A Hybrid Wave-Pipelined Network Router

José G. Delgado-Frias
Department of Electrical and Computer Engineering
University of Virginia
Charlottesville, VA 22904-4743
delgado@virginia.edu

Jabulani Nyathi
Department of Electrical Engineering
State University of New York
Binghamton, NY 13902-6000
jabu@binghamton.edu

Abstract

In this paper a novel hybrid wave-pipelined bit-pattern associative router is presented. A router is an important component in communication network systems. The bit-pattern associative router (BPAR) allows for flexibility and can accommodate a large number of routing algorithms. Wave-pipelining is a high performance approach which implements pipelining in logic without using intermediate registers. In this study a hybrid wave-pipelined approach has been proposed and implemented. Hybrid wave-pipelining allows for the reduction of the delay difference between the maximum and minimum delays by narrowing the gap between each stage of the system. This approach yields narrow "computing cones" that allow faster clocks to be run.

This is the first study in wave-pipelining that deals with a system that has a substantially different set of pipeline stages. The bit-pattern associative router has three stages: condition match, selection function, and port assignment. Each stage's data delay paths are tightly controlled to optimize the proper propagation of signals. The simulation results show that using hybrid wave-pipelining significantly reduces the clock period and circuit delays become the limiting factor, preventing further clock cycle time reduction.

1 Introduction

Communication channels between any two nodes regardless of their physical location within a communication network system can be established by using routers at each node. The purpose of the router would be to receive, forward, and deliver messages. The router system transfers messages based on a routing algorithm which is a crucial element of any communication network [1]. Given the reconfiguring requirements for many computing systems, a number of routing algorithms as well as network topologies must be supported. This leads to a need for a very high performance flexible router to support these requirements. To maximize a machine's overall performance and to accommodate reconfiguration in a distributed environment requires matching the application characteristics with a suitable routing algorithm and topology. Having the capability of changing the routing algorithm at run time could facilitate smart interconnects and adaptability that allow changes on the machine's topology for different applications.

A router intended to be used for a number of routing algorithms and/or network topologies has to be able to accommodate a number of routing requirements. It is of great importance that the routing algorithm execution time be extremely short. This time dictates how fast a message can advance through the network, since a message cannot be transferred until an output port has been selected by the routing algorithm. Thus, the routing algorithm execution time must be reduced to decrease message delays. Other requirements may include: flexibility to accommodate modifications to a network, algorithm and/or topology switching with minimum delay, and programmability to support a large number of routing algorithms and network topologies.

In this paper we present a high performance hybrid wave-pipelined VLSI router that constitutes of several modules and uses dynamic circuitry within these modules. Wave-pipelining is a design method that enables pipelining in logic without the use of intermediate registers [2]. In order to realize practical systems using wave-pipelining, it is a requirement that accurate system level and circuit level timing analysis be done. At system level, generalized timing constraints for proper clocking and system optimization need to be considered. At the circuit level, performance is determined by the maximum circuit delay difference in propagating signals within a given module of the system. Accurate analysis and strict control of these delays are required in the study of worst case delay paths of circuits. Data dependent delays also present a problem that needs to be considered in the use of wave-pipelining. In this study, *hybrid wave-pipelining*; a different approach to minimizing the clock period is undertaken.

Section 2 provides an overview of the bit-pattern associative router, describing the router operation and the circuits that form the basic blocks of each module. In Section 3 we describe hybrid wave-pipelining and demonstrate how the delay differences are narrowed, resulting in clock period reduction. Some concluding remarks appear in Section 4.

2 Router Functional Organization

The bit-pattern associative router (BPAR) scheme supports the execution of routing algorithms that are used in most communication switches. The associative router uses a content addressable memory as its bit-pattern associative unit, and this enables the destination address alternatives to be considered in parallel. The destination address is presented as the input to the bit-pattern associative unit for comparison with the stored data. The patterns stored in the bit-pattern associative unit allow the router to make a decision about the destination port based on the routing algorithm. The address of a destination node D $(d_{n-1}, \ldots d_0)$ needs to be compared to the current node's address C $(c_{n-1}, \ldots c_0)$. A routing algorithm compares the bits of the two addresses; some bits are ignored since they do not affect the current routing decision. These "don't care" bits usually occur at different positions for each potential path being considered and need to be customized according to the routing algorithm requirements. To provide the flexibility required to support multiple interconnection networks and routing algorithms the bit-pattern associative router must be programmable. The results of the comparison are passed to the selection function, which then passes the match output with the highest priority to the port assignment to select the word corresponding to the selected output port. Figure 1 shows the bit-pattern associative router organization.

