

# Built-In Constraint Resolution

Grady Giles

Joel Irby

Daniela Toneva

\*Kun-Han Tsai

Advanced Micro Devices  
5900 East Ben White Blvd  
Austin, TX 78741, USA

\*Mentor Graphics Corporation  
8005 S.W. Boeckman Road  
Wilsonville, OR 97070, USA

## Abstract

A new scan circuit construction obviates ATPG constraints for the prevention of tristate contention on internal tristate busses and one-hot muxes. The circuit is supported by enhanced ATPG.

## 1. Introduction and Discussion of Prior Work

In order to provide the highest performance features in a very competitive marketplace, contemporary microprocessor designs typically employ aggressive circuit techniques not commonly used in more conventional VLSI. One such technique is the prevalent use of pass-gate muxes, one-hot muxes, and internal tristate busses. Such use entails much additional design effort and analysis to ensure that contention is prevented from occurring on circuit nodes that have multiple drivers. For example, contention must be prevented even during transient timing conditions and in the case of non-typical and non-ideal fabrication processing. Contention, should it occur, may likely result in a reliability failure. Diligence in the prevention of contention is also required during test generation and test application.

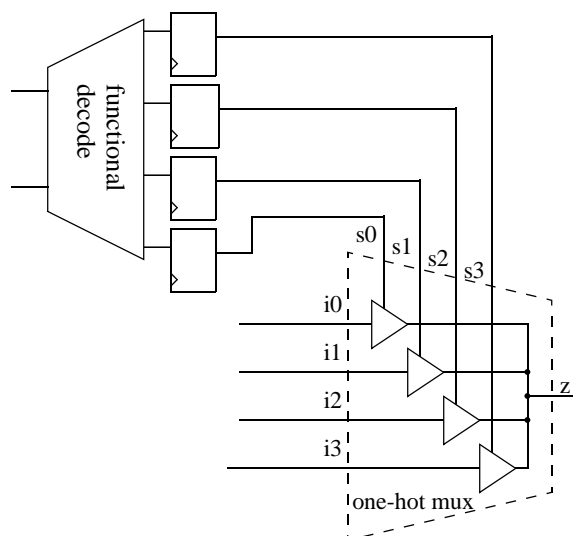


Figure 1: Problem of Scan Flops for Selects of One-hot Muxes

Figure 1 depicts a common test problem where the select inputs of a one-hot mux are driven directly from flip-flops. (One-hot is defined as exactly one of N.) If these are conventional scan flipflops [8], then during scan shift oper-

ation, contention will likely occur on circuit node z. (Strictly speaking, contention also depends on the data inputs of the muxes but generally the data inputs are not correlated with the select inputs.)

One remedy for this problem is to use a decoder for the selects in the same logic cone as the mux. However this is not always feasible because of circuit timing issues. In the case of Figure 1 the decoding may be on the left side of the flipflops for such timing reasons. In fact, this arrangement is sometimes preferred in order to minimize transient overlap timing on the select inputs.

Other ways of coping with this problem are to insert gates (and performance penalty) that enforce the one-hot behavior on the selects during scan shift between the flipflops and the selects, or a scheme such as is taught in [2] whereby the state elements driving the selects do not update until the end of the scan load sequence. These methods prevent contention during scan shift and rely on ATPG to ensure that scan loaded values satisfy the one-hot constraints for the group of flipflops that control the selects. We have practiced these methods and have been troubled by the poor performance of ATPG when it must resolve many constraints for every pattern. Resolving constraints can be particularly vexing for sequential tests such as the launch on capture type of delay test where two system clocks are applied [5]. In those tests the constraints must be satisfied for the initial loaded value and after each of the two capture clock pulses. The number of patterns and ATPG execution times tend to increase. Also, these methods are inapplicable for random patterns such as are applied with logic BIST.

We wanted to obviate constraints for ATPG by construction, hence the term *Built-In Constraint Resolution* (BICR, pronounced *bicker*). In literature we found the *Schmookler* device [1] which is an early BICR by our definition. This involves inserting a state machine into the scan shift path and using a clever retimed model substitution for ATPG. However, we judged that the *Schmookler* BICR had the following unattractive aspects: a) it is difficult to insert in a design with assured correctness; b) it confounds tools designed to trace scan chains; c) it is difficult to simulate (x states are perpetuated); and its most unfavorable property, d) it encodes captured fault effects with information loss, needlessly complicating fault diagnosis.

Motorola has used an offline decoder in conjunction with a resistive pull-down device on the tristate output of

the mux [3,4]. (See Figure 2.) The decoder generates a one-hot mask vector that is ANDed with the outputs of the constrained set of flipflops to produce the mux selects. There is an input to force the decoder to be inactive for normal operation. This scheme, which is also a kind of BICR, entails a functional performance penalty due to the AND gates in the functional path. It is tolerant of random patterns, but most random patterns simply activate the resistive pull-down device, resulting in diminished random pattern testability. Because the pull-down device is active only in test mode, the timing of transitions at the tristate output is altered: falling edges are faster, but rising edges are slower than in normal mode. There is a qualitative difference in the kinds of signal transitions that the offline decoder scheme can impart in test application versus in functional operation. In functional operation the select inputs can only switch from one one-hot combination to a different one-hot combination. The offline decoder scheme can only switch from a one-hot state to a zero-hot state, or vice versa.

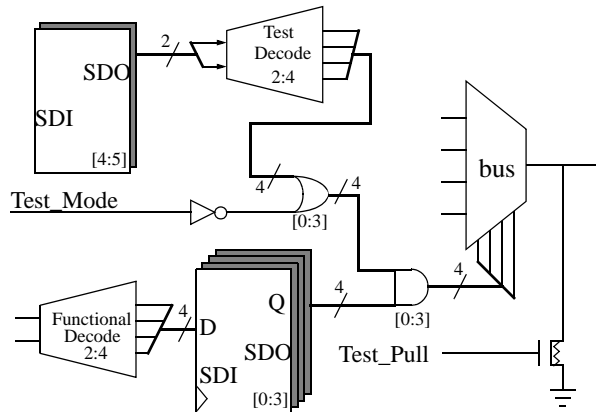


Figure 2: Offline Decoder BICR Scheme

Very late, a reviewer kindly informed us of a relevant prior work [12]. Excerpted from that paper is Figure 3 shown below. This scheme is an extension of the L3 shadow registers of [13]. The registers F1 in Figure 3 are a constrained set and are (sort of like) shadows of the scan

