A Wave-Pipelined Router Architecture Using Ternary Associative Memory

José G. Delgado-Frias* State University of New York
Jabulani Nyathi* State University of New York
Laxmi Bhuyan† Texas A & M University

Abstract

In this paper a wave-pipelining scheme is used to increase the performance of a router architecture. Wave-pipelining has a potential of significantly reducing clock cycle time and power. The design approach considered in this paper allows the propagation of data from stage to stage to occur without the use of intermediate latches. Control signals are used to ensure that intermixing of *data waves* does not occur. The results of the study show that wave-pipelining helps to reduce the clock period.

1 INTRODUCTION

A critical component in the new class of computing and communication systems is the router which receives, forwards, and delivers messages. The router system transfers messages based on a routing algorithm which is a crucial element of any communication network [1, 2]. Given the reconfiguring requirements for many computing systems, a number of routing algorithms as well as network topologies must be supported. This in turn leads to a need for a very high performance flexible router to support these requirements. To maximize machine's overall performance and to accommodate reconfiguration in a distributed environment requires matching the application characteristics with a suitable routing algorithm and topology. It has been shown that machine performance is fairly sensitive to the routing function and traffic [3]. In addition, having the capability of changing the routing algorithm at run time could lead to facilitate smart interconnects and adaptability that allow changes on the machine's topology for different applications.

The major approaches to the realization of flexible routers are: dedicated processors and look-up table [4]. A dedicated processor allows bit manipulation that most routing algorithms require to be easily executed. This approach however results in slow routers due to the sequential execution of the routing algorithms. The lookup table approach requires storing in memory a predetermined routing path for all possible destination nodes; the time for determining the output port is a memory access delay. However, large tables are required to store the routing information. The bit-pattern approach uses an associative approach to determine in parallel the potential output ports (according to the routing algorithm). The output port is selected by means of a selection function. This scheme has been shown to be able to accommodate a large number of routing algorithms and to have fast execution [5, 6]. This scheme has been mapped onto a pipelined VLSI architecture [4].

The use of latches in pipelines limits the reduction of the clock period due to the additional time during which the latch is sensitive to any change in the input data. In any pipeline system, the clock period is set by the longest delay produced by the combinatorial logic and wiring paths between input and output latches of a given data path pipeline stage [7]. These latches need be synchronized using a clock; as clock frequencies increase clock distribution becomes a more difficult task. Wave-pipelining offers the potential for high performance, since its maximum clock frequencies are limited by the path delay differences as opposed to the longest path delays [7, 8]. This approach provides a method for significantly reducing clock cycle time, silicon real estate, power, and latency. The waves of electrical signals are used to realize a pipeline [9]. Since there are delays due to the logic and wiring paths, these delays are used to isolate one pipeline stage from the adjacent ones. Wave-pipelining could potentially increase the pipeline rate without using additional registers (latches).

In this paper, a novel router approach is presented; this approach provides flexibility to execute a number of routing algorithms as well as extremely fast execution. This paper is organized as follows: Section 2 provides a description of the router architecture and some of the major circuits that form the router modules. In section 3 we discuss the wave-pipelining approach. The signals of importance within the systems' operation are presented. This section also includes a description of how the signals and data propagate through the system. Some concluding remarks appear in Section 4.

2 BIT-PATTERN ASSOCIATIVE ROUTER SCHEME

The bit-pattern associative router scheme supports intrinsic routing algorithms used in most router schemes. In intrinsic routing the network topological characteristics are inherent in the network addressing scheme. The approach is explained in detail in [5].

