Emulation of Scan Paths in Sequential Circuit Synthesis

Bernhard Eschermann, Hans-Joachim Wunderlich
Institut für Rechnerentwurf und Fehlertoleranz (Prof. D. Schmid)
Universität Karlsruhe, Zirkel 2, 7500 Karlsruhe, F. R. Germany

Abstract – Scan paths are generally added to a sequential circuit in a final design for testability step. We present an approach to incorporate the behavior of a scan path during circuit synthesis, thus avoiding to implement the scan path shift register as a separate structural entity. The shift transitions of the scan path are treated as a part of the system functionality. Depending on the minimization strategy for the system logic, either the delay or the area of the circuit can be reduced compared to a conventional scan path, which may be interpreted as a special case of realizing the combinational logic. The approach is also extended to partial scan paths. It is shown that the resulting structure is fully testable and test patterns can be efficiently produced by a combinational test generator. The advantages of the approach are illustrated with a collection of finite state machine examples.

1 Introduction

Scan path techniques are widely used to facilitate test generation for sequential circuits: Hardware added after the circuit is synthesized guarantees that all combinationally irredundant faults, which do not introduce sequential behavior, can be detected with a number of test patterns increasing no more than linearly with the circuit size. Structural test generation programs for combinational logic are sufficient to produce these patterns. In some cases, however, the additional hardware and delay incurred by incorporating a scan path is considered undesirable.

Recently algorithms for synthesizing circuits without combinational and sequential redundancies have been proposed [DMNS 90]. They guarantee that all single stuck-at faults can be detected. Unfortunately they are computationally quite expensive. They also do not increase the controllability and observability of memory elements. Partial scan paths, in which all the cycles in the circuit structure are cut [KuWu 90, ChAg 89a], improve the situation, but cannot help for circuits with complex structural dependences like controllers [KuWu 90]. First approaches to specify test functionality before synthesis and to accordingly modify the synthesis process were presented in [EsWu 90, AgCh 90].

In this paper a different method for considering testability during synthesis is described, which avoids post-design circuit modifications like adding a shift register structure. It starts from the behavioral description of a sequential circuit and produces a circuit structure, in which the memory elements are both directly controllable and observable ("emulated scan path", see section 2). In effect the structure an emulated scan path is not fixed a priory, it is customtailored to the circuit to be tested instead. Depending on the minimization strategy, either the delay or the area of the circuit can be optimized and reduced compared to a conventional scan path (section 3). The testability of the resulting circuit structures is investigated in section 4. Test patterns for faults in the combinational logic can be generated using a combinational logic test generator and the test application time may turn out to be shorter than for scan paths. An extension to the case, where a partial scan path is sufficient, is investigated in section 5. Results for a collection of benchmark examples are presented in section 6.

2 Emulation of Scan Paths

The following characteristics of a scan path are vital to facilitating the test of a sequential circuit with r flipflops:

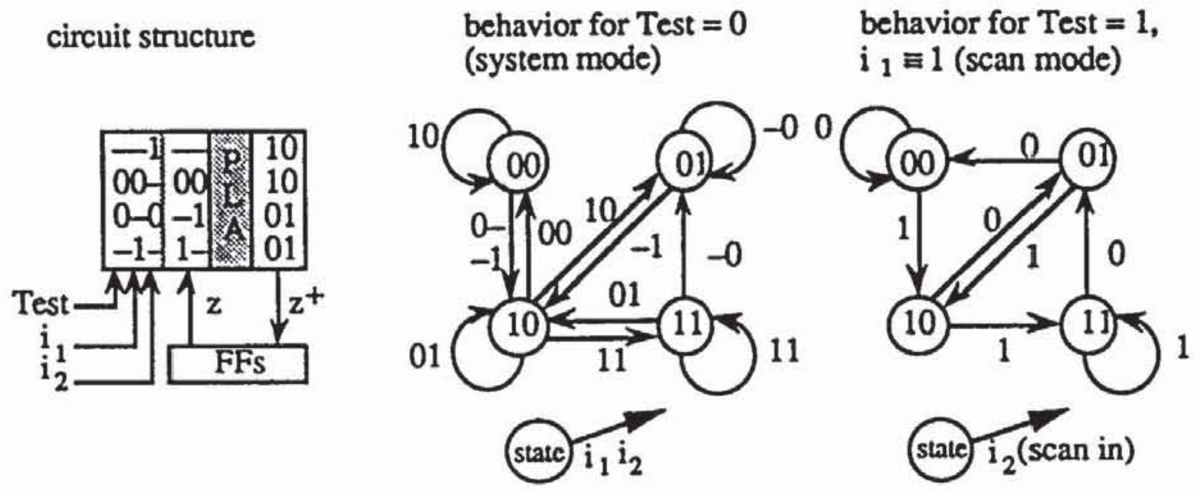
- Every state is reachable in at most r steps from every other state (shift in).
- The current state of a circuit can be identified after at most r steps (shift out).
- The correctness of these two functions can be easily verified (O(r) steps).

