

**DYNAMIC ALLOCATION OF DATAPATH RESOURCES
FOR LOW POWER***

**Dmitry Ponomarev, Gurhan Kucuk, Kanad Ghose
Department of Computer Science
State University of New York
Binghamton, NY 13902–6000**

Workshop on Complexity Effective Design, June 30, 2001

*** supported in part by DARPA through the PAC–C program and NSF**

MOTIVATION

Current trends in microarchitecture:

- Aggressive out-of-order execution, use of register renaming, multiple FUs, sizable on-chip caches, large register files, ROB etc.

Impact on Energy/Power Dissipation:

- Absolute power dissipation of processor is high
- Areal energy/power density of high-end superscalar processors is becoming an immediate, serious concern – will soon become comparable to that of nuclear reactors

Consequences:

- Intermittent and permanent failures on the die and serious challenges for the cooling facilities

Thus:

- Energy dissipation needs to be controlled through technology independent techniques

Related Work – Dynamic Resource Allocation for Instruction Scheduling

Adaptive Issue Queue (Rochester, Albonesi et al):

- multi-partitioned issue queue
- number of partitions dynamically allocated based on number of ready flags set in entries within active partition
- IPC drop triggers growth

Resizable Issue Queue (UPC, Barcelona, Folegnani/Gonzalez)

- FIFO issue queue, multi-partitioned
- resizing based on number of instruction committed from "youngest" partition: used for downsizing
- Number of active partitions grown periodically to avoid performance loss

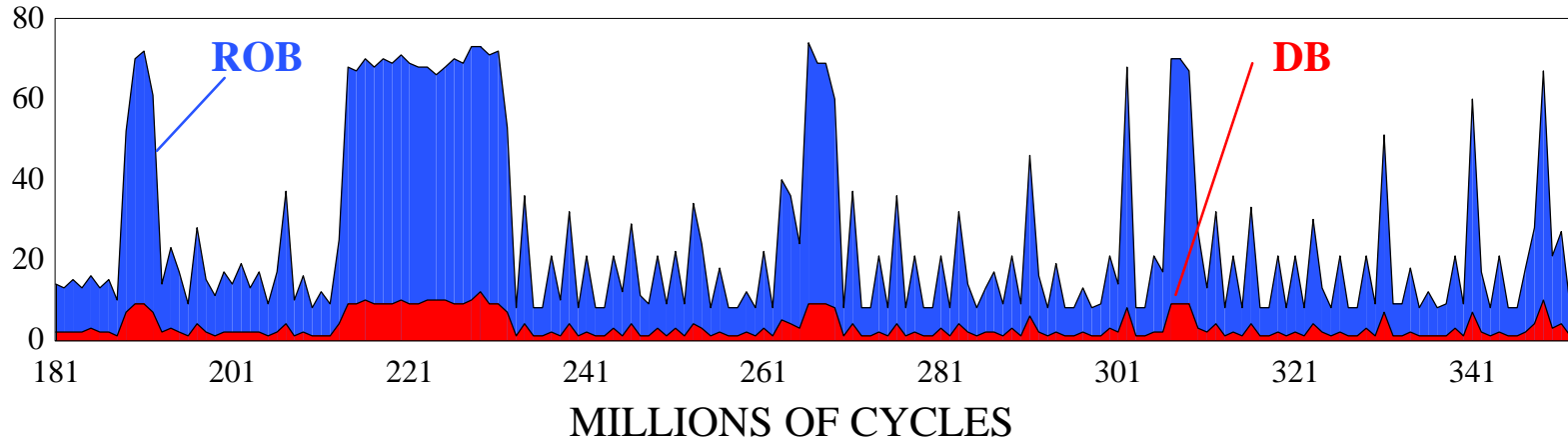
Pipeline Balancing (Compaq, Bahar & Manne)

- For multiclustered datapath organizations
- Dynamic Resizing of Issue Queue & Dynamic Cluster Activation
- IPC monitored to allow clusters/issue queue partitions to be turned off with minimal impact on performance

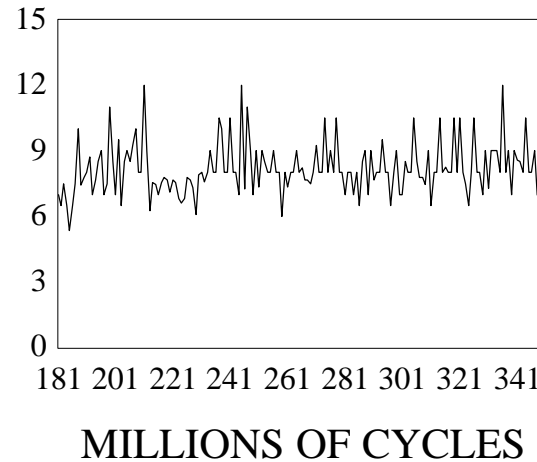
+ others (IPC monitoring & resource control by OS, dynamic profiling)

RESOURCE USAGE IN SUPERSCALAR DATAPATH: EXAMPLE

Occupancies of the DB and the ROB for FPPPP benchmark



RATIO OF OCCUPANCIES
OF THE ROB AND THE DB



RESOURCE USAGE IN SUPERSCALAR DATAPATHS: OBSERVATIONS

- Resource requirements of applications change dramatically
- Resource usage is positively correlated– where the usage of the ROB increases, the usage of the DB increases as well
- The ratio of the ROB and the DB occupancies changes widely – in the above example between 4 and 12.
- Can reduce power dissipation by adjusting resource sizes dynamically. Need separate monitoring of each resource.

