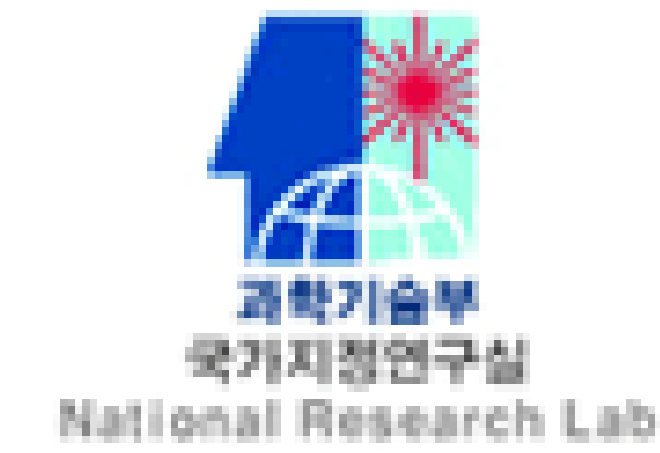




ePRO-MP: energy PРоfiler and Optimizer for Multi-Processor

Wonil Choi, Hyunhee Kim, Wook Song, Jiseok Song, and Jihong Kim
School of Computer Science and Engineering, Seoul National University, Seoul, Korea



Introduction

- Two main design goals for mobile embedded applications
 - High performance
 - Low energy consumption
- Requirements for development tools
 - Easy profiling of the performance and energy consumption of applications
 - Easy identifications of energy and performance bottlenecks in the application level

Motivations

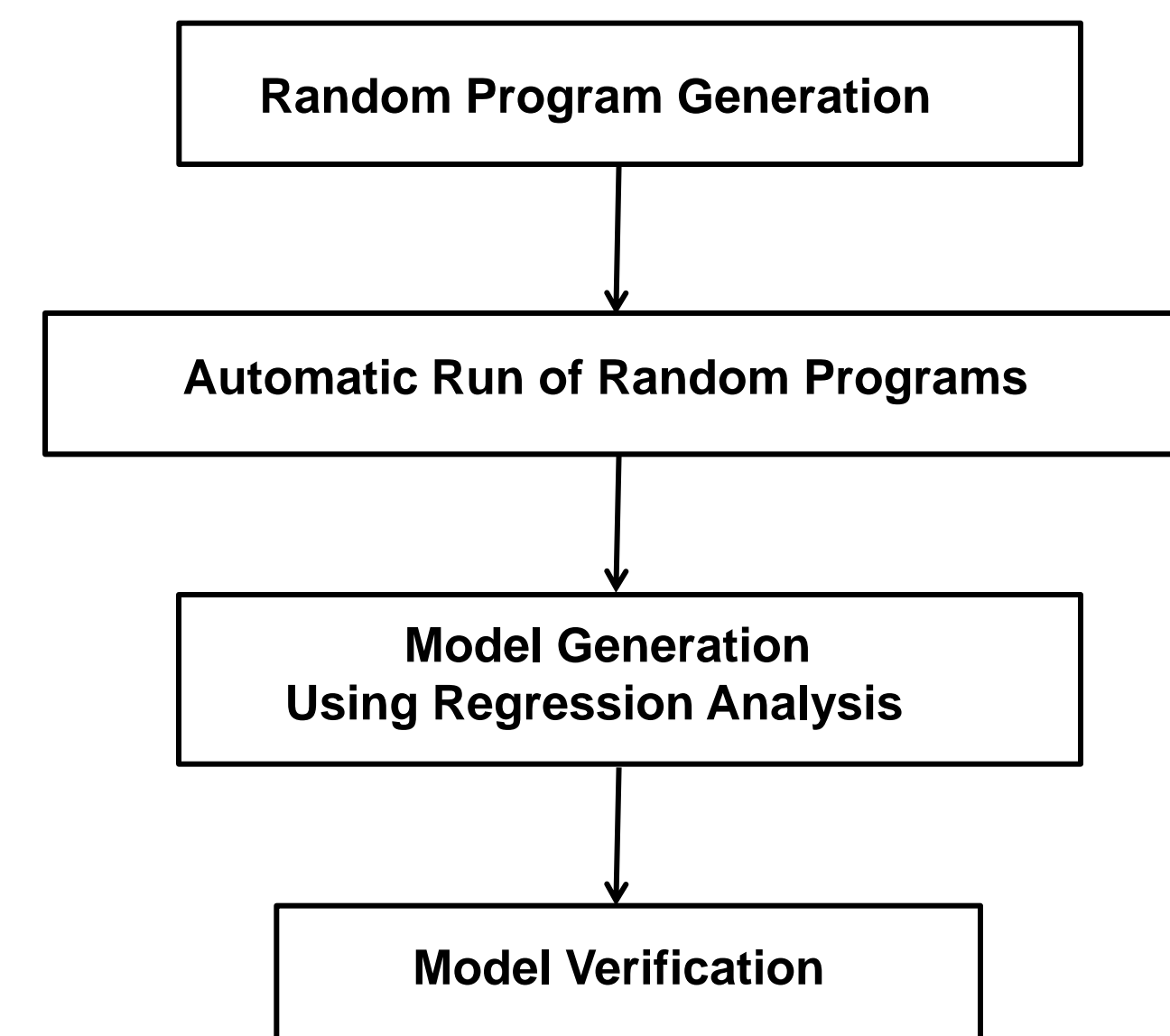
- Use of power measurement equipments for power profiling
 - Average embedded software developers are reluctant to use them
 - They are often too expensive to be widely used
- Absence of automatic optimization support
 - An efficient implementation often requires to explore a large design space
 - E.g., determining the optimal number of threads for co-running applications