The BPAR supports three basic operation modes: Normal or matching, programming or data loading and refreshing [3]. In normal mode the previously loaded data is compared to data presented at the search argument register and the results passed to the selection function. The programming mode initializes the memories with destination addresses, while refreshing enables replenishing of the loaded data during the normal operation.

2.1 Dynamic CAM Cell (DCAM)

The DCAM cell is the basic building block of the bit-pattern associative unit array. It implements a comparison between an input and the ternary digit condition stored in the cell. A single DCAM cell shown in Figure 2, consists of eight and a half transistors; transistor T_e is shared by two cells. The read, write, evaluation, and match line signals are shared by the cells in a word while the BIT_STORE, NBIT_STORE, BIT_COMPARE

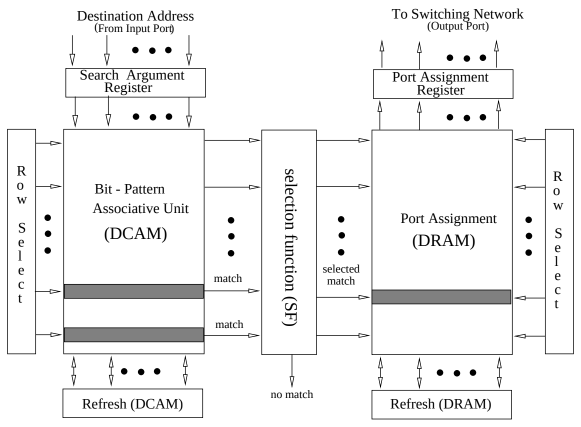


Figure 1. The bit-pattern associative router organization.

and NBIT_COMPARE lines are shared by the corresponding bit in all words of the matching unit. The design uses a precharged match line to allow fast and simple evaluation of the match condition.

In Table 1 the possible DCAM cell stored values are listed. Transistors T_{ref0} and T_{ref1} are used for refreshing. Normal operation 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 is presented on the BIT_COMPARE line and its inverse value on the NBIT_COMPARE 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 implemented by transistors T_{c1} and T_{c0} whose gates hold the stored value. The match line is discharged through a series transistor pair (T_m and T_e) and a logic 0 on the match line indicates a non matching condition while a logic 1 indicates a matching condition.

Table 1. Representation of stored data in a DCAM cell.

$Sb1$	$Sb0$	state
0	0	X(don't care
0	1	1(one)
1	0	0(zero)
1	1	not allowed

2.2 Selection Function and Port Assignment

The selection function should be designed to ensure the deterministic execution of the routing algorithms. For a given input (i.e. destination address) and a set of patterns

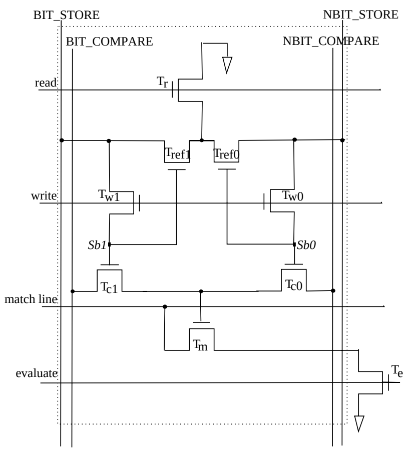


Figure 2. CMOS circuit for a DCAM cell.

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 at the current bit-pattern and the priority for this row. If both match and priority are "1"; then the encoded priority is true. A priority lookahead scheme has been proposed and implemented; it has been reported in [5]. The port assignment memory or RAM holds information about the output port that has to be assigned after a bit-pattern that matches the current input is found. This memory is proposed to be implemented using a dynamic approach. The selected row address is passed from the priority encoder, the cells in this row read and their data latched in the port assignment register. 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 explained in [3].

