Gao X, Chen YJ, Wang HD *et al.* System architecture of Godson-3 multi-core processors. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 25(2): 181–191 Mar. 2010

System Architecture of Godson-3 Multi-Core Processors

Xiang Gao[1] (高 翔), Yun-Ji Chen[1] (陈云霁), Huan-Dong Wang[1,2] (王焕东), Dan Tang[1,2] (唐 丹) and Wei-Wu Hu[1] (胡伟武)

[1] *Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China*

[2] *Graduate University of Chinese Academy of Sciences, Beijing 100049, China*

E-mail: {gaoxiang, cyj, wanghuandong, tangdan, hww}@ict.ac.cn

Received March 30, 2009; revised October 9, 2009.

Abstract Godson-3 is the latest generation of Godson microprocessor family. It takes a scalable multi-core architecture with hardware support for accelerating applications including X86 emulation and signal processing. This paper introduces the system architecture of Godson-3 from various aspects including system scalability, organization of memory hierarchy, network-on-chip, inter-chip connection and I/O subsystem.

Keywords multi-core processor, scalable interconnection, cache coherent non-uniform memory access/ non-uniform cache access (CC-NUMA/NUCA), MESH, crossbar, cache coherence, reliability, availability and serviceability (RAS)

1 Introduction

Godson-3 is the multi-core generation of Godson microprocessor family[1-2]. It targets to the low cost servers and high-performance computers. The four-core Godson-3A, tapping out in the late 2008, is the first multi-core microprocessor design in China.

Philosophy and objectives guiding the design of Godson-3 are focused on the following points:

System Scalability. With the scaling of IC integration density, more processor cores could be accommodated on a single chip[3-6]. The interconnections of the Godson-3 system must be designed in a scalable manner so as to manage design complexity and to provide a uniform topology to system developers. The Godson-3 system was designed from the ground up as a multiprocessing system capable of scaling to both small and large processor counts with little bandwidth, latency, or cost cliffs. The scalable directory-based cache coherence protocol is implemented to support cache coherence among a scalable multi-core/multi-chip system.

Broad Applicability. As a multi-core processor with high-performance, low cost, low-power, Godson-3 targets to high throughput computing in data center servers, high density computing in digital signal processing, and high performance computing in PetaFlops high performance computers. Hardware and software supports are provided in Godson-3 to satisfy the demand for broad applicability. It provides 200+ instructions in addition to the MIPS64 instructions to speedup X86 emulation. To speedup DSP applications, its DMA engine supports automatic data prefetching and matrix transposing and its L2 cache can be configured to be on-chip SRAM.

Low Power Consumption. Systems based on Godson-3 processors require green solutions for high performance applications. Therefore low power consumption is an important design objective of Godson-3. The power consumption of the first quad-core Godson-3 chip fabricated in a 65 nm CMOS process is targeted at dissipating only 10∼15 W at 1.0 GHz.

Balanced Design. While increasing transistor budgets can accommodate large numbers of processing cores on a single chip, the number of signal I/O pins does not scale with the number of transistors. The communication between the chip and other parts of system is both critical for performance and expensive to scale. Multiple cores will contend for on-chip memory and I/O bandwidth. Performance of memory and I/O subsystem is crucial to the multi-core based system. In Godson-3, a great proportion of system resource, including on-chip transistors, I/O pins and off-DRAM modules, have been devoted to the memory subsystem so as to make high-performance available.

Pico-Architectural Design and Optimization. In a nanometer technology, architectural design and

Regular Paper

Supported by the National High Technology Development 863 Program of China under Grant No. 2008AA010901, the National Natural Science Foundation of China under Grant Nos. 60736012 and 60673146, and the National Basic Research 973 Program of China under Grant No.2005CB321601.

optimization have to take into account wire delay and place and route information from the very beginning. We call the architecture design and optimization related to physical implementation with the name of pico-architecture design[4]. In the development stage of Godson-3, the architecture design team and physical design team had collaborated closely for efficient coordination, which made the signoff requirements for clock rate and area cost achievable.

Pre-Silicon Performance Evaluation. The design space of Godson-3 is so large that a minor perturbation of micro architecture could significantly affect the overall performance. In order to accurately locate where performance bottlenecks lie, we extract key code segments from various realistic applications and examine their performance at RTL-level in full system environments. The selected applications cover a broad range of area such as server application, high-performance computing, signal processing and desktop application. The methodology of full system simulation using realistic applications helps us locate and fix many error-free but performance-degrading defects.

The remainder of this paper is organized as follows. Section 2 introduces the system architecture of Godson-3. Section 3 briefly depicts the four-core Godson-3A, which is the first version of Godson-3 multi-core processor. A more detailed discussion of the Godson-3 chip may be found in [7]. Sections 4, 5, 6 describe the memory subsystem, interconnection subsystem and I/O subsystem respectively. Section 7 introduces features for RAS (reliability, availability, and serviceability) and debugging. Section 8 presents the preliminary performance results. Section 9 gives the concluding remarks and our future work.

2 Godson-3 System Structure

As shown in Fig.1, the Godson-3 system is a highly modular and hierarchical shared-memory multiprocessing design based on multi-core processors. The design employs a scalable 2D MESH as interconnection topology. The basic building block of the system is the node, which has four inter-node ports for communication with adjacent nodes in the east, south, west and north. In a typical configuration, each node contains up-to-4 high performance processor cores, 4 MB of banked L2 cache, a crossbar based router, its local memory and I/O controllers.

The node is also the basic building module at chip-level. A multi-core chip of Godson-3 may include one node for four cores, two nodes for eight cores and so on. Due to the area constraint of CMOS fabrication process, however, the count of nodes that can be integrated into a single chip is limited. For example, the Godson-3 chip can contain a maximum of 2 nodes (eight cores) in 65 nm process. Systems going beyond the bounded number of nodes should be formed by interconnecting multiple chips on board. Therefore, the system specification defines only the abstract layer of the inter-node connection protocol, leaving the implementation to practice. The interconnection between nodes may be implemented through either network-on-chip or I/O-link-off-chip. This mechanism provides much flexibility to the development of large-scale systems. The current system implementation supports as much as 16 nodes, for a maximum configuration of 64 processor cores and 64 MB L2 cache totally. Even larger system can be easily and smoothly formed based on the structure.

