The High Performance Interconnection Network of the Parallel Inference Machine PIE64

Hanpei Koike, Eiichi Takahashi, Tsukasa Yamauchi and Hidehiko Tanaka
The Tanaka Lab., Dept. of Electrical Engineering,
The University of Tokyo, Hongo 7-3-1, Bunkyo-ku, Tokyo 113, JAPAN

Abstract

PIE64 is a parallel inference machine. It is designed mainly for fast execution of the parallel symbol manipulating programs written in a committed-choice language FLENG. 64 Inference Units (IUs) are connected using two high speed interconnection networks; Process Allocation Network (PAN) and Data Allocation/Accessing Network (DAAN). Both PAN and DAAN are 3-stage networks and have an automatic load balancing facility.

The base of a parallel machine is an interconnection network. The performance and the property of the interconnection network directly affects the total architecture and processing method of the system. So, we start designing PIE64 from designing its network system. We have already designed the switching unit (SU) chip which is the elementary component of the network. We also start building the network hardware.

In this paper we present the network design of PIE64. In the first section, a brief introduction to the architecture of PIE64 is presented. We describe the SU chip in Section 3. In Section 4, we discuss how to implement the interconnection network using the SU chips.

1 Introduction

PIE64 is a parallel inference machine. It is designed mainly for fast execution of the parallel symbol manipulating programs written in the *reduced primitive set* committed-choice language FLENG [1]. 64 Inference Units (IUs) are connected using two high speed interconnection networks; Process Allocation Network (PAN) and Data Allocation/Accessing Network (DAAN). The global architecture of PIE64 is shown in Fig. 1.

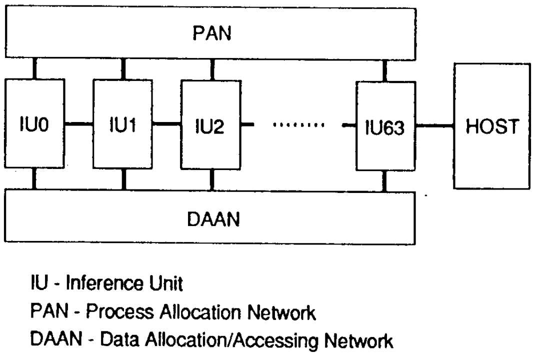
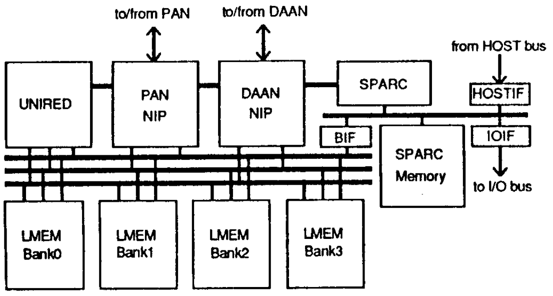

PAN
IU0
IU1
IU2
....
IU63
HOST
DAAN
IU - Inference Unit
PAN - Process Allocation Network
DAAN - Data Allocation/Accessing Network

Figure 1: The Global Architecture of PIE64

Each IU consists of:

1. Unifier/Reducer (UNIRED)
2. Network Interface Processor (NIP)
3. SPARC Processor
4. Local Memory (LMEM)
5. SPARC memory
6. Host interface
7. I/O interface



UNIRED - Unifier/Reducer
NIP - Network Interface Processor
LMEM - Local Memory

Figure 2: The Internal Architecture of IU

The internal architecture of the IU is shown in Fig. 2.

UNIRED and NIP are co-processors of the SPARC processor. They enhance the inference function and the parallel processing function of the IU respectively.

UNIRED is a support hardware for inference. UNIRED is a silicon interpreter of logic programs and executes goal rewriting primitive operations shown below using pipelines [3].

- Passive Unification
- Active Unification
- Goal Reduction
- Overwrite Goal Reduction

These operations are executed on multiple *contexts* (goal and clause pairs) concurrently supported by the fast context switching mechanism in order to maintain high throughput of processing against relatively slow remote data access and frequent suspension. UNIRED is also to support primitive operations for garbage collection and compaction.

NIP is a support hardware for parallel processing. Basic operation of the NIP is to transfer contents of the local memory to/from other IUs through PAN and DAAN on a request from the UNIRED or the SPARC processor. The destination of transfer is either addressed explicitly by the requester or selected automatically to one whose load value is the lowest. The latter is supported by the load balancing facility of the interconnection network described below.

SPARC processor executes system predicates and program modules written in conventional lauguages such as C. It also schedules the goal execution in UNIRED and manages load distribution.

UNIRED, NIP and SPARC share LMEM through three fast pipeline-arbitrated synchronous buses. Each memory access port of the processors can request memory access every 100 ns and the result will be obtained after 200 ns. Arbitration and data access are performed in a pipelined manner. Two phase of arbitration, i.e. bus arbitration and memory bank arbitration are done in the first stage of the pipeline. In the next stage, data are accessed on each bus.

Both PAN and DAAN are 3-stage networks and have an automatic load balancing facility [2]. Load information of each destination is sent through unused path of the network. On a load distribution request, the network automatically selects the connectable path to the processor whose load is the lowest. PAN uses this facility to balance processing load among IUs. DAAN uses this facility to balance allocation of data among IUs and to reduce access contention [5].

A Sun-4 workstation is used as a host processor of PIE64. Its main functions are program loading, user interface and system maintenance.

Three gate array chips; UNIRED, NIP, and a switching unit (SU) for the interconnection networks, are under designing as the elementary components of PIE64.

2 Interconnection Network of PIE64

The base of a parallel machine is an interconnection network It must fulfill requirements such as:

- high performance enough not to be the bottle neck of the system
- moderate cost and esiness of implementation

The performance and the property of the interconnection network directly affects the total architecture and processing method of the system. The goal of PIE64 is a general-purpose MIMD machine. It requires high performance interconnection network. So, we start designing PIE64 from designing its network system. We have already designed the SU chip which is the elementary component of the network. We also start building the network hardware.

In this paper we present the network design of PIE64. We describe the SU chip in Section 3. In Section 4, we discuss how to implement the interconnection network using the SU chips.

3 SU chip

3.1 Main features of the SU chip

The SU chip is the elementary component of the network. The main features of the SU chip are as follows:

1. Besides normal destination-addressed communication, the SU chip has an *automatic load balancing facility*. The SU chip automatically selects the route to the destination whose load is the lowest, by comparing load information sent on free paths in reverse direction.
2. The SU chip contains a 4 × 4 crossbar switch with 8-bit data width. The chip has two modes. In the *master mode*, the chip has a routing function. In the other, *slave mode*, the chip only works as a switching circuit controlled by the master chip. If one chip is used in master mode and the other chips in slave mode, a network with arbitrary data width can be built.
3. The SU chip supports circuit switching, synchronous or asynchronous communication. It also supports multicasting.

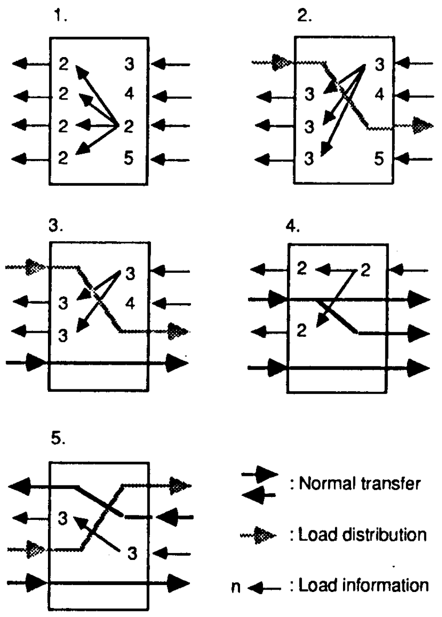


Figure 3: The Examples of Communication

Examples of several kind of communication offered by the SU chip are shown in Fig. 3.

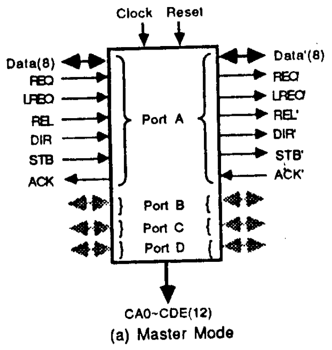
1. shows transfer of load information.
2. shows load distribution.
3. shows normal connection.
4. shows multicasting.
5. shows reverse communication.

3.2 Function of the SU chip

SU chips in master and slave mode are shown in Fig. 4.

The SU chip has data/control lines shown below on each port.

- D0-7 ... bi-directional data lines
- REQ ... connection request
- LREQ ... load distribution connection request
- DIR ... direction of data transfer
- REL ... connection release



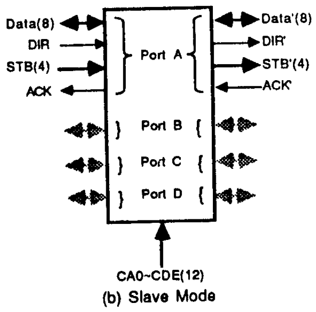


Figure 4: The SU chip in two modes

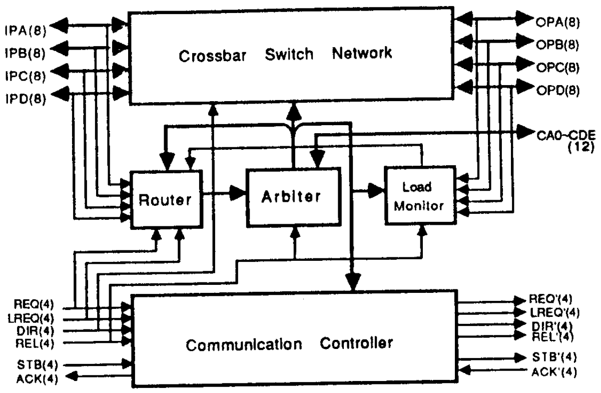


Figure 5: The Configuration of the SU chip

- STB ... forward (source to destination) communication control line
- ACK ... reverse (destination to source) communication control line

It also has control lines such as:

- STAGE0-1 ... indicate which stage the chip is and which bits on address to use
- CHMODE ... select master/slave mode
- ARMODE ... select 1 clock/2 clocks arbitration
- CA0-CDE ... connection information from a master chip to slave chips

Ether *Destination-addressed connection* or *load distribution connection* can be requested to each input port of the SU chip using REQ and/or LREQ. After each connection, input port and addressed output port is connected in a circuit switching manner. Data can be transferred data synchronously/asynchronously in a one to one / multicasting way. To multicast data, connection request may be repeated as long as needed. If signals are sent in reverse direction on multicast connection, they are ORed together.

3.3 Configuration of the SU chip

The SU chip is divided into five blocks shown in Fig. 5.

1. crossbar switch
   The crossbar switch has four input ports and four output ports. Half duplex communication is possible using control line DIR. The load values of the destinations are supplied from the next stage to unused output ports of the chip in reverse direction. The lowest of them are passed to the previous stages through unused input ports.
2. router
   On destination-addressed connection request, the router decodes address on the input port and requests the connection to the addressed output port. On load balancing connection request, the router requests the connection to the output port whose load value is the lowest.

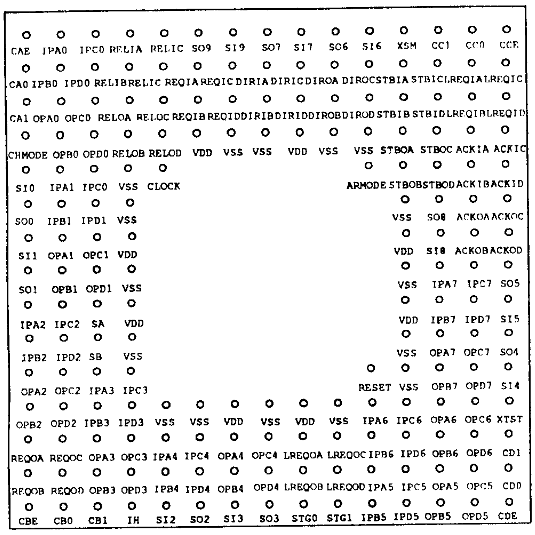

CAE IPA0 IPC0 RELIA RELIC SO9 SI9 SO7 SI7 SO6 SI6 XSM CC1 CC0 CCE
CA0 IPB0 IPD0 RELIB RELIC REQIA REQIC DIRIA DIRIC DIROA DIROC STBIA STBIC LREQIA LREQIC
CA1 OPA0 OPC0 RELOA RELOC REQIB REQID DIRIB DIRID DIROB DIROD STBIB STBID LREQIB LREQID
CHMODE OPB0 OPD0 RELOB RELOD VDD VSS VSS VDD VSS VSS STBOA STBOC ACKIA ACKIC
SI0 IPA1 IPC0 VSS CLOCK ARMODE STBOB STBOD ACKIB ACKID
SO0 IPB1 IPD1 VSS VSS SO8 ACKOA ACKOC
SI1 OPA1 OPC1 VDD VDD SI8 ACKOB ACKOD
SO1 OPB1 OPD1 VSS VSS IPA7 IPC7 SO5
IPA2 IPC2 SA VDD VDD IPB7 IPD7 SI5
IPB2 IPD2 SB VSS VSS OPA7 OPC7 SO4
OPA2 OPC2 IPA3 IPC3 RESET VSS OPB7 OPD7 SI4
OPB2 OPD2 IPB3 IPD3 VSS VSS VDD VSS VDD VSS IPA6 IPC6 OPA6 OPC6 XTST
REQOA REQOC OPA3 OPC3 IPA4 IPC4 OPA4 OPC4 LREQOA LREQOC IPB6 IPD6 OPB6 OPD6 CD1
REQOB REQOD OPB3 OPD3 IPB4 IPD4 OPB4 OPD4 LREQOB LREQOD IPA5 IPC5 OPA5 OPC5 CD0
CBE CB0 CB1 IH SI2 SO2 SI3 SO3 STG0 STG1 IPB5 IPD5 OPB5 OPD5 CDE

Figure 6: Pin Assignment of the SU chip

3. arbiter
   The arbiter arbitrates the connection requests occurred at the same time to the same output port. The arbiter uses the ring counters for fair arbitration. Synchronous(one clock)/asynchronous(two clocks) arbitration can be selected according to if each stage works synchronously or not.

4. load monitor
   The load monitor compares load information supplied from free output ports and determines the output port through which the lowest load processor can be reached.

5. communication controller
   The communication controller generates control lines to be sent to the next stage.

3.4 Implementation of the SU Chip

The SU chip is implemented using a 8700-gate gate array. The package is Pin Grid Array of 179 pins. Pin assignment of the SU chip is shown in Fig. 6.

4 Hardware Implementation of the Network

4.1 Construction of the Network

The interconnection networks of PIE64 are both 64 × 64 3-stage networks. These networks are built using the SU chips as follows.

Using four SU chips (one for master and three for slave), a 4 × 4 crossbar switch with:

- 4 network control lines (REQ, LREQ, REL, DIR)
- 32 bi-directional data lines
- 13 forward communication control lines (STB0-12)
- 4 reverse communication control lines (ACK0-3)

can be built.

To build 64 × 64 3-stage network, 16 crossbar switchs shown above (64 SU chips) are required for one stage. The overall number of the chips to build a 3-stage network is 192.

64 × 64 3-stage network can be divided into 16 × 16 2-stage networks and a 64 × 64 shuffle (Fig. 7).

We use four 16 × 16 network boards and one 64 × 64 shuffle board to build 64 × 64 network (Fig. 8). One 16 × 16 network board contains 32 SU chips, 16 connectors to/from IUs and connector to/from 64×64 shuffle board. If the connector to/from 64 × 64 shuffle board is short-circuited, this board can be used alone as a small 16 × 16 2-stage network. 64 × 64 shuffle board contains 64 SU chips and 4 connectors to/from 16 × 16 network boards.

We estimated that the number of connection lines passing between neighboring two pins of a SU chip on the network board is about 6. So, example of wiring is using 2 wiring layers with 3 connection lines per layer. This wiring density is easily realizable using current technology.

We also estimated that the maximum length of lines connecting two SU chips can be almost limitted to the width of the board. The deviation

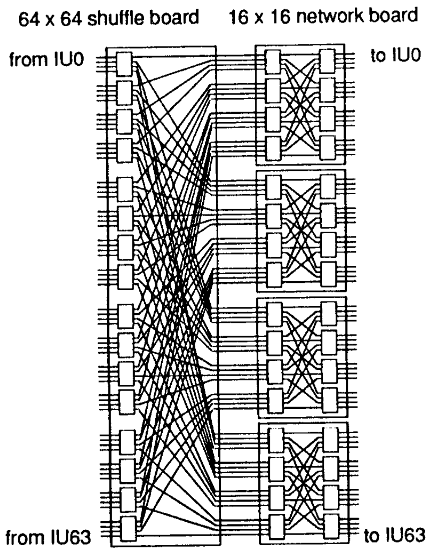


Figure 7: 64 × 64 3-stage Network

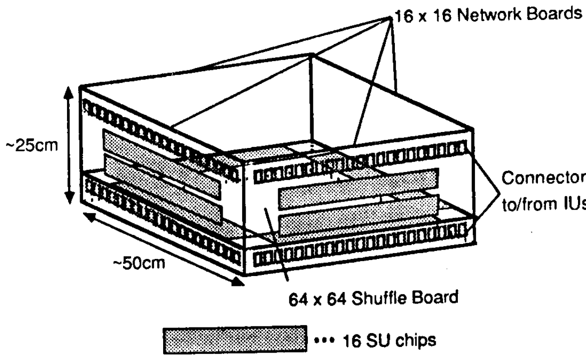


Figure 8: Hardware Implementation

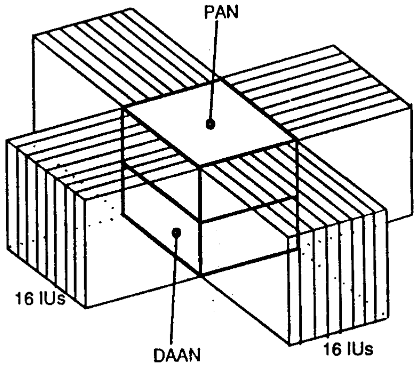


Figure 9: Two Networks of PIE64

of the length can be very small, if it is properly wired. Small deviation of delay time is important for high throughput.

The number of contacts between one 16×16 network board and a 64 × 64 shuffle board is about 1300 ~ 1600. This means the network hardware requires contact density of 80 contacts per inch. Such a high density connector has become available today.

4.2 Performance of the Network

Data are expected to be transferred at 40M Byte/s per port synchronously with common clock. Total maximum throughput of two networks of PIE64 will be about 5G Byte/s.

The delay time for each stage of the SU is estimated to be about 30 ns. Since data are transferred in a circuit switching manner, network delay may be around 100 ns.

To assure safe communication, the output registers and the input latches of the network interface will be driven by two different clocks whose phases are precisely trimmed.

Connection takes 4 clocks. To shorten the connection time, each stage of the SU is driven properly phase-trimmed clocks. Using this technic, connection time can be shortened.

5 Discussions

5.1 Trade-off of the Number of Ports and Data Width

Our design choice on the number of ports and data width is 4 × 4 and 8 bit. Another choice may be 8 × 8 and 4 bit. This seemed attractive in the early stage of designing. It requires same number of pins for data lines of IP and OP, i.e. 64. But the number of the other lines increases substantially. Control lines may be twice as much, i.e. the number increases from 48 to 96. Connection control lines from tme master to the slave may increases from 12 to 32. The number of pins exceeds the pin limitation of the package. Another problem in this choice is comparison of load value. 8 bit is at least necessary for the width of the load value. If the data width is reduced to 4 bit, comparison must be performed on 2 master chips. This requires not only additional pins, but also the speed of comparison may become slower. Accordingly, our choice is the optimal one as long as current technology is used.

5.2 Wrong Load Distribution

If the path to the lowest load processor is blocked, load can be distributed to the processor whose load is not necessarily lowest. This seems cause wrong load distribution. But the sender can decide whether it is worth distributing its load or not, by comparing its own load and load information sent through the network. The sender does not send load to the destination whose load is higher than the load of the sender. Very rarely, if the load distribution requests to the same lowest load processor occur at the same clock, this case happens, although, because senders but one cannot reach the processor whose load information all senders use to decide to send. Avoidance of this case such as time-outing may be necessary.

5.3 Future Expansion

For more than 64 inference units, hierarchical network will be used. For example, PIE256 is composed of four PIE64s connected by 4 × 4 level-2 network, and PIE1024 is composed of 16 PIE64s connected by 16 × 16 level-2 network as shown in Fig. 10.

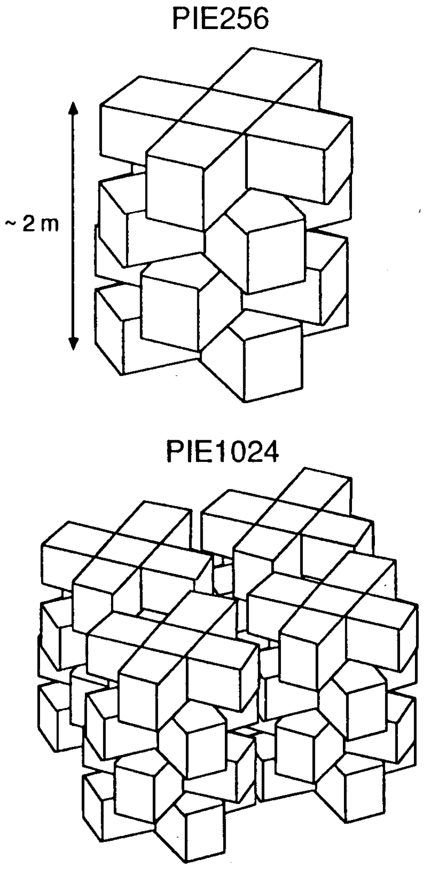


Figure 10: Future Expansion of PIE64

6 Conclusions

In this paper the network design of PIE64 is presented. The SU chip, the elementary component of the network, is a 4 × 4 cross-bar switch with 8-bit data width. It has automatic load balancing facility and works in either master or slave mode. The interconnection network hardware uses these 192 SU chips and its size can be only $50cm \times 50cm \times 25cm$. The performance of the network is expected to be 40 MByte/s per port. We have already desined the SU chip and start testing it. We also start implementing the network hardware.

7 Acknowledgement

The authors would like to thank many people who helped us, especially Dr. Kei Hiraki at ETL who showed us his large scale dataflow machine Sigma-1 and inspired us, Dr. Shuichi Sakai at ETL by whom the original idea of the load balancing facility is conceived, Mr. Kouichi Kumon and Mr.

Hideo Miyake at Fujitsu Lab. who advised and helped us on gate array design and Toshiaki Tarui at Hitachi Central Lab. for his helpful advice on high density connectors. This work is supported by Grant-in-Aid for Specially Promoted Resarch of the Ministry of Education, Science and Culture.

References

[1] Nilsson, M. and Tanaka, H.: *Fleng Prolog - The Language which turns Supercomputers into Prolog Machines.* In Wada,E. (Ed.): Proc. Japanese Logic Programming Conference. ICOT, Tokyo, 1986. p 209-216. Proceedings also printed as Springer LNCS 264.

[2] Sakai, S., Koike, H., Tanaka, H. and Motooka, T.: *Interconnection Network with Dynamic Load Balancing Facility,* J. IPS Japan, Vol.27, No.5, 1986, p.518-524 (In Japanese).

[3] Noda, H., Koike, H., Yamauchi, T. and Tanaka, H.: *Parallel Inference Engine Experimental Environment PIEEE - Inference Unit,* Proc. 34th Annual Convention IPS Japan, 4P-6, 1987, (In Japanese).

[4] Yamauchi, T., Koike, H., Noda, H. and Tanaka, H.: *Parallel Inference Engine Experimental Environment PIEEE - Automatic Load Balancing Network,* Proc. 34th Annual Convention IPS Japan, 4P-5, 1987, (In Japanese).

[5] Koike, H. and Tanaka, H.: *Data Allocation Mechanism for Low Access Conflict on PIEEE,* Proc. 35th Annual Convention IPS Japan, 3C-10, 1987, (In Japanese).

[6] Takahashi, E., Koike, H., Yamauchi, T. and Tanaka, H.: *Design of an LSI Switching Unit for the Highly Parallel Inference Engine PIE,* Proc. 36th Annual Convention IPS Japan, 1D-9, 1988, (In Japanese).