2.1 VLSI Organization

A novel dynamic content addressable memory (DCAM) cell has been designed as part of the bit- pattern router VLSI implementation [4]. The DCAM design includes per entry unique bit masking to provide the parallel evaluation of all the bit-patterns on the input data. The design uses dynamic storage cells to hold the matching data condition. This condition requires a ternary digit (with values: 0, 1, and X-don't care) per comparison bit. The refreshing requirement can be made transparent by interleaving the refresh operation with the match operation to maintain a high level of performance. The hardware resources are shared at different times during a clock cycle in a non-conflicting manner. The resulting DCAM cell is significantly smaller than the static cell while maintaining the same level of performance and functionality. The organization of the pipelined router is shown in Figure 1. The CAM cells perform the bit-comparison between the stored patterns and the argument (destination address, network status and algorithm parameters) sent by the search argument register (SAR). Once a comparison with all the stored patterns has been performed the condition of the match lines is latched; this accommodates the requirements for a pipeline. If more than one match line has been selected, the selection function circuitry allows only one selected line to pass to the port assignment memory (RAM cells). Then, the selected row of RAM cells is read and passed to the port assignment register.

*Department of Electrical Engineering, State University of New York,Binghamton, NY 13902-6000, delgado@binghamton.edu

†Department of Computer Science, Texas A & M University, College Station, TX 77843

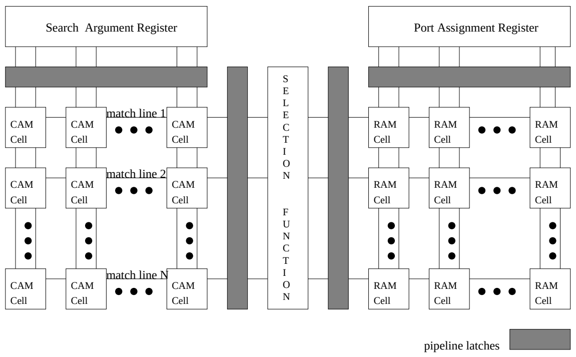

Figure 1: Bit-pattern router organization.

In order to provide programmability and refresh to the dynamic CAM and RAM cells, two sub-systems row select shift register and refresh register (not shown in figure 1) are used. The row select shift register generates signals (row 1...row n) to select a row in the CAM array to either read from or write to. This pointer is used for both programming and refreshing modes. In the refreshing mode the refresh register holds and passes back data to the proper row. The following subsections explain the major components of this organization.

2.2 Dynamic Content Addressable Memory Cell

The DCAM cell implements a comparison between an input and the ternary digit condition stored in the cell. The proposed cell has a small transistor count. The circuit of a single DCAM cell, shown in Figure 2 , consists of six and half transistors; transistor T_e is shared by two cells. The read (denoted as $r\&x$ in Figure 2), write, evaluation and match line signals are shared by the cells in a word while the BIT and NBIT lines are shared by the corresponding bit in all words of the matching unit. The design uses a precharged match line to allow fast as well as simple evaluation of the match condition. This condition is represented by the DCAM cell state which is set by $Sb1$ and $Sb0$ node status. This state is stored in the gate capacitance of the transistors T_{c1} and T_{c0}. Representation of stored data in a DCAM ($Sb0$ and $Sb1$) is as follows: $00 = don't\ care$, $01 = 1$, $10 = 0$ and $11 = not\ allowed$. The DCAM cell performs a match between its stored ternary value and a bit input (provided by BIT and NBIT which represent a bit and its opposite value, respectively). The match line is precharged to “1”; thus, when a no match exists this line is discharged by means of transistor T_m. To read the contents of the DCAM cell transistor T_{rx} is set on. If the stored bit at $Sb1$ (or $Sb0$) is a “1” then the BIT line (or NBIT line) will be set to “0”. If both $Sb1$ and $Sb0$ are “0” (don’t care condition) transistor T_{rx} sets a “0” at the gate of T_m; this in turn serves as refresh for the don’t care condition at transistor T_m. Match operation mode involves comparing the input data to the patterns stored in the DCAM and determining if a match has been found. During match operation, the input data to be compared with the stored data is presented on the BIT line and its inverse value on the NBIT line. Before the actual matching of these two values is performed, the match line is precharged to “1” which indicates a match condition. The matching of the input data and the stored data is performed by means of an exclusive-or operation which is implemented by the two transistors (T_{c1} and T_{c0}) that hold the stored value. The refresh operation occurs at a portion of the clock cycle when the BIT and NBIT lines are not used for the match operation. This makes the refresh operation transparent to the match operation.