BOTTOM LINE:

The one-size-fits-all approach results in overcommitting datapath resources and considerable waste in energy/power

DYNAMIC RESOURCE TUNING:

OPTIMIZING ACROSS THE POWER–PERFORMANCE SPECTRUM

■ TUNABLE PARAMETERS

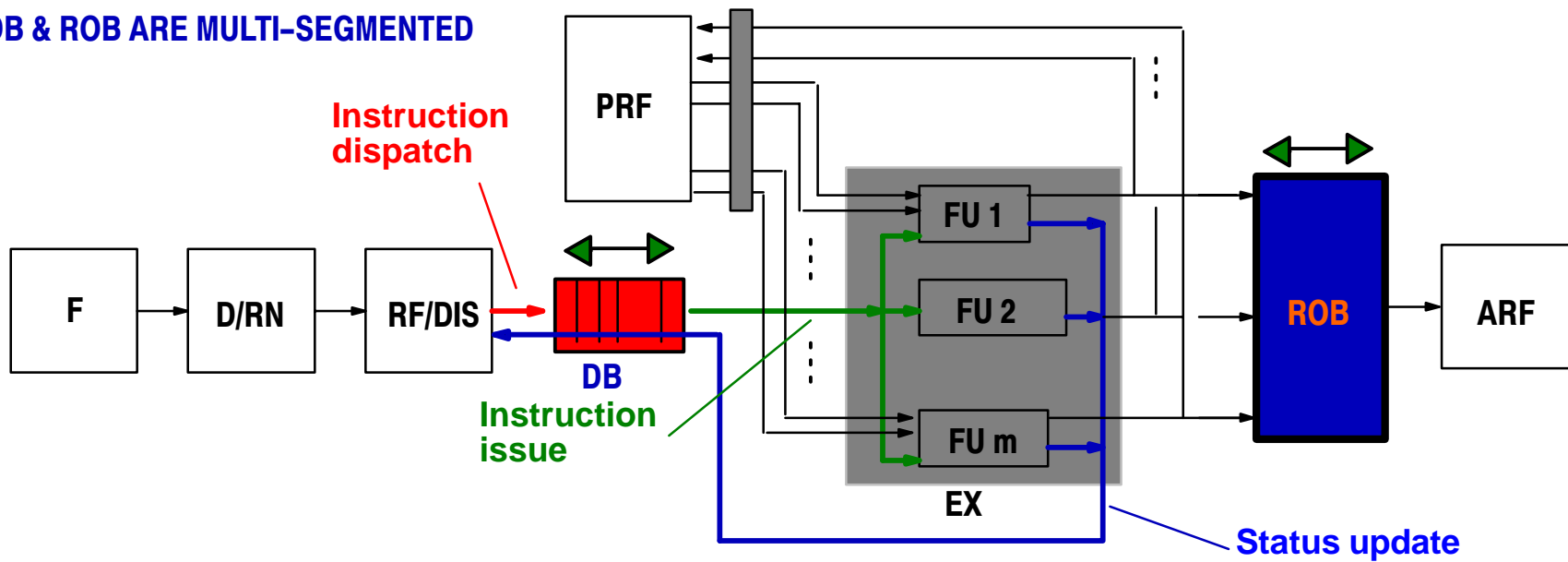
- 1) DYNAMIC SCHEDULING RESOURCES: RESIZE DYNAMICALLY
- 2) INSTRUCTION DISPATCH RATE: VARY DYNAMICALLY

