THETA EDA

Efficient Wake-Up Scheduling for Multi-Core Systems

Ming-Chao Lee*, Yiyu Shi**, Yu-Guan Chen*,
Diana Marculescu***, Shih-Chieh Chang*

* Dept. of CS, National Tsing Hua University, Taiwan

** Dept. of ECE, Missouri University of Science and Technology

*** Dept. of ECE, Carnegie Mellon University

Outlines

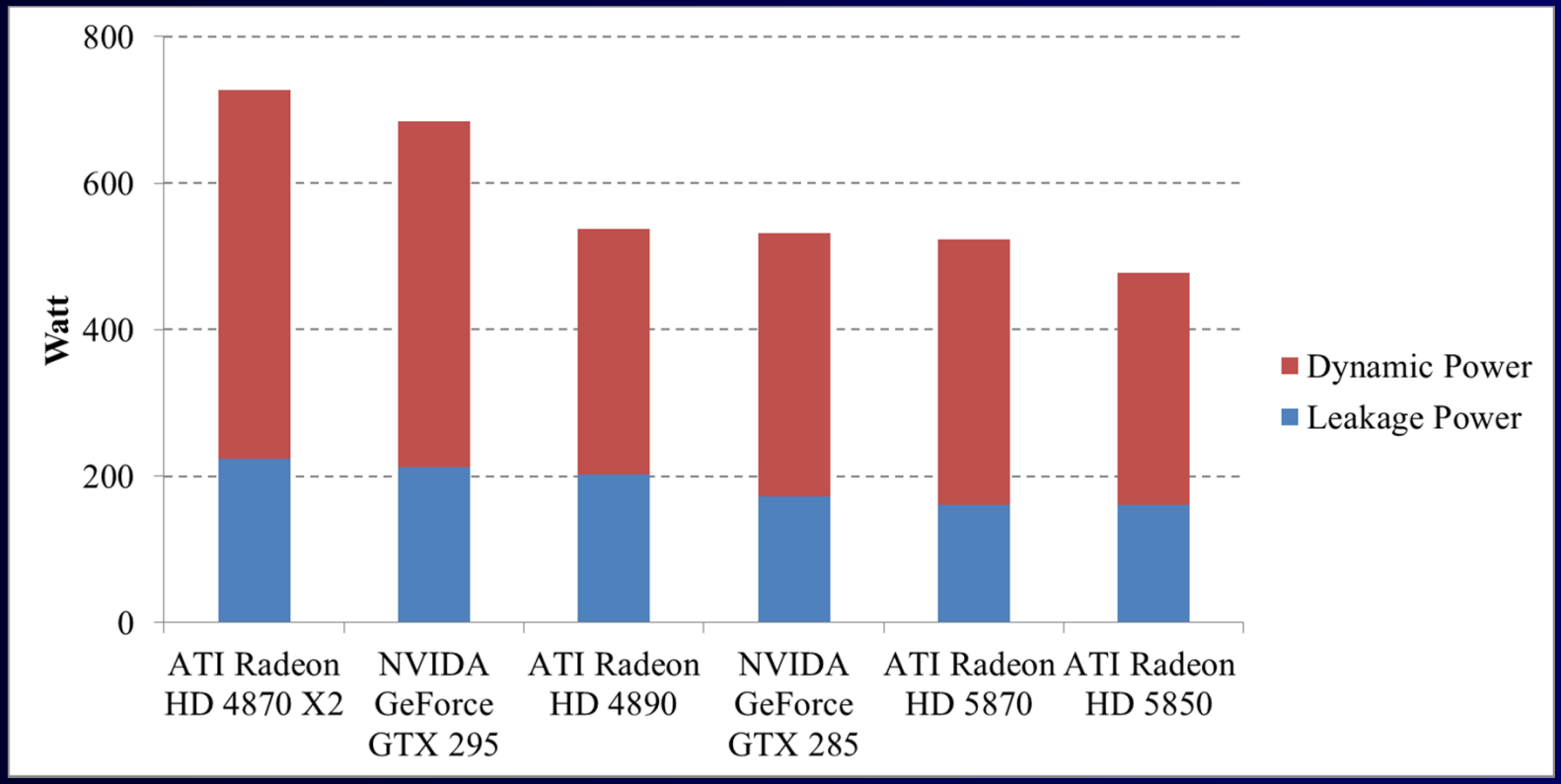
- Introduction
- Problem Formulation
- Efficient Wake-up Scheduling Algorithm
- Experimental Results
- Conclusions

Outlines

- Introduction
- Problem Formulation
- Efficient Wake-up Scheduling Algorithm
- Experimental Results
- Conclusions

Multi-core System

- Multi-core architecture is widely adopted for high performance computing.
- A serious problem for multi-core designs is their large power consumption.