3 Hybrid Wave-Pipelining

In this section we outline some of the challenges of wave-pipelining first and then describe the timing constraints for the proposed hybrid wave-pipelining approach. Conventional circuit pipelining uses intermediate latches in addition to the input and output registers. Intermediate latches (registers) ensure that when the leading edge of the system clock comes data gets propagated from one stage to the next in a synchronous manner. In a system setup like this there is only one set of data between register stages. Wave-pipelining is an approach aimed to achieve high-performance in pipelined digital systems by removing intermediate latches or registers [4]. Idle time of individual logic gates within combinational logic blocks can be minimized using wave-pipelining.

Some of the challenges of designing wave-pipelined systems are: *Preventing data collision;* there must be no data overrun in each circuit block, and it must be ensured that there is no over committing of the data path. *Designing dedicated control circuitry;* control logic circuits must be designed to operate synchronously with the circuitry of the pipeline stages. *Balancing delay paths;* delay paths must be controlled or equalized to reduce major discrepancies or differences between maximum and minimum delays [4]. The requirements stated above are not inclusive but represent some of the most important design issues in wave-pipelining. Timing constraints for the proposed hybrid wave-pipelining approach are derived in the same fashion as in [4]. In many computer/digital systems each stage has a significantly different function and circuitry; wide variations in delays (D_{min} and D_{max}) may not be tolerated. A common engineering practice is to consider the worst case delay (D_{max}), to ensure that the system runs properly. D_{max} plays a very important role in the system's performance and safe regions of operation. D_{min} (the shortest delay path), on the other hand, gives information about when the results will begin to emerge.

The equations derived for the hybrid wave-pipelining are denoted by the subscript h. To derive the equations that describe the timing constraints for the hybrid wave-pipeline, the temporal/spatial diagram representing this scheme is presented first. The shaded regions of Figure 3 indicate that data is not stable, therefore, register outputs cannot be sampled. The computational cones in this diagram have been arranged to represent each stage within the design. We define some of the variables appearing on the figure. D_{min} and D_{max} are the minimum and maximum propagation delays through the stages with T_{clk} defining the clock period. T_s and T_h are the register setup and hold times, Δ refers to the constructive clock skew while Δ_{clk} is the register's worst case uncontrolled clock skew. D_R is the register's propagation delay with $d_{min}(n)$ being the minimum delay encountered in propagating data within a single stage n and D_{min_hold}, the overall minimum delay of all the stages.

The time it takes for data to emerge at the output register after N clock cycles is T_L and it is given by:

$$T_{L(h)} = NT_{clk} + \Delta \tag{1}$$

Clocking the earliest data associated with $wave_i$ requires the following condition:

$$T_{L(h)} < T_{clk} + D_R + D_{min\_hold} - (\Delta_{clk} + T_h) \tag{2}$$

where $D_{min\_hold} = d_{min}(0) + d_{hold}(0) + d_{min}(1) + d_{hold}(1) + d_{min}(2) + d_{hold}(2)$
This equation takes into consideration the intermediate stages of the design. From the above equation it can be determined that $D_{min\_hold} \geq D_{min}$ implying that this delay difference is less than $D_{max} - D_{min}$ of the wave-pipelining

scheme. The latest possible time at which data associated with $wave_i$ can be clocked is given by:

$$T_{L(h)} > D_R + D_{max} + T_s + \Delta_{clk} \quad (3)$$

The clock period for the hybrid approach is determined to be:

$$T_{clk(h)} > (D_{max} - D_{min\_hold}) + T_s + T_h + 2\Delta_{clk} \quad (4)$$

The hybrid wave-pipelined approach allows for the clock signal's period to be reduced, hence an increase in performance. A complete analysis of the hybrid wave-pipelining scheme must include clock cycle minimization, taking into consideration the constraints of the internal nodes of the system and the register constraints. The minimum delay of the hybrid approach can be written to include the stage hold times as follows:

$$T_{min(h)} = D_R + D_{min\_hold} - \Delta_{clk} - T_h - \Delta \quad (5)$$

Also from Figure 3 it can be noticed that the region in which data is not stable, i.e. the difference between $D_{max} - D_{min\_hold}$, is short. It can then be safely stated that $D_{max} \approx D_{min\_hold}$. The signal latching time, expression becomes: $D_R + D_{min\_hold} + T_s + \Delta_{clk} - \Delta < NT_{clk} < T_{clk} + D_R + D_{min\_hold} - (\Delta_{clk} + T_h) - \Delta$.