■ GOALS

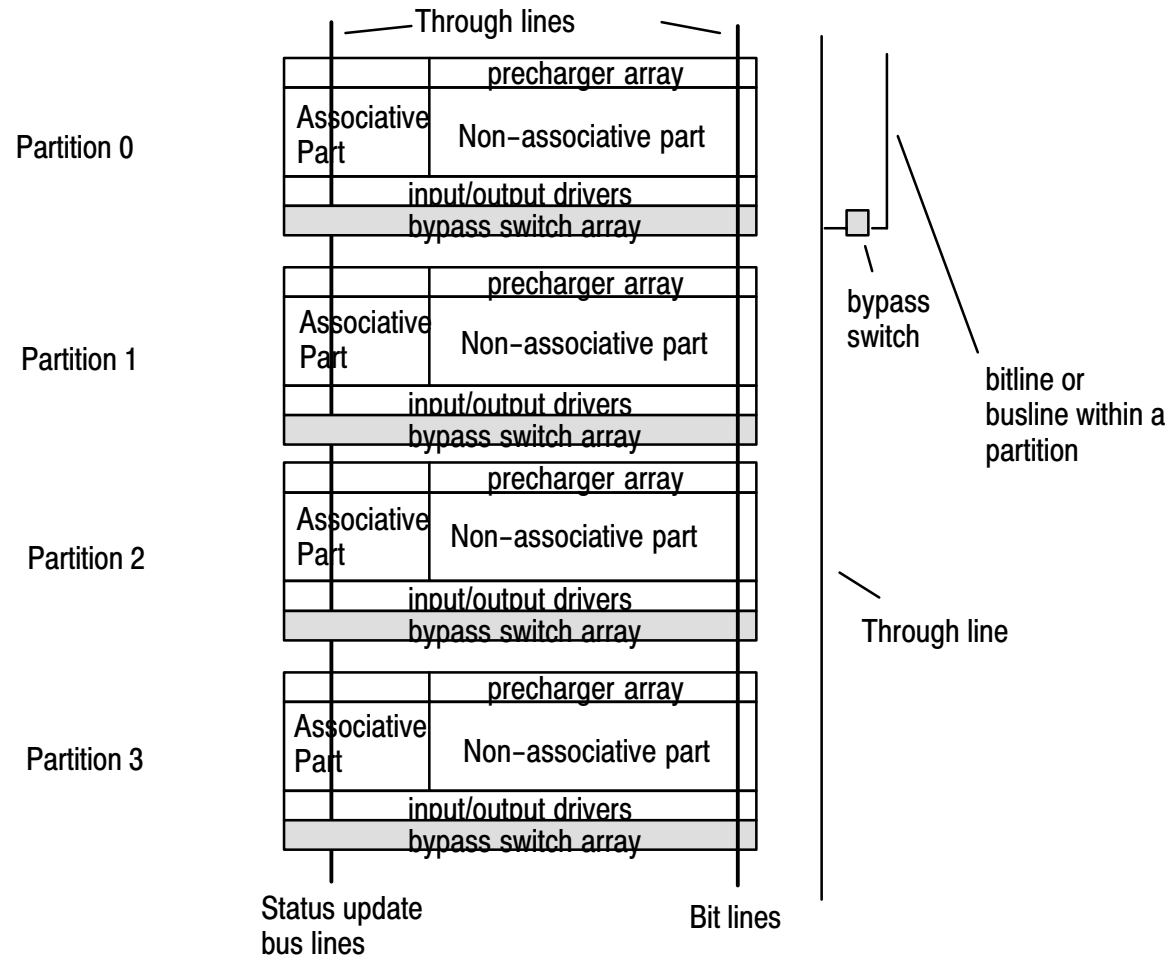
- 1) MUST USE SIMPLE INSTRUMENTATION
- 2) EASY TO RETROFIT TO EXISTING DATAPATHS
- 3) STABLE CONTROL

DYNAMIC RESIZING OF DISPATCH BUFFER (DB) AND REORDER BUFFER (ROB)

DB & ROB ARE MULTI-SEGMENTED



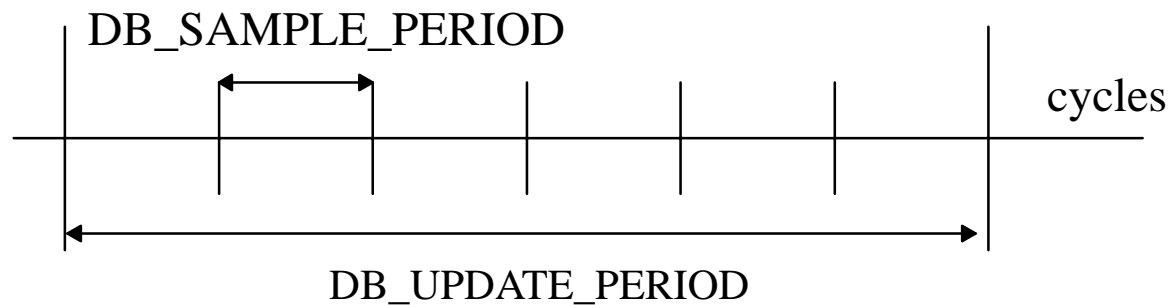
SEGMENTED DB ORGANIZATION



DB RESIZING: 2 PHASES

DB DOWNSIZING:

- Sample DB occupancy every *db_sample_period* cycles: reduces counting overhead
- Take the average of these samples at the end of *db_update_period*: approximates DB occupancy throughout the update period
- If the measured average is less than current size of the DB by more than partition size → turn-off one (or more) partition(s). Otherwise, leave DB size unchanged.



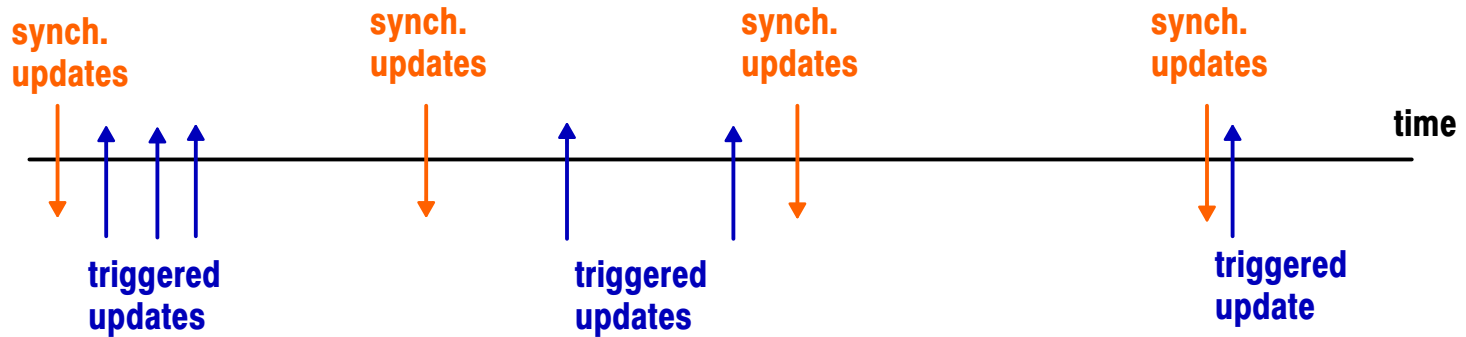
DB RESIZING: 2 PHASES

DB UPSIZING:

- Count the number of cycles when dispatch blocks because DB is full
- If this counter exceeds *db_overflow_threshold* → add one partition
 - upsizing is more aggressive than downsizing – reduces hit on performance
- Reset *db_overflow_threshold* to 0 at the beginning of a new *db_update_period*

SUMMARY OF THE CONTROL STRATEGY

synch. updates are used for resizing DOWN



triggered updates are used for resizing UP

SUMMARY OF THE CONTROL STRATEGY (continued)

Only three parameters used for control:

db_overflow_threshold

db_update_period

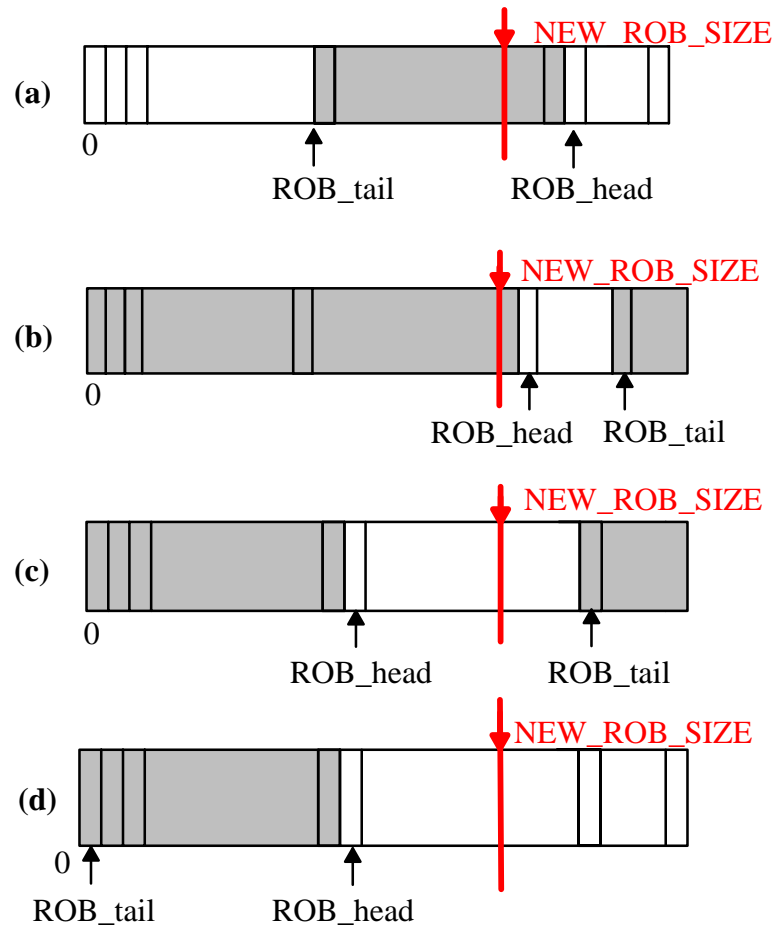
db_sample_period

Advantages:

- Can easily achieve a desired power/performance tradeoff by adjusting *db_overflow_threshold* and *db_update_period*
- Monitoring on a cycle-by-cycle basis is avoided – done once every *db_sample_period* cycles

