Shuqi Wu 18301412 CS7NS1 Data Science

Nile: A Programmable Monitoring Coprocessor

1.1 **HPCs** are a set of registers built into processors used to count and monitor related activities with flaws on fixed-sized-blocks for data allocation.

1.2 **Nile** is a computable hardware monitor with a hardware detector architecture and an **RISC-V processor.**

1.3 The limitation of collecting events data caused by **HPCs** may miss the needful information for optimizing computer instructions.

1.4 **SimOS** has advantages in optimizing and analyzing in run-time.

1.5 **MUs** are set to process the commit log and are capable of communicating with each other.

2.1 **Commit log** displays the process of the execution of instruction based on RISC-V processor.

2.2 The received **commit log** is transmitted to **MUs**.

2.3 The **activation signal** was trigged after Nile gets its programmed threshold.

2.4 In the case of **Stack Buffer Overflow Protection**, MUs protect one piece of memory which then be reported to Nile.

2.5 **Stack buffer overflow** can be overcome by the use of **Shadow Stock** implemented by Nile.

3.1 **Commit log** data collection requires **parallel computing** to improve its performance.

3.2 Scalable computing may help the **dequeue and enqueue** allocation of activation pockets.

3.3 **Configuration** is able to control the unit to compute actions based on different values.

3.4 The implementations of **MUs** and communications between MUs order to overcome Stack Buffer overflow may contain Scalable computing.

3.5 **Nile** is used for security, performance improvement and utilization of power, etc.

The Quest for Energy-Efficient I$ Design in Ultra-Low-power Clustered Many-Cores

1.1 **Near-Threshold computing** was designed to improve the performance thorough parallel computing of supply voltage in a low condition.

1.2 The restriction of NT in **PVT** motivated the design of **Ultra-Low-Power.**

1.3 Based on the advantage of **SCMs** (with respect to low-voltage, good management of energy) and **memory sharing**, multiple architectures were evaluated in this report.

1.4 The increment the space of **MP** configuration was caused by the augment of **TCCC**.

2.1 Private I$ generates the problem of **data redundancy** and inefficiency of energy and space.

2.2 In order to control the access of **CB**, round-robin policy has been used.

2.3 The address of misses issued by master cache controller can be placed in a **CAM**.

2.4 Different **misses** generated within the same cache line will be given different unique identities for storage in a **CAM**.

2.5 **L0** is used for minimizing the stress on **CBs** which can keep cache lines.

3.1 Those architectures based on NT to overcome **voltage scaling chock point** can be achieved by scalable computing (SC).

3.2 Scalable computing can be used to solve the restriction caused by **lock-step execution** along with **GP-GPU**.

3.3 S**upply voltage** can be changed to **NT** with the operation of **SOC** and clusters along with **FLLs**.

3.4 Managing different read and write ports for different bands in MPI$ needs SC.

3.5 **PULP** is a scalable architecture which can work with multi-clusters.