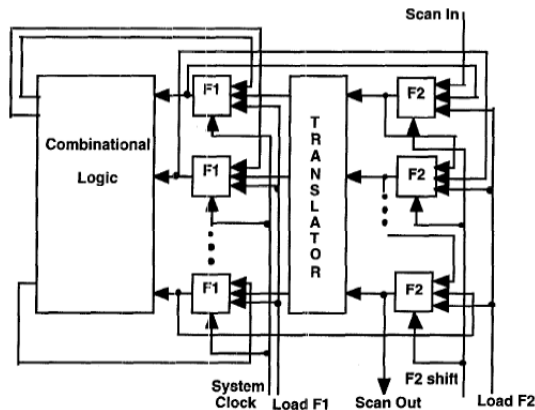


Figure 3: Shadow Register Based Technique to Handle One-Hot Signals During Scan

registers F2. The *translator* is a test decoder that ensures that the shadows can only load values that result in a contention-free state, and so this scheme too is a BICR. The BICR that we have developed and deployed is inserted into a design in the same place as this shadow register based scheme but it doesn't entail the global L3-type control signals depicted in Figure 3 as Load\_F1 and Load\_F2.

In this paper we will describe an improved BICR that is fully supported by enhanced ATPG and LBIST tools. In section 2, a novel BICR apparatus and models are discussed. Sections 3 & 4 discuss the Mentor Graphics' implementation of DRC and ATPG enhancements respectively for supporting the new BICR. Section 5 discusses experimental comparisons. Section 6 discusses the practical application of BICRs in designs. Diagnosis with BICRs is the subject of Section 7. Section 8 concludes the paper.

## 2. BICR Apparatus and Models

Our approach provides an alternative scan *flipflop* (hereafter referred to as *flop*) and design augmentation for instances where the flop is to be member of a set of constrained flops. The insertion of the BICR into the design should be simple and result in negligible performance impact. Finally the solution should be amenable to commercial tools for design rule checking (DRC), test generation and fault diagnosis.

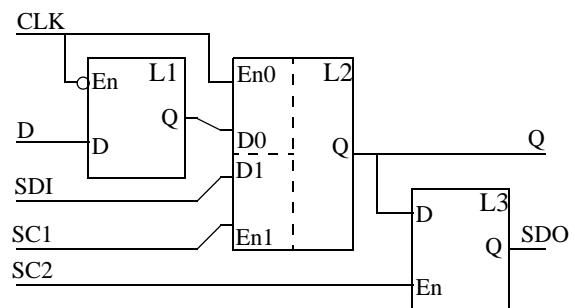


Figure 4: Standard Scan Flop

Figure 4 is a simplified diagram of a standard scan flop used in many microprocessor designs which is similar to [11]. Left and center are the master and slave latches, L1 & L2, comprising the system flop. The output of the system slave latch L2 is the state node of the flop. The system slave latch has a second port clocked by SC1 (Scan Clock 1) and thus is the master latch with respect to scan operation. The scan slave latch L3 is on the lower right. Scan shifting is performed by alternately pulsing SC1 and then SC2. Values are captured into the state node of the flop by pulsing the system clock CLK. Prior to the scan unload operation, the master observe operation is performed by pulsing SC2 to transfer the captured value from the state node into the scan slave latch. Our modification of this standard flop is the BICR flop. The BICR flop has three alternate representations, type-I, type-II & type-III.

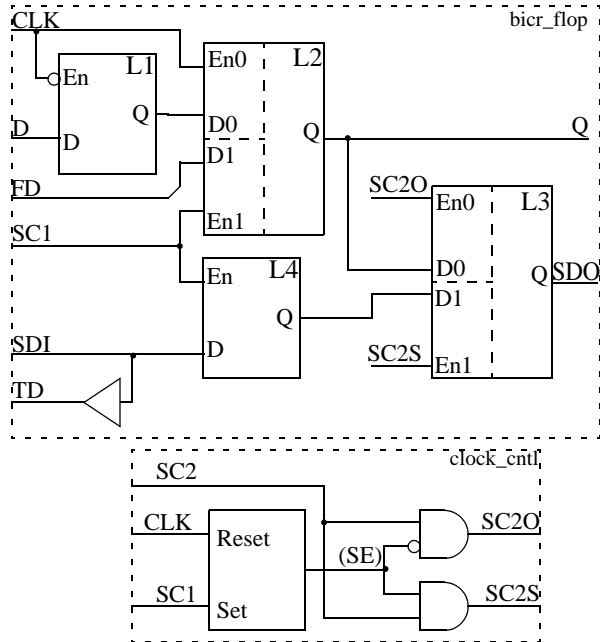


Figure 5: BICR Flop & Local Clock Control with Type-I Models

## 2.1. Type-I BICR

Figure 5 shows a modified scan flop called a BICR flop and a companion local clock control cell. Note that the system functional path D-to-Q is identical to the corresponding path in the standard flop. The data input for the SC1 port of the system slave latch L2 is FD (*From Decoder*). A separate shift path is provided through the lower two latches, L4 & L3. The scan slave latch L3 is clocked by two different gated versions of SC2 provided from the clock control cell. The RS latch output, SE, is zero following a capture clock pulse and is one following an SC1 pulse. So SC2 applied for a master observe operation will generate SC2O (*Scan Clock 2 Observe*) and SC2 applied immediately following a pulse of SC1 will generate SC2S (*Scan Clock 2 Shift*). Comparing the BICR flop cell to figure 3, Latches L1 and L2 correspond to a register F1 and latches L4 and L3 correspond to a register F2. (This labeling scheme is not to be confused with that of [13].) TD (*To Decoder*) is an output which is a buffered version of SDI.