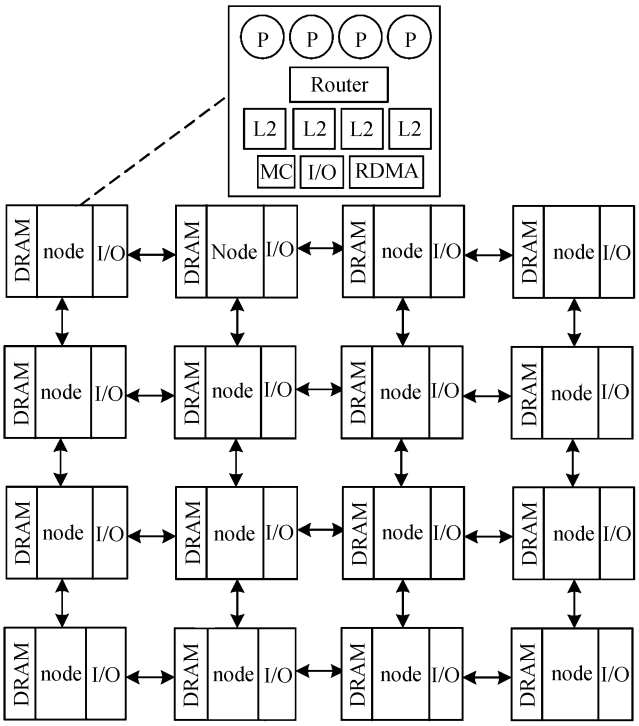


Fig.1. Large-scale Godson-3 system consists of a mesh of interconnected nodes, each with four processor cores, four L2 cache modules, its local memory and I/O. In this 16-node system of 64 cores, any processor core can access any other node's L2 cache and memory with no more than six hops.

Technically, a single-node system of Godson-3 is a traditional SMP (Symmetric MultiProcessing). A multi-node system of Godson-3 is a non-uniform memory and cache access (NUMA, NUCA) design based on distributed shared memory architecture (DSM). The main memory and L2 cache are distributed across nodes to provide scalable memory bandwidth and cache capacity. The single-address-space system architecture provides global addressability of all memory, all L2

caches and the I/O subsystem. I/O devices can DMA to and from all memory in the system, not just their local memory. A scalable directory-based cache protocol is employed to maintain the global data coherence over the whole system.

In a classic CC-NUMA system[8], however, communications between local processors and remote memory incur long latency penalty, forcing software to take these delays into account. To address the issue of remote access latency, the Godson-3 system design emphasizes both low latency and high bandwidth for inter-node communication. The network-on-chip is designed to operate at the same frequency as the processor core. The width of the fully-pipelined datapath is as high as 128 bit. Inter-chip connections are based on a high-speed I/O link, which operates at a nearly equal frequency of the processor core. For communications between nodes on the same chip, the access latency is only a fewer cycles higher than local access. For inter-chip communications, the latency is less than 100 cycles higher. Furthermore, Godson-3 integrates a remote RDMA engine, which serves as a software-initiated prefetcher independently of processor cores. The remote DMA can get bulks of data from remote memory and put them into local L2 caches, or vice versa. This facility can significantly reduce the waiting cycles of processor cores due to remote access.

3 Godson-3A Chip Overview

Godson-3 adopts the SMOC (Scalable Mesh of Crossbar) on-chip network topology. Each node in the mesh includes an 8×8 AXI crossbar which connects four processor cores, four shared L2-cache banks and four adjacent nodes in the east (E), south (S), west (W) and north (N) respectively. One (or multiple) memory controllers are connected to each node. HT (HyperTransport) I/O controllers are connected through free crossbar ports of boundary nodes. Based on the SMOC architecture, a 2×2 mesh network can support a sixteen-core processor, and a 4×4 mesh network can support a sixty-four-core processor.

The first member of Godson-3 series is called Godson-3A, which is physically designed on 65 nm CMOS technology. The block diagram of Godson-3A is shown in Fig.2. The chip consumes 425 million transistors to integrate one four-core node, 4 MB L2 cache, two DDR2/3 controllers, two HyperTransport 1.0 controllers, one PCI/PCI-X controller, one LPC interface and one SPI interface. The chip measures 174 mm^2 in size. The peak frequency is up to 1 GHz and the power dissipation is 10∼15 W depending on applications. The eight-core Godson-3 and the sixteen-core one are still in physical implementation.

The processor core of Godson-3 (GS464) implements the MIPS64 instruction set in a four-issue, out-of-order execution way. It fetches and decodes four instructions per cycle and dynamically issues them to five fully pipelined functional units (two fix-point, two floating-point and one memory access unit). Instructions are issued out of order by two 16-entry reservation stations and are committed in program order by a 64-entry reorder queue. Two 64-entry physical register files are