Power Gating Technique

- Power gating
  - is a well-known technique to suppress the leakage power
  - inserts sleep transistors between logic devices and ground rail.
- During the power mode transition from “off” to “on”, a sudden current may flow through sleep transistors.
  - thereby degrading signal and power integrity in the nearby operating devices.
- Wake-up scheduling is to design a wake-up sequence of sleep transistors
  - the total wake-up time is minimized ,
  - and without causing IR-drop violations.

Motivation

- For a multi-core design, the number of active cores and their locations vary with applications.
- An effective multi-core wake-up scheduling should best decide the wake-up order at runtime.
  - This puts high demand on the on-line scheduling algorithm.

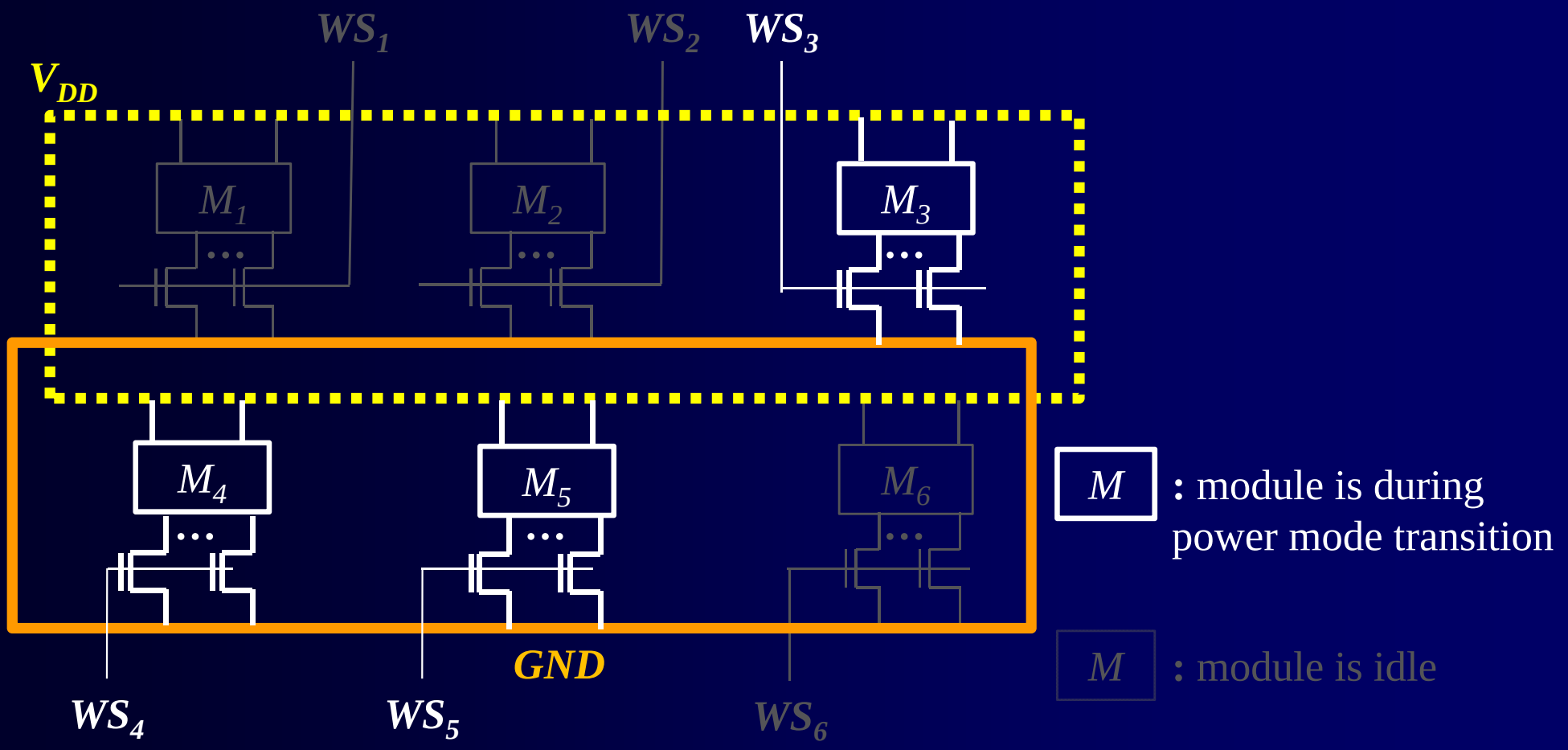
Contributions