## 2.2. BICR ensemble

Figure 6 depicts an example of a BICR ensemble. Scan connections and clocks are not shown for the sake of clarity, but typically a single clock control cell is provided for each ensemble. Connections to and from a test decoder are shown with arrows. Note that in this example not all of the TD terminals of the BICR flops are connected. In this arrangement, as scan data are shifted through the scan path latches of instances 0 & 1, the state nodes of instances 0-3 (which are not on the scan path) receive decoded values from the test decoder. The test decoder is designed to satisfy the one-hot constraint for this set of four flops and so

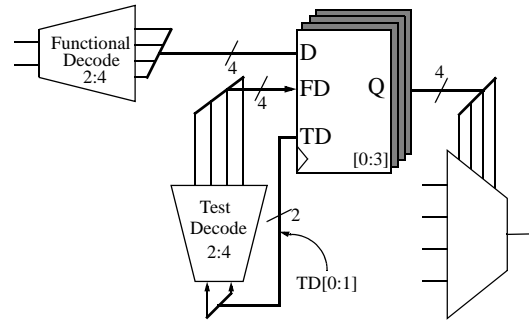


Figure 6: Example BICR Ensemble

fulfills the built-in constraint resolution criterion. Test decoder operation happens only during scan shift. Values captured in the BICR flops' state nodes are moved into the scan path to be shifted out according to the same master observe protocol that is used for the great majority of the standard scan flops in the design.

Though we recommend a type-I physical implementation as has been described, there is a design trade-off choice that we should highlight. Instead of generating SE as a derivative signal, there are some advantages to providing SE as an independent primary input. Fault effects from the test decoder are captured in the state node of the BICR flops by pulsing SC1, but the RS latch generation of SE requires that the system clock first be pulsed in order to enable captured values to be transferred into the scan path. Therefore test decoder fault effects are only observable forward from the Q outputs. As BICRs are extensions of the scan architecture for both control and observation, it would be desirable to have a set of patterns that verify correct operation of the extended architecture, much as a chain test verifies the scan chain integrity. This is much more feasible if SE is a primary input. Despite these considerations, we chose the derivative generation method because of the undesirability of an additional globally routed signal. Strategies for testing the test decoders are presented later in this paper.

## 2.3. Type-II and Type-III BICR Models

There are some logically equivalent retimed BICR models which are noteworthy. The type-I BICR cell defini-

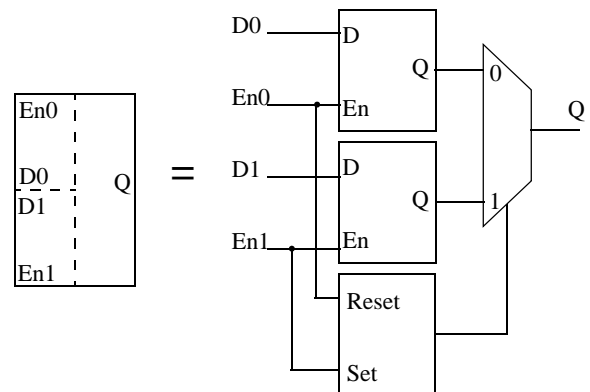


Figure 7: Retiming Equivalent Circuit for Dual-port Latch

tions were carefully chosen in order that model substitutions could be made at the cell level. The reader may be aided in understanding the model transformations by considering the retiming equivalence trick shown in Figure 7.

Figure 8 depicts the type-II model of the BICR flop, which when used with the clock control cell model depicted in figure 10, is functionally equivalent to the corresponding models of type-I. In fact these models are phys-

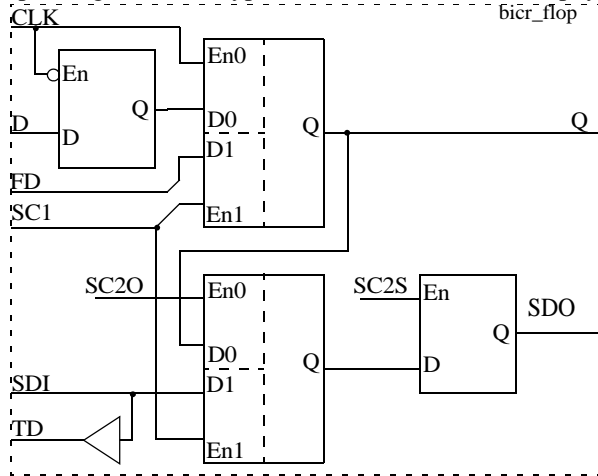


Figure 8: Type-II Model of BICR Flop

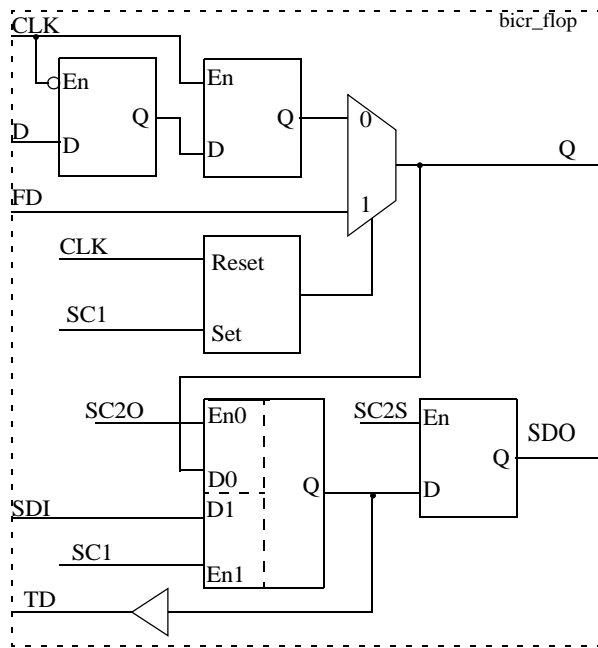


Figure 9: Type-III Model of BICR Flop

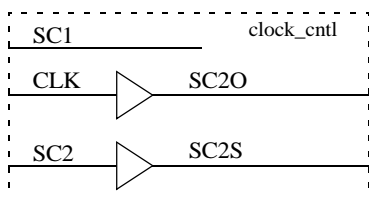


Figure 10: Model of Clock Control Cell used for Type-II and Type-III

ically realizable alternative implementation styles (though we do not do that). Exhaustive simulation has shown that type-I and type-II models are equivalent except in the illegal case where CLK, SC1, & SC2 are all asserted. Our ATPG vendors have indicated that type-II models are somewhat easier for ATPG to accommodate than full type-I models. Figures 9 and 10 show another functionally equivalent retimed model, type-III. This model is derived by the application of the retiming trick of figure 7 to the type-II model. Physical implementation of this style of BICR would entail the performance penalty of the mux in the output path. This model is amenable to conventional ATPG because the test decoder appears as though it is in the same cone as the one-hot mux. Also, as discussed in section 7, this model may facilitate testing and diagnosis of test decoder faults.