Contributions

- Design of a system development tool, ePRO-MP
 - Profiles both performance and energy consumption of multi-threaded applications
 - Presents the analysis results in the program, thread, and function levels
- Energy profiling without requiring a power measurement equipment
 - Performance data and a regression-based energy model are used
- A limited support for automatic optimization functions
 - Two optimization examples are shown

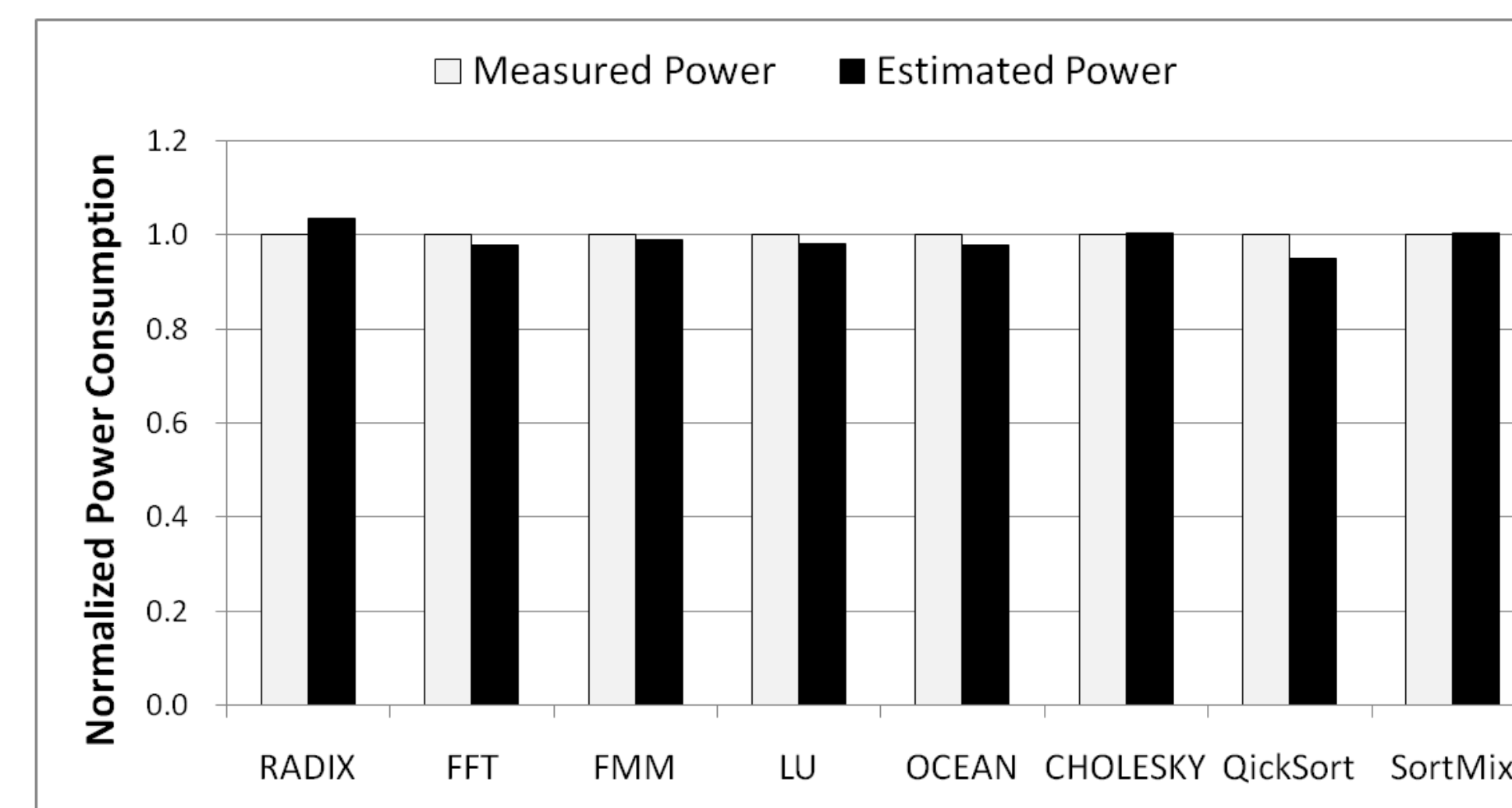
Model-Based Energy Profiler

- Power model development procedure



Procedure for Deriving Energy Model

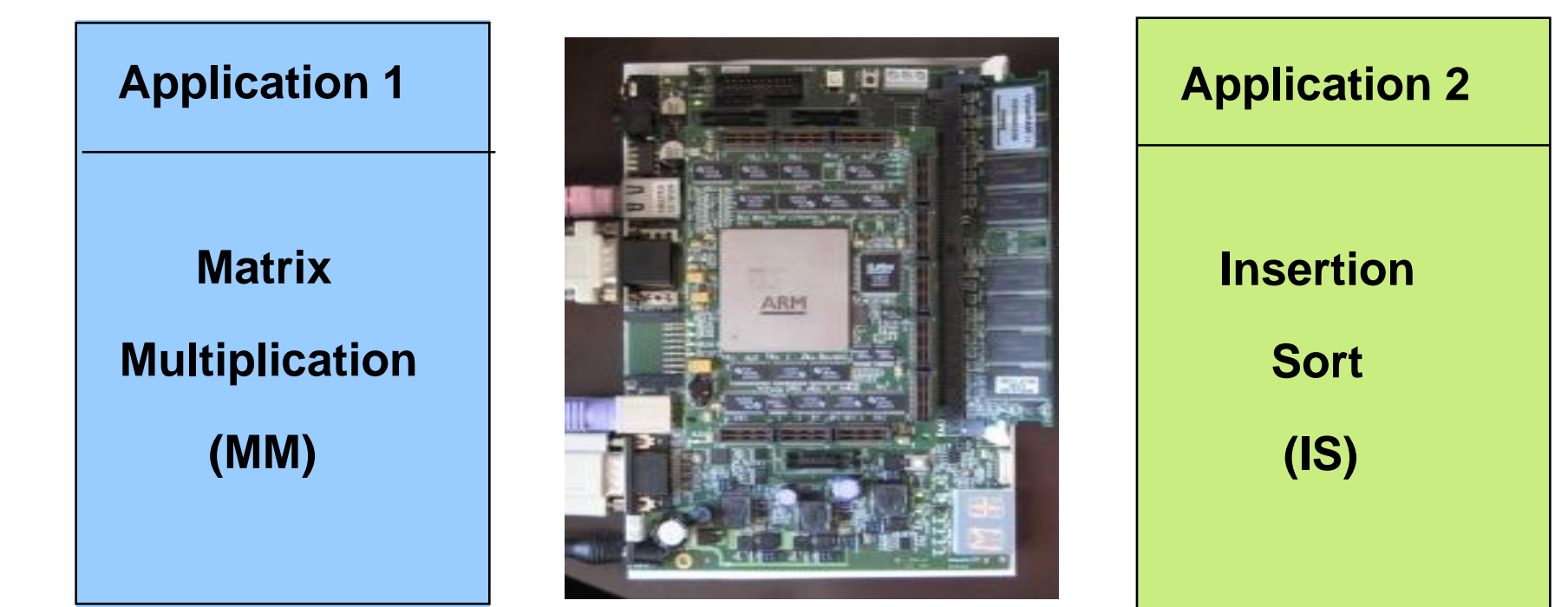
- Power model for ARM11 MPCore
 - Constraint: no per-core level power measurement
 - $Power = A \times (Instr / time) + B \times (DL1Access / time) + C \times (L2Access / time) + D \times (DataDep / time) + E \times (CohTrans / time) + F_{const}$
 - Average error 3% for SPLASH-2 benchmark programs