- We propose an on-line wake-up scheduling framework for the multi-core architecture.
  - We first use a heuristic algorithm to construct a *multi-conflict graph* (*MCG*) from the multi-core design off-line.
  - Based on the MCG, we develop a fast linear-time on-line scheduling algorithm to determine the optimal wake-up order for a given set of cores from a task scheduler.
- Our approach can achieve 46.01% speedup on average over the industrial approach [4] in the wake-up latency without violating the noise constraint.

Outlines

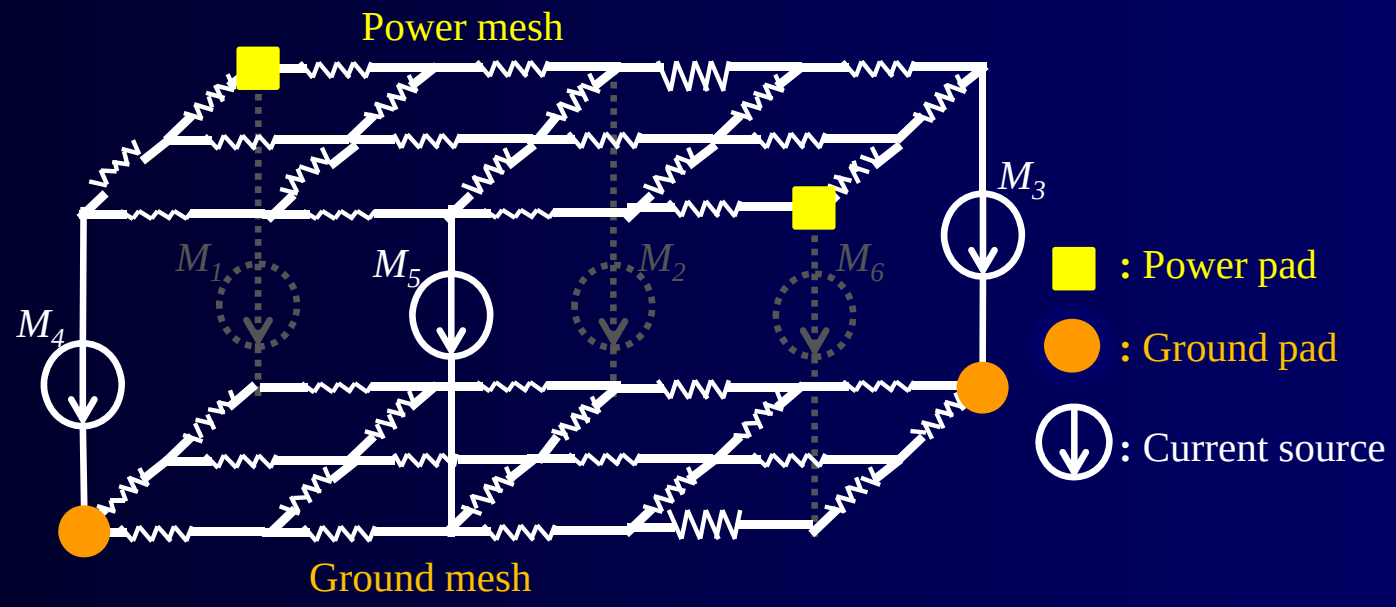
- Introduction
- Problem Formulation
- Efficient Wake-up Scheduling Algorithm
- Experimental Results
- Conclusions

Problem Formulation



Find the optimal order for turning on a given set of cores at *runtime*, such that the total wake-up time is minimized under the given IR drop bound.