## 2.4. Multi-stage BICR Ensemble

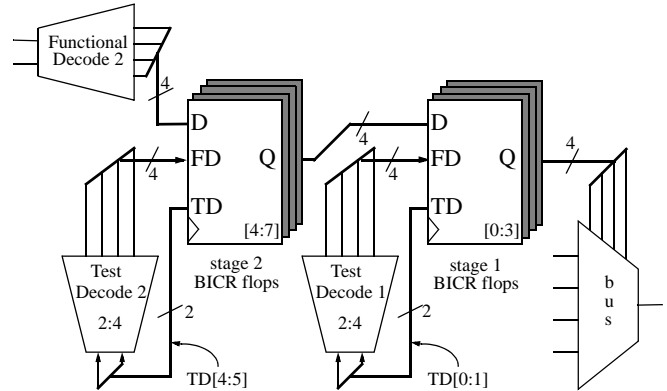


Figure 11: Example of Pipelined BICR Ensemble

Sometimes the functional logic does not provide fully decoded values to the BICR flops in the direct manner shown in Figure 6. There may be reasons to build the functional decode in a pipelined fashion such that the decoding occurs over a number of system clock cycles. In such cases the BICR ensemble must be designed in a pipelined manner as well. Each stage of such a pipelined arrangement comprises a set of constrained flops. The constraints may be different from simple one-hot constraints. Test decoders must be designed to satisfy the constraints appropriate for each set of constrained flops in the pipelined arrangement. If it is determined, prior to test generation, that there is a maximum limit to the number of capture clock pulses that are to be applied for scan tests, then the number of pipeline stages that must be assessed for the potential need for test decoders is one more than that limit. For most of our circuits, we plan and commit to the application of just two capture clocks.

Figure 8 shows a trivial example of a dual-stage BICR ensemble. In this example, BICR flop instances 4-7 are a constrained set of flops and instances 0-3 are a separate constrained set, but both sets' constraints are due to the same one-hot mux. Straight wire connections in the functional paths between the second and first stages are shown. In this example, these connections constitute a degenerate

case stage 1 functional decoder, but more generally there could be any less-than-full decoder here for a multi-stage ensemble. If the stage 2 functional decoder shown satisfies the one-hot constraint for instances 0-3, (as translated through stages 2 & 1) then this pipelined BICR ensemble is complete and safe for any number of capture clock cycles. If, however, the functional decoder shown in Figure 11 is not a full one-hot decoder, then this pipelined BICR is safe for the application of only a single capture clock pulse.

### 3. Design Rule Checks (DRCs) for BICR

There are two main objectives of the DRC for BICR. The first is to learn the BICR logic automatically in order to simplify the ATPG process. The second is to ensure that the BICR logic is inserted correctly so that the associated bus is always contention free during the scan operation as well as the system operation. In addition, to check that the design is BIST-ready, DRC analyzes the BICR logic to ensure there are no X sources in the design. It is critical to identify the special BICR off-shift-path storage elements that receive the decoded values and to recognize their special properties. In this ATPG context we refer to the special storage elements as BICR *registers*. Herein, this is distinguished from design context references to the BICR *flops* which are the design cells that contain both the special off-shift-path storage elements and scan cells.

#### 3.1. BICR Structure Assumptions

The BICR assumptions are designed to make the DRC efficient without restricting the BICR hardware structure.

*Assumption 1:* A BICR register contains two clock ports, one controlled by a system clock and the other controlled by a scan clock.

*Assumption 2:* The number of pipeline stages of a BICR ensemble is bounded by a user-specified number  $p$ .

*Assumption 3:* The test decoder inputs are driven by either scan cells or primary inputs, with the total number of inputs not exceeding  $n$ . In reality  $n$  is set to 16, which bounds the test decoder to be up to 16 by 65536.

*Assumption 4:* Each BICR is associated with at least one bus.

*Assumption 5:* The drive enables of a bus that necessitates a BICR are combinationaly reachable from only BICR registers. This assumption is primarily used to reduce the complexity of bus contention analysis.

*Assumption 6:* BICR registers comprising a pipeline stage receive the same system clock.

#### 3.2. BICR DRC Flow

The BICR DRC process is designed to automatically learn the BICR logic during the scan cell identification process and to fit into the existing DRC. Note that this scan cell identification process, often referred to as scan tracing, is the fundamental step of identifying all scan elements in the design in many ATPG tools. Scan tracing identifies the scan path from the user-specified scan output pin back to

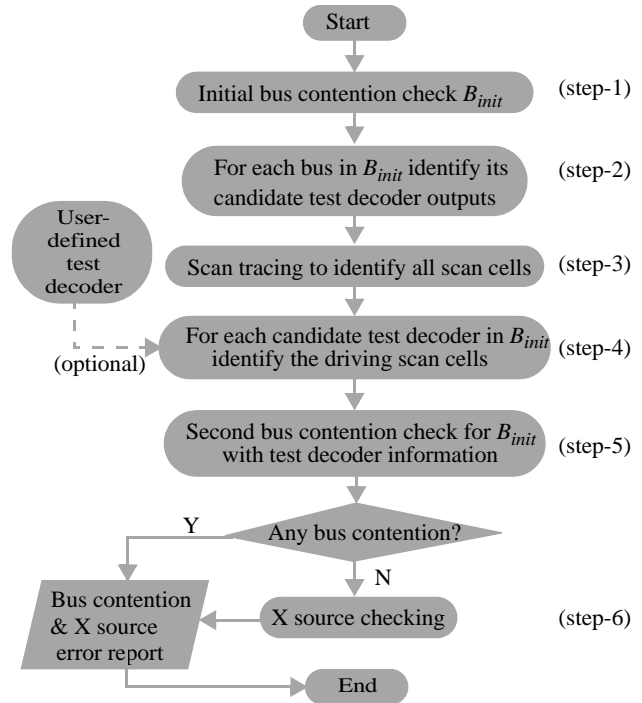


Figure 12: DRC flow for BICR

the scan input pin using the user-specified scan procedures [9]. After the scan elements are identified, the test decoder inputs of BICR registers can be located, since each input of the test decoder is either a scan element or a primary input. The output of the BICR test decoder is assumed to always be connected to the test port of a dual port sequential element, the BICR register; the other port of which is driven by the system logic. The BICR output can be identified by tracing the bus select lines backwards to combinationaly reachable dual port sequential elements. The system port of the dual port sequential element is distinguished from the test port during the scan tracing by monitoring the clock port activity of candidate BICR registers during the scan shift operation. Only the test decoder port has switching activity. The system clock port should remain quiet during this period.