POSSIBLE DELAYS IN ROB RESIZING: SCENARIOS

■ valid entries = current extent of the ROB □ free entries



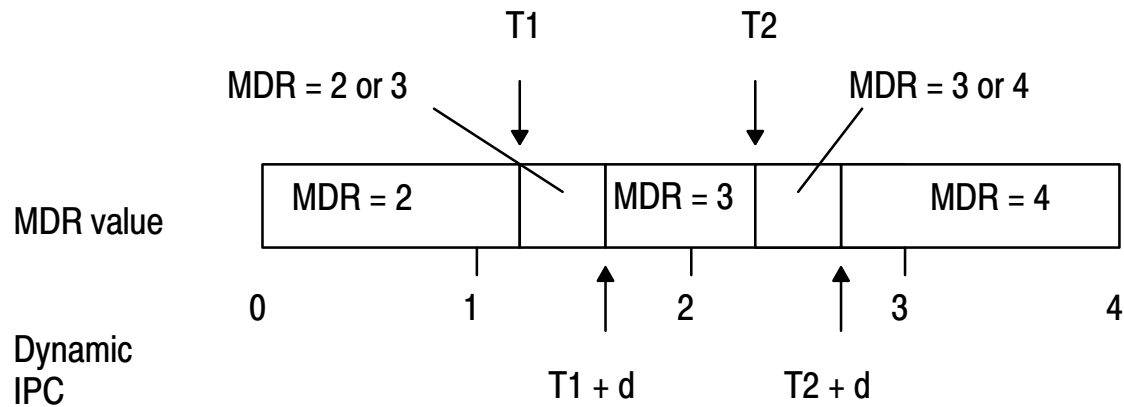
ADDITIONAL LEVEL OF CONTROL: ADAPTIVE DISPATCH RATE (ADR)

Maximum Dispatch Rate is adapted periodically based on the measured IPC in this period and the values of two thresholds

Thresholds can float within ranges:

