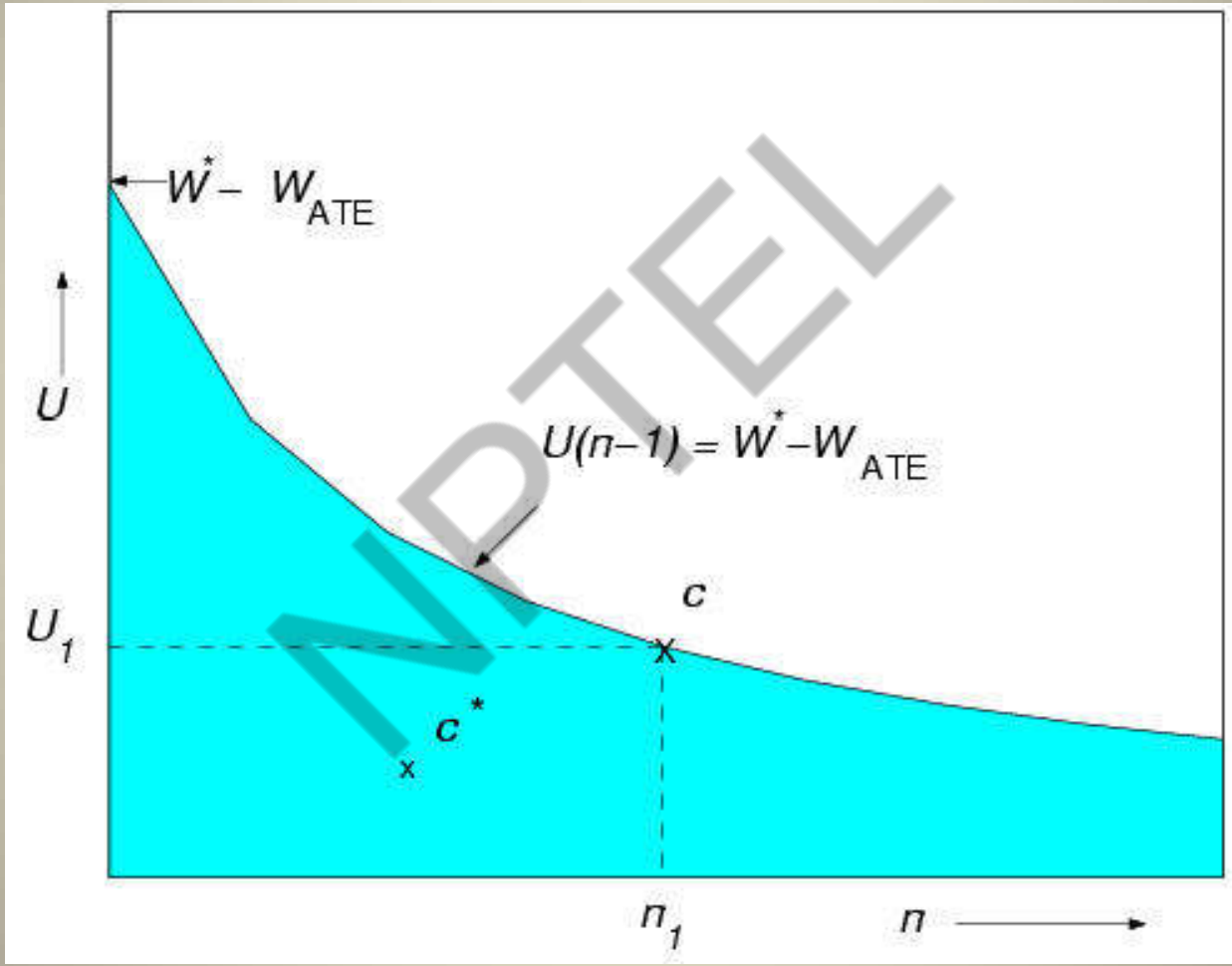
- *U* and *n* should be chosen such that total virtual TAM width *W* does not exceed *W**