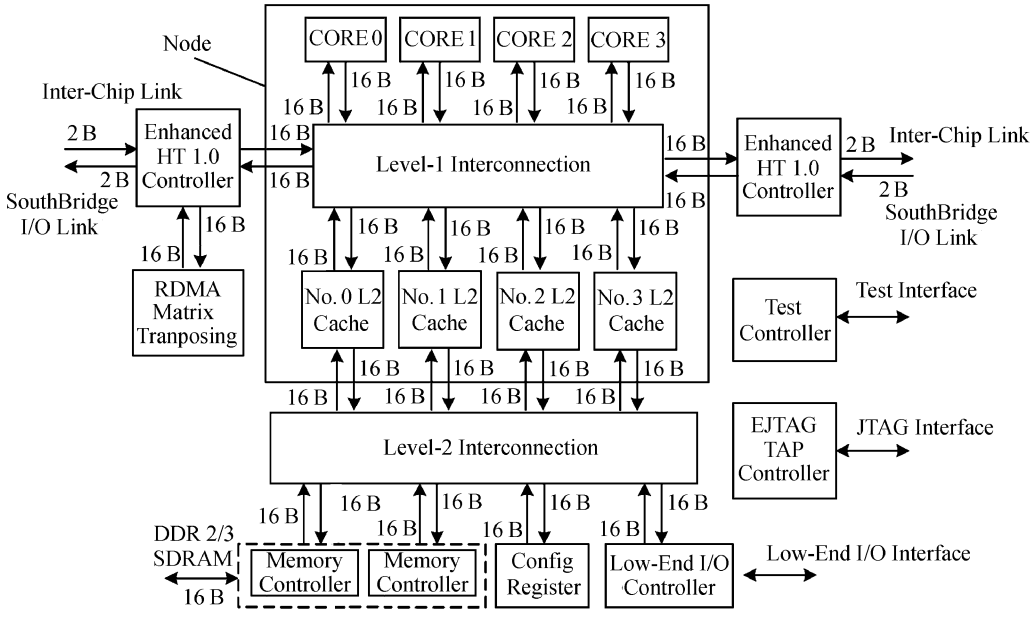


Fig.2. Godson-3A block diagram.

used for register renaming of the general purpose and floating point registers. An 8192-entry pattern history table, a 9-bit global history register, a 16-entry branch target buffer and a 4-entry return address stack keep the branch history information for prediction. GS464 has a 64 KB instruction cache and a 64 KB data cache. A 24-entry memory access queue that contains a content-addressable memory for dynamic memory disambiguation supports out-of-order memory access and non-blocking cache in GS464.

An important feature of GS464 is X86 emulation support. The XBAR (X86 Binary translation Acceleration on RIS processors) technique is proposed to speed up binary translation from X86 to MIPS binary codes with dedicated hardware support. In software-based binary translation[9], some X86-related features, which are not presented in MIPS, such as EFLAGs, floating-point register stack, segment addressing mode and so on, make the MIPS binary be translated from X86 inefficiently. In many cases, one X86 instruction may be translated into tens of MIPS instructions. To smooth the translation from X86 binary to MIPS binary, GS464 defines new instructions or runtime environments through MIPS64 UDI (User Defined Interface) to bridge the semantic gap between X86 ISA (Instruction Set Architecture) and MIPS64 ISA. More than 200 new instructions are defined and implemented with little hardware cost. Based on these hardware supports, a system-level and a process-level virtual machine monitor (VMM) can be implemented on top of Linux for being X86 compatible in both ISA and ABI (Application Binary Interface) levels.

In Godson-3A, four processor cores share a unified L2 cache, which is banked into four 1 MB modules on the same chip, through the Level-1 crossbar. Each L2 cache module can serve up to 8 coherence requests in flight from processor core and refill 16B of data per cycle. The directory for maintaining the identity of sharers is located in the L2 cache module. In addition, an uncached access queue (ucq) is also included in the L2 cache module for handling uncached operations to the memory.

4 Memory Subsystem

4.1 Memory Hierarchy

More than 70% of transistors of Godson-3 are expended by the memory subsystem. As shown in Table 1, the memory subsystem of Godson-3 consists of three levels, including L1 cache, L2 cache and DDR2/3 controllers. The communication between different components in the memory subsystem adopts AXI (Advanced extensible Interface) protocol with cache coherence extension.

Both L1 instruction cache and L1 data cache are 64 KB, four-way set-associative, using virtual index and physical tag and adopting random replacement strategy. Each line of L1 cache is 32 B. The latency of a memory access hitting L1 data cache is 3 cycles for fixed point and 4 cycles for floating point. Refilling data to L1 cache employs 16 B datapath, i.e., refilling one cache line needs two cycles. L1 instruction cache adopts the reliability scheme of parity checking, and L1 data cache adopts the reliability scheme of ECC (Error Checking and Correcting).

In the four-core Godson-3A, there are four L2 cache modules and the size of each module is 1 MB. These L2 cache modules are globally addressed and equally accessed by processor cores. To support cache coherence, the L2 cache also contains the directory information for each cache line. The L2 cache, which adopts the random replace strategy, is four-way set-associative, using physical address as index and physical address as tag. Godson-3 provides cache locking scheme to guarantee that the specific cache lines can stay in L2 cache forever. The latency of a memory access hitting L2 cache is about 20 cycles. Refilling data to L2 cache also employs 16 B datapath, i.e., refilling 1 cache line needs 2 cycles. L2 cache adopts the reliability scheme of ECC.

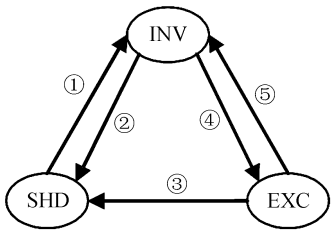
In order to provide sufficient bandwidth and capacity for on-chip processing, Godson-3A integrates two on-chip DDR2/3 memory controllers. The built-in memory controllers of Godson-3A conform to DDR SDRAM industry standard JESD79C, with the highest clock frequency of 400 MHz.

Table 1. Memory Hierarchy of Godson-3

Component	Configurations (Feature, Reliability, Access Latency, Capacity)
L1 data cache	Four-way, 32 B per line, virtual index, physical tag, random replace, ECC, 3 cycles for fix point, 4 cycles for floating point, 64 KB per core.
L1 instruction cache	Four-way, 32 B per line, virtual index, physical tag, random replace, parity, 1 cycle, 64 KB per core.
L2 cache	Four-way, 32 B per line, physical index, physical tag, random replace but lockable, ECC, ∼20 cycles, 4 MB shared by four cores.
DDR2/3	64 B datapath, up to 400 MHz, ECC, ∼100 cycles, 128 GB per controller.