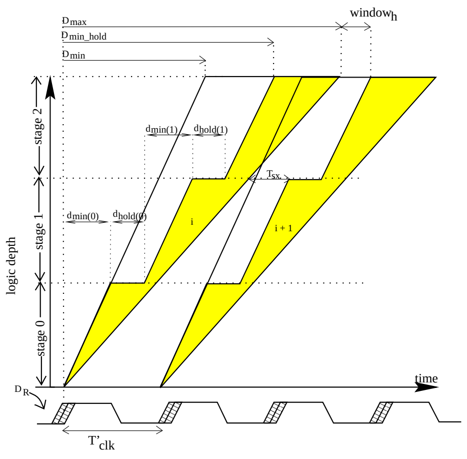


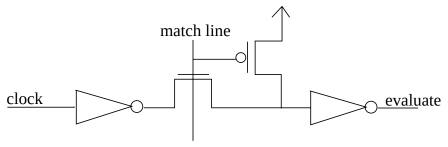
Figure 3. Temporal/spatial diagram after clock period reduction.

Hybrid wave-pipelining allows for the reduction of clock cycle time using the delays to propagate data from stage to stage without the use of either intermediate latches or distributed clocks.

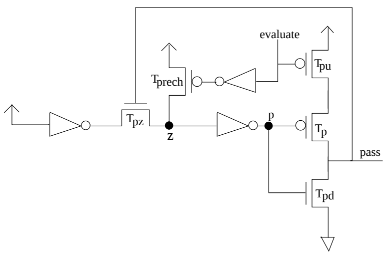
3.1 Control Signals for The Bit-Pattern Associative Unit

In this study propagating coherent data waves from one pipeline stage to the next without the use of a distributed system clock is achieved by using the delays in the design. These delays are manipulated to allow data to ripple from one stage to the next without any collision. The difference between the maximum (worst case) delay and the minimum delay is of particular interest in designing the timing scheme of a wave-pipelined bit-pattern associative router [2]. The buffer insertion method to balance the delays described in [4] is not used, instead specialized circuitry is designed to provide very precise timing sequences.

The evaluate signal must go to "1" after the data passed to the DCAM for comparison has stabilized on the lines BIT_COMPARE and NBIT_COMPARE. The circuit that generates this signal once all the lines have been set to their appropriate input values is shown in Figure 4(a). It is an AND gate whose inputs are the match line and the clock signals. The delays encountered in charging and discharging the match line are indicative of stable data on the bit lines, hence the use of the match line signal to generate an evaluate signal. Once evaluation has been completed the match



(a) Generating the evaluate signal.



(b) Generating the pass signal.

Figure 4. Circuits to generate BPAR control signals.

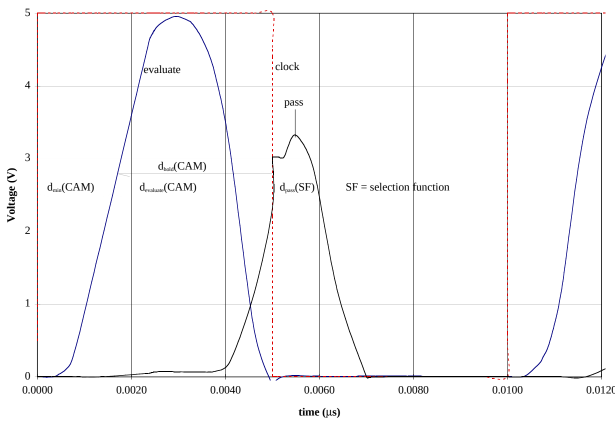
line outputs are passed to the selection function. Thus, the pass signal must immediately be active following completion of the evaluation process. The circuit used to generate the pass signal is shown in Figure 4(b). The pass signal is designed to mimic the path a zero on the match line (non-matching condition) takes once evaluation completes. Passing a "0" to the selection function provides the maximum delay that can be experienced in passing the matching unit's outputs to the selection function and, therefore, constitutes the worst case propagation delay for this operation. The circuits in Figures 4(a) and 4(b) have been designed to sense the duration and voltage levels of these signals. All the signals presented to this point depend on the system clock.