$$W \leq W^*$$

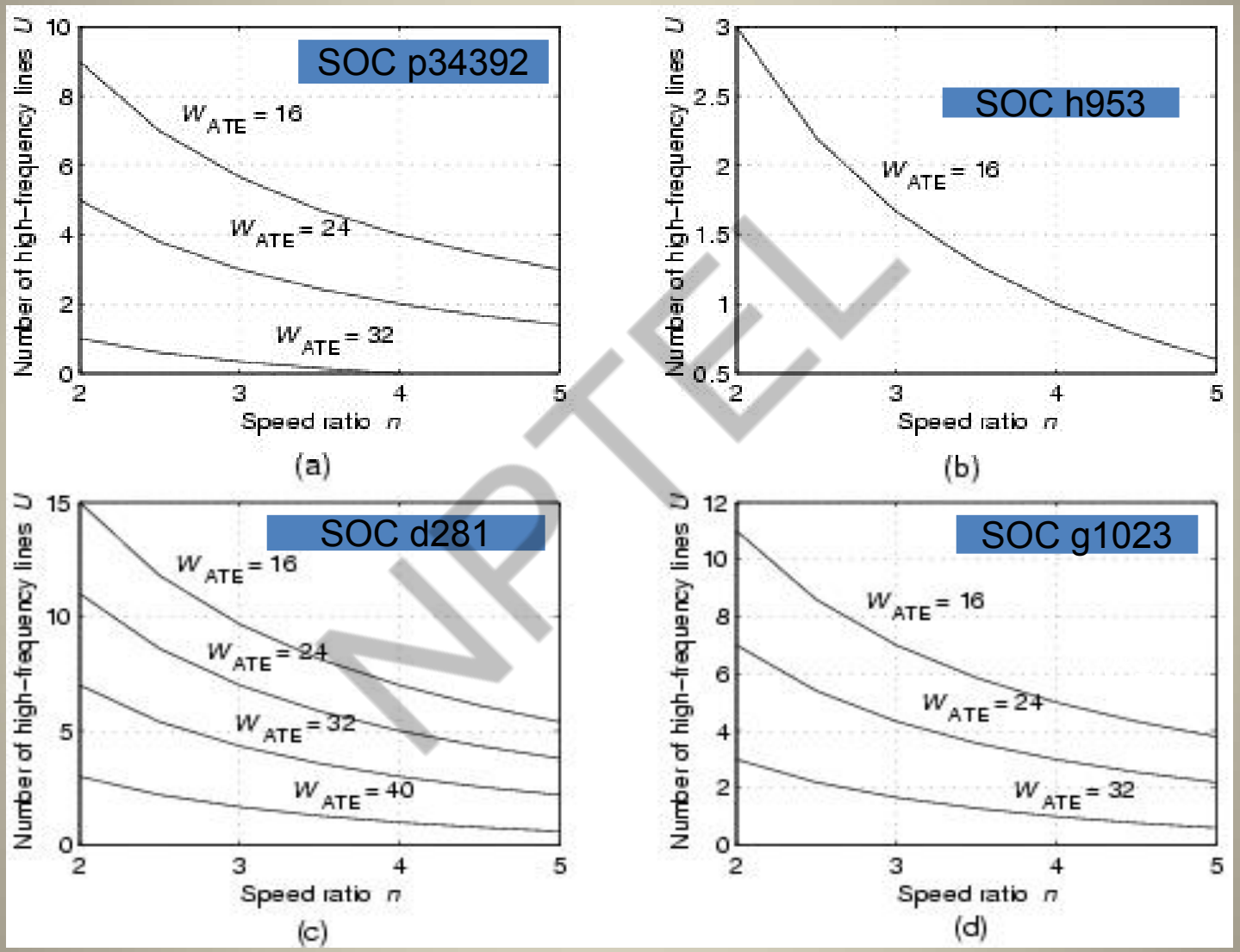
$$W = nU + (W_{ATE} - U)$$

$$U(n-1) \leq W^* - W_{ATE}$$

Variation of U with n

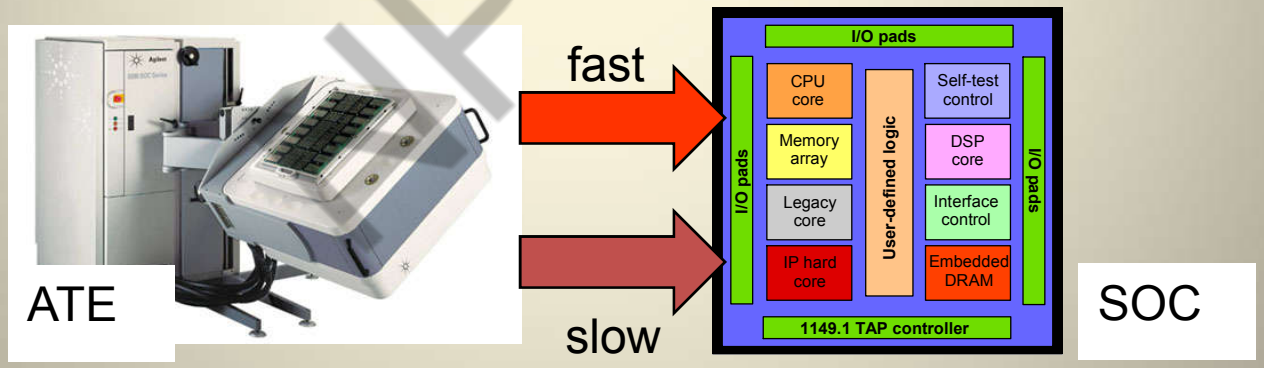


U vs n for ITC'02 Benchmarks



Multiple-Speed TAM Architectures