4.2 Cache Coherence

Godson-3 employs directory based MSI-like cache coherence protocol. The directory of a cache line, which uses 32 bits to represent sharers of the cache line in L1 caches, is maintained in the corresponding L2 cache module. Each cache line has a home L2 cache module. There are three possible states for a cache line in L1 cache: INV (invalid), SHD (shared), and EXC (exclusive). The transitions between these three states are depicted in Fig.3.



① Reqinv, Reqreplace ② Reqread ③ Reqinvwtbk
④ Reqwrite ⑤ Reqinvwtbk, qreplace

Fig.3. Transition of cache state in Godson-3.

Table 2. Cache Coherence Protocol Through AXI Bus

Request of AXI Master and Replies of AXI Slave				
Reason	Request	AXI Channel	Replies	AXI Channel
L1 cache miss of load or fetch	Reqread	AR	Repread	R
L1 cache miss of store	Reqwrite	AR	Repwrite	R
Replacing L1 cache	Reqreplace	AW+W	Repreplace	B
DMA read request	Dmaread	AR	Repread	R
DMA write maintain cache coherence	Prewrite	AR	Preresp	R
DMA write request	Dmawrite	AW+W	Wresp	B
Request of AXI Slave and Replies of AXI Master				
Reason	Request	AXI Channel	Replies	AXI Channel
Invalidating L1 cache	Reqinv	R	Repinv	AW+W
Writing back L1 cache	Reqwtbk	R	Repwtbk	AW+W
Invalidating and writing back L1 cache	Reqinvwtbk	R	Repinvwtbk	AW+W

When load operations or instruction fetching cause L1 cache miss, the processor core sends a *Reqread* request to the corresponding L2 cache module; the L2 cache module sends *Repread* reply containing an SHD cache line backup to the corresponding processor core. When store operations cause the L1 cache miss, the processor core sends a *Reqwrite* request to the corresponding L2 cache module; the L2 cache module sends *Repwrite* reply containing an EXC cache line back to the corresponding processor core. When L1 cache replaces out some cache line, the processor core sends a *Reqreplace* request to the corresponding L2 cache module; the L2 cache module sends *Repreplace* reply to the corresponding processor core. An L2 cache module can send a *Reqinv* request to the corresponding processor core to invalidate SHD L1 cache backup; the processor core informs the completion of invalidation to the corresponding L2 cache module with *Repinv* reply. An L2 cache module can send *Reqwtbk* request to the corresponding processor core to degrade EXC L1 cache backup; the processor core informs the completion of degrading to the corresponding L2 cache module with *Repwtbk* reply. An L2 cache module can send a *Reqinvwtbk* request to the corresponding processor core to invalidate and write back EXC L1 cache backup; the processor core informs the completion of invalidation to the corresponding L2 cache module with *Repinvwtbk* reply.

The AXI protocol of Level-1 crossbar switch is extended to support cache coherence in Godson-3. There are five channels in AXI protocol, which are AR (reading address), R (reading data), AW (writing address), W (writing data), and B (writing response) respectively. Traditional read and write transactions can be realized on these five channels. Besides, cache coherence protocol needs coherence requests to send invalidations to processor core, and coherence replies to send invalidations to L2 cache modules. In Godson-3, coherence requests are realized on R channel, coherence replies are realized on AW+W channels. The usage of AXI channels for communications between processor cores, I/O and L2 cache are listed in Table 2.

5 Interconnection

5.1 Network-on-Chip

Godson-3 adopts the scalable mesh of crossbar topology. At the heart of the Network-on-chip is an 8×8 crossbar based router, in which every port conforms to the same 128-bit enhanced AXI interface. The novelty of the network lies in that it connects both function components (processor cores and L2 caches) and other nodes through a uniform interface.

The crossbar features low latency, enormous bandwidth, and support for directory-based cache coherence. The architecture of the crossbar is presented in Fig.4. It contains an 8×8 multiplexing matrix which connects

eight AML (AXI Mater Link) ports and eight ASL (AXI Slave Link) ports. Among the above ports, the left four AMLs are responsible for connecting four processor cores as master components and the right four ASLs are responsible for connecting four L2 cache modules as slave components.

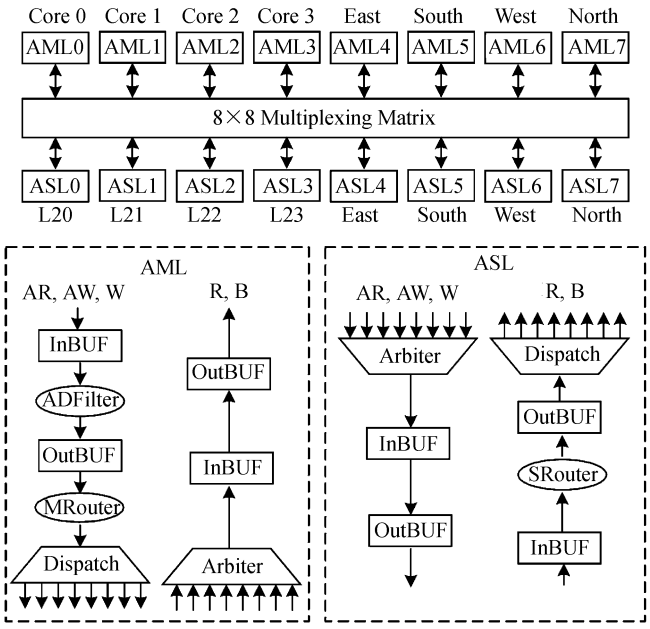


Fig.4. Microarchitecture of crossbar.

The rest of the AMLs and the ASLs are used to connect the adjacent nodes in four directions. There are two pipeline stages in both AML and ASL, i.e., the crossbar has the latency of four hops.

Each master-slave link in the crossbar has five channels according to the AXI protocol. The read and write requests (the AW/W and AR channels) are routed by the AML according to the access address. Each AW and AR channels have eight address windows. Each window assigns a request to a matching destination ASL port. The ASL routes read and write replies (R and B) according to the id of the corresponding request.