3.2 Hybrid Wave-Pipelining Signals

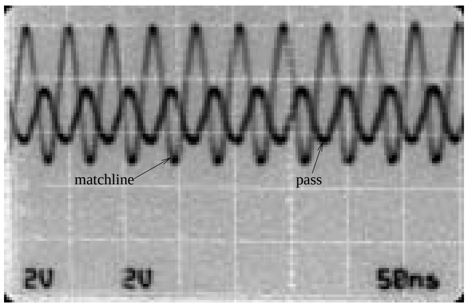
The signals generated by the above circuits appear in Figure 5(a) along with the clock. The delays of the first two stages are clearly marked in the figure to show a correlation with the hybrid wave-pipelined approach. On Figure 5(a) the minimum delays, (d_{min}) and hold times (d_{hold}) of each stage are shown. We have also included some results from the BPAR chip test, fabricated in a 0.5μm technology. These results appear in Figures 5(b) and 7(b).

Once the output of the match lines have been received by the selection function a decision need to be made when more than one matching condition has been received. The selection function is designed to propagate a priority status P_i to the entry below it indicating whether it has registered a match. This priority must be propagated very fast to prevent false starts. Some of the signals of importance in this scheme are shown in Figure 6. The selection function's critical operation occurs when its first and last entries simultaneously receive inputs indicating that matching conditions have been found in the corresponding bit-pattern associative unit entries. The first entry has to propagate a "0" to the last entry in order to prevent generation of a pointer to the DRAM by the last entry. Signals to lessen the possibility of false starts are generated and these are labeled $enable$. They are designed to ensure that even if a false start occurs the pointer to the DRAM array is not established. In Figure 6 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 are shown. The plots in Figure 7(a) are those of the DRAM pointers to its first and last entries. They serve to show that an output port assignment can be read from the DRAM array every one and a half clock cycles.

The *computational cones* of Figure 8 show the three stages of the bit-pattern associative router and their associated delays. Each stage accommodates the latest data associated with the current wave by allocating additional time to process this data. The bit-pattern associative unit completes worst case operation in 5.1 ns. This time includes the input latch delays, clock skew and hold and setup times for this stage. The hold time for the selection function is very short, almost equal to the minimum delay of this stage. This short hold time is a direct result of the selection function design, which ensures propagation of the priority status to entries below the current one very fast by means of the priority lookahead. The port assignment has a minimum delay of 0.4 ns and requires 3 ns of hold time. Reduction of the gap between D_{max} and D_{min} at each stage is presented here graphically with the delays displayed to show how the clock period is made shorter in this design.



(a) Plot of the bit-pattern associative unit control signals.



(b) Evaluate and match line signals.

Figure 5. SPICE simulations and chip test results for evaluate and pass circuitry.