Consequently, a test consists of two phases. First the correct function of the scan path is verified. Then the scan path is used to scan in test patterns, which were generated by only considering the combinational logic, and to scan out the test responses. Since the number of test patterns is generally O(r) [Goel 80], the test application time roughly grows quadratically with the number of flipflops.

To obtain this functionality the storage elements are generally reconfigured to form a shift register. If, however, a scan path is not characterized by this shift register *structure*, but by its *behavior*, i. e. its main characteristics listed above, more degrees of freedom exist to implement it. Such generalized scan path structures will be called "emulated scan paths" in the sequel and an approach to synthesize such structures will be presented. The additional test state transitions can be specified together with the overall circuit behavior and considered in the synthesis process, resulting in a more economical implementation. Shift inputs and outputs may be mapped to system inputs and outputs, reducing the need for pins solely devoted to testing. Since in an emulated scan path no switching between different flipflop modes is necessary, it is also possible to detect dynamic faults relevant to the circuit behavior.

The basic idea can be realized in two different ways:

- One primary input is chosen to act as the shift input in test mode, a primary output is used as shift output. The additional state transitions realizing the scan path behavior are activated by a test control signal. This possibility is examined in the rest of this paper. A pertinent structure (for simplicity without outputs) is illustrated in Fig. 1 and explained in more detail later on. The input i_2 is used as shift input, the signal Test is the test control signal and input i_1 is kept to constant 1 during shifting.
- Several primary inputs and outputs are used to shorten the test sequences ("parallel scan"). A disadvantage of this approach is that the realization effort for the additional state transitions may become very large.



circuit structure
behavior for Test = 0 (system mode)
behavior for Test = 1, i 1 ≡ 1 (scan mode)
PLA
Test
i1
i2
z
z+
FFs
state
i1 i2
i2(scan in)

Fig. 1: Sequential circuit with "emulated scan path".

In some special cases the circuit already covers the behavior of a shift register or such a behavior can be easily obtained by using unspecified state transitions [AgCh 90]. Then the additional external connection "Test" can be saved and normal state transitions can be used to force the circuit into a given state or observe the state of the circuits at the primary outputs, both in r steps.

3 Synthesis of Circuits with Emulated Scan Path

To synthesize a circuit with the architecture of Fig. 1, several design decisions have to be made:

- A certain sequence of state variables has to be determined to be able to specify the shift transitions for the scan mode[1].
- An input and output variable have to be chosen as shift input and output.
- The inputs (except for the shift input) can be set to a fixed value in scan mode. This is not a necessary prerequisite, but increases the minimization potential and facilitates the test as shown in section 3.5.

3.1 Basic Idea for Shift Sequence Determination

Emulated scan paths can be efficiently implemented by making good choices during the synthesis process. Considering these issues on the behavioral level like in [AgCh 90] leads to a complexity growing at least linearly with the number of states, e. g. exponentially with the number of flipflops. It is also difficult to estimate the realization complexity for different alternatives on this level. On the other hand it is not possible to wait until the end of the synthesis process, because at that point only a conventional scan path can be incorporated. Therefore we first produce an intermediate structural representation of the circuit, then modify the logic and afterwards proceed with the rest of the synthesis process. This approach also facilitates the incorporation of emulated scan paths in existing designs by starting with the synthesized circuit, extracting the intermediate structural description and resynthesizing the

[1] Since the scan behavior is hidden in the system logic, it is not necessary to use storage elements which are in close proximity as consecutive elements of the scan path.

logic with the test hardware. An intermediate structure is obtained by carrying through a provisional synthesis (including logic minimization). The additional state transitions then should be chosen such that this initial structure is complicated as little as possible.