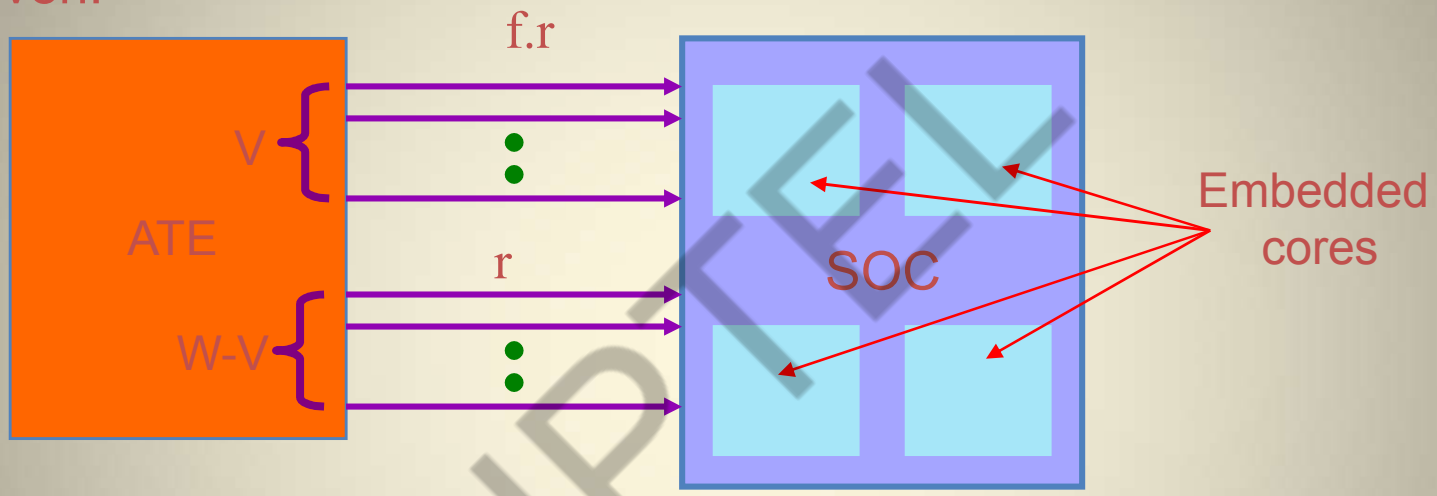
- Exploit port-scalability of ATEs
- Facilitate efficient use of high data-rate tester channels
- Unlike virtual TAMs, avoid on-chip hardware overhead
- Reduce testing time of bottleneck cores