5.2 HT Based Inter-Chip Interconnection

The HyperTransport interfaces implemented in Godson-3 are connected to the free ports of the Level-1 crossbar. When used as I/O link, the HyperTransport interface uses a write-invalidate approach to maintain the cache coherence between I/O and processor cores. When used for multi-chip interconnection, the HyperTransport interface serves as a channel for cache coherence traffic between chips.

On the basis of the scalable mesh architecture of Godson-3 processor, a multi-chip system can be constituted. The two 16-bit HyperTransport interfaces can be taken apart into four 8-bit HyperTransport interfaces, providing enough flexibility for interconnecting with the neighborhood chips in mesh architecture.

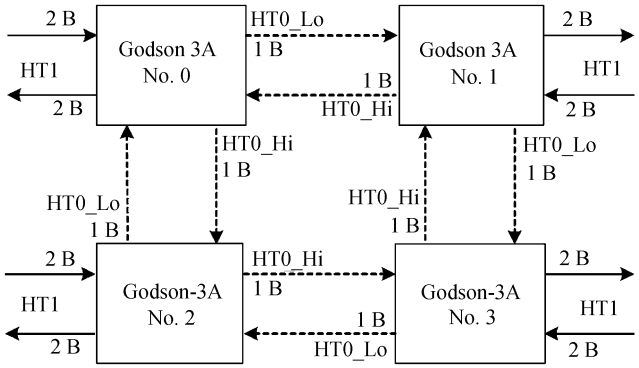


Fig.5. Interconnection of a four-chip CC-NUMA system based on Godson-3A. The dotted lines indicate the enhanced HyperTransport interfaces with cache coherence protocol support, while the real lines indicate the standard HyperTransport interfaces.

Fig.5 shows an interconnection paradigm for a 16-core CC-NUMA system with four Godson-3 chips. Extended HyperTransport protocol is used on HT0_Lo and HT0_Hi for coherent interconnection. In such a system of four chip, only one HyperTransport controller per chip (called as HT0, can be taken apart into two 8-bit interfaces, marked as HT0_Lo and HT0_Hi in Fig.5) is needed to be configured into cache coherent mode and the other one (which we call as HT1) can be used as I/O connection.

The routing in the system adopts the X-Y algorithm, which means that any message, no matter a request or a response it is, will first route along the X direction then along the Y direction until it has reached the target. For example, there is a read request from Chip 0 targeting at Chip 3. This request will first pass through the HT0_Lo in Chip 0 to Chip 1, and then pass through the HT0_Lo in Chip 1 to Chip 3, which is the target. The corresponding read response generated by Chip 3 will first pass through the HT0_Lo in Chip 3 to Chip 2, then pass through the HT0_Lo in Chip 2 to Chip 0 and finally complete the whole request.

The present implementation of directory-based cache coherence protocol in Godson-3A can support at most 4 chips (16 cores). Interconnecting more than four Godson-3A chips can be achieved in a non-cache-coherent manner. Up to 16 chips (64 cores) NCC-NUMA system can be implemented. Fig.6 shows the method of interconnecting two CC-NUMA systems (4 chips each) to form a much larger NCC-NUMA system (8 chips totally). The HT0 controller works in cache coherent mode, as depicted in Fig.5. The HT1 controller is used for interconnection between the two CC-NUMA

systems. The routing of communication across CC-NUMA systems should be determined by configuring the address access windows in the Level-1 crossbar. A more detailed discussion about multi-chip interconnection of Godson-3 can be found in [10].

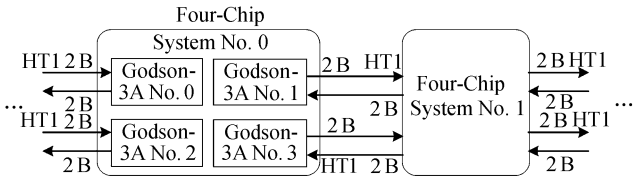


Fig.6. Interconnection of an 8-chip NCC-NUMA based on Godson-3A.

5.3 Extension with Dedicated Network

Much larger shared memory system can also be built through the "hierarchical directory extension" technology of Godson-3. Since both CPU cores and interconnection controllers (i.e., HyperTransport Controller) are connected to the L2 cache modules through a unified crossbar, each L2 cache block can regard remote access request from a crossbar port as that from a virtual processor connected to the port, and a bit in the bit vector directory of an L2 cache block can represent either a local processor or a virtual processor. Fig.7 shows a bit vector which indicates an L2 cache block is kept by CPU cores P0, P2 and P4 which connects to ports 0, 2 and 4 of the crossbar. Port 4, however, is connected to the HyperTransport controller, and hence P4 is a virtual CPU core which represents all CPU cores that access that L2 cache block through port 4 of the crossbar. An external directory entry can be maintained in the inter-chip interconnection network to record CPU cores which access the L2 cache block through port 4 of the crossbar. Fig.8 shows a large-scale shared memory system which connects a huge number of Godson-3 chips with a scalable network and extended directory.

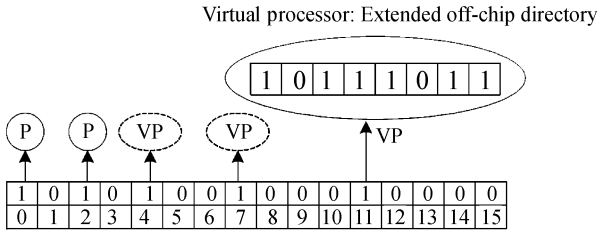


Fig.7. Directory extension through virtual processor.

