A Compact Array Processor Based on Self–Timed Simultaneous Bidirectional Signalling

Ghassan Y. Yacoub, Tarun Soni and Walter H. Ku
Electrical and Computer Engineering
University of California, San Diego
La Jolla, CA 92093–0407

Abstract **– In this paper a bundled self–timed simultaneous bidirectional signalling protocol is used to remove the clock dependency and minimize latency in the communication network of array processor. The use of the same data bus for bidirectional data transfer effectively doubles the I/O bandwidth of such a communication network. This also permits making the input data transfer cycles independent of the output data transfer cycles, thus decoupling the data and result waves in a wavefront array processor. As a vehicle to demonstrate the merit of this protocol and to compare it to the more conventional WAP protocols, we describe the design of a two dimensional array processing structure for matrix multiplication using such a protocol. A computation element based on a bit–serial multiplier accumulator is chosen with a data dependent computation time. This permits the design of an array where neither the computation nor the communication speeds are bound by any clock speeds. It is shown that such an approach reduces the latency of the array structure without increasing I/O pin count.**

I. Introduction

Two approaches have been proposed for the design of computational arrays [1]. The first is the systolic approach which makes use of the global clock's regularity to compute and communicate synchronously. However, problems like global synchronization led to the development of the second approach namely, wavefront array based processing (WAP), where computation depends on data arrival, thus creating a globally asynchronous system. This approach substituted the timing constraint on the interprocessor communication network with a sequencing constraint, wherein the communication network needs to only maintain a correct sequence of data and need not step in synchronization with any global clock.

However, even in this asynchronous array structure, there is a certain overhead involved in the data flow, with the transfer of data from one processor to another being ruled by an asynchronous transfer protocol and dependence on local clocks, unlike self–timed protocols [3]. Furthermore, the unidirectional bus structure also constrains the communication network to schedule the result output in sequence with the input data since they are both using the same unidirectional network.

In section 2, we describe a self–timed interface and compare it to the conventional WAP interface. We show the dependence of the WAP handshaking logic on the local clock and the absence of any such dependence in the self–timed logic. In section 3, we describe the application of this bidirectional interface to an elementary matrix multiplier array.

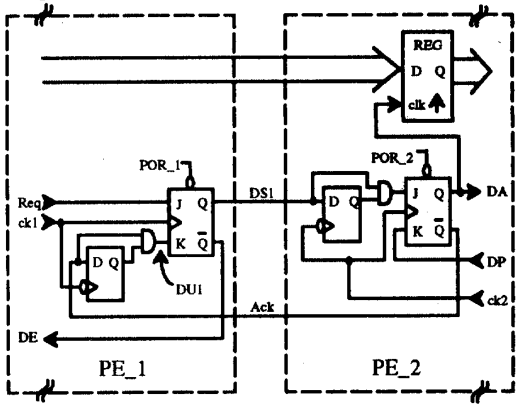

Figure 1a: Conventional interprocessor block handshaking circuit. Nodes Req and Ack correspond to transitions 2 and 3 respectively in Figure 1b.

II. Interprocessor Communication

2.1 Description of WAP Interface

Figure 1a shows a typical neighboring interprocessor I/O communication interface used in a wavefront array processor system [1]. In this case, though the interprocessor communication is not dependent on any global clock it does depend on the local clocks of the two processors. Thus, in the case of data from the processors being ready much faster than the communication bandwidth of this interface, the speed of data transfer is constrained by the latching times shown in Figure 1b as the time interval between transition 2 and transition 3. This time interval, τ_{23} consists of several delay factors. The superscript index on

0-7803-1254-6/93$03.00 © 1993 IEEE

the delay symbol, τ, denotes that the associated delay is triggered off a corresponding indexed clock edge

$$\tau_{23} = \tau^{1}_{jk} + \tau^{1,2}_{ck} + \tau^{2}_{jk} + \tau_{p} \qquad (1)$$

where τ^{1}_{jk} is the JK flip flop delay in PE1, $\tau^{1,2}_{ck}$ is the phase difference between clock1 and clock2, τ^{2}_{jk} is the JK flip flop delay in PE2, and τ_p is the propagation delay along the acknowledge line originating from processor 2. Essentially, τ_{23} consists of the cumulative addition of at least one clock1 period, the clock1 to clock2 phase difference, half a clock2 period, and the propagation delay of the acknowledge signal.

2.2 Description of Proposed Interface

Figure 2a depicts the self–timed simultaneous bidirectional I/O circuit utilized by two processors communicating simultaneously via a common data bus where each wire carries superposed encoded descrete signals traveling in opposite directions [3]. For each transaction the sender transmits new data followed by a transition on the Request line. Each transaction is initiated by a transition on the Acknowledge line and assures data stability when both the Request and Acknowledge controls are in opposite phases. The two main building blocks, the receiver and the transmitter, will be described.