Problem Formulation

- Dual-speed optimization problem

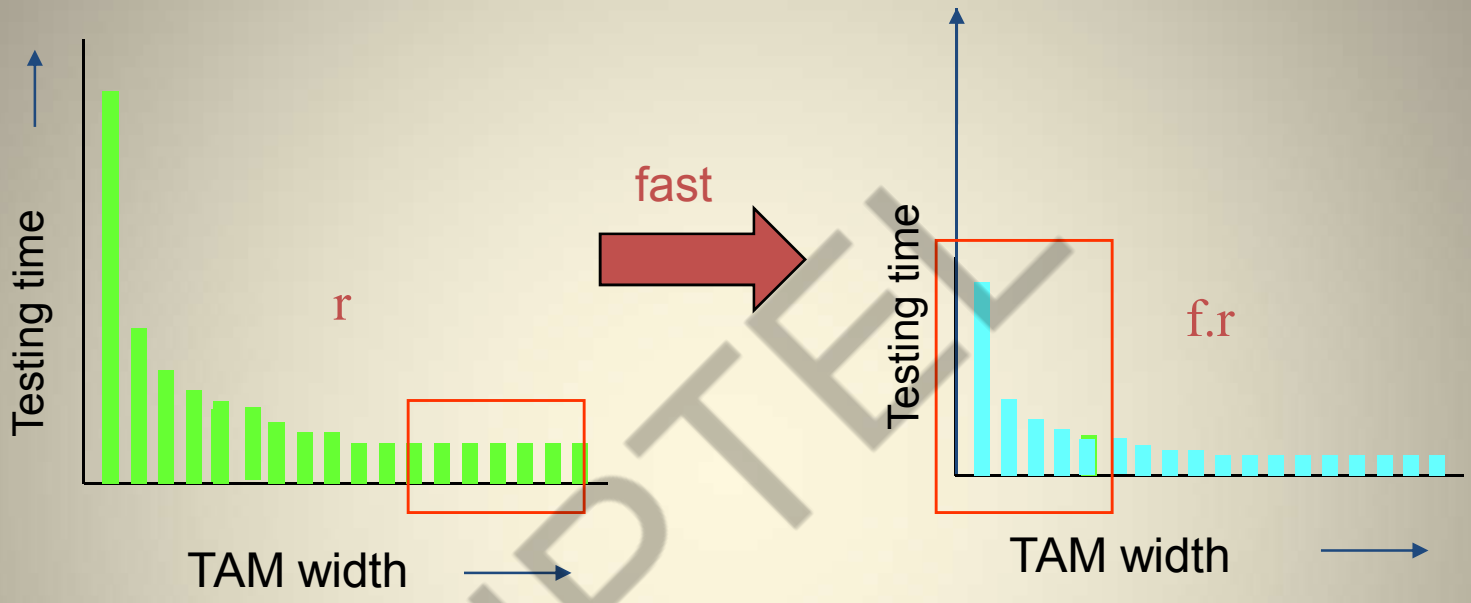
Given:



Determine the wrapper design, TAM width and test data rate for each core, and the SOC test schedule such that:

- the total number of TAM wires utilized at any moment does not exceed W
- the number of TAM wires driven at the high data rate does not exceed V
- the SOC testing time is minimized

