Bus Architecture Optimization Method Based on System-Level Profiling

Masaharu Imai and Kyoko Ueda

Graduate School of Information Science and Technology
Osaka University, Japan
imai@ist.osaka-u.ac.jp

Outline

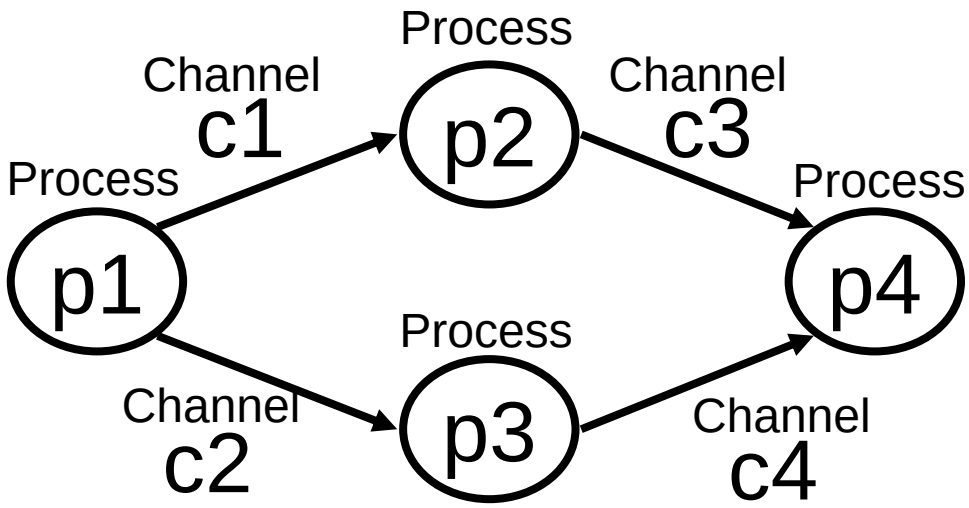
- Introduction
  - Requirements to Embedded System Design
  - Limitation of Conventional Estimation Method
- Proposed Solution
  - Performance Estimation
  - HW Area Estimation
  - Power Consumption Estimation
  - Architecture Exploration
- Experiments
  - Design Quality Estimation
  - Architecture Exploration
- Conclusion

Introduction

- Requirements to Embedded Systems
  - High Performance
  - Small HW Area
  - Small Power Consumption (Static and Dynamic)
- Requirements to Design Method
  - Short Time to Market
  - Design Optimization (Performance, Area, Power)
- Limitation of Conventional Design Method
  - Iteration of description and evaluation
  - RTL Evaluation: Accurate but Slow
  - System-Level Estimation: Quick but Less Accuracy