Transmitter Block : This block consists an event–driven storage element controlled by a Muller C–element similar to those in [5] and [6] whose outputs go to current drivers where impedance matching occurs. Digital data is accepted from the left or right sides which can be synchronous interfaces possibly having a stretchable clock as described in [7]. The current drivers and active termination devices serve to match the transmission lines depicted in Figure 2a. If the delay times are τ_0 and τ_h for the data and request signals respectively, then the condition, $\tau_0 \le \tau_h$, must hold (bundling constraint).

Receiver Block : The receiver block contains a sense amplifier which decodes the signals received from the other end as shown in Figure 2a. The digital outputs of these latched amplifiers are stored in the event–driven registers.

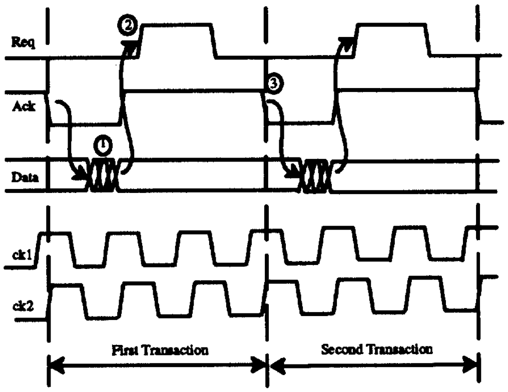

Figure 1b: The time interval between transitions 2 and 3 are dictated by the clock rates inside each of processors 1 and 2.

This receiver–transmitter pair can be configured in two dimensions and can be extended to n–bit wide buses. The inclusion of a 3–input C–element, as shown in Figure 2a, ensures correct dataflow scheduling in both dimensions.

A typical data transfer cycle with such an interface is shown in Figure 2b. Here, the time interval between transition 2

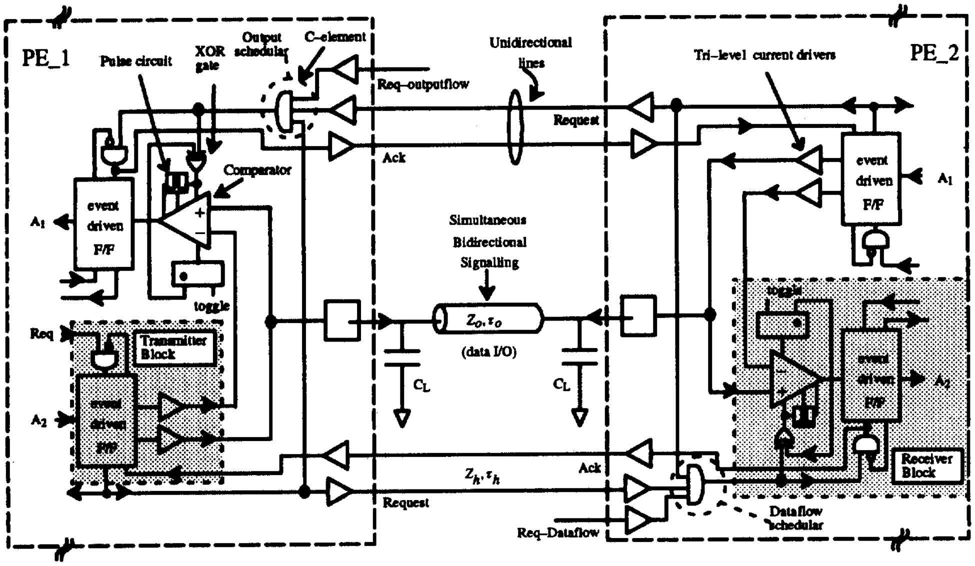

Figure 2a: Architecture of the proposed self–timed simultaneous bidirectional interprocessor I/O circuit used in the design of the compact array processor.

and transition 3, denoted as t_{23}. also consists of several delay factors.

t_{23} can then be expressed as

$$t_{23} = t_{c,1} + 2t_{c,2} + 2t_p \qquad (2)$$

where $t_{c,1}$ is the C–element delay in PE1, $t_{c,2}$ is the C–element delay in PE2 (there are two such gates), and t_p is the interprocessor line delay (Request and Ack). As can be seen, the complete data transfer depends only on the various propagation delays in the data path and has no dependence on the local clocks of the two processing elements.