Comparison of Measured Power and Estimated Power

Profile-Based Automatic Optimizer

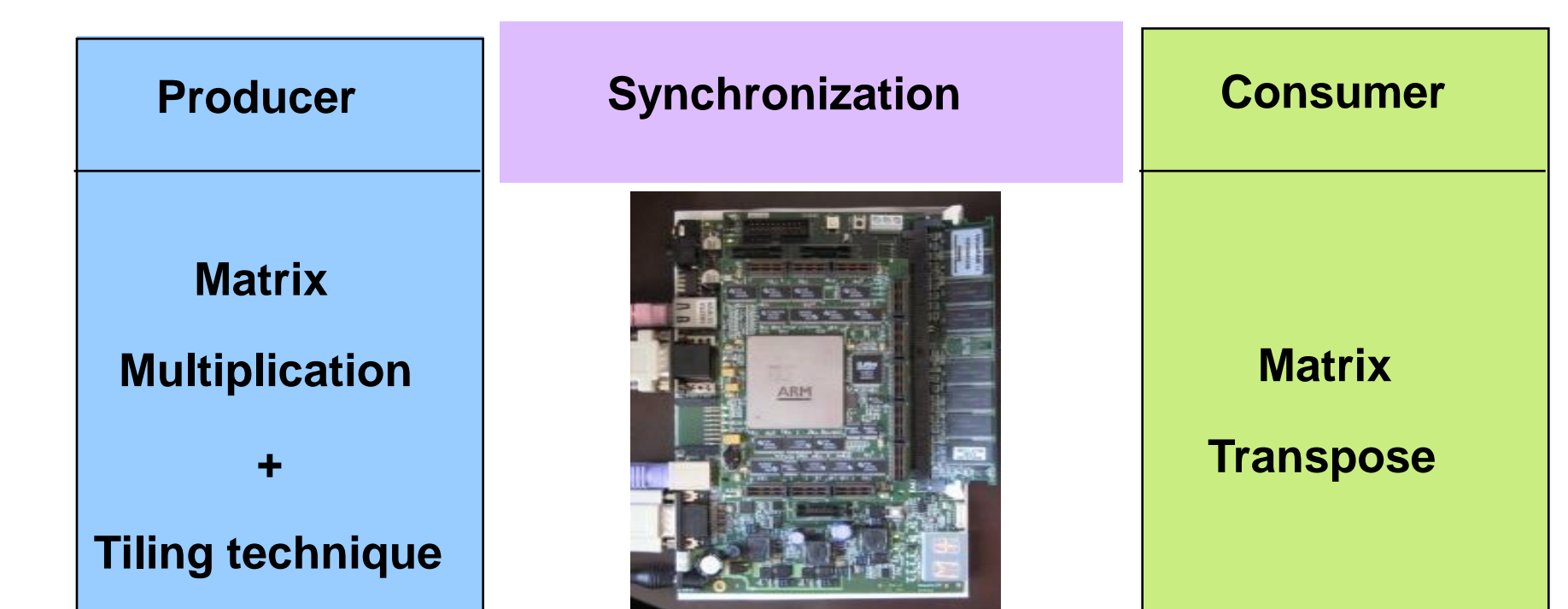
Optimization Example 1: Co-Running Applications Optimization



How many threads? 4 cores How many threads?

- Results
 - Three threads for MM(A) and one thread for IS(B)
 - Performance improvement 6.1% and energy improvement 4.1%

Optimization Example 2: Producer-Consumer Application Optimization



How many threads? 4 cores How many threads?
How much tile size? 32 KB \$L1

- Results
 - Four threads for MM, one thread for MT, and the tile size close to L1 cache size
 - Performance improvement 60.5% and energy improvement 43.3%

