



# Asynchronous Arbiter for Micro-threaded chip Multiprocessors

Nabil Hasasneh & Ian M. Bell  
The University of Hull  
Department of Engineering

Chris R. Jesshope  
The University of Amsterdam  
Department of Computer Science

## Abstract

This poster presents a scalable and partitionable asynchronous bus arbiter for use with microthreaded chip multiprocessors (CMP). The arbiter exploits the advantage of a concurrency control instruction Brk provided by the micro-threaded microprocessor model to set the priority processor and move the circulated arbitration token at the most likely processor to issue the create instruction. This mechanism provides latency hiding during token circulation by decoupling the micro threaded processor from the ring's timing, and together with its asynchronous operation and communication ensure rapid response to thread creation requests. The arbiter can be extended easily to support large number of processors on-chip and can be used for a variety of chip multiprocessor arbitration requirements. This poster describes arbiter organization and its state machine. Arbiter partitioning and scalability are also introduced. The arbiter has been simulated using VHDL with arbitrary communication delays and clock differences in the processor array.

## 1. Micro-threaded Chip Multiprocessor Architecture Model

Figure 1 shows the micro-threaded CMP architecture model [1,2,3,4]. This model exploits the advantages of a GALS design approach by using a set of global buses all of which use fully asynchronous communication. One of these buses is the global create bus, for which each micro-threaded processor uses in order to create a new family of micro-threads. The concurrency controls used in this model provide a flexible and efficient mechanism for thread creation, context switching and synchronization.

### 1.1 Global Create Bus

- Each processor contends to access the global create bus to create a new family of micro-threads.
- The break (Brk) concurrency control instruction forces a break from a loop executed concurrently.
- The asynchronous arbiter is used to handle processors' contention and to provide fair and starvation-free bus access.

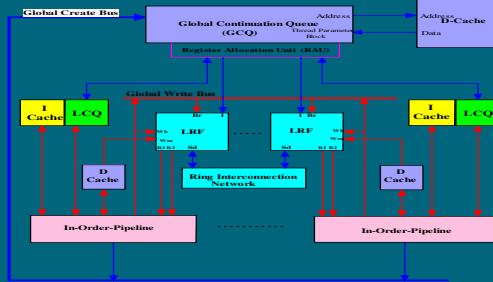
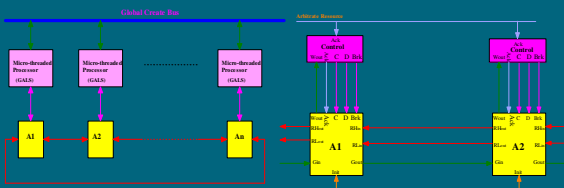


Figure 1. Micro-threaded CMP architecture Model

## 2. Arbiter Organization

Figures below shows the novel arbiter organization. Each processor has its own local control and a separate arbiter module in order to allow processor partitioning. Our arbiter exploits the concurrency control instruction provided by the micro-threaded microprocessor model to set a priority policy based on the processor that has succeed in executing the Brk instruction. This mechanism provides latency hiding of the token circulation time.



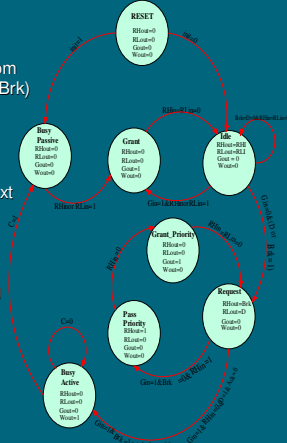
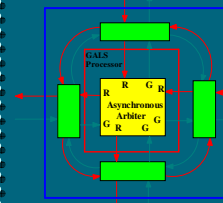
## 3. Arbiter State Machine

- Asynchronous State machine
- Priority policy based on prediction from the concurrency control instruction (Brk)
- Asynchronous Communication
- Delay insensitive model.

## 4. Arbiter Partitioning

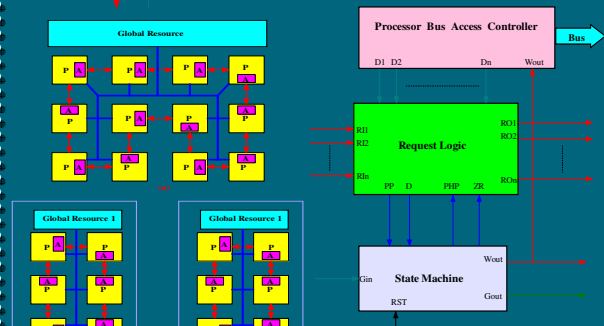
Each arbiter module is linked to the next one in a ring Arrangement and the processors are arranged in a Grid layout as shown below.

- Configurable arbitration signal routing
- Point-to-Point connection
- Less Delay
- Local Control



## 5. Arbiter Scalability

- N-levels of priority
- Processors Array
- Routing Architecture



## 6. Simulation Results

- Delay Insensitive Model
- Simulated using VHDL Language.
- Processor's having different clock phases and different frequencies.

## 7. CONCLUSION

The arbiter provides a very simple arbitration design methodology, where each module has a few wires connecting a next one and the last linked to the first module in a circular fashion. A delay-insensitive methodologies with unbounded wire and gate delays are considered in the arbiter simulation procedures. The arbiter has the advantages of GALS communication design and include the following features:

- Point-to-point communication between adjacent modules.
- Partitionable Configuration
- Scalable Solution
- Insensitive Delay model
- Local Control signal for each arbiter module.
- Simple Mechanism
- Ring Configuration to arbiter module
- Fair Priority Mechanism.

## 8. REFERENCES

- C.R.Jesshope, Multi-threaded Microprocessors Evolution or Revolution, ACSAC 2003.
- C.R.Jesshope, A Concurrency Model for Instruction-Level Distributed Computing, NPC04, 2004.
- C.R.Jesshope, Scalable Instruction-Level Parallelism SAMOS, July, 2004.
- C.R.Jesshope, Micro-grids the exploitation of massive on-chip concurrency, Proc. HPC, June, 2004.

Contact  
NABIL M. HASASNEH  
DEPARTMENT OF ELECTRONIC ENGINEERING  
THE UNIVERSITY OF HULL  
HUB 7BX  
Email: N.Hasasna@eng.hull.ac.uk