Problem Formulation

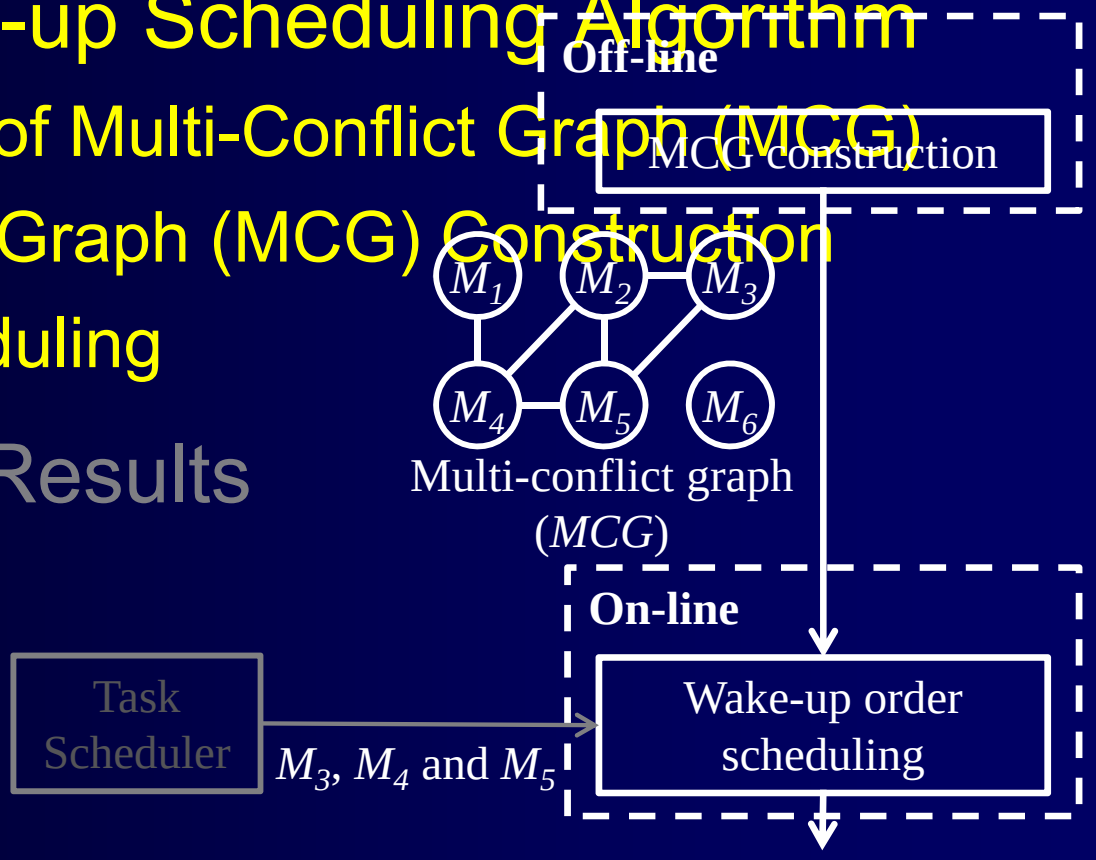


This model can be pessimistic.

Determining the optimal wake-up scheduling at runtime is still not an easy task.

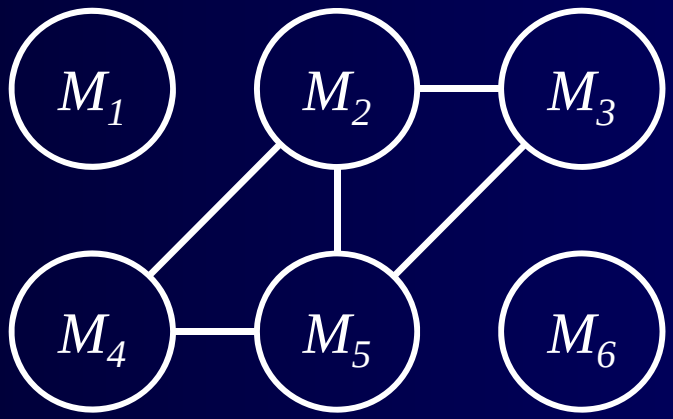
Outlines

- Introduction
- Problem Formulation
- Efficient Wake-up Scheduling Algorithm
  - The Concept of Multi-Conflict Graph (MCG)
  - Multi-Conflict Graph (MCG) Construction
  - On-line Scheduling
- Experimental Results
- Conclusions