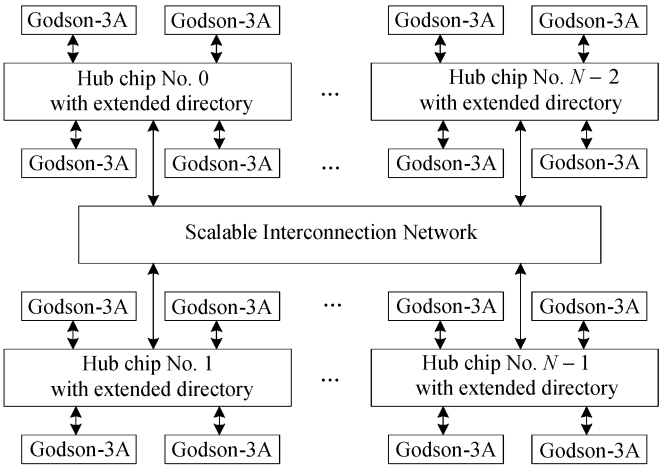


Fig.8. A large-scale system of Godson-3 using dedicated network.

6 I/O Subsystem

6.1 Extensive I/O Support

Fig.9 shows the I/O structure of the Godson-3 system. The system supports a broad range of I/O interfaces. These peripherals can be classified into two categories: the high-speed IPs and the low-end IPs.

High-speed I/O controllers are critical for efficient executions of throughput based workloads, while

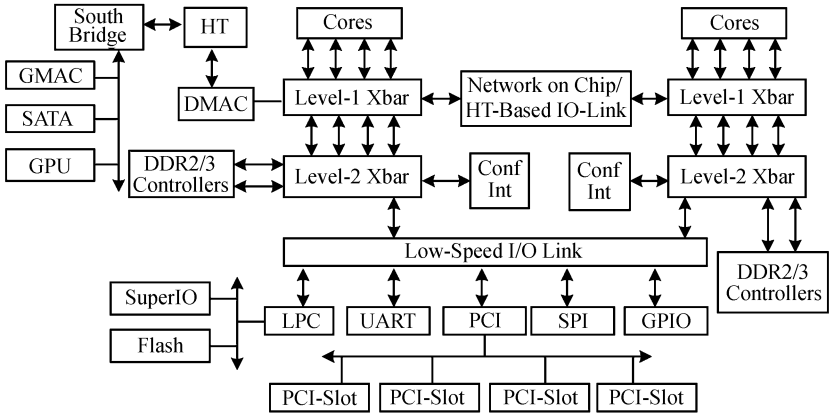


Fig.9. I/O subsystem overview.

low-end I/O controllers provide flexibility to form a functionally-complete system. The first version of Godson-3 currently encompasses two high-speed I/O controllers of HyperTransport.

Low speed I/O interfaces in Godson-3A include PCI/PCI-X controller, LPC controller, UART controller, SPI controller and GPIO. In addition, there are 12 pins available for chip interrupt, including 4 dedicated interrupt pin, 4 PCI interrupts and 4 GPIOs capable of interrupt triggering. Each pin can be individually configured as level/edge sensitive and masked out. Additionally, each interrupt pin can be routed to interrupt a certain processor core as dictated by software programmable registers.

6.2 Cache Coherence Support for I/O DMA

In multiprocessing systems based on Godson-3, which are equipped with multiple cache hierarchies, DMA operations from I/O devices may lead to cache coherence problem. When an I/O device accesses a location in the global memory, the most up-to-date copy may lie in off-chip memory modules or the on-chip L2 cache or L1 cache inside a certain processor core.

There are some distinct discrepancies between the I/O and the processor core, making it difficult to effectively implement cache coherence for I/O access. The cache protocol is inevitably biased toward the processors core, resulting in maintaining cache-coherence on granularity of one cache line (32 B in Godson-3), which is the maximum length for processor access. The request length of one I/O request, however, typically spans multiple cache lines. Additionally, an inherent difference between processor initiated accesses and I/O initiated accesses is the cacheable property of reading. While the processor core will keep a copy of the retrieval data in its L1 cache for later use, the I/O will consume and then discard the data due to poor locality of the I/O access sequence.

Godson-3 system provides a simple and elegant solution to address this issue. The cache coherence is still maintained on granularity of one cache line, therefore a multi-line I/O access is split into multiple single-line accesses for handling. As shown in Fig.10, a DMA coherence module lies between the HT controller and the Level-1 crossbar. Requests from the HT controller will be accepted and served by the internal logic of the DMA coherence module. Due to the split nature of read and write channel in the interface, DMA coherence contains a read queue for handling read requests, and a write queue for handling write requests respectively.

The state transition of write queue is illustrated in Fig.11. The write request enters and retires queue in their sending order, but the write queue can handle them out-of-order so as to allow better performance. The mechanism of in-order retirement but out-of-order handling is very similar to the reorder queue in classic superscalar processors capable of out-of-order executions.

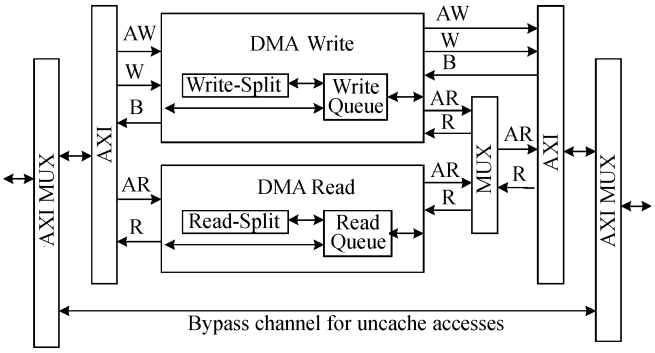


Fig.10. Structure of DMA controller.

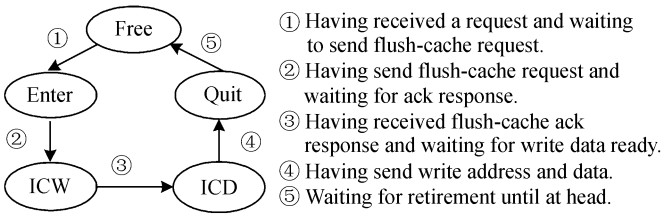


Fig.11. State transition in DMA write queue.