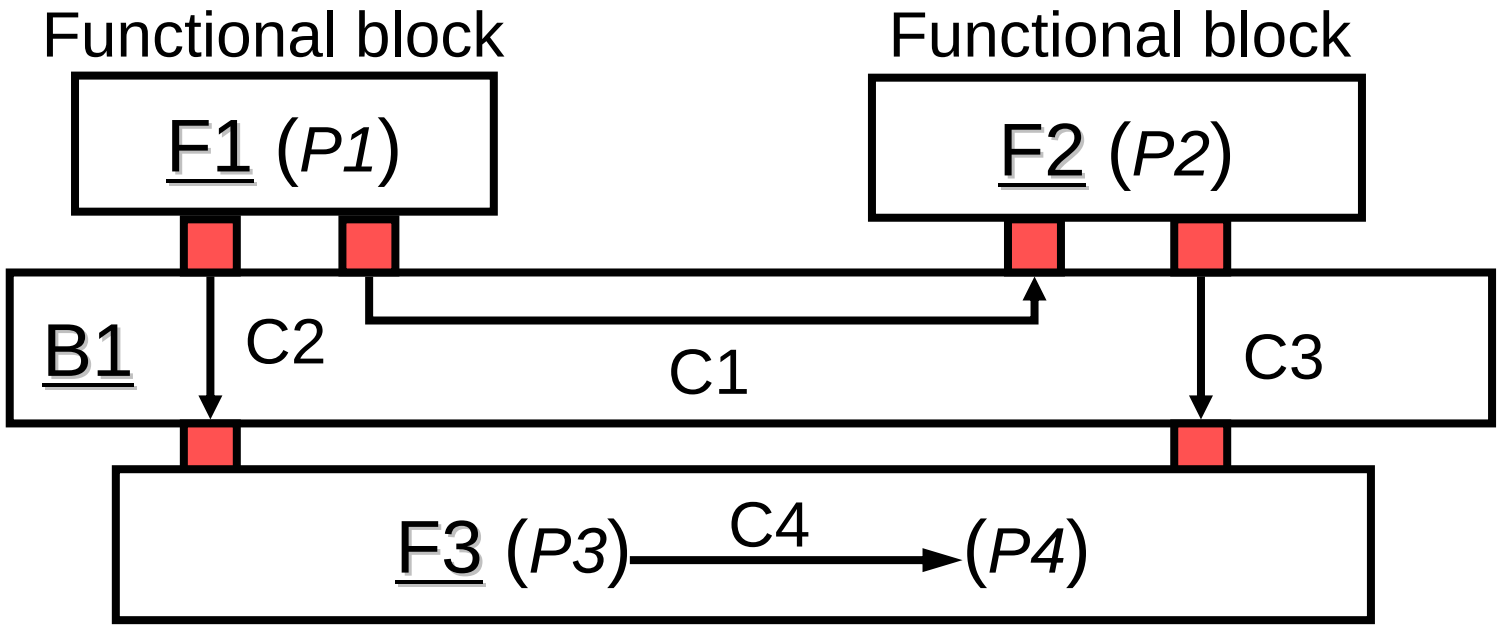
System-Level Model

- Components
  - Process: unit representing data processing
  - Channel: unit representing data transfer
- Assumption
  - Data processing and data transfer take 1 unit time each
- Example:



Architecture-Level Model

- Component
  - Functional block: data processing unit
  - Bus: data transfer unit
  - Buffer: data storage unit associated with bus
- Example



Outline

- Introduction
  - Requirements to Embedded System Design
  - Limitation of Conventional Estimation Method
- Proposed Solution
  - Performance Estimation
  - HW Area Estimation
  - Power Consumption Estimation
  - Architecture Exploration
- Experiments
  - Design Quality Estimation
  - Architecture Exploration
- Conclusion