Conflict Graph (CG)

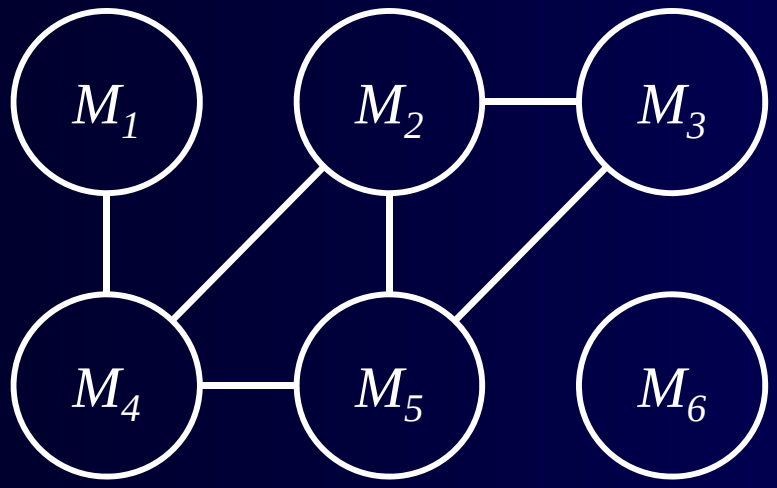
- A conflict graph (CG),
  - Each vertex represents core, and
  - each edge represents a conflict between the two vertices



- We construct a CG by adding an edge between a pair of vertices/cores that should not be turned on simultaneously.

Multi-Conflict Graph (MCG)

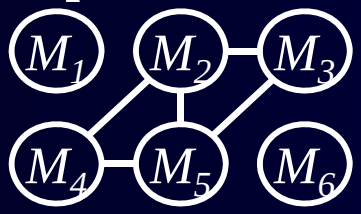
- A multi-conflict graph (MCG) is an undirected graph
  - the vertices form an independent set (*IS*) which is a set of vertices without any edge between them
    - the group of corresponding modules can be turned on simultaneously



M_2, M_4 and M_6 form an IS
→ M_2, M_4 and M_6 can be not **turned on simultaneously.**

MCG Construction

Step 1



conflict graph (CG)