For each state variable z_i in the initial structure a cone of dependence can be determined, which includes all the input and state variables directly influencing the value of z_i. A shift transition, which transfers the contents of a storage element z_k into the storage element for z_i, makes variable z_i dependent on z_k. If z_i is independent of z_k in the initial structure, this complicates the combinational logic no matter how the logic is implemented. Therefore, if possible, no additional dependences in shift mode should be created.

3.2 Formal Model for Shift Sequence Determination

Definition 1: The *dependence graph* of a circuit structure is a directed graph DG = (V, E), with vertex set $V = \mathcal{I} \cup \mathcal{F} \cup O$, where $\mathcal{I}$ denotes the primary inputs, $\mathcal{F}$ the flipflops and O the primary outputs of the circuit, and an edge set $E \subseteq \{(v_1, v_2) \mid v_1 \in \mathcal{I} \cup \mathcal{F}, v_2 \in \mathcal{F} \vee v_1 \in \mathcal{F}, v_2 \in O\}$, which represents the dependences of the state and output variables v_2 on inputs or state variables v_1.

Let $|\mathcal{I}| = p$, $|\mathcal{F}| = r$, $|O| = q$. A suitable sequence of state variables in shift mode corresponds to a sequence of edges through all vertices of $\mathcal{F}$ in G, starting in an arbitrary vertex of $\mathcal{I}$ and ending in an arbitrary vertex of O (see Fig. 2). There are at most $p \cdot r! \cdot q$ such sequences. More formally one can formulate the following decision problem:

Problem SPSS (scan path shift sequence)

Instance: Let a dependence graph DG = (V, E), $V = \mathcal{I} \cup \mathcal{F} \cup O$, of a circuit structure be given.

Question: Does there exist an edge sequence $(v_0, v_1), (v_1, v_2), \ldots, (v_r, v_{r+1})$ with $(v_i, v_{i+1}) \in E$, $0 \leq i \leq r$, $v_0 \in \mathcal{I}$, $v_{r+1} \in O$, such that $\{v_1, \ldots v_r\} = \mathcal{F}$?

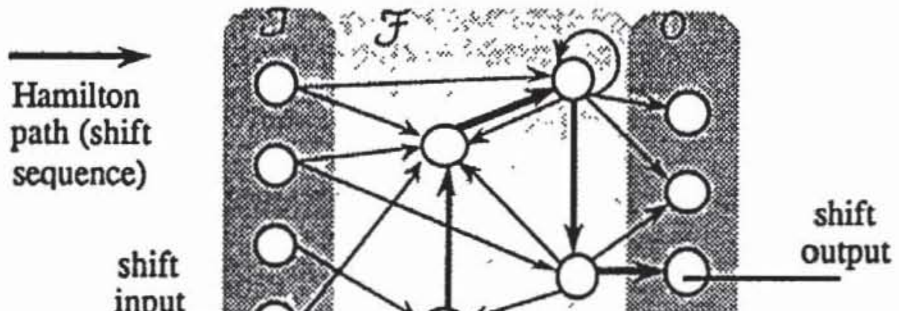


Fig. 2: Dependence graph and Hamilton path.

Theorem 2: SPSS is NP-complete.

Proof: α) SPSS ∈ NP: trivial, by following a given shift sequence and marking the vertices.

β) By restricting SPSS to problems with $|\mathcal{I}| = |O| = 1$ it is reduced to the well-known problem of determining a directed Hamilton path with given start and end point. But this problem is NP-complete [GaJo 79]. ■

In spite of the problem's NP-completeness, even for larger circuits optimal solutions can be found, since the number of vertices in $\mathcal{F}$ only grows logarithmically with the circuit's number of states. This is a major advantage of inserting the shift sequences on an intermediate structural level and not directly e. g. into a state transition diagram. When DG does not contain cycles, Hamilton paths can be found in polynomial time [Lawl 76]. If DG does not contain a Hamilton path, additional edges have to be inserted in DG to obtain a shift sequence. The task then is to determine a sequence of vertices, such that the least number of such edges (corresponding to the minimal number of additional dependences) has to be added.

Definition 3: The *weighted dependence graph* of a circuit structure is a digraph $DG_g = (V, