The Design of Delay Insensitive Asynchronous 16-bit Microprocessor

Byung-Soo Choi, Dong-Wook Lee and Dong-Ik Lee
Department of Information and Communications
Kwang-Ju Institute of Science and Technology(K-JIST)
572, Sangam-Dong, Kwangsan-ku, Kwangju, 506-712, KOREA
Tel:+82-62-970-2266
Fax:+82-62-970-2204
e-mail:{bschoi,dwlee,dilee}@kjist.ac.kr

Abstract — In recent, asynchronous design has been resurged to exploit potential advantages of asynchronous VLSI such as; high-performance, low power consumption, timing fault tolerance and design cost reduction. This paper describes our first design and implementation of DINAMIK project which aims to show realizability of potential merits of asynchronous VLSI and to establish the design methodology. In the design, ease of design(high modularity) and delay insensitivity was especially emphasized while power consumption, performance and area optimization were ignored as the first stage of the project. To achieve our main purpose simple architecture and a pessimistic delay assumption has been selected. DINAMIK has been fabricated as a SOG using 0.6 μ technology.

I. INTRODUCTION

Advances in semiconductor processing technology and increasing social demands for ultra-high speed information processing caused by emerging multimedia communication technologies require revolutionary change of VLSI design style. In other words, these advances in various technologies force VLSI to exhibit higher speed operation and thus better performance, more functionality, lower power consumption in shorter developing cycle time. Though synchronous VLSI is dominant in current digital circuits, it is nearly impossible to satisfy above requirements due to the existence of global clock. More specifically, while better performance requires higher operation frequency, higher frequency makes the clock distribution more difficult or even impossible. Moreover, the more functionality requires the more transistors in a chip and the bigger area which results in not only more serious clock distribution problem but more clock driving power and hence more power consumption, since clock driving power increases in proportion to the area of a chip. If we adopt a different algorithm in a part of chip or processing technology for better performance or any other reason, whole chip has to be redesigned due to the timing dependency between front-end design and back-end design, in consequence long developing cycle time is required. An asynchronous VLSI that is not driven by a global clock but is driven by transition of signals has been resurrected as an alternative of synchronous ones to overcome above shortcomings. An asynchronous VLSI has the following potential advantages over synchronous counterpart; high-speed operation with ultra high-speed devices, ease of design(high modularity), low power consumption. In recent, many researches have been performed to show the advantages of asynchronous circuits, in universities and advanced companies, AMULET1, 2 and 2e for low power consumption[1], TITAC1 and 2[2], and microprocessors designed in Caltech[3] for high performance are some of examples. A 16-bit microprocessor called DINAMIK(Delay INsensitive Asynchronous MIcroprocessor in K-JIST) has been designed and implemented. The DINAMIK project aims to show the feasibility of asynchronous microprocessor. Section II describes the data transfer methods using dual-rail encoding method and modified four phase protocol. Basic control scheme of DINAMIK, say micropipeline, is explained in section III. Working modules and C-element adopted in DINAMIK are shown in section IV. Architecture, Design and Implementation flow are shown in section V. Evaluation results and conclusions are explained in section VI and VII.

II. DATA TRANSFER METHODS IN ASYNCHRONOUS CIRCUIT

An asynchronous circut can be considered as a set of working modules transferring control information and data each other through not global synchronization but local synchronization. In an asynchronous circuit, to communicate between two modules, 'request' and 'acknowledge' signals are used, so called handshaking protocol, for local synchronization. There are several protocols used in an asynchronous circuit. Two-phase protocol and four-phase protocol are commonly used[4]. Since gates and wires have unbounded delay in a delay insensitive asynchronous circuit, we use dual-rail encoding method in data line and data processing logic[4]. To transfer data between two modules, a combination of modified four-phase protocol and dual-rail encoding method is used in DINAMIK.

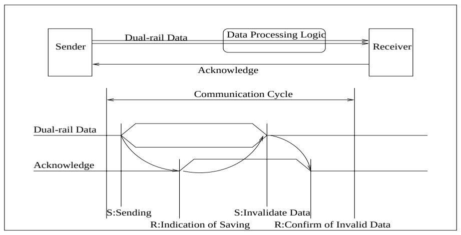

Fig. 1. Modified four-phase protocol

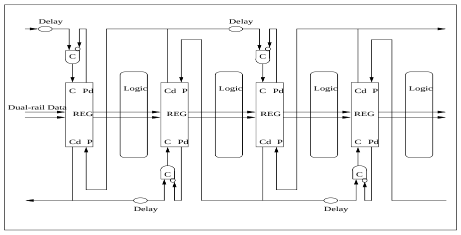

Fig. 2. Micropipeline

A. Dual-Rail Encoding Method

To represent 1-bit data in dual-rail encoding method, two physical lines are used[4]. For example, a logical data, D is represented by two physical data lines, D1 and D0. The following equation shows this encoding scheme;

D = 0 $\Longleftrightarrow$ (D1,D0) = (0,1)

D = 1 $\Longleftrightarrow$ (D1,D0) = (1,0).