Handling of DMA read is relatively simple. The main task is to split a single multi-line request into multiple single-line requests. Each entry of the read queue is responsible for handling a one-line request. The read queue issues DMA read requests to the destination L2 module as normal transactions. The DMA read is a special access for L2 cache controller. Compared with handling of the normal processor-initiated read, handling the DMA read has the only difference in that there is no need to modify the identity vector of directory. Since the data response from L2 cache may arrive out of order, request enters and retires queue in their sending order, so the read queue contains enough space to buffer the data and maintains the atomicity and the order of data response to I/O.

7 RAS Features

7.1 Treatment for Hardware Error

Memory subsystem consumes most transistor resources of a multi-core processor, therefore error detection and recovery of the memory subsystem is important to provide high reliability. In Godson-3, all components of the memory subsystem are protected by parity checking or ECC. There are two rules for the reliability strategy of Godson-3. First, if a component can

recover its content from other components of the memory subsystem, it uses parity checking, otherwise it uses ECC. Second, error should be automatically found and recovered by hardware.

For example, once a cache line read from L1 instruction cache fails in parity checking, the cache line is flushed out of L1 instruction cache, and a cache miss happens to get correct value from L2 cache. Once a cache line read from L1 data cache fails in ECC check, corrected value of the cache line is automatically generated, and both the value read out and the value stored in data cache are automatically corrected without software intervention.

Godson-3 also provides enough flexibility to turn off its components. For example, processor cores, HT controllers, or DDR controllers can be turned off by operating system if they are proved to be instable or idling. Other components can continue their work unaffectedly.

7.2 Debugging

Debugging either hardware or software on a multi-core processor is difficult and time-consuming. Godson-3 adopts MIPS EJTAG standard to facilitate debugging. As a standard for hardware/software subsystem, EJTAG provides comprehensive debugging capabilities, including debug exception and debug mode, off-board EJTAG memory, debug breakpoint, instruction and data hardware breakpoints, single-step execution, program counter scan and so on. The IEEE 1149.1 JTAG Test Access Port (TAP) standard is employed to provide an external interface for EJTAG. Every processor core in Godson-3 has an EJTAG TAP controller, and all TAP controllers are connected as a chain. A debug host can communicate with each processor core through their TAP controllers.

8 Preliminary Performance Evaluation

At the time of paper writing, the first-silicon sample chips of Godson3a had just returned from fabrication.

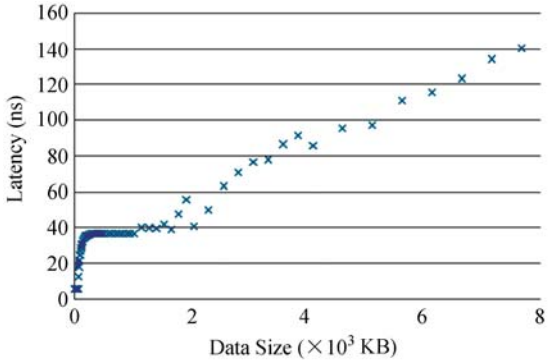


Fig.12. Dependent load latency.

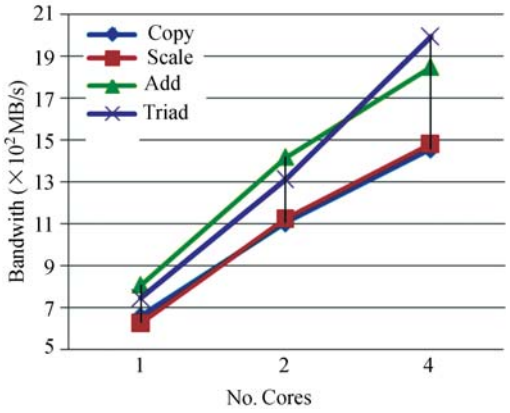


Fig.13. Stream bandwidth.

Table 3. SPEC CPU2000 Comparison Between Godson and Pentium Series Processors
(All programs are compiled with GCC compilers)

	Godson SPEC Ratio			Pentium SPEC Ratio	
	2E-750	2F-800	3A-800	PIII-800	PIV-1.4
	Mhz	Mhz	Mhz	Mhz	Ghz
164.gzip	209	251	324	344	397
175.vpr	237	239	391	261	246
176.gcc	282	329	369	241	350
181.mcf	271	232	421	229	255
186.crafty	356	362	415	352	386
197.parser	202	152	225	231	331
252.eon	289	441	526	90.7	125
253.perlbmk	235	321	330	397	547
254.gap	238	243	229	260	441
255.vortex	236	274	297	383	478
256.bzip2	247	241	268	249	314
300.twolf	313	331	486	269	287
SPECint2000	**256**	**275**	**345**	**260**	**326**
168.wupwise	307	308	325	248	474
171.swim	247	273	336	218	244
172.mgrid	156	155	184	99.2	320
173.applu	188	268	200	154	333
177.mesa	373	438	400	265	265
178.galgel	-	345	583	-	-
179.art	349	693	1254	115	109
183.equake	250	303	278	190	493
187.facerec	-	111	177	-	-
188.ammp	277	283	364	174	200
189.lucas	-	284	251	-	-
191.fma3d	-	108	128	-	-
200.sixtrack	131	217	184	137	224
301.apsi	172	197	225	190	199
SPECfp2000	**232**	**254**	**289**	**171**	**263**

System-level verification and testing had affirmed success of the first taping-out of Godson-3. In this section, we present some preliminary performance results.

Fig.12 lists the dependent-load latency measured by

Table 4. SPECint_rate2000 and SPECfp_rate2000 of Godson3 at 700 MHz

SPEC Programs	Description	Copies	Rates
164.gzip	Compression	4	10
175.vpr	FPGA circuit placement and routing	4	14
176.gcc	C programming language compiler	4	14
181.mcf	Combination optimization	4	11
186.crafty	Game playing: chess	4	21
197.parser	Word processing	4	11
252.eon	Computer visualization	4	23
253.perlbmk	Perl programming language	4	12
254.gap	Group theory, interpreter	4	10
255.vortex	Object-oriented database	4	15
256.bzip2	Compression	4	11
300.twolf	Place and route simulator	4	17
SPEC_INT2000rate			**13**
168.wupwise	Physics: Quantum chromodynamiscs	4	21
171.swim	Shadow water modeling	4	12
172.mgrid	Multigrid solver: 3D potential field	4	7
173.applu	Partial differential equations	4	20
177.mesa	3D graphics library	4	16
178.galgel	Computational fluid dynamics	4	34
179.art	Image recognition/neural networks	4	82
183.equake	Seismis wave propagation simulation	4	16
187.facerec	Image processing: Face recognition	4	15
188.ammp	Computational chemistry	4	11
189.lucas	Number theory/primality testing	4	11
191.fma3d	Finite-element crash simulation	4	10
200.sixtrack	Nuclear physics accelerator design	4	11
301.apsi	Meteorology: pollutant distribution	4	9
SPEC_FP2000rate			**15**

the LMbench[11]. The processor core has a 64 KB Level-1 Data cache and a 4 MB Level-2 uniform cache.

The load-to-use latency is lower than 6 ns for data range below 64 KB and no more than 90 ns for data range between 64 KB and 4 MB. Fig.13 presents the sustainable memory bandwidth in megabytes per second gauged by the STREAM benchmark[12]. It can be drawn from the figure that Godson-3 provides nearly-linear bandwidth scaling as the core number increases.

Table 3 shows the spec ratios of Godson and Intel Pentium series processors[13]. In order to provide a fair comparison, all the SPEC programs are compiled with GCC compiler. It can be seen from the table that performance superiority of Godson-3 is obvious. Table 4 lists the SPEC CPU2000 rates of Godson-3 at operating frequency of 700 MHz. The SPEC programs are compiled with the ORC compiler from ICT.

It should be noted the above evaluation is still very preliminary. Sophisticated system-level and compiler-level performance optimizations for Godson-3 are still in process and will last several months. Afterwards, there will be a dedicated paper for detailing the performance of Godson-3.

9 Conclusions

This paper introduces the system architecture of the Godson-3 multi-core processor from the aspects of system scalability, organization of memory hierarchy, network on-chip, inter-chip connection and I/O subsystem.

Our recent work includes designing eight-core and sixteen-core Godson-3 chip with improved frequency, building a PetaFlops high performance computer based on Godson-3 processors. Research on TeraFlops scale many-core chips is also in progress.

References

[1] Hu W W, Tang Z M. Microarchitecture design of the Godson-1 Processor. *Chinese Journal of Computers*, April 2003, 26(4): 385-396. (in Chinese)

[2] Hu W W, Zhang F X, Li Z S. Microarchitecture of the Godson-2 processor. *Journal of Computer Science and Technology*, March 2005, 20(2): 243-249.

[3] Hammond L *et al.* A single-chip multiprocessor. *IEEE Computer*, Sept. 1997, 30(9): 79-85.

[4] Tendler J, Dodson S, Fields S, Le H, Sinharoy B. Power4 system microarchitecture. IBM Technical White Paper, October 2001.

[5] Kalla R *et al.* IBM POWER5 chip: A dual-core multithreaded processor. *IEEE Micro*, March 2004, 24(2): 40-47.

[6] Kongetira P *et al.* Niagara: A 32-way multithreaded Sparc processor. *IEEE Micro*, March 2005, 25(2): 21-29.

[7] Hu W W, Wang J, Gao X, Chen Y, Liu Q, Li G J. Godson-3: A scalable multicore RISC processor with X86 emulation. *IEEE Micro*, 2009, March/April, pp.17-29.

[8] Laudon J, Lenoski D. The SGI origin: A ccNUMA highly scalable server. In *Proc. the 24th International Symposium on Computer Architecture*, Denver, USA, June 2-4, 1997, pp.241-251.

[9] Klaiber A. The technology behind CrusoeTM processors. White paper, Transmeta Corporation, Jan. 2000.

[10] Wang H, Gao X, Chen Y, Hu W. Interconnection of Godson-3 multi-core processor. *Journal of Computer Research and Development*, Dec. 2008, 45(12): 2001-2010. (in Chinese)

[11] http://www.bitmover.com/lmbench/.

[12] http://www.cs.virginia.edu/stream.

[13] Hu W W, Zhao J Y, Zhong S Q, Yang X, Guidetti E, Wu C. Implementing a 1 GHz four-issue out-of-order execution microprocessor in a standard cell ASIC methodology. *Journal of Computer Science and Technology*, 2007, 22(1): 1-14.

Xiang Gao received his B.S. degree in 2002 and his Ph.D. degree in 2007, both in computer science from the University of Science and Technology of China. He is currently an assistant professor in the Institute of Computing Technology, Chinese Academy of Sciences. His research interests include high performance computer architecture, parallel processing and operating system.

Yun-Ji Chen received his B.S. degree from the University of Science and Technology of China in 2002 and his Ph.D. degree from the Institute of Computing Technology, Chinese Academy of Sciences in 2007, both in computer science. He is currently an assistant professor in the Institute of Computing Technology. His research interests include hardware verification and high performance computer architecture.

Huan-Dong Wang received his B.S. degree from the University of Science and Technology of China in 2004, majored in computer science. He is currently a Ph.D. candidate of the Institute of Computing Technology, Chinese Academy of Sciences. His research interests include high performance computer architecture, memory and IO system.

+

Dan Tang is a Ph.D. candidate in computer science at the Institute of Computing Technology, Chinese Academy of Sciences. His research interests include computer architecture, parallel computing and operating system. He received an M.S. degree in computer science from Huazhong University of Science and Technology.

Wei-Wu Hu received his B.S. degree from the University of Science and Technology of China in 1991, and his Ph.D. degree from the Institute of Computing Technology, Chinese Academy of Sciences in 1996, both in computer science. He is currently a professor in the Institute of Computing Technology. His research interests include high performance computer architecture, parallel processing and VLSI design.