An OpenCL Implementation Supporting Task Parallel Execution on Embedded Many-core Architecture

Takuji Hieda, Naoki Nishiyama, Ittetsu Taniguchi, and Hiroyuki Tomiyama
College of Science and Engineering, Ritsumeikan University, Japan
{takuji.hieda, naoki.nishiyama}@tomiyama-lab.org, {i-tanigu, ht}@fc.ritsumei.ac.jp

Koji Inoue
Department of Advanced Information Technology, Kyushu University, Japan
inoue@ait.kyushu-u.ac.jp

ABSTRACT
Many-core architectures have attracted attention for embedded systems, not only high performance computing. A typical many-core architecture has tens or hundreds of processing elements (PEs). Many-core architectures can achieve low power consumption and highly computation performance by optimizing task allocation on the PEs. GPU architectures have become used for general-purpose computing in recent days. However, a typical GPU consists of a number of PE groups and all PEs in a group must execute the same task. This constraint is unsuitable for embedded systems, so parallel programming environments which can customize task allocation are required.

We have developed an OpenCL environment which supports both data- and task-parallel programming models. The implemented OpenCL environment consists of OpenCL runtime library and many-core simulation library. The OpenCL runtime library implements APIs to control program execution. The many-core simulation library simulates many-core architecture. With the simulation library, each core is simulated by a thread on the host machine. Using the OpenCL environment, programmer can develop applications without target many-core hardware.
An OpenCL Implementation Supporting Task Parallel Execution on Embedded Many-core Architecture

Takuji Hieda†, Naoki Nishiyama†, Ittetsu Taniguchi†, Hiroyuki Tomiyama†, and Koji Inoue‡

† College of Science and Engineering, Ritsumeikan University, Japan
‡ Department of Informatics, Kyushu University, Japan

Overview of the OpenCL Implementation on Virtual Environment

- Simulates many-core architecture on the host processor
- Executes OpenCL program
- Supports task parallel execution described in kernel functions
- System designer can decide task mapping on the core array

OpenCL Program

```c
main() {
    ... 
    clEnqueueTask(...);
    ... 
    clEnqueueTask(...) {
        ... 
        open("/dev/smyle"");
        ... 
    }
    ...
}
```

Runtime Library

- Core Simulation Thread
- Core Simulation Thread
- ... 
- Many-core Simulation Library

Many-core Architecture Supporting Task Parallel Execution

- A general-purpose embedded many-core architecture to achieve task and data parallel execution
  - Allocates a number of PEs for each task
  - The allocated tasks work in parallel

Highly parallelizable task
- Use all PEs for the task

Cooperative tasks
- Map tasks on each PE

Low workload tasks
- Power off the unused PEs

OpenCL on GPGPU Architecture

- A task can be executed only in the units of clusters which has a number of processing elements (PEs)
  - There may exist unused PEs in the clusters
  - The maximum number of tasks is limited by the number of clusters in the GPU

A mechanism of highly flexible task parallel execution is required for embedded many-core systems