T1 range: 0.5–1.6

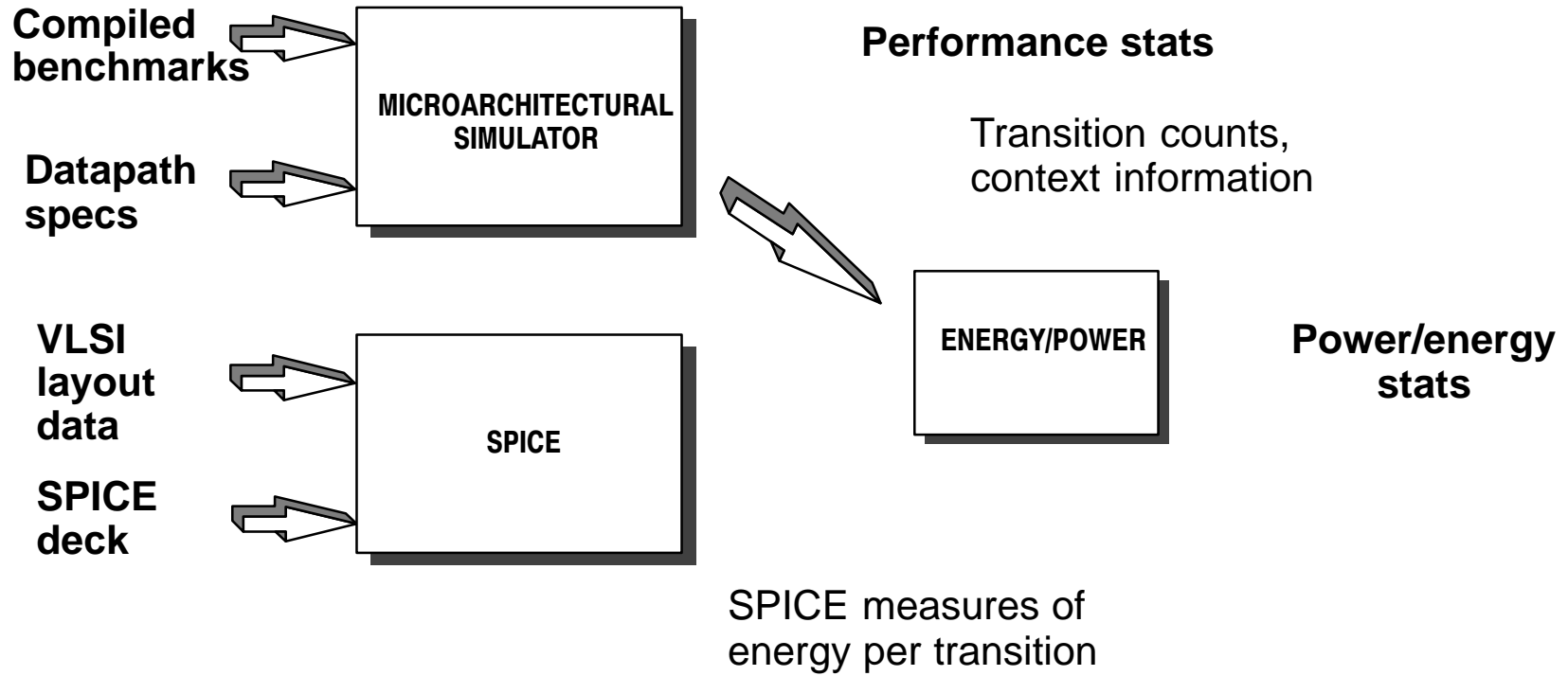
T2 range: 2.1 – 2.5



APPROACH TO POWER ESTIMATION

- Used a **true cycle-by-cycle register-level simulator for a typical superscalar pipeline**. SimpleScalar has been substantially modified for this purpose
- Simulated the execution of **SPEC 95 benchmarks**.
- Collected transition counts for each major datapath component
- Used **SPICE measurements for the VLSI layout** of dispatch buffer and reorder buffer in a 0.5 micron, 4-metal layer process to estimate the power dissipated for each type of transition within each major component (migrating to 0.18 micron soon!)
- Computed total energy dissipation within the DB and the ROB based on transition counts and energy dissipated per transition for each component, for each benchmark
- **Overall accuracy is as good as it gets short of an implementation**

DATAPATH POWER ESTIMATOR



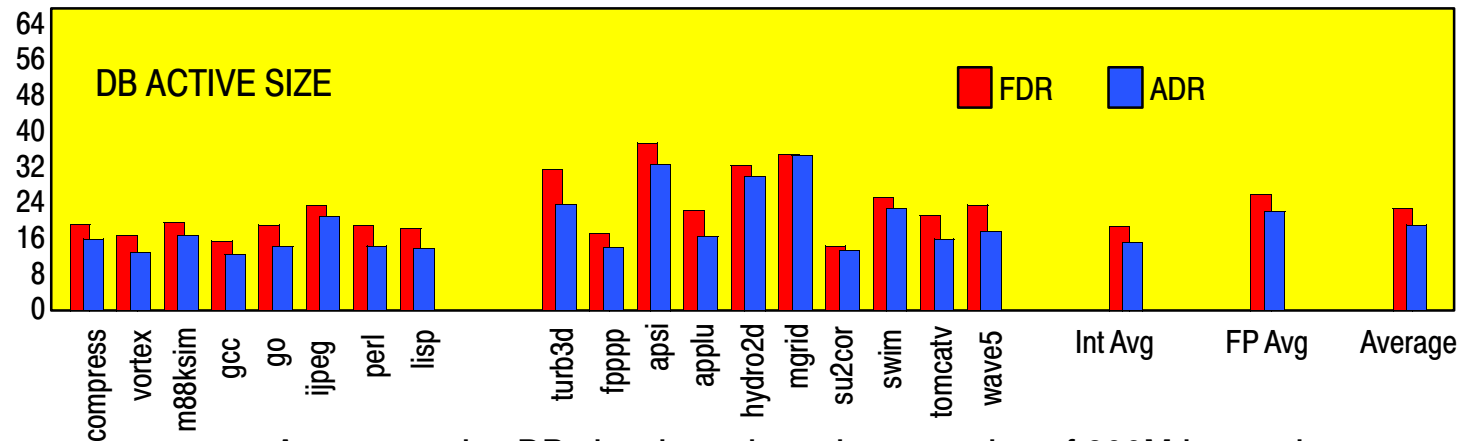
EXPERIMENTAL RESULTS: SUMMARY

- Behavior is very sensitive to the ratio of *update_period* to *overflow_threshold*
- Optimum power/performance tradeoff is achieved when the above ratio is 4 → a partition is added to a resource **if more than one overflow occurs in four cycles**.
- The following results are for *overflow_threshold* of 512, *update_period* of 2048 cycles and *sample_period* of 32 cycles for both DB and ROB. This allows for the acquisition of 64 samples in one *update_period*.