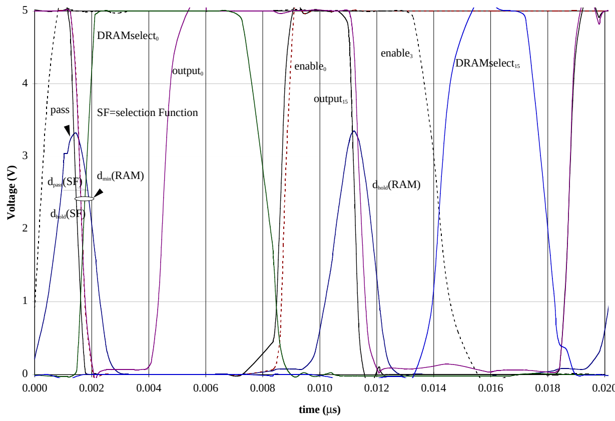
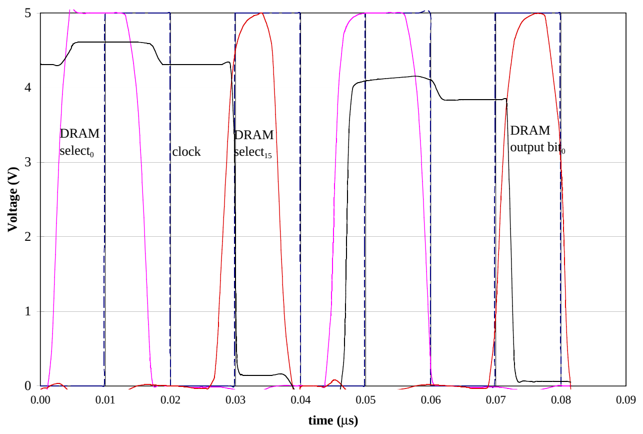
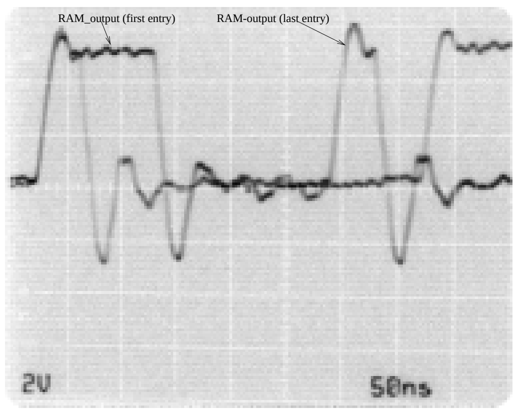


Figure 6. Plots of the enable signals.



(a) Plot of the DRAM pointers.



(b) Port Assignment output.

Figure 7. SPICE simulations and chip test results for output port assignment.

4 Concluding Remarks

An extremely fast flexible router capable of accommodating a number of routing algorithms and networks has been presented in this paper. The bit-pattern associative router is designed using a novel scheme that combines wave-pipelining and conventional pipelining producing hybrid wave-pipelining. The hybrid wave-pipelining approach narrows the gap between D_{min} and D_{max} resulting in a reduction of the clock cycle time. To further improve the performance of the bit-pattern associative router dynamic circuits along with a high performance selection function to reduce priority status propagation delays are used. This study is the first of it's kind; it examines the delays within each module of the design and minimizes them individually.

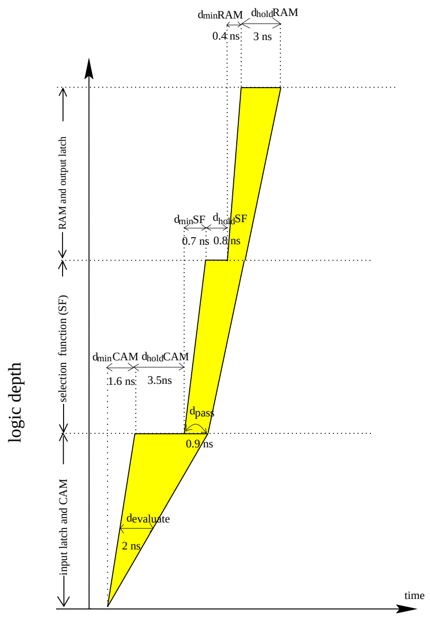


Figure 8. Delays of the BPAR translated to a computational cone diagram.

Issues such as signal generation, uneven delays, and timing have been addressed to synchronize the pipeline.

ACKNOWLEDGMENT

This work was supported in part by the National Science Foundation under contract CCR 9900643.

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] C. Thomas Gray, Wentai Liu, Ralph K. Cavin, III, *Wave Pipelining: Theory and CMOS Implementation*, Kluwer Academic Publisher 1994.

[3] 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. 1182-90, Nov. 1998.

[4] W. P. Burleson, M. Ciesielski, F. Klass and W. Liu, "Wave-Pipelining: A Tutorial and Research Survey," *IEEE Trans. on Very Large Scale Integration (VLSI) Systems*, Vol. 6, No. 3, pp. 464-474, September 1998.

[5] J. G. Delgado-Frias and J. Nyathi, "A High-Performance Encoder with Priority Lookahead," *IEEE Trans. on Circuits and Systems, I: Fundamental Theory and Applications*, Vol. 47, NO. 9, September 2000.