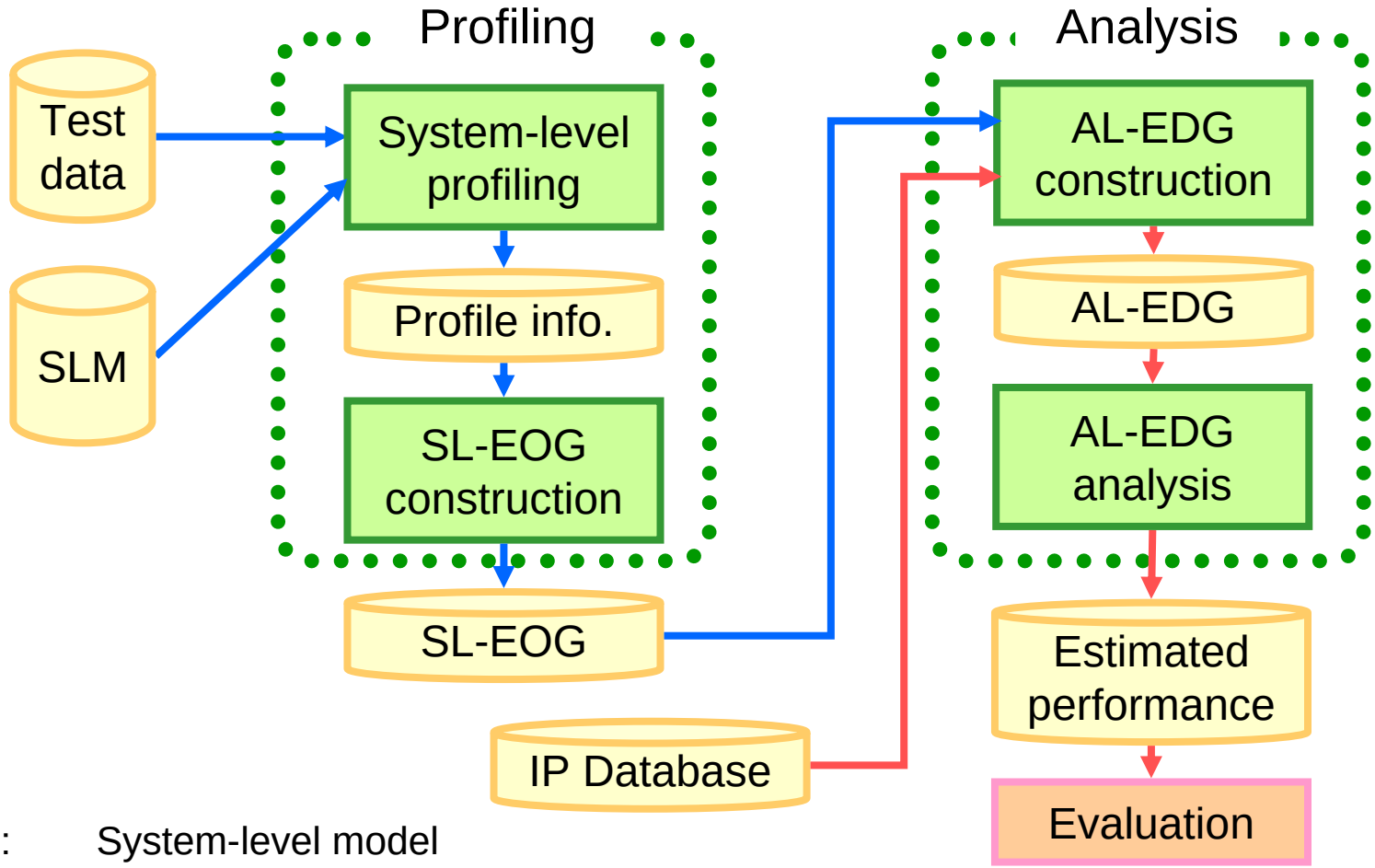
Proposed Solution

- IP-Based Design Method with IP database
  - IP database that contains HW area, performance, and power consumption information
- Quick and Accurate Performance Estimation using System-Level Profiling (system-level profiling before architecture-level performance estimation)
- Key Idea
  - Focusing on two kinds of dependencies
    - System-level dependency (execution order)
    - Architecture-level dependency
  - Separate computation and data transfer
  - Data flow amount between architecture-level components is the equivalent to that between system-level component

System-Level Profiling

- Target system written in SystemC
- What to profile
  - The time instance when data processing is completed
  - The time instance when data transfer is completed
  - Transferred data size
- How to profile
  1. Operate data processing or data transfer
  2. Record the data processing or data transfer completion time
  3. Wait for 1 unit time

Performance Estimation Flow



SLM: System-level model
ALM: Architecture-level model
SL-EOG: System-level execution order graph
AL-EDG: Architecture-level execution dependency graph

HW Area Estimation

- Estimated as the sum of hardware areas of
  - Functional Blocks
  - Buffers
  - Buses (wire area and control logic)
- Bus wire length estimation
  - Minimum Rectilinear Steiner Tree (MRST) model

Power Consumption Estimation (1)

- Estimated as the sum of power consumption of
  - Functional Blocks
  - Buffers
  - Buses (bus wire & logic)
- Functional Blocks
  - Static power consumption (leakage)
    - Product of static power registered in IP database and application execution time duration
  - Dynamic power consumption
    - Sum of product of dynamic power consumption and number of calls of the function

Power Consumption Estimation (2)

- Buffer
  - Estimated using
    - Transferred data size through channel
- Bus
  - Estimated using
    - Transferred data size through channel
    - Electric capacity of bus
    - Power consumption of bus controller