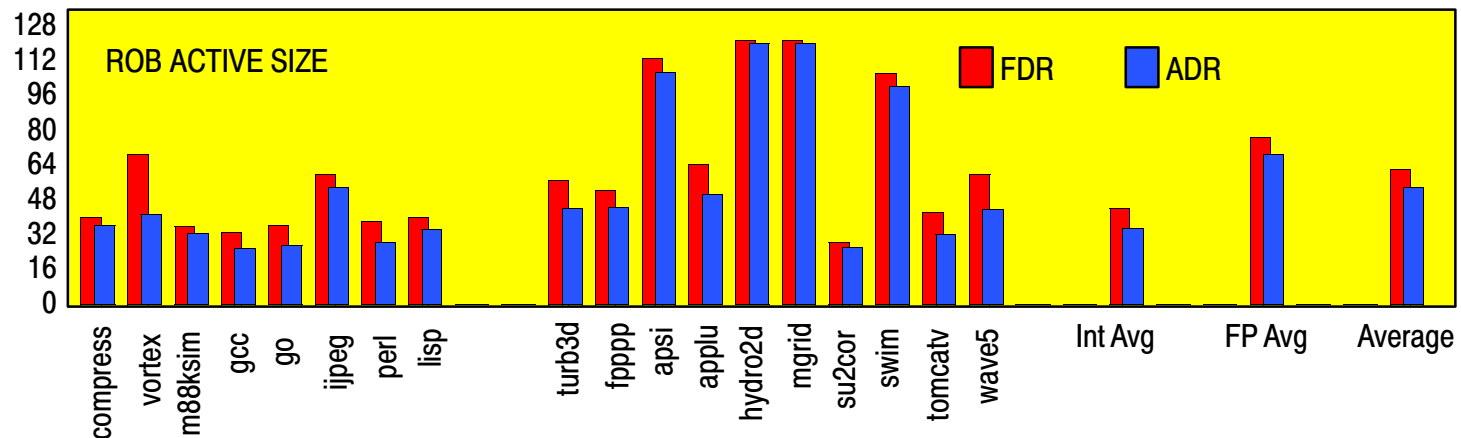
EXPERIMENTAL RESULTS: ACTIVE SIZE REDUCTION

FDR: 64% for DB AND 51% for ROB

ADR: 70% for DB, 58% for ROB



(a) Average active DB size throughout the execution of 200M instructions

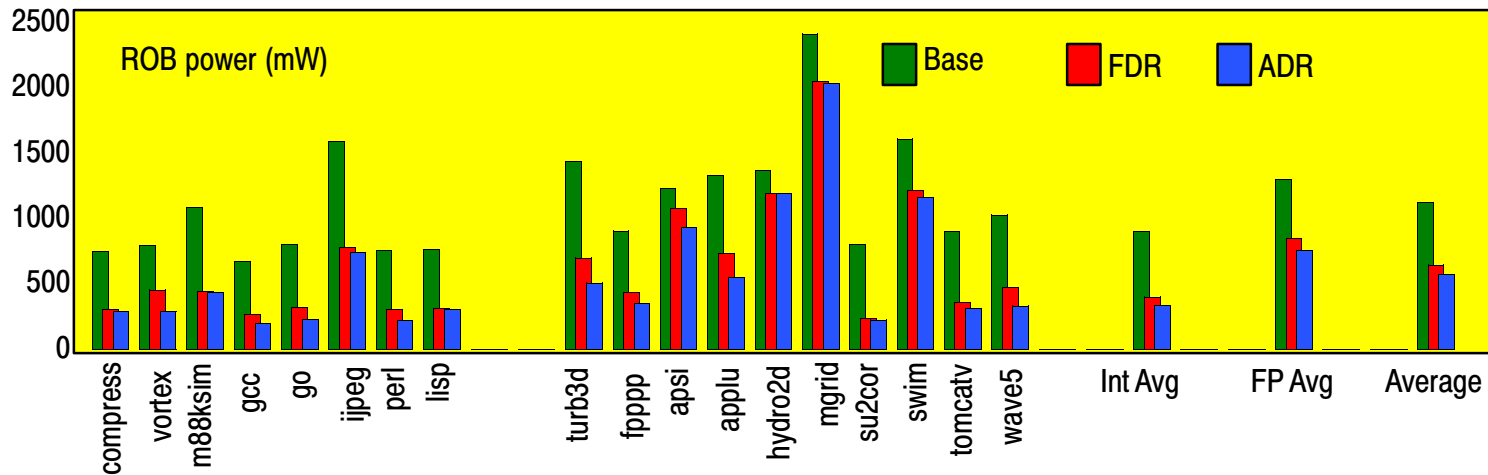
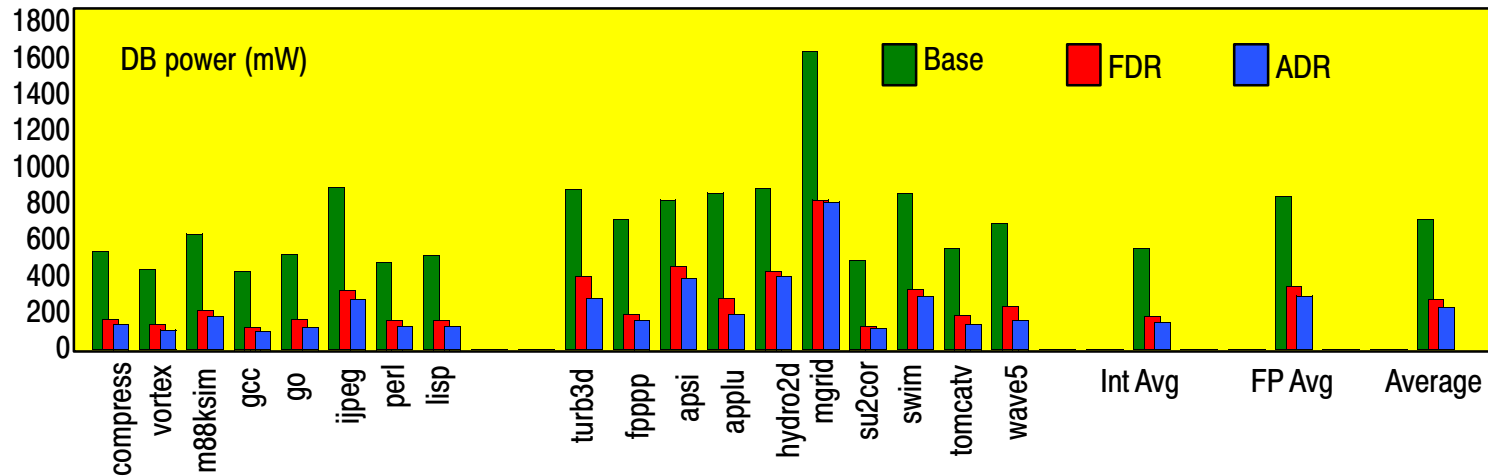