Selection of Data Rate for a Core

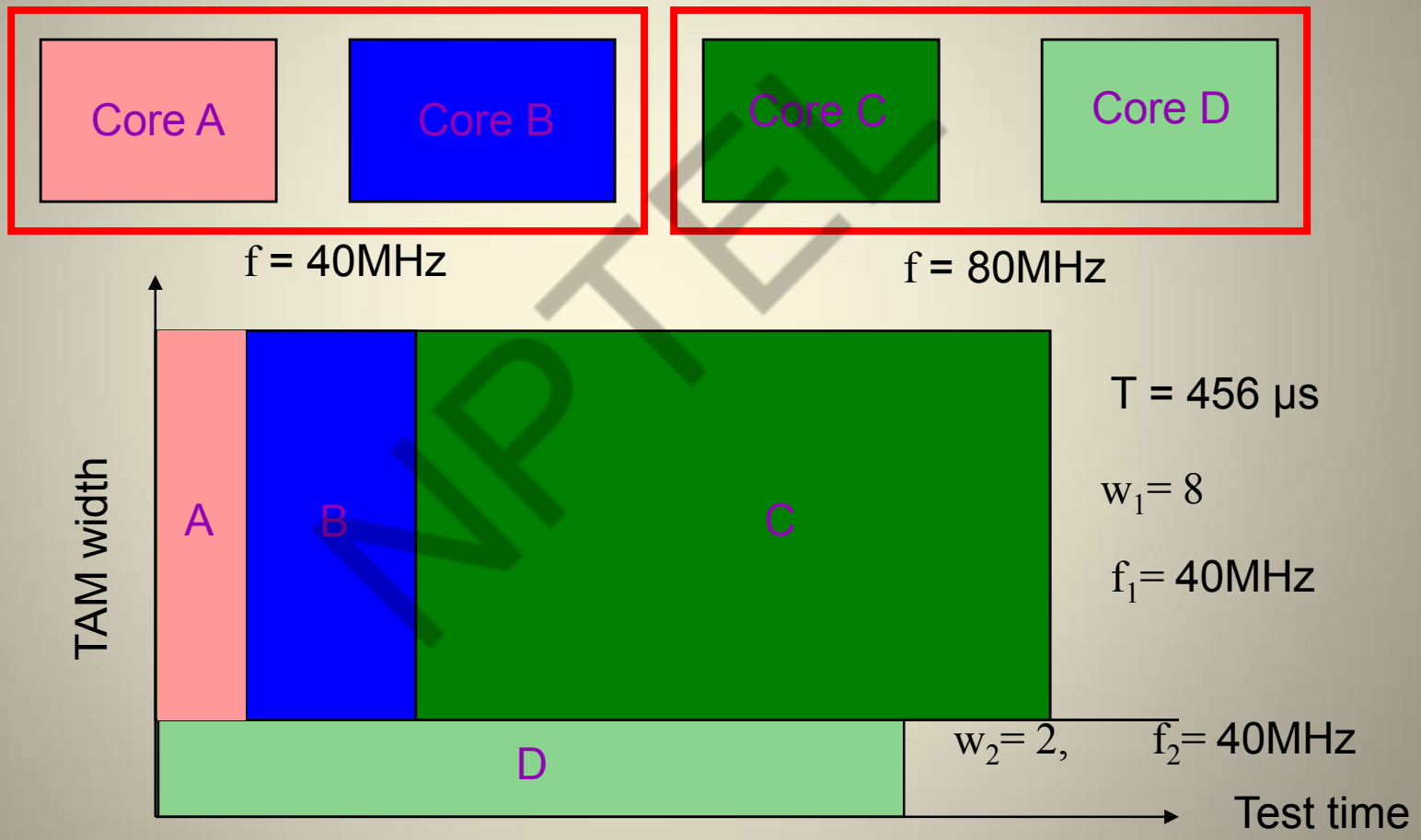


Core 5 in SOC p93791

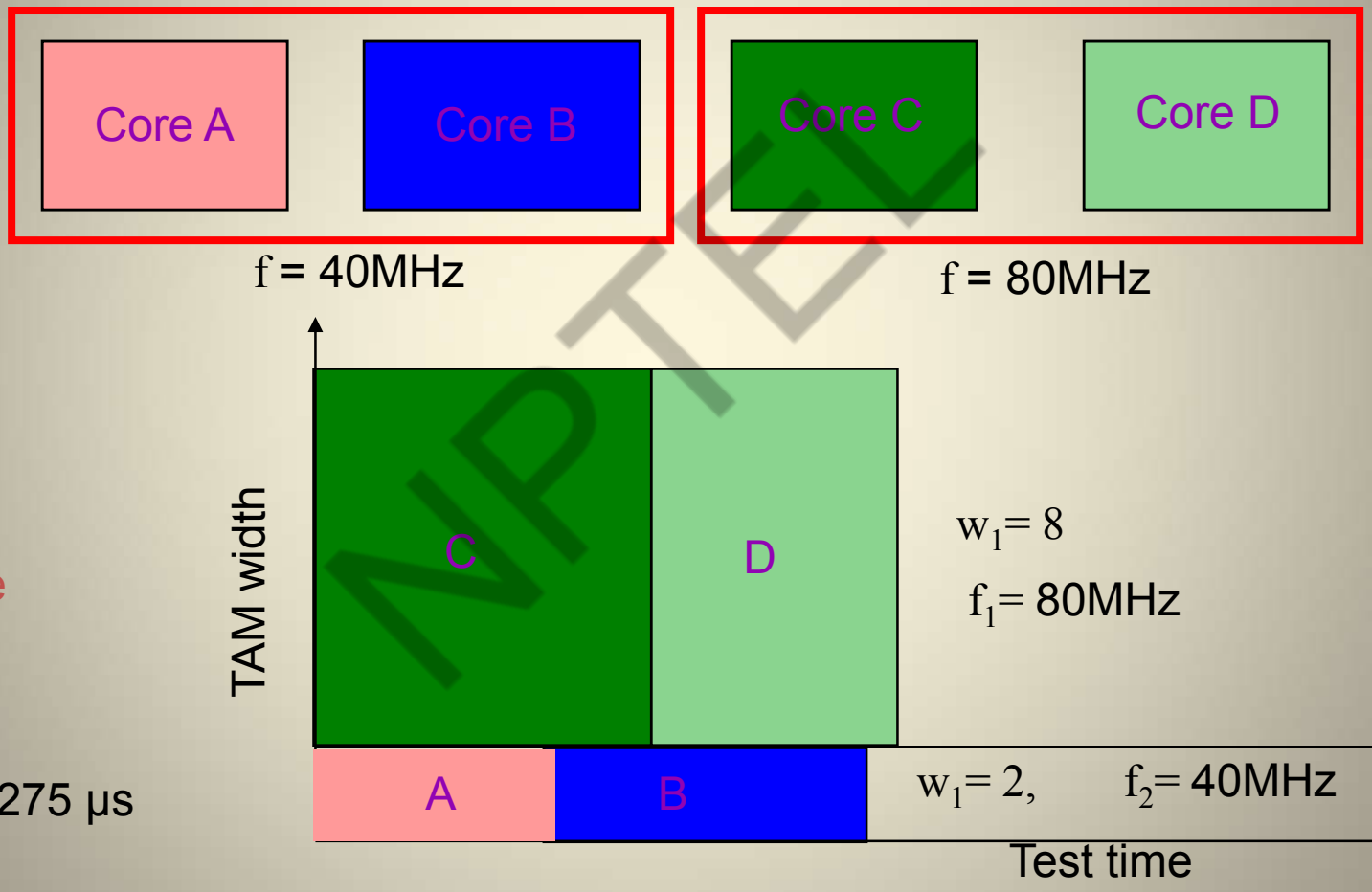
f =2	V=10	T = 14026.9µs
f =1	W-V=23	T = 11398.9µs