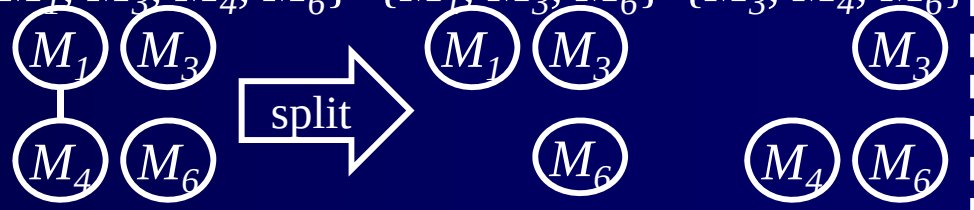
Step 2

ϕ

$\{M_1, M_2, M_6\}$
$\{M_1, M_5, M_6\}$
$\{M_1, M_3, M_4, M_6\}$

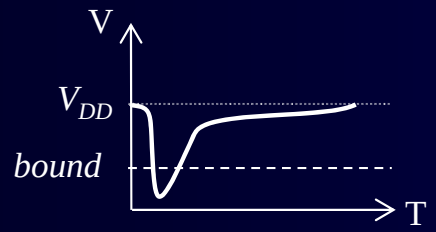
Step 4

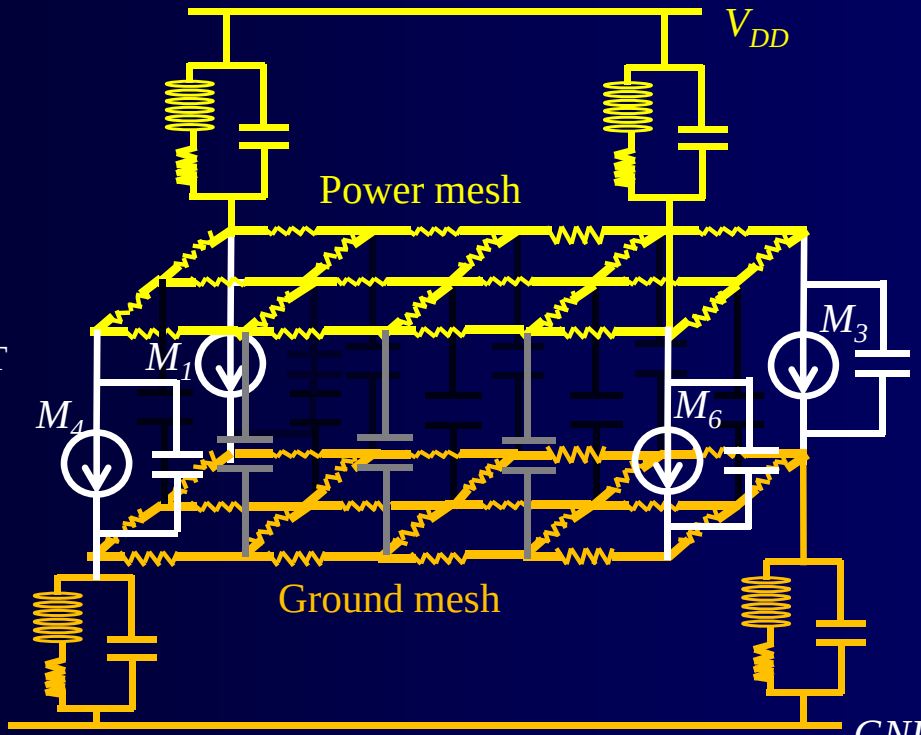
$\{M_1, M_3, M_4, M_6\}$ $\{M_1, M_3, M_6\}$ $\{M_3, M_4, M_6\}$