Figure 12 depicts the flow of DRC for BICRs.

**Step 1.** The BICR DRC starts by checking each bus in the design assuming that all sequential elements are non-scan elements and derives a bus set  $B_{init}$  containing all of the busses that may result in bus contention. This step removes any bus that is already one-hot combinationaly, to reduce the DRC overhead. If the bus is not in  $B_{init}$  there is no need for further BICR analysis.

**Step 2.** This step takes each bus in  $B_{init}$  and traces the corresponding bus select lines backward to identify the dual-port registers as candidate BICR registers driving the bus. For example in Figure 6, by tracing back from the bus select lines, the BICR registers of instances 0, 1, 2 and 3 will be identified.

**Step 3.** The scan tracing is then performed to identify all scan cells. While performing the scan tracing, the clock

ports of the candidate BICR registers are monitored. The ports with activity are marked as test decoder ports and other ports are marked as system ports.

**Step 4.** The test decoder logic is extracted by tracing back from the test decoder ports of BICR registers, identified in step 3, to outputs of scan cells or primary inputs. For example in Figure 11, test decoder 1 is derived by tracing back from the FD ports of BICR flops 0, 1, 2 and 3. For the case of a multi-stage BICR structure, the BICR DRC is able to identify test decoders up to a user-specified number of pipelined stages when considered with system port information. Again in Figure 11, by tracing back from the system (D) ports, of stage 1 BICR registers (3-0), the stage 2 BICR registers (7-4) will be identified. Similarly, by tracing back from the test (FD) ports of stage 2 BICR flops in Figure 11, the cone of test decoder 2 are learned.

An optional step allows user-defined test decoders to be manually declared for BICRs that are not associated with any bus or that otherwise do not conform to the six assumptions. This functionality takes a list of user-specified signals and treats them as bus select lines where candidate BICR registers can be derived by backward trace. Once the candidate BICR registers are identified, the rest of DRC for the user-defined test decoders is the same as those that are automatically identified.

**Step 5.** After all BICR structures are identified, DRC performs more accurate bus contention check with the BICR registers constrained by both test decoders and functional decoders. A divide-and-conquer approach is used to activate a single clock port of each BICR register at a time. For the BICR structure with  $i$  stages, it is sufficient to check  $(i+1)$  times of the single port pipelined structure. For example, in Figure 11 which contains two stages, three paths are involved: a first path through test decoder 1 (by enabling only test ports of stage 1 BICR registers); a second path through functional decoder 1 (by enabling system ports of stage 1 BICR registers and test ports of stage 2 BICR registers); finally through functional decoder 2 (by enabling system ports of all BICR registers).

There are two benefits of using a divide-and-conquer approach for the bus checking on multi-stage BICR structure. First, it reduces the complexity of the problem from exponential search (with respect to the number of flops in each stage) into a linear search. If there are  $i$  pipelined stages and each stage has  $k$  flops, without simplifying the problem, it will take  $2^{k \cdot i}$  clock port combinations to verify no bus contention is possible. By taking advantage of the pipelined clocking assumption that test ports or system ports of each stage are always turned on simultaneously, the divide-and-conquer approach reduces the problem into  $(i+1)$  combinations.

The second advantage of the divide-and-conquer approach is in debugging BICR insertion errors. Since the DRC process is able to identify each stage's test decoder and system decoder and perform contention checking accordingly, any potential bus contention after BICR insertion can be spotted by DRC. The corresponding test decoder as well as the pipeline stage are reported by DRC

to assist in debugging DFT insertion or design errors. Since the tool uses ATPG to justify bus contention, whenever a potential bus contention is proven, an assignment is also reported to show one condition where bus conflict happens. In addition, the clocking assumption is checked by ATPG to ensure that all test ports are activated together and all system ports are activated simultaneously as well. Any violation of the clocking assumption will be reported along with a description of where the violation happens.

**Step 6.** To ensure the design is BIST-ready, X source checking is performed by leveraging the bus contention determination of step 5. A bus can produce X if ATPG justification shows it can have a conflict during either scan shift or system operation or the bus can be undriven during scan shift or system operation.

Traditional ATPG is designed to handle bus contention or a floating bus design as well as X sources. The drawback is the pattern volume and performance overhead as will be explained in section 4.

#### 4. ATPG with BICR

When handling a design with a bus (such as node Z in Figure 1), ATPG engines commonly have two strategies to prevent contention. One is to prevent contention up front by explicitly making assignments at the bus select lines before targeting any fault, and then generating patterns for the targeted fault list. The second strategy is to target faults first and work on the bus contention later. Typically, when a design contains only a few instances of potential bus contention, the post process approach is a good trade-off with negligible impact on ATPG performance. However, the up front approach tends to work best for designs with numerous sites of potential contention because of the increased likelihood of assignment conflicts between fault detection and contention prevention. Either approach imposes significant run time and pattern volume overhead as each pattern needs to explicitly prevent contention by ATPG justification. With the introduction of BICR, once DRC has proven that no assignment can result in a bus being in a contention state or floating during the capture period, the ATPG engine is relieved of the burden of ensuring contention prevention and may focus exclusively on fault detection.

There are two major steps to support BICR that are different from traditional ATPG. First, the BICR registers are treated as special scan cells that are directly controllable by ATPG. The only extra effort for ATPG is to justify from BICR registers back to scan cells, combinationally, through the (automatically learned) test decoders. This process needs to be done only for the faults that require an assignment on a BICR register and is relatively fast since it is combinational logic with bounded fanin (by  $n$  which is set to 16 in practice). The other BICR registers not involved with the fault detection will remain unspecified and later be randomly filled and are still ensured to be contention free. This nice property not only reduces ATPG effort but also leaves ATPG more flexibility to perform dynamic compaction [6] to reduce the pattern volume.

The second different step, defined as the *BICR observation step*, is to treat BICR registers as observation points. A fault effect that propagates to the system port of a BICR register is considered to be detected. The master observe procedure, performed before the scan unload operation to propagate the value captured at a BICR register into a scan cell, typically just needs to pulse the scan slave clock to capture the value at the scan slave latch. For example, the master observe procedure for the BICR flop in Figure 8 can consist of simply pulsing SC2S. To guarantee that this step is consistent with the hardware behavior, an additional DRC is performed after step 4 in Figure 12. The purpose of this step is to simulate the master observe procedure and ensure that the BICR register value can be transferred safely into the scan slave latch.