In particular, (0,0) represents a space which allows us to identify consecutive 0's or 1's. (1,1) state is not used. Data transferring starts from the (0,0) state. If a state is changed from (D1,D0) = (0,0) to (0,1)/(1,0), which notices the arrival of logical data '0'/'1'.

B. Modified Four-Phase Protocol

Modified four-phase protocol is a combination of four-phase protocol and dual-rail encoding as shown in Figure 1. Four phase protocol consists of *sending data*, *receiving data*, *invalidating data* and *confirming the invalidation of data* phases. In *sending data phase*, a sender loads valid data on dual-rail data lines. These data are processed in the data processing logic between the sender and the receiver. After transmission and processing time, the receiver knows that all the data lines has valid data. At this time, the receiver saves the data in the local storage. When data are completely saved, the receiver sends an acknowledgement signal to the sender using a physical acknowledgement line. This phase is a *receiving data phase*. After receiving of acknowledgement, the sender invalidates the current data loaded in dual-rail data lines. We call this phase as an *invalidating data phase*. And after propagation delay time of data lines, the receiver knows that there is no valid data. At the same time, the receiver sends acknowledgement signal to the sender, as a *confirming the invalidation of data phase*. By this procedure, two working modules can communicate independently with delay variation of data lines and control signals. In DINAMIK *2n+1* wires are required, *2n* for *n*-bit data and *1* for acknowledgement signal.

III. MICROPIPELINE

Though we do not use the micropipeline[5] as it originally proposed, micropipeline gives us basic idea on control mechanism between working modules. Here micropipeline and its difference from control scheme used in DINAMIK will be briefly discussed. In micropipeline it is assumed that upper bounds of wire and gate delays are known and hence maximum local delay between two consecutive working modules can be estimated. Thus local synchronization can be realized by delay padding insertion rather than dual-rail encoding. In micropipeline *2+n* wires are used, *n* wires for *n*-bit data, *1* for request and *1* for acknowledgement signals. Compared with a synchronous pipeline, micropipeline needs local synchronization through handshaking between higher and lower stages. A C-element, described as an AND gate marked 'C' in Figure 2, plays very important role in an asynchronous circuit. If all inputs of C-element is '1'('0'), the output is '1'('0'), respectively. Otherwise the previous output is preserved. In Figure 2, 'C' of each 'REG'(register) represents that 'REG' can save the current input data. 'Cd' means that data capture has been completed. 'P' means that the state of the register is changed to transparent mode so that the register can receive next data. 'Pd' means that the state of the register is changed completely into transparent mode. When input data are valid and value of 'C' is changed, the register saves current input data and notices that to higher and lower stage by changing of 'Cd'. After delay time specified in 'Delay' in Figure 2, lower stage meets the same condition and saves processed data. Lower stage also works with same procedure. Thus, the input value of 'P' of the higher register is changed. Consequently, the higher register goes to transparent mode and prepares to save next new data.

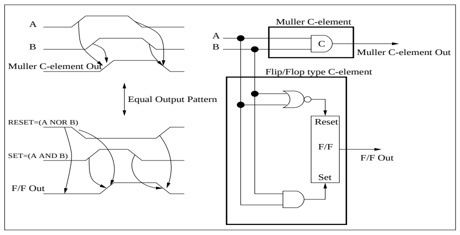

Fig. 3. C-element

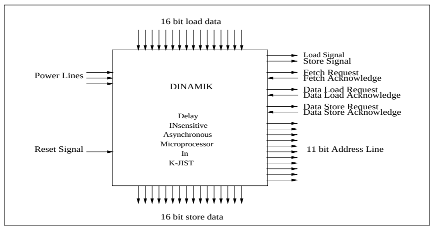

Fig. 4. Pin description of DINAMIK

IV. Working Module and C-element

A working module consists of registers, register control logic and data processing logic. An asynchronous circuit can be divided by several working modules. Each working module has a unique interface protocol. Hence each working module can be designed independently. In merging working modules, simple interfaces among them, actually only wiring, need to be taken into accounts rather than considering complex timing issues for the interface. This feature discloses high modularity hence ease of design, in asynchronous design.

As mentioned before, Muller C-elements are indispensable in asynchronous circuits. However, Muller C-elements were not available in the implementation, we implement these based on R/S Flip/Flop as shown in Figure 3. In Figure 3 , lower part shows Flip/Flop C-elements used in DINAMIK. From timing diagram shown in Figure 3, we know that the output of Flip/Flop type C-element is the same as that of Muller C-element.

V. Architecture, Design and Implementation Flow

Pin description of DINAMIK is shown in Figure 4. All instructions are shown in Table I. DINAMIK uses three operands for each operation. The size of each instruction is 16-bit. Specially, 'HALT' instruction can stop all working modules of DINAMIK. Figure 5 shows the floor-plan of DINAMIK. We used VHDL for logic design. COMPASS ASIC Synthesizer and Gate-level simulator were used for design and simulation respectively. After that, SYNOPSYS Synthesizer tool was used for synthesis with 0.6μ, KG75000 gate library supported by SamSung Electronics. With CADENCE VerilogXL tool we performed simulation. Before layout, we did not know the wire delay information. Thus, we verified DINAMIK with only gate delay information. After gate level design is completed, layout of DINAMIK was performed in SamSung. After layout, we verified DINAMIK with gate and wire delay information. Delay insensitivity allowed DINAMIK to pass the simulation in one time verification with wire delay.

TABLE I
INSTRUCTIONS

Type	Instructions
Data Transfer	MOVI, MOVE, LD, ST
Arithmetic	ADD, ADDC, SUB, SUBB
Logic	AND, OR, NOT, XOR
Shift,Rotate	SHL, SHR, ROL, ROR
Branch	JMP, JZ, ZB, JC, JO
Status	CLC, STC, CLO, STO
Halt	HALT

Characteristics of DINAMIK are shown in in Table II.

VI. Performance Evaluation

The first chip of DINAMIK works correctly with variable supply voltages from 2.0V through 8.0V and temperature from $0°$C through $100°$C. This fact shows that delay insensitivity characteristic of DINAMIK and hence timing fault tolerant with delay variance in layout design, fabrication process and operating environment. Performance of DINAMIK is summarized in Table III. For the performance evaluation we set a testing board shown as Figure 6. During the evaluation, 'MOVI' instruction is loaded on instruction line. Then, instructions loaded on instruction line are always valid. An oscilloscope in Figure 6 checks the cycle time of 'MOVI' instruction. At the same time, the input voltage and temperature are changed to verify the delay insensitivity. DINAMIK shows the 10% performance variation with temperature. Figure 7 shows the demonstration board.

TABLE II
CHARACTERISTICS OF DINAMIK

Process	0.6 μ CMOS, 2-layer, SOG
Die size	5mm * 5mm
Gate Count	52,168

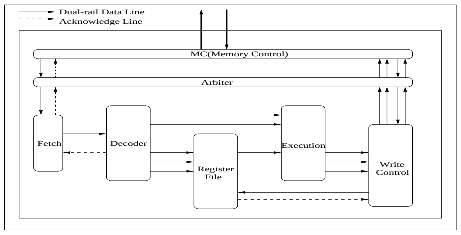

Fig. 5. Floor plan of DINAMIK

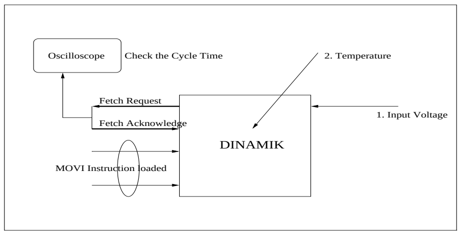

Fig. 6. Performance evaluation method

TABLE III
PERFORMANCE OF DINAMIK

Vol.	I(mA)	C(ns)	MIPS	W(mW)	MIPS/W
2.0	0.03	200	5	60	83.33
3.0	0.06	125	8	180	44.44
4.0	0.11	100	10	440	22.73
5.0	0.15	85	11.76	750	15.68
6.0	0.20	75	13.33	1200	11.11
7.0	0.26	75	13.33	1820	7.32
8.0	0.31	75	13.33	2480	5.37

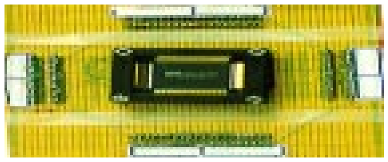

Fig. 7. Demonstration board of DINAMIK

VII. CONCLUSIONS

We have designed and implemented a delay insensitive asynchronous 16-bit microprocessor. Delay insensitivity is shown in design phase(specially from VHDL-level to gate-level and from gate-level to layout level), fabrication phase(difference between expected and actual delay information), and performance evaluation phase(with the variation of the input voltage and temperature). Measured speed and power consumption of DINAMIK is not comparable to synchronous counterparts. This is caused by unreasonably pessimistic delay model, while this delay assumption shows high reliability to timing variance. This observation suggests that there is tradeoff between performance indices and reliability.

REFERENCES

[1] Furber,S.B., Day,P.,Garside,J.D., Paver,N.C. and Woods,J.V., "AMULET1: A Micropipelined ARM," Proceedings of the IEEE Computer Conference, March 1994.

[2] Akihiro Takamura, Masashi Kuwako, Masashi Imai, Taro Fujii, Morokazu Ozawa, Izumi Fukasaku, Yoichro Ueno, and Takashi Nanya, "TITAC-2: An asynchronous 32-bit microprocessor based on Scalable-Delay-Insensitive Model," Proceedings of ICCD'97, pp. 288-294, October 1998.

[3] A.J.Martin, S.M.Burns, T.K.Lee, D.Borkovic, and P.J.Hazewindus, "The Design of an Asynchronous Microprocessor," Advanced research in VLSI; Proceedings of the Decennial Caltech Conference on VLSI, MIT Press, pp.351-373, 1989.

[4] Scott Hauck, "Asynchronous Design Methodologies : An Overview," Proceedings of the IEEE, Vol.83, No.1, pp.69-93, January 1995.

[5] Sutherland,I.E., "Micropipelines," Communications of the ACM, Vol.32, No.6, pp. 720-738. January 1989.