Matching Core Scan Frequencies to ATE Data Rates

Baseline
Case 1



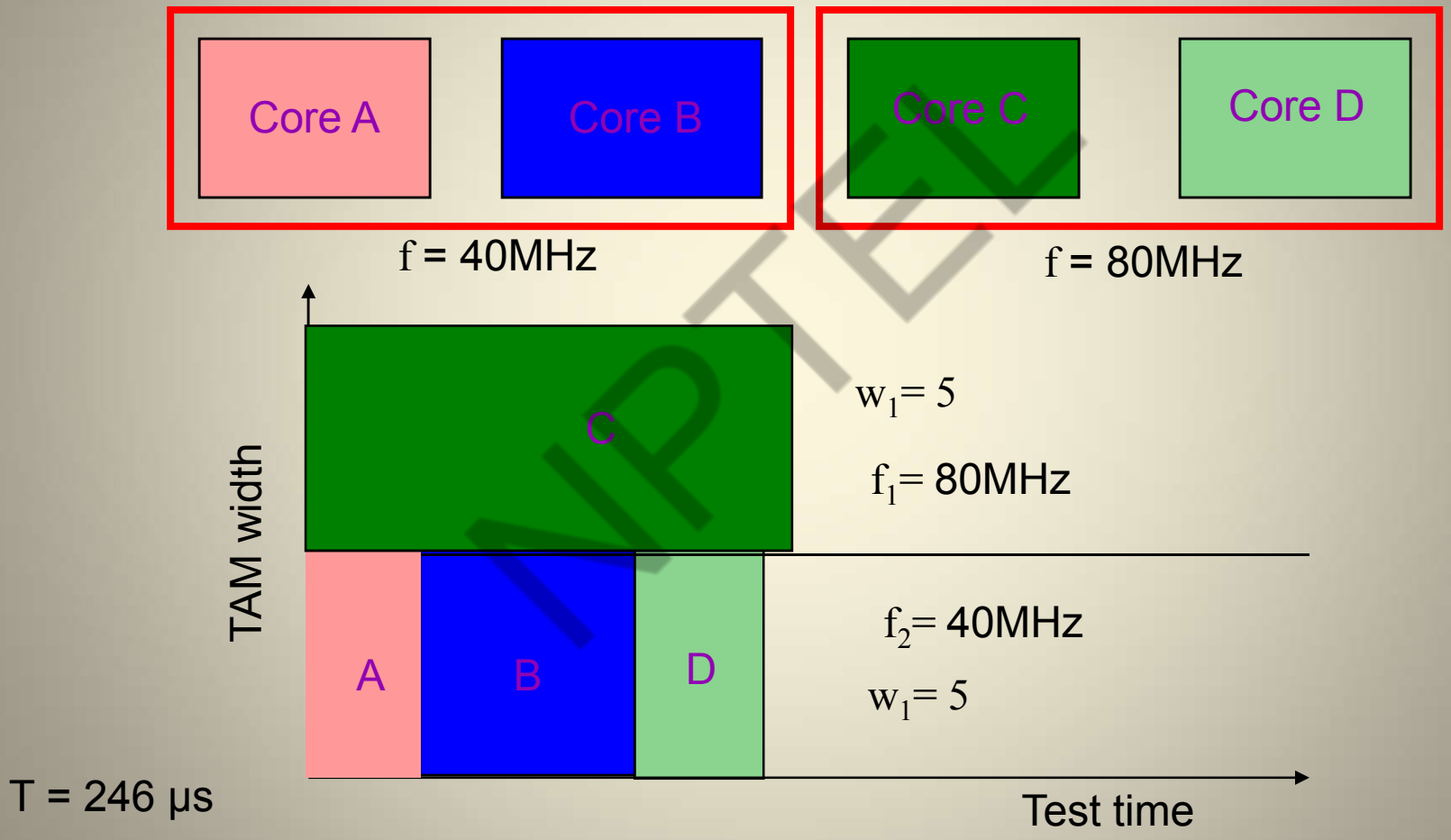
Matching Core Scan Frequencies to ATE Data Rates