## 5. Experimental Comparisons

We have conducted experiments on three benchmark circuits with constrained sets of flops. These benchmark circuits were originally designed with BICRs inserted, but we made modified versions of the respective netlists with the BICRs removed and with offline decoders inserted in place of the BICRs. The shadow register based technique [12] and our BICR are equivalent w.r.t. ATPG but the DRCs would be very different, and those DRCs were not developed, so that scheme is not included in our experiment. For the versions of the benchmark circuits without BICRs, ATPG is constrained to load only one-hot combinations into the constrained set of flops and to disallow these flops to capture anything but one-hot values. In our first experiment, transition delay ATPG was performed on each of the three versions (no BICR, BICR, offline decoder) of each of the three benchmark circuits in a two-pass flow. In the first pass, the one-hot muxes are modeled with tristate buffers and BICRs are modeled as type-II. We use the

tristate models in the first pass in order to take advantage of ATPG’s bias for choosing observation paths that do not pass through the select inputs of the one-hot muxes, if such alternate paths exist. The patterns from this first pass are saved. In the second pass, the muxes are modeled combinatorially and the BICRs are modeled as type-III. The patterns from the first pass are transition fault simulated. Top-off ATPG is then performed for any remaining testable transition faults. This step allows ATPG to propagate fault effects through the mux select inputs. These patterns are saved and the total transition delay fault grade statistics recorded. These transition delay patterns are then fault simulated against the stuck-at fault model and that fault grade result is recorded. Top-off stuck-at pattern generation is not part of the experiment. The results of this experiment are presented in the first three columns for each benchmark circuit in Table 1.

The CPU effort values are normalized w.r.t. the BICR case. This includes both DRC and ATPG but in all cases the DRC effort was negligible compared to the ATPG effort. Generally the results are that each method can achieve comparable test coverage but that the AMD BICR method requires fewer patterns. While it is true that the no BICR cases get better transition coverage, this is accounted for by the fact that ATPG was prohibited from using the scan clocks to launch transitions. The BICR test decoders and the offline decoders only operate in the SC1 clock domain which is not provided with the at-speed launch capability. Thus, though ATPG could, if allowed, generate transition tests for faults in the decoders, such tests can not be applied at-speed, and so are not included.

We were concerned that these results, which appear to show that the BICR and the offline decoder are essentially equivalent, are misleading because of the non-functional nature of the transitions provided by the offline decoders.

	Circuit A				Circuit B				Circuit C			
	w/o BICR	BICR	offline dec I	offline dec II	w/o BICR	BICR	offline dec I	offline dec II	w/o BICR	BICR	offline dec I	offline dec II
faults	216840	219364	222460		112700	113544	114854		1178460	1179128	1180026	
scan cells	5918		5965		3300		3342		12719		12731	
constrained sets of flops	21				4				4			
average flops/set	4.95				8				6.5			
buses needing treatment	549				260				82			
<b>Fastscan Results</b>												
transition coverage	97.8%	96.6%	96.9%	96.6%	90.9%	90.2%	90.3%	90.2%	89.7%	89.6%	89.6%	89.2%
stuck-at coverage	98.8%	98.5%	98.4%	98.4%	93.1%	93.0%	92.8%	92.8%	97.7%	97.7%	97.7%	97.7%
pattern size	2368	1682	1941	3512	611	585	604	950	21953	17191	17351	22103
CPU effort	1.20	1.00	0.96	1.98	1.07	1.00	0.86	1.72	1.38	1	1.69	1.95
<b>10X TestKompess Results</b>												
transition coverage	97.7%	96.6%	96.9%	96.6%	90.9%	90.2%	90.3%	90.1%	89.7%	89.6%	89.6%	89.2%
stuck-at coverage	98.7%	98.5%	98.4%	98.4%	93.1%	92.9%	92.8%	92.8%	97.8%	97.8%	97.8%	97.8%
pattern size	2801	1838	2094	3422	755	669	683	915	27098	23426	27001	27199
CPU effort	1.32	1.00	1.11	1.75	1.20	1.00	1.45	2.06	1.11	1.00	1.11	1.41

Table I: ATPG Result Comparisons

ATPG credits detection of transition faults in the cones of the one-hot mux select inputs for the one-hot to zero-hot and zero-hot to one-hot transitions. The AMD BICR launches transitions in a manner much more like transitions that occur in normal operation. We designed a second experiment in order to highlight the impact of this qualitative difference.

From the offline decoder versions of the benchmark netlists, we extracted the fault lists of the one-hot mux select inputs' cones. Importantly, the lists include faults on the outputs of the constrained sets of flops. ATPG was performed on the offline decoder benchmarks with these faults deleted. ATPG was free to make any assignments for the selects but was prevented from crediting detection in those cones. In a separate run, ATPG processed the select cone faults. In this treatment, a one-hot ATPG constraint was applied to the select inputs. This forced ATPG to create the kind of transitions that occur in normal operation and it was the only opportunity for ATPG to credit detection of these faults. The results of the separate treatments of the two partitions of faults were combined and are shown in the Table 1 in the columns labeled offline dec II. A comparison of the two sets of offline decoder results indicates the strong tendency of the offline decoder method to produce the qualitatively less desirable signal transitions. Overall, we regard that the results show that, though the offline decoder BICR method is very satisfactory for stuck-at testing, our BICR method is superior for at-speed transition testing. The superiority of BICR over ATPG constraints is especially striking; 0.2% constrained flops in circuit C resulted in a 28% pattern size impact.

## 6. BICR Usage

BICRs may complement an overall microprocessor DFT scheme. A comprehensive DFT approach for the digital circuitry is summarized as follows. The multitude of custom designed special purpose embedded memory arrays, referred to as *macros*, are tested by MBIST functions that are tightly woven into the normal functional access paths. The MBIST has a rich feature set and is designed to operate at least at the rated frequency of the product. For structural test of non-memory logic using ATPG, the input registration of the macros is scanned for observation of macro inputs. The output registration is scanned for control of macro outputs, but in addition these macro output registers are provided with means to launch arbitrary transitions duplicating the timing of functional accesses without actually reading the embedded memory data. The rest of the logic is full scan. Embedded Deterministic Test (EDT) [10] is used for test data compression and a custom LBIST is integrated in such a manner that it shares much of the EDT circuitry. We have versatile clock control for the application of at-speed scan tests. Transition delay is the primary fault model used for ATPG, and this is supplemented by stuck-at fault and path delay ATPG. BICRs are the critical factor for our ability to do LBIST and we have found that, without having to resolve constraints for every pattern, ATPG is able to produce more

compact pattern sets, especially with hardware compression.