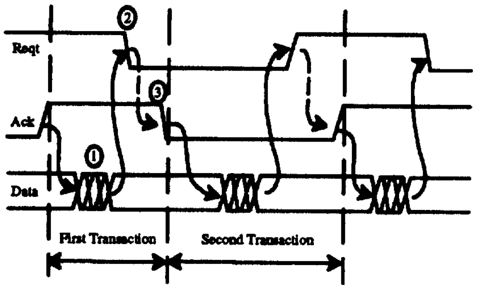

Figure 2b: The time interval between transitions 2 and 3 in the self–timed interface are seen to be independent of the local clocks.

III. Compact Array Processing

3.1 Conventional WAP

For demonstrating the effectiveness of such a protocol we consider the matrix multiplication array used in [2]. The computation element is taken to be a bit–serial multiplier accumulator [4] with control circuitry. The communication modules between processors can be made to be either a WAP or a self–timed simultaneous bidirectional one. If such an array is configured as a square array of N x N processing elements where A={a_{ij}}, B={b_{ij}}, and C=AxB={c_{ij}} are all N x N matrices and the matrix A consists of columns A_i and matrix B of rows B_j, such that the matrix multiplication can be carried out in N recursions, then computing

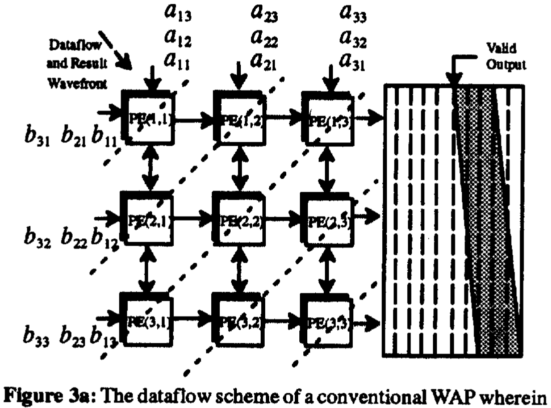

Figure 3a: The dataflow scheme of a conventional WAP wherein the result becomes available after 2N–1 computational cycles.

$$C^{(k)} = C^{(k-1)} + A_kB_k \qquad (3)$$

occurs recursively for k = 1,2, . . . N.

The result of such a recursion starts becoming available at the processing elements after N computation cycles, (where N is the size of the array and consequently the size of the matrix being operated upon). However, in a conventional wavefront array processor, since the result has to travel in the same direction as the input data and on the same data bus, it is available at the output only after (2N–1) computation cycles as can be seen in Figure 3a. The dataflow in such a scheme shows N clock cycles at the output being unused due to the computations being performed and another N–1 cycles being unused due to the output of PE (1,1) travelling across the array and reaching the rightmost output node.

3.2 Reduced Latency WAP

In case of the Two Wave WAP shown in Figure 3b, though the configuration of processors is still the same, the communication network uses the bidirectional protocol described in section 2. A block diagram of the core processing module is depicted in Figure 4 while Figure 5 shows a high–level relationship with the self–timed simultaneous bidirectional I/O interfaces. In this case, since the result is being output in the opposite direction from the input data, the computation result can be made available at the left ports of the array after N computation cycles, thus saving (N–1) cycles of latency and effectively reducing the latency of the array structure by half as can be seen in Figure 3b.

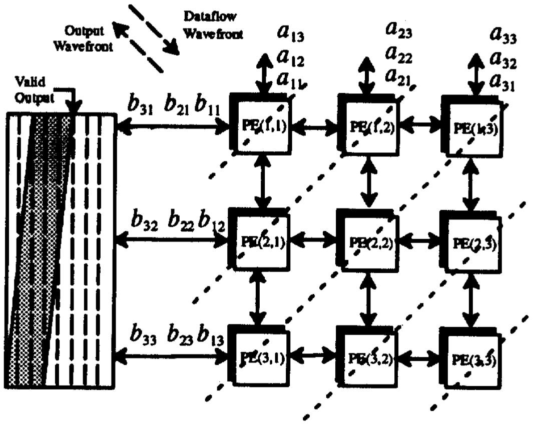

Figure 3b: The dataflow scheme of a Two Wave WAP wherein the result becomes available after N computational cycles.

Another advantage of such a communication scheme is that it maintains the same number of I/O pins as the conventional WAP, therefore effectively doubling the I/O bandwidth. Each PE can flow its result out in the left direction as soon as it is ready since the output wavefront, travelling diagonally from right to left, is independent of the data wavefront travelling diagonally from left to right.