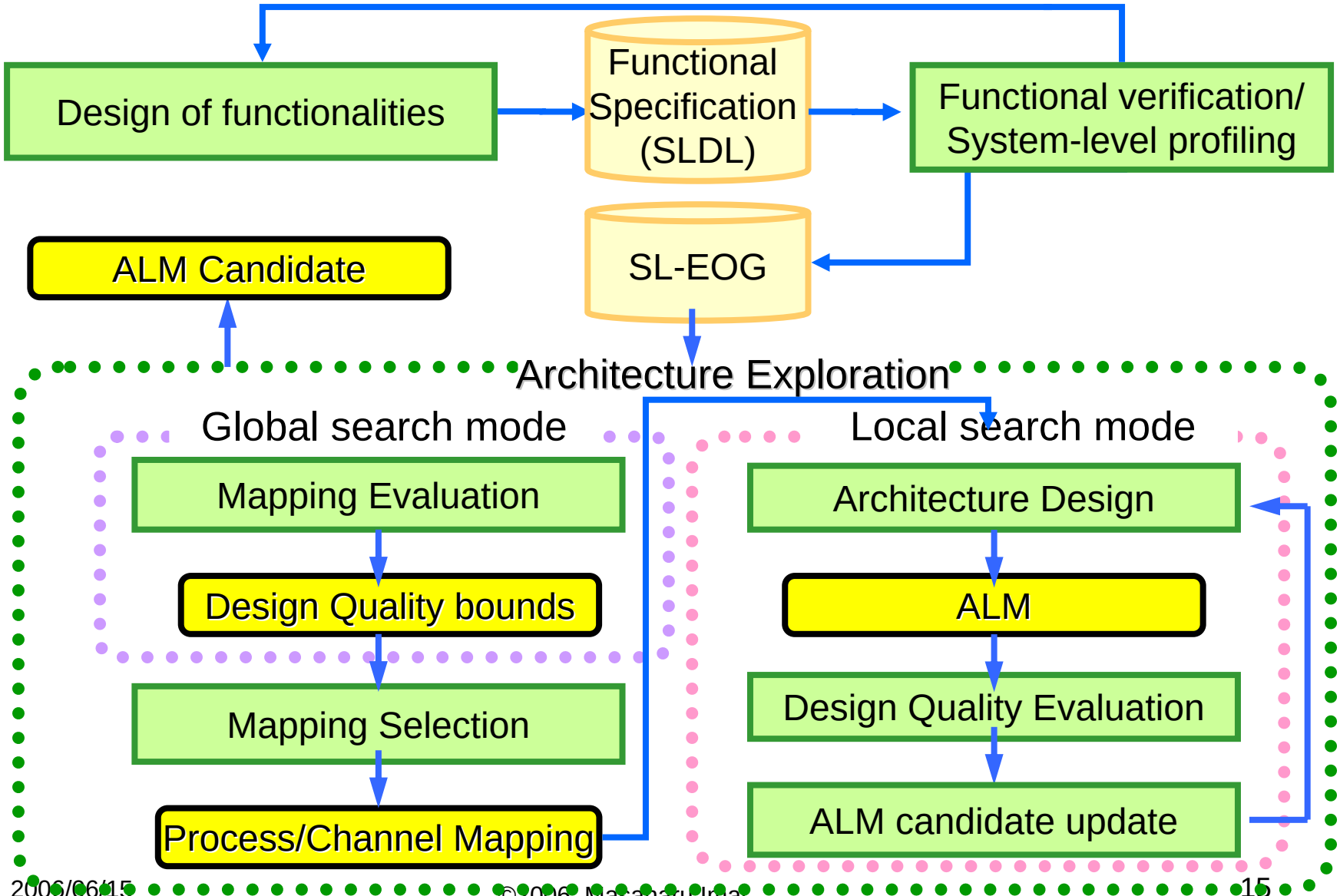
Architecture Exploration

- Objective
  - Explores architecture-level models and output those candidates that have a trade-off relation between HW area and application execution time
- Input to architecture exploration
  - System-Level Model
  - System-Level Execution Order Graph
  - HW area constraint. (can be set to infinity)
  - Application execution time constraint. (can be set to infinity)
  - Bus clock frequency candidates
  - Bus bit-width candidates
  - Number of buffer candidates

Global and Local Search Modes

- Global Search Mode
  - Part of the system-level components (processes) are fixed to map to given architecture-level components (functional blocks).
  - Estimate the upper- and lower-bound of performance, HW area, and power consumption for the mapping
- Local Search Mode
  - Choose a promising mapping
  - Perform branch-and-bound search within the mapping