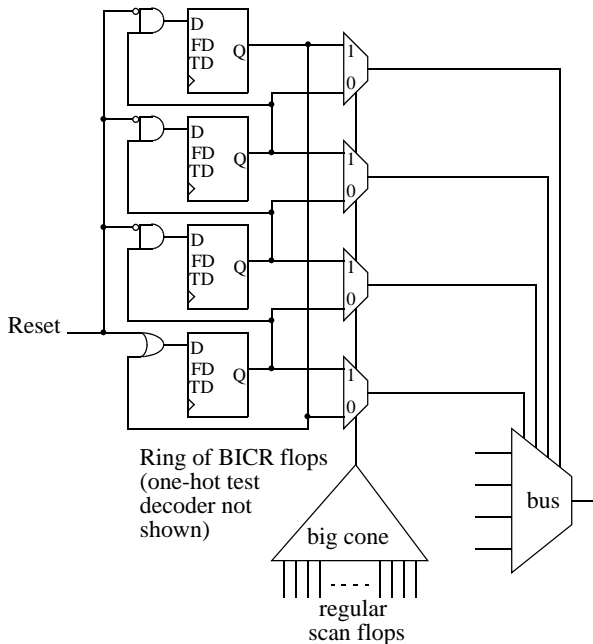
In the normal course of their work, AMD design engineers usually recognize instances where a BICR ensemble is necessary. Every normal type scan flop in the design cell library has a BICR type counterpart. The designers insert the BICR flops in instances where the flops are members of a constrained set of flops, and connect a test decoder appropriate for the particular ensemble. Only rarely have multi-stage ensembles been required. ATPG is instructed to consider that only two capture clock pulses are to be used, limiting the theoretical maximum required depth of an ensemble to three stages; that depth has yet to be necessary.

For most purposes in our work flow, the BICR flops are modeled as type-II and the one-hot muxes are modeled with tristate devices. Design rule checking is performed to verify the adequacy of the contention prevention measures in the design and to learn the test decoder mapping functions. Any BICR deficiencies are flagged at this step. The most common error encountered is a test decoder, the outputs of which are all exactly of the opposite polarity of what is required. Once the DRCs have passed, ATPG and/or LBIST simulation is performed. ATPG patterns and/or a sample of LBIST patterns are resimulated in a logic simulator. This process yields the vast majority of our patterns but fault simulation at this stage typically reports poor coverage in the areas of the test decoders and the select cones of the one-hot muxes.

This poor coverage is usually due to the fact that though the test decoder outputs are captured in the off-shift-path state node of the BICR flops (the BICR register in the ATPG context), those captured values are not directly transferable into the scan slave latch by the master observe procedure. (The functional captured values are transferable.) Test decoder fault effects are observable forward from the Q state node outputs of the BICR flops. Often this means that these Q state node outputs are only observable through the selects of the one-hot muxes, and with the tristate modeling of those muxes the best that can be had is possible detection. To address these remaining faults we fault simulate the patterns against a model with type-III BICR models and combinational models of the one-hot muxes. Stuck-at ATPG may then be performed if there remain any test decoder and mux cone faults. Because the test decoders operate only with the (relatively) slow scan clocking, we are satisfied with stuck-at coverage for these faults.

We engaged with our ATPG providers to ensure sufficient support of the novel structure. It was anticipated that there might be exceptional circumstances where it would be impractical for ATPG to automatically identify a valid BICR ensemble. Hence we requested the user-defined test decoder ATPG feature. Of the approximately 500 BICR ensembles with which we have had experience to date, there have been but two ensembles that have necessitated the use of the user-defined test decoder. Both of those ensembles are identical. Figure 13 shows this case which occurs in the benchmark circuit C. The four BICR flops are

arranged in a ring with the only functional data input to the ring being Reset. Functionally, a one-hot value is continually recirculated. There is a large cone of conventionally scanned logic that controls the four combinational muxes. Those muxes select a one-hot vector from the ring or that same vector shifted by one position. Clearly, the flops of the ring are the only constrained flops, but this structure violates the fifth assumption of the test decoder learning and DRCs. We provided ATPG the pin-path names of the four outputs of this test decoder and had no further difficulties.



**Figure 13: BICR ensemble required user-defined test decoder feature of ATPG**

## 7. Diagnosis Considerations

Diagnosis impact is one of the important factors to be accounted for with any DFT logic insertion. The primary diagnosis consideration for BICR is to ensure that any fault in the BICR logic is distinguishable from faults in the system logic. This condition plays a very important role during the failure analysis to quickly narrow down the root cause of a chip failure. As described in section 2.2, if the SE signal for type-I BICR is driven by a dedicated pin, the *chain test*, defined as the test pattern set that applies only the scan load, unload and master observe without system capture clock, can be used to both test and diagnose the failure on the scan chain and BICR. Since chain test does not apply any system clock, the cause of the chip failure must come from the scan path or BICR. In such a case, a few random chain test patterns are sufficient to test BICR with a small test decoder, e.g. the test decoder with less than 6 inputs. If the test decoder is larger, deterministic test pattern generation can be used to ensure that all combinations of the test decoder states are covered during the scan

load and unload process of chain test. The faults on BICR can be further distinguished easily from the faults on the scan path in most cases since the BICR faults are only observed by patterns with a master observe procedure.

In practice, we choose not to use a dedicated pin for SE, hence the chain test is not applicable. For the type-I BICR without dedicated SE pin and for type-II BICR model, the only way to detect test decoder faults is through the system logic, usually via a bus select line. In such cases, the test decoder faults are treated the same as the system faults, which are targeted explicitly by the deterministic ATPG engine and a traditional fault diagnosis algorithm [7] can be used.

One practical issue for detecting BICR faults in type-II BICR design is due to a general ATPG assumption that the fault propagation is not considered during the scan loading period but only during capture cycle(s). The test decoder fault effects need to propagate through the test port of the BICR registers before they can be captured at downstream observation points. Thus, it always requires ATPG to pulse the SC1 clock before pulsing a system clock. Consequently, the detection criteria of BICR faults are more restrictive, resulting in much more ATPG effort as well as extra test patterns.