Optimized target software

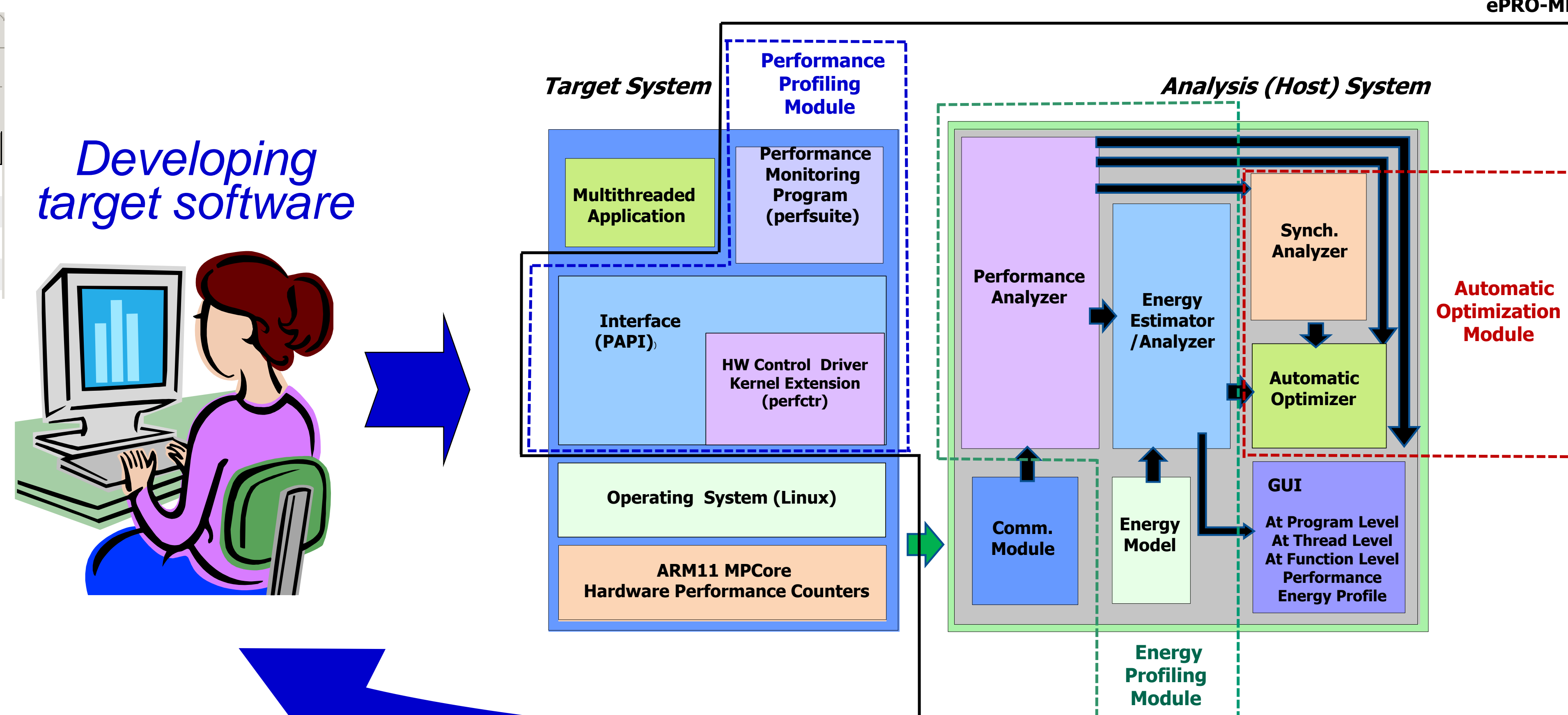
Thread ID	Cycle	IPC	DL1MissRatio(%)	SPI	Energy(mJ)
thread1	228,772,422,056	0.227	3.850	3.390	211,664.800
thread2	229,130,279,563	0.228	3.850	3.399	211,646.840
thread3	228,897,090,718	0.229	3.850	3.365	211,949.190
thread4	423,198,349,513	0.497	0.940	1.013	472,384.910

Optimization Result of MM-IS Applications



Optimization Result of MM-MT Application

Executing target software & analyzing performance and energy



Developing target software

Detecting bottlenecks and hot-spots

Thread ID	Cycle	IPC	DL1MissRatio(%)	SPI	Energy(mJ)
thread1	181,402,407,285	0.215	3.870	3.678	163,708.420
thread2	181,258,645,863	0.216	3.880	3.498	166,260.150
thread3	180,923,039,271	0.216	3.870	3.653	163,642.690
thread4	190,283,707,395	0.205	3.890	3.505	174,946.740
thread5	112,002,049,989	0.470	0.990	1.159	121,288.450
thread6	111,668,147,760	0.471	0.980	1.134	121,507.060
thread7	112,392,556,940	0.469	0.990	1.131	122,356.300
thread8	112,420,600,848	0.468	0.990	1.162	121,818.490

Profiling Result of (4, 4) Thread Allocation Shown in Thread Level