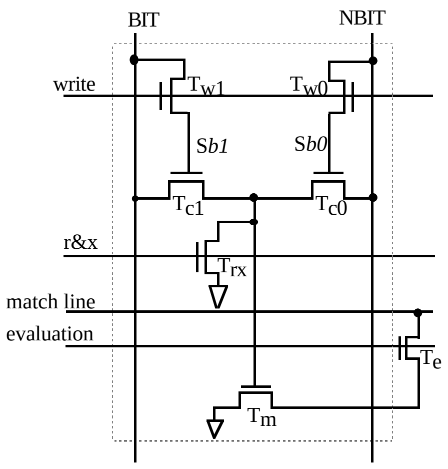

Figure 2: CMOS circuit for a DCAM cell.

2.3 Selection Function (Priority Encoder) and Port Assignment Memory

The priority function should be designed to ensure the deterministic execution of the routing algorithms. With a given input (i.e. destination address) and a set of patterns stored in the matching unit, the port assignment should always be the same. The priority allows only the highest priority pattern that matches the current input to pass on to the port assignment memory. The encoded priority (EP_i) output depends on the match (M_i) at the current bit-pattern and the priority (P_i) for this row. If both match and priority are “1”; then the encoded priority is true. The priority (P_i) is “1” when no other row with higher priority has had a match. Assuming that high priority is given to rows with low numbers; this priority function can be expressed as $P_i = \overline{M}_{i-1} \cdot P_{i-1}$. It should be pointed out that determining dynamically the priority can be seen as a problem where the priority is propagated or disabled. When M_i is “0”, P_i is propagated to the next cell (P_{i+1}); on the other hand, if M_i is “1”, P_i is disabled and P_{i+1} is set to “0”. Using the priority lookahead scheme, the delay in a sequential approach is drastically reduced. The priority lookahead scheme has been proposed and implemented; it has been reported in [10].

The port assignment memory holds information about the output port that has to be assigned after a bit-pattern that matches the current input is found. The selected row address is passed from the priority encoder. The cells in this row are read and their data is latched in the port assignment register. This structure has a hidden refresh approach which is accomplished by means of a DRAM refresh and the row select shift registers. The DRAM structure is able to perform an OR function per column when multiple RAM rows are selected; this is when multiple matches are passed on. The DRAM structure is fully explained in [4].

3 WAVE-PIPELINED VLSI ARCHITECTURE

In order to increase the performance of the proposed associative router, a wave-pipelined architecture is studied and simulated. Wave-pipelining has the potential of significantly reducing the clock cycle time, the area associated with inter-stage latches, power, and latency [7, 9]. As clock rates continue increasing, clock distribution is becoming a major problem; wave-pipelining could help to alleviate it by significantly reducing the clock load as well as the number of inter-stage latches [8].

3.1 Wave-Pipelining Requirements and Circuits

Wave-pipelining involves propagation of data from the input register to the output register without the use of intermediate latches. Wave-pipelining allows for several unrelated *data waves* to be within the system, but at different stages of the system. Removal of intermediate latches introduces the possibility that two or more *data waves* can be intermixed. A wave-pipelined design must then ensure that this intermixing of data does not occur. We have designed circuitry that ensures that data propagates within its own wave. The signals from these circuits allow data to ripple to the next stage without any intermixing. The first signal of importance is the signal used to precharge the match line.

The requirement for precharging the match line is that this occurs before evaluation takes place and after the output of the match line has been passed to the next stage. We, therefore, design the precharge signal using the signals used to evaluate and pass the match line output to the next stage. At evaluation time it is required that the match line precharge signal be at “1”, to prevent a conflict of operations on the match line (attempting to precharge the match line while evaluating it). Also care must be taken to prevent precharging the match line before its output is passed to the next stage (the hit and priority logic -HPL). The match line precharge signal, therefore, has to be low only when both the evaluate and pass signals are at “0”.