(b) Average active ROB size throughout the execution of 200M instructions

EXPERIMENTAL RESULTS: POWER REDUCTION

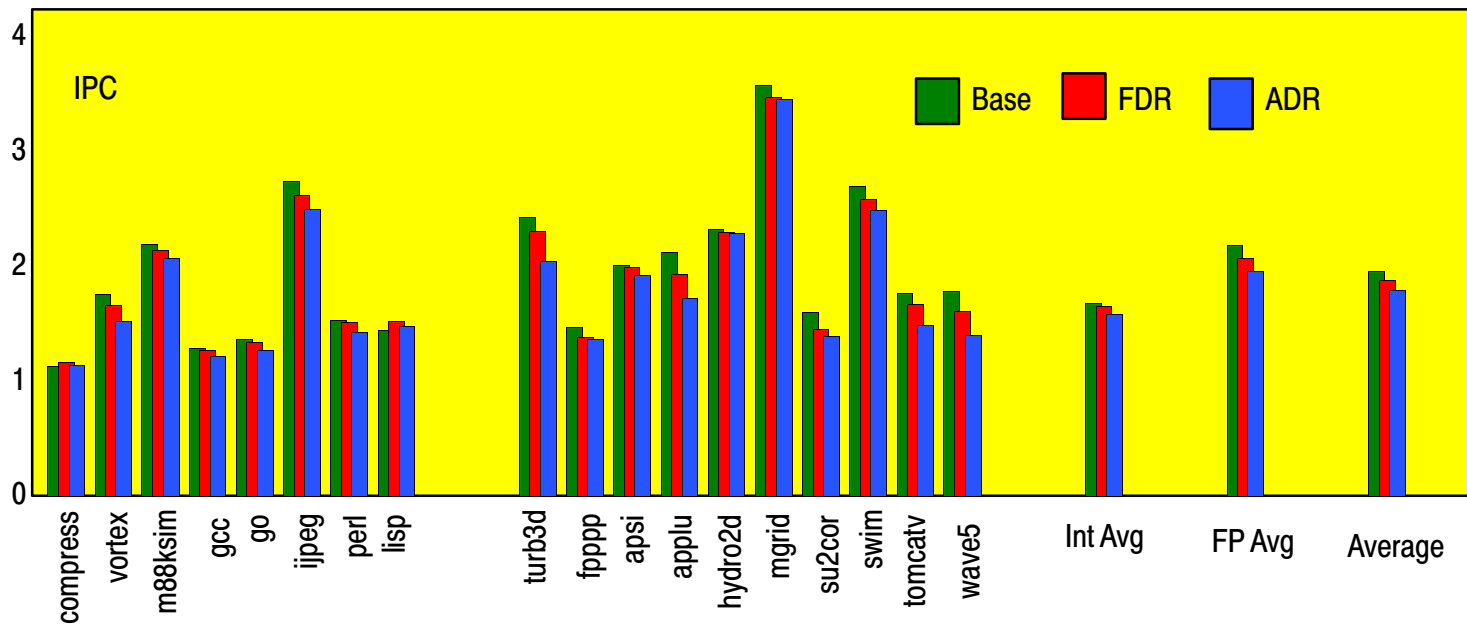
FDR: 62% for DB AND 41% for ROB

ADR: 68% for DB, 49% for ROB



EXPERIMENTAL RESULTS: EFFECT ON PERFORMANCE

AVERAGE IPC DROP: FDR – 3.5%, ADR – 8.5%



CONCLUDING REMARKS

- Significant power savings with minimal impact on performance are achieved by dynamically resizing **multiple** datapath resources
- Simple control strategy is used that avoids resource monitoring on a cycle-by-cycle basis
- Basic techniques are orthogonal to other power reduction strategies like selective bit-slice activation, frequency and voltage scaling and additional circuit techniques.
- Current efforts are to extend this to more resources and datapath styles and the design of the MORPH processor, where these and other power reduction techniques will be integrated.