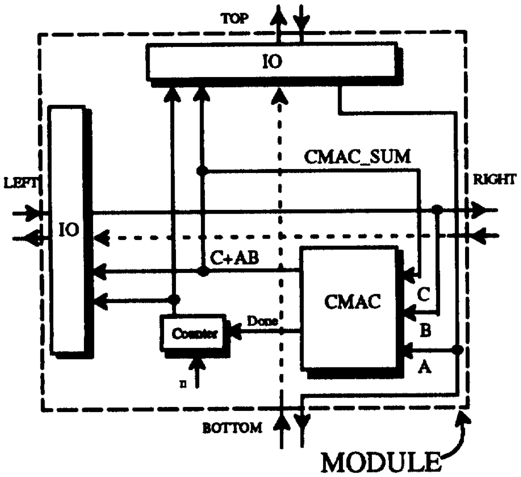

Figure 4: The data–dependent computation module used in a processing element of the array as shown in Figure 5.

3.3 Area–Efficiency of Reduced Latency WAP

The proposed two wave WAP design is based around the complex multiplier chip which is an area efficient bit–serial floating point complex multiplier accumulator [9][10]. This chip has been successfully fabricated and tested. The bit–serial architecture has resulted in more than 100% area reduction (the chip was designed in 1.25 micron CMOS and is 300 mil on a side) over parallel architectures which perform the same task. The trade–off for this area gain has been an increase in the computational latency. However the same bit–serial nature also makes the computational time of the chip data dependent. Part of the data latency delay is also regained by the use of the proposed self–timed simultaneous bidirectional interface.

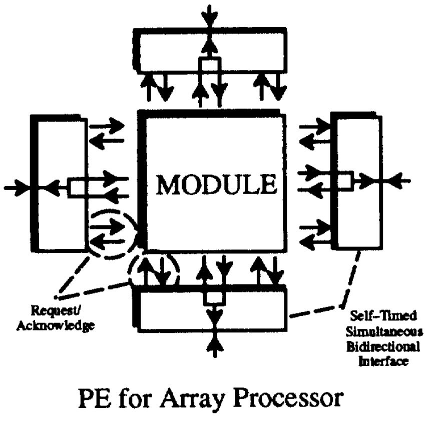

Figure 5: The simultaneous self–timed bidirectional communication is controlled by interface modules and is transparent to the computational element.

V. Conclusion

The use of the same data bus for simultaneous bidirectional data transfer was shown to effectively double the I/O efficiency of an interprocessor communication network. The bidirectional nature of such an interface protocol was shown to make the input data transfer independent of the output data transfer in wavefront array processors, thus decoupling the data and result waves in such architectures. A compact array matrix multiplier design was described using a two dimensional array processing structure. A computation element based on a bit–serial multiplier accumulator was used with a data dependent computation time. Such a computation element coupled with a self–timed interface permits the design of an array where neither the computation nor the communication speeds are bound by any clock speeds. It was shown that the latency of such an array can be reduced by a factor of two by the use of a bidirectional two wave WAP. The conventional WAP and the two wave WAP results were validated through behavioral simulations.

References

[1] S. Y. Kung, VLSI Array Processors, *Prentice Hall,* New Jersey, 1988.

[2] G. M. Jacobs, "Self–Timed Integrated Circuits for Digital Signal Processing," *Ph. D. Thesis, University of California, Berkeley,* UCB/ERL Memorandum No. M89/128, November 30, 1989.

[3] G. Y. Yacoub and W. H. Ku, "Self–Timed Simultaneous Bidirectional Signalling for IC Systems," *Proceedings of IEEE International Symposium on Circuits and Systems,* pp. 2957–2960, May 1992.

[4] P. Denyer and D. Renshaw, VLSI Signal Processing: A Bit–Serial Approach, *Addison–Wesley,* Reading, 1985.

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

[6] I. E. Sutherland, R. F. Sproull and I. Jones, Standard Asynchronous Modules, *Technical Memo 4662,* Sutherland, Sproull and Associates, 1986.

[7] D. M. Chapiro, "Globally–Asynchronous, Locally–Synchronous Systems," *Ph.D. Dissertation, Computer Science Department, STAN–CS–1026, Stanford University,* October 1984.

[8] H. B. Bakoglu, *Circuits, Interconnections, and Packaging for VLSI,* Addison–Wesley, Reading, Massachusetts, 1990.

[9] K. C. Chew, "High Performance Adaptive and Non-adaptive Digital Signal Processors," *Ph. D. Thesis, University of California at San Diego,* La Jolla, 1991.

[10] K. C. Chew, W. H. Ku and J. A. Eldon "Applications of a Bit–Serial Floating–Point Complex Multiplier–Accumulator for High–Speed Digital Signal Processing," *22nd Asilomar Conference on Signals, Systems and Computers,* Asilomar, CA, 1988.