The evaluate signal must go to “1” after the data passed to the DCAM for comparison has stabilized on the bit lines (BIT and NBIT). There is circuitry that generates this signal once all the lines have been set to their appropriate input values. This circuitry takes into account setting a “0” or “1” in the particular CAM array and is shown in Figure 3. Its operation is governed by the status of the match line. The inverse of the clock gets passed to an intermediate node when the the match line is at “1” and this intermediate node gets precharged to “1” whenever the match line is at “0”. The inverse of the intermediate node is the evaluate signal. Using the match line to pass the clock signal and set the evaluate signal to “0” when evaluation is completed provides the proper duration for the evaluate signal to stay at “0” or “1”. Once evaluation has been completed the match line outputs are passed to the HPL. Thus, the pass signal must immediately be active following completion of the evaluation process. The circuit used to generate the pass signal is shown in Figure 4. The pass signal is designed to mimic the path that a zero on the match line (non-matching condition) takes once evaluation completes. Passing a “0” to the HPL provides the maximum delay that can be experienced in passing the matching unit’s outputs to the HPL and, therefore, constitutes the worst case propagation delay for this operation. The node pass is initialized to “1” by precharging node z when the evaluate signal is “1”. A “1” at the pass node ensures that transistor T_{pz} passes a “0” which gets inverted and becomes input to transistor T_{pd}. When transistor T_{pd} conducts discharging the pass node, this becomes an indication that the pass signal has stayed at “1” long enough to pass the output of the match line to the HPL. The duration and voltage level of both the evaluate and pass signals need be just long and high enough to accomplish their purpose. The circuits in Figures 3 and 4 have been designed to sense the duration and voltage levels of these signals.

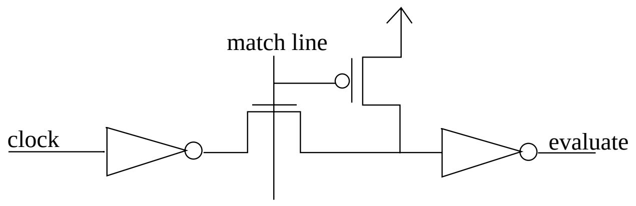

Figure 3: Evaluate signal circuit.

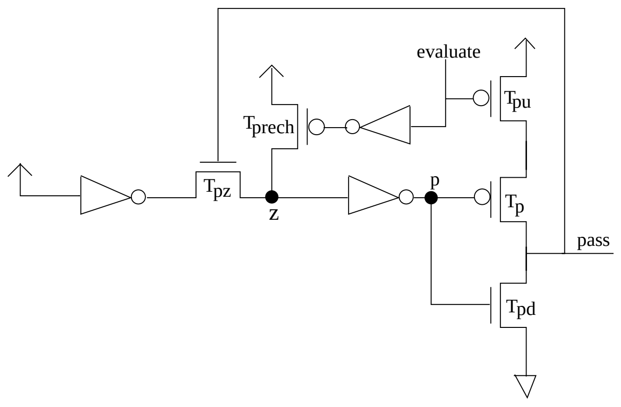

Figure 4: Pass signal circuit.

3.2 Wave-Pipelining Signals

The signals generated by the above circuits appear in Figure 5 along with the clock. Once the output of the match lines have been received by the hit and priority logic (HPL) a decision needs be made when more than one matching condition has been received. The HPL is designed to propagate a priority status (P_i) to the entry below it indicating whether it has registered a match. Some of the signals of importance in this scheme are shown in Figure 6. The selection function’s critical operation occurs when the first and last entries of the HPL simultaneously receive inputs indicating that matching conditions have been found in the corresponding matching unit entries. The first entry of the HPL has to propagate a “0” to the last entry of the HPL, to prevent generation of a pointer to the DRAM by the last entry. In Figure 6 we show signals used to enable the DRAM pointers for the first and last entries from a setup in which the last entry always finds a match and the first entry finds a match every other clock cycle. We show the delays involved. The plots in Figure 7 are those of the DRAM pointers to the first and last entries of the DRAM. They serve to show that an output port assignment can be read from the DRAM array every one and a half clock cycles.

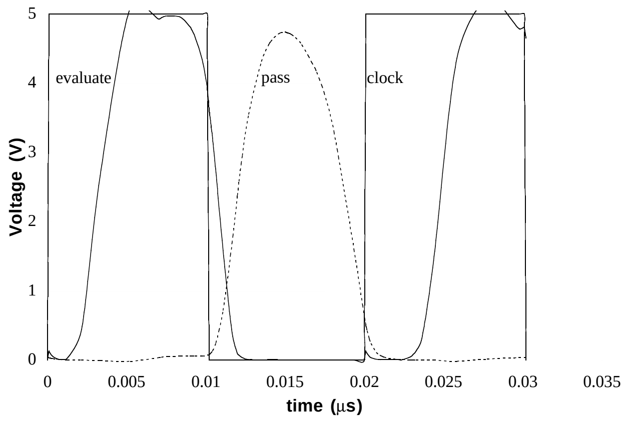

Figure 5: Plot of the matching unit control signals.

4 CONCLUDING REMARKS

An extremely fast flexible router will be very useful in the new class of computing/communication systems, since it is able to accommo-

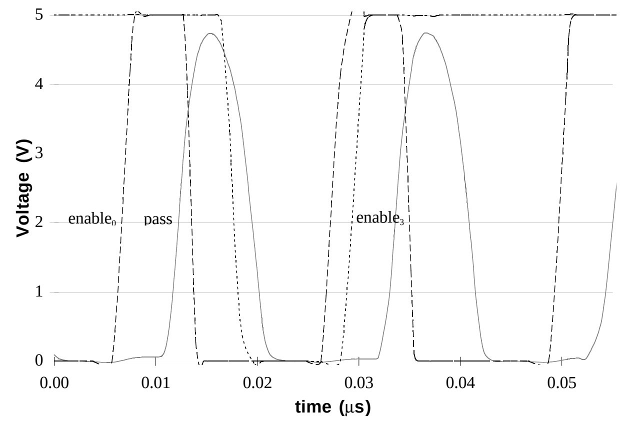

Figure 6: Plots of the enable signals.

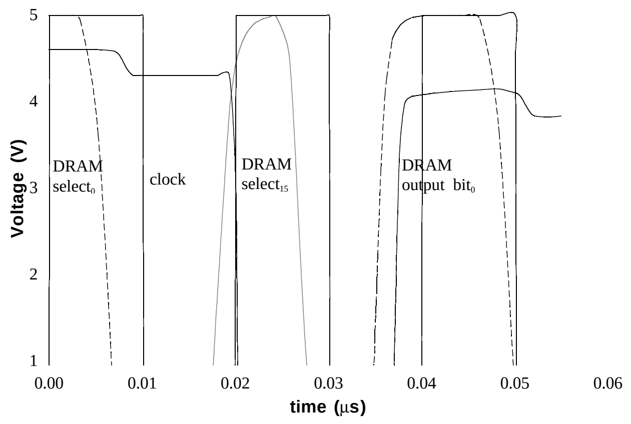

Figure 7: Plot of the DRAM pointers.

date a number of routing algorithms and networks. The execution of the routing algorithm is critical in all the network communications because messages or packets cannot be sent until the destination is determined. To realize smart interconnects it is necessary to execute different algorithms that change due to load of the network or the application being run. Thus, there is a great need for an extremely high performance reconfigurable router such as the one described in this study. In this paper we have presented a novel wave-pipelined router that is capable of executing routing algorithms in one and a half clock cycles. As mentioned earlier, wave-pipelining has the potential for increasing further the performance of computer systems [7, 9]. The first prototype shown in Figure 8 has been realized in 2μm technology. Issues such as signal generation, uneven delays, and timing have been addressed to maintain the pipeline. This is one of the first studies that attempts to address these issues using an implementation of a somewhat complex pipelined architecture as an exploratory means to gain knowledge in this field.