Architecture Exploration Flow

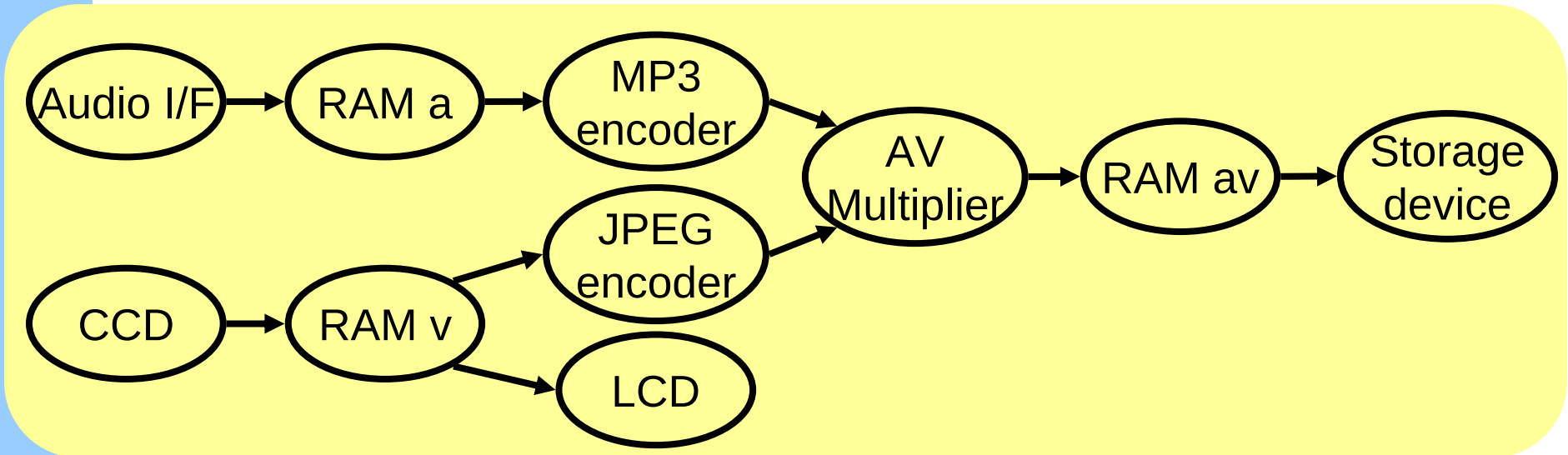


Outline

- Introduction
  - Requirements to Embedded System Design
  - Limitation of Conventional Estimation Method
- Proposed Solution
  - Performance Estimation
  - HW Area Estimation
  - Power Consumption Estimation
  - Architecture Exploration
- Experiments
  - Design Quality Estimation
  - Architecture Exploration
- Conclusion

Experiment (1): Performance Estimation

- Comparison between the estimation time of conventional method and proposed method
  - Conventional method: architecture-level simulations
  - Proposed method: profiling and graph analyses
- Target system
  - Audio/Video encoding system of a DVD