Step 3

$\{M_1, M_3, M_4, M_6\}$



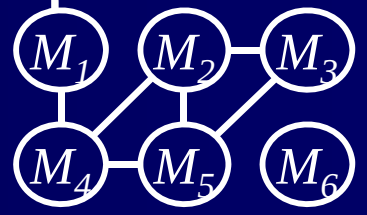


Step 5

ϕ

$\{M_1, M_3, M_6\}$
$\{M_3, M_4, M_6\}$

Step 6

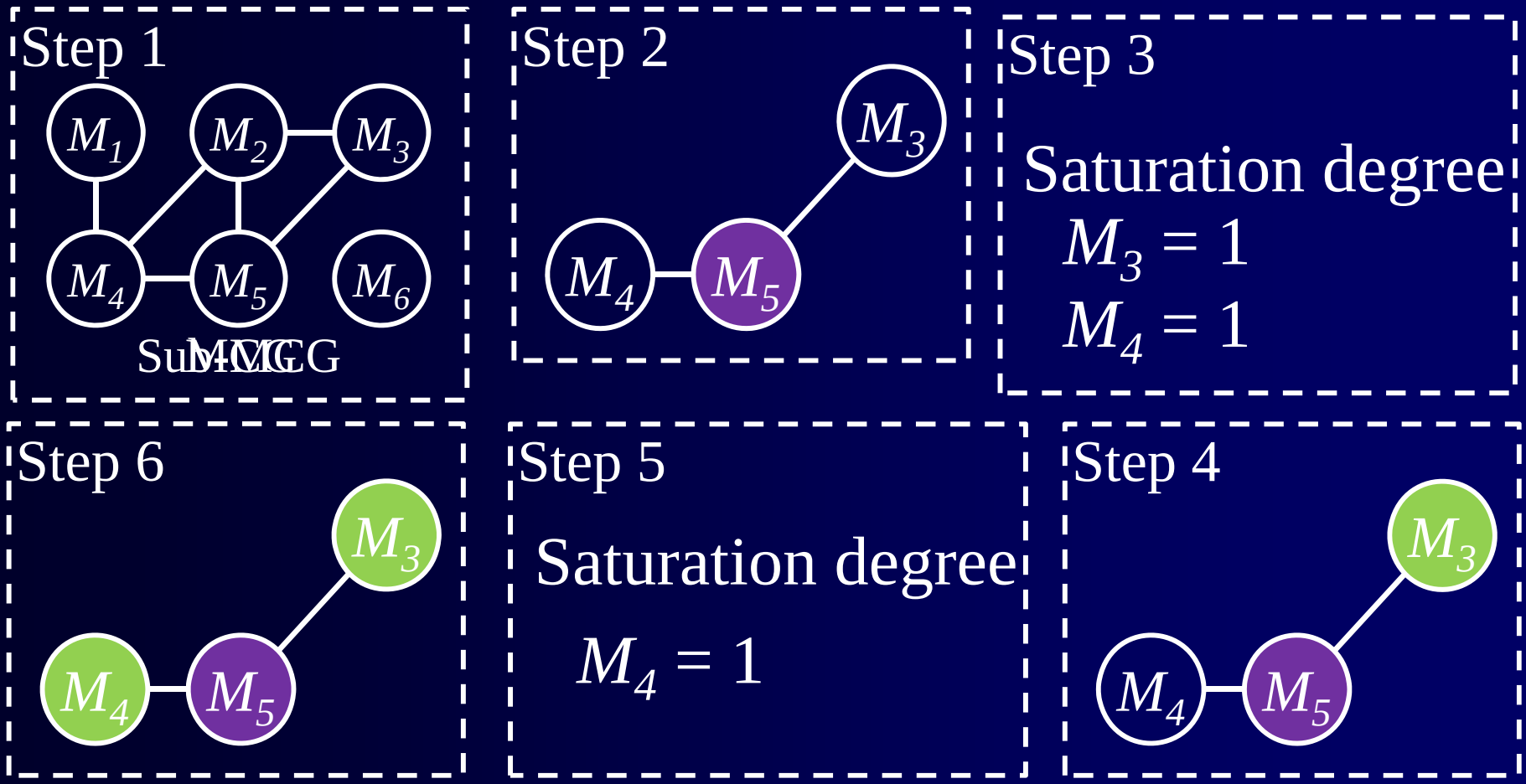


multi-conflict graph (MCG)

ϕ

empty

On-Line Scheduling Algorithm



**Saturation degree* of a vertex is the number of different colors which is connected

Outlines

- Introduction
- Problem Formulation
- Efficient Wake-up Scheduling Algorithm
- Experimental Results
- Conclusions

Experimental Setup

- We use the TSMC 90nm CMOS technology.
  - 55,794 gate count for each core
  - The size and the wake-up scheduling of sleep transistors for the core are designed appropriately.
  - The maximum wake-up surge current during the wake-up operation and the wake-up time for the core are 240mA and 18ns, respectively.
  - 16-core, 64-core, and 256-core, three different multi-core systems so that the topology of cores, the number and the locations of power/ground pads on the multi-core designs are properly designed.

Experimental Results

# of modules in multi-module design	# of modules to turn on (mc: Monte Carlo search, og: off-line grouping)											
	8				16				32			
	mc	ours	og	[4]	mc	ours	og	[4]	mc	ours	og	[4]
16	72	72	72	144	144	144	144	288	-	-	-	-
64	72	72	72	144	144	144	162	288	306	306	360	576
256	108	108	144	144	216	216	288	288	450	450	576	576
Average	1.00	1.00	1.11	1.78	1.00	1.00	1.15	1.78	1.00	1.00	1.23	1.58
# of modules in multi-module design	# of modules to turn on (mc: Monte Carlo search, og: off-line grouping)											
	64				128				256			
	mc	ours	og	[4]	mc	ours	og	[4]	mc	ours	og	[4]
16	-	-	-	-	-	-	-	-	-	-	-	-
64	720	720	864	1152	-	-	-	-	-	-	-	-
256	954	972	1134	1152	1998	2016	2268	2304	4140	4176	4608	4608
Average	1.00	1.01	1.19	1.40	1.00	1.01	1.14	1.15	1.00	1.01	1.11	1.11

Outlines

- Introduction
- Problem Formulation
- Efficient Wake-up Scheduling Algorithm
- Experimental Results
- Conclusions

Conclusions

- In this paper, we discuss the wake-up scheduling for multi-core systems.
- We propose
  - an off-line algorithm to characterize the multi-conflict graph (MCG) for the multi-core design, and
  - an on-line scheduling algorithm to efficiently decide the order of turning-on cores.
- For a 256-module system, the wake-up latency for our approach on average achieves 23.06% and 23.67% wake-up latency reduction compared with [16] and [4], respectively.

Thanks for your attention