Baseline Case 2

T = 275 µs

Matching Core Scan Frequencies to ATE Data Rates



Problem Statement

Given

- Test data parameters for N embedded cores
- Maximum scan frequency f_i^* for each core i
- SOC-level TAM width W

Determine

- The number of TAM partitions B
- Width w_j and scan frequency f_j of each TAM partition j
- Assignment of cores to TAM partitions

Such that

- TAM frequency does not exceed the maximum scan frequency of any core assigned to that TAM partition
- The overall test time is minimized
- The sum of the widths of all the TAM partitions does not exceed W

Solution Techniques

- Lower bound on test time based on geometric arguments (rectangle packing)
- Integer linear programming
  - Exact optimization method, limited to small problem instances
- Fast heuristic method
  - Scalable, close to optimal results

Lecture 51

Conclusions

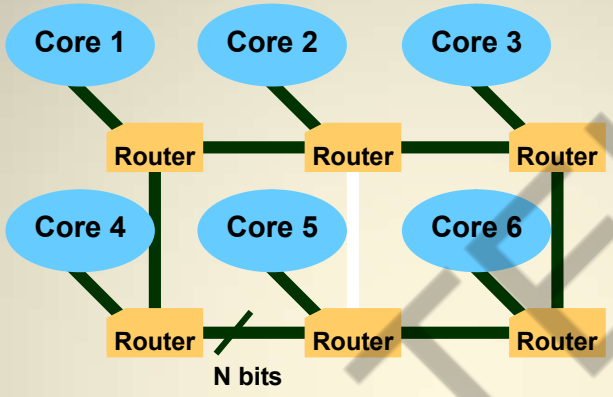
- Test reuse, test time minimization, and test compression are necessary to reduce test cost for SOCs
- Wrapper/TAM optimization and test scheduling can reduce test time for core-based SOCs
- Virtual TAMs offer several advantages for SOC testing
  - On-chip TAM wires are not limited by the number of available pins on the SOC
  - Better utilization of high-speed ATE channels reduces testing times
- TAM architectures can match port-scalable ATE channels to different scan frequencies of embedded cores

Introduction to Network-On-Chip Testing

- For future SoCs with large number of cores and increased interconnect delay, traditional point-to-point or bus-based communication architecture becomes new bottleneck.
- Traditional communication architectures cannot meet system requirements of bandwidth, latency, and power consumption.
- Integrated switching network has been proposed as an alternative approach to interconnect cores in SoC.
- Such networks rely on a scalable and reusable communication platform, called network-on-chip (NoC) system, to meet two major requirements: reusability and scalable bandwidth.

Conceptual Architecture of a NoC System

- The figure shown below represents a 2-D mesh NoC.



- Cores are connected to NoC by routers or switches.
- Data are organized by packets and transported through interconnection links.
- Various network topologies and routing algorithms can be used to meet requirements of performance, hardware overhead, power consumption.

Special Features of NoC Testing

- The greatest difference between NoC testing and SoC testing is on test access mechanism design.
- On-chip-network of a NoC can be reused as a TAM for test packet delivery. Theoretically, no TAM interconnects are required to be invested.
- Test time can be reduced by network reuse even under power constraints, with minimized pin count and area overhead.
- Generally, more cores can be tested in parallel than TAM-based SoC testing, due to large NoC channel bandwidth.