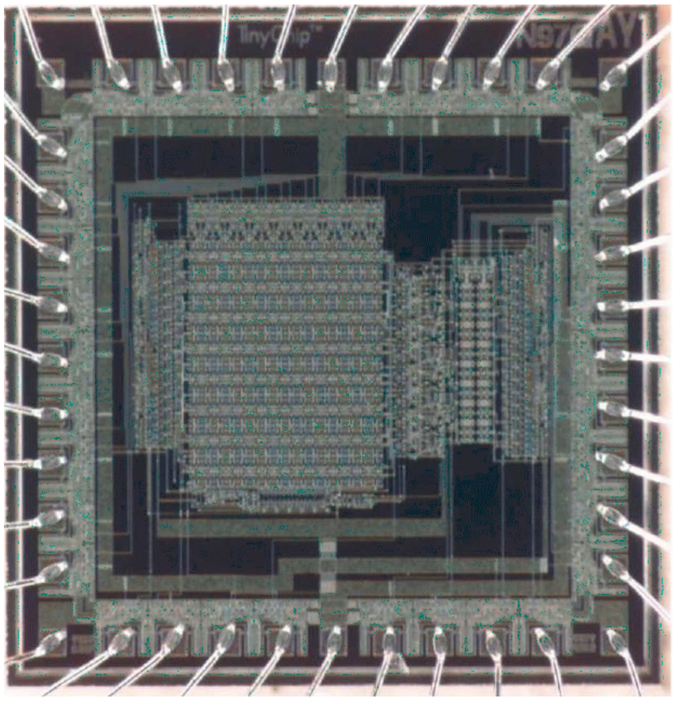

Figure 8: The router chip

References

[1] D. Clark and J. Pasquale, "Strategic Directions in Networks and Telecommunications," *ACM Computing Surveys,* vol. 28, no. 4, pp. 679-690, Dec. 1996.

[2] T. Leighton, "Average Case Analysis of Greedy Routing Algorithms on Arrays," *2nd Annual ACM Symposium on Parallel Algorithms and Architectures,* pp. 2-10, 1990.

[3] R. Cypher, A. Ho, S. Konstantinidou, and P. Messina, "Architectural Requirements of Parallel Scientific Applications with Explicit Communication," *20th International Symposium on Computer Architecture,* pp. 2-13, May 1993.

[4] J. G. Delgado-Frias, J. Nyathi and D. H. Summerville, "A Programmable Dynamic Interconnection Network Router with Hidden Refresh" *IEEE Trans. on Circuits and Systems, I:* vol. 45, no.11, pp. 1182-1190 Nov. 1998.

[5] D. H. Summerville, J. G. Delgado-Frias, and S. Vassiliadis, "Executing Tree Routing Algorithms on a High-Performance Pattern Associative Router," *Journal of Systems Architectures,* vol. 44, no. 11, pp. 849-866, August 1998.

[6] D. H. Summerville, J. G. Delgado-Frias, and S. Vassiliadis, "A Flexible Bit- Associative Router for Interconnection Networks," *IEEE Trans. on Parallel and Distributed Systems,* vol. 7, no. 5, pp. 477-485, May 1996.

[7] W. P. Burleson, M. Ciesielski, F. Klass, and W. Liu, "Wave-Pipelining: A Tutorial and Research Survey," *IEEE Trans. on VLSI Systems,* vol. 6, no. 3, pp. 464-474, September 1998.

[8] C. T. Gray, W. Liu, R. K. Cavin, III, "Timing Constraints for Wave-Pipelined Systems," *IEEE Trans. on Computer-Aided Design,* vol. 13, no. 8, pp. 987-1004, August 1994.

[9] W. K. C. Lam, R. K. Brayton, and A. L. Sangiovanni-Vincentelli, "Valid Clock Frequencies and Their Computation in Wave-pipelined Circuits," *IEEE Trans. on Computer-Aided Design,* vol. 15, no. 7, pp. 791-807, July 1996.

[10] J. G. Delgado-Frias and J. Nyathi, "A High Performance Encoder with Priority Lookahead," *IEEE Sixth Great Lakes Symp on VLSI,* pp. 59-64, Lafayette, Louisiana, Feb. 1998.