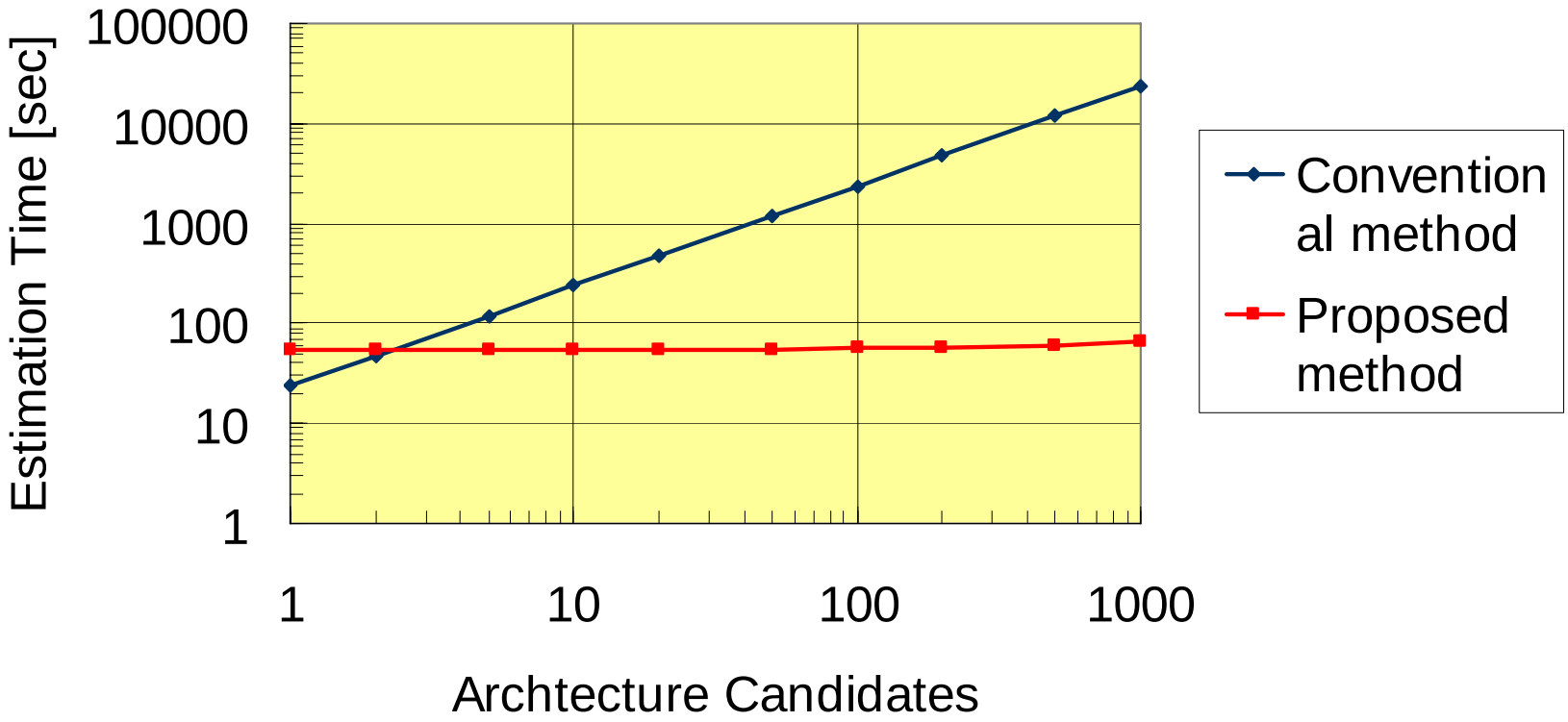


Comparison of Estimation Time (1)

Method		Estimation time [sec]		
		2,328 arch.	1 arch.	*N* arch.
Conventional method		5.53×10^4	23.8	$23.8 \times N$
Proposed method	Profiling & SL-EOG const.	54.8		
	AL-EDG const. & analysis	25.6	0.011	0.011 x N
	Total	80.4	54.8	54.8 + 0.011 x N

Comparison of Estimation Time (2)

- Result of estimation time for *N* architecture candidates



- Proposed method can evaluate many architecture candidates almost without additional time

Experiment(2): Architecture Exploration

- Comparison between the number of explored bus architectures and the time for exploration between an exhaustive search and proposed method
- Target system
  - Audio/video encoding system
- Inputs to architecture exploration
  - Bus data transfer candidates (1 MHz, 2 MHz)
  - Bus bit width candidates (16 bits, 32 bits)
  - Number of buffer candidates (1, 2)
  - HW area constraint (Infinity)
  - Application execution time constraint (Infinity)
  - HW area and execution frequency of IPs registered in a database
  - Execution cycle and IP functionality information (Mappable processes)

Comparison of exploration time

Method	# of nodes	# of leaves	# of est.	Comp. Time
Exhaustive	1.59×10^{13}	7.95×10^{12}	7.95×10^{12}	2,521 Years
Exhaustive w/pruning	1.01×10^{8}	7.77×10^{6}	1.87×10^{7}	52 Hours
Global mode	1.98×10^{6}	1.58×10^{6}	3.17×10^{6}	12 Hours
Local mode	44,521	156	29,245	30 Min.

- Experiments on a PC with Pentium 4 (2GHz, 512MB)
- Local mode denotes local search mode with fixed process and channel mapping (10 functional blocks and five buses)
- It is impossible to explore design space by exhaustive search even if pruning is done
- Simultaneous use of global and local search modes can greatly reduce the time needed for exploration

Outline

- Introduction
  - Requirements to Embedded System Design
  - Limitation of Conventional Estimation Method
- Proposed Solution
  - Performance Estimation
  - HW Area Estimation
  - Power Consumption Estimation
  - Architecture Exploration
- Experiments
  - Design Quality Estimation
  - Architecture Exploration
- Conclusion

Conclusion

- Quick and accurate architecture-level performance estimation method
  - Based on system-level profiling
  - Performance estimation for *N* architecture candidates can be performed *N*/2 times faster than conventional method (architecture-level simulation)
- Efficient architecture exploration method
  - Uses proposed performance estimation method
  - Explores architecture candidates using branch-and-bound method
  - Proposed Global Mode and Local Mode, when used sequentially, enables designers to explore a huge architecture design space very effectively

Thank you for your attention!