Type-III models are useful to ease these criteria since the test decoder fault effects can be propagated combinatorially to an observation point without first pulsing the SC1 clock. In reality, this is true of the physical type-I circuit as well. For diagnosis, type-III models make the test decoder faults more distinguishable from the system decoder faults, because test decoder fault effects are more clearly inferred to be captured at an observation point forward from the BICR register output, whereas functional decoder fault effects are captured in the BICR register.

## 8. Conclusion

### 8.1. Limitations and Costs

The silicon cost of our BICR over a conventional scan implementation with ATPG constraints is an extra latch element per constrained scan cell, and a test decoder and clock control cell per constrained set of flops. The constrained flops are a small minority of the scannable elements in our microprocessor designs, on the order of 1%. This method obviates another common cost associated with constrained data elements, that is the necessity of a globally routed test signal to make one-hot muxes safe during scan shift.

We have discussed the theoretical necessity for multi-stage BICRs and those extra stages represent additional cost in those rare instances. The logical problem of the multi-stage BICR is easily addressed, and there is ample timing margin in our BICR design. There is though at least a possibility of an implementation problem with the multi-stage BICR. As the constraints are pushed back into functional pipeline stages farther from the one-hot mux, the members of the constrained sets of flops of those more

remote stages may be physically more dispersed from their respective other set members, and yet each member must communicate with a common test decoder. The offline decoder style BICR does not have this potential routing issue.

The BICR learning algorithm has the limitations of its six simplifying assumptions. These limitations are easily mitigated by use of the user-defined test decoder ATPG feature. Also BICR learning, as an exceptional process, can be avoided entirely by the use of type-III models.

## 8.2. Benefits

Built-In Constraint Resolution provides the following benefits:

- The ability to accommodate the use of one-hot muxes in our designs,
- The ability to apply random patterns as in LBIST,
- More compact and compressible pattern sets than patterns with comparable coverage produced by use of ATPG constraints,
- Realistic at-speed signal transitions in test application, and
- Virtually identical functional performance as a conventional scan implementation (i.e. no performance penalty).

## 8.3. Summary and Review of Results

We have described a novel Built-In Constraint Resolution DFT flipflop and three alternate models of that flipflop from both the design and ATPG perspectives. We have taught how to use these flipflops in instances where there are inherent constraints and how to connect them with test decoders to form single-stage and multi-stage BICR ensembles to prevent tristate contention in one-hot muxes. ATPG has been enhanced to check for proper BICR insertion and to take full advantage of the novel structure for testing and diagnosis. Experiments have shown that our BICR method enables ATPG to produce high quality and compact patterns. For designs incorporating one-hot muxes, BICR is a useful DFT feature, facilitating at-speed application of both deterministic ATPG patterns and LBIST.

## 8.4. Opportunities for Further Development

- Insertion of BICR circuitry is today a manual process. We believe that it should be feasible to develop automatic BICR synthesis and insertion techniques.
- We have an idea of how to make the test decoders more directly testable. If the system clock connected to the BICR registers can be gated off while a system clock connected to the local clock control circuit is allowed to pulse, then the test decoder outputs captured in the BICR registers may be directly transferred to the scan slave latch by the master observe operation. This sort of circuit would violate the sixth simplifying assumption of the BICR DRC so currently it is not supported.

- Herein, we have presented a (more or less) LSSD style BICR. We have thought of a way to do a muxed-D style BICR similar to this treatment, but have not pursued developing that idea.

## REFERENCES

- [1] S. Pateras; M.S. Schmoekler, "Avoiding unknown states when scanning mutually exclusive latches", *International Test Conference*, 1995. Pages 311 - 318
- [2] Kuppuswamy, R.; DesRosier, P.; Feltham, D.; Sheikh, R.; Thadikaran, P. "Full Hold-Scan Systems in Microprocessors: Cost/Benefit Analysis." *Intel Technology Journal*. <http://developer.intel.com/technology/itj/2004/volume08issue01/> (February 2004).
- [3] C. Pyron; M. Alexander; J. Golab; G. Joos; B. Long; R. Molyneaux; R. Raina; N. Tendolkar; "DFT Advances in Motorola's MPC7400, a PowerPC TM Microprocessor", *International Test Conference*, 1999. Pages 137 - 146
- [4] R. Raina, R. Bailey, D. Belete, V. Khosa, R. Molyneaux, J. Prado, A. Razdan, "DFT Advances in Motorola's Next-Generation 74xx PowerPC Microprocessor", *International Test Conference*, 2000. Pages 131 - 140
- [5] T. Kant, "Special Issues with Generation of Contention-Free Vectors in Transition Test", [http://www.mugweb.org/members/conferences/2004/papers/kant\\_sun.pdf](http://www.mugweb.org/members/conferences/2004/papers/kant_sun.pdf)
- [6] P. Goel and B.C. Rosales, "Test Generation and Dynamic Compaction of Tests," *Digest of Papers 1979 Test Conf.*, pp. 189-192, 1979.
- [7] J. Waicukauski and E. Lindbloom, "Failure Diagnosis of Structured VLSI", *IEEE Design & Test of Computers*, 1989, pp. 49-60.
- [8] T. W. Williams and K.P. Parker, "Design for Testability - A Survey," *IEEE Trans. on Computers*, Vol. C-31, No. 1, pp 2-15, January, 1982.
- [9] Design-for-Test Common Resources Manual, chapter 8, pp.329-370, v8.2004\_6, Mentor Graphics, 2004.
- [10] J. Rajski, J. Tyszer, M. Kassab, N. Mukherjee, R. Thompson, K.-H. Tsai, A. Hertwig, N. Tamarapalli, G. Mrugalski, G. Eide, J. Qian; "Embedded Deterministic Test for Low Cost Manufacturing Test", *International Test Conference*, 2002, pp. 301 - 310.
- [11] U.S. Patent 4,277,699.
- [12] S. Mitra, L. J. Avra, E. J. McCluskey; "Scan Synthesis for One-Hot Signals", *International Test Conference*, 1997, pp.414-422
- [13] S. Das Gupta, R. G. Walther, T. W. Williams, E. B. Eichelberger, "An Enhancement to LSSD and some Applications of LSSD in Reliability, Availability and Serviceability," *11th International Symposium on Fault-Tolerant Computing Proc.*, 1981, pp. 32-34.

The authors wish to thank Eric Mahurin, Tim Wood, Brion Keller and Don Ross for their contributions to this work, Carol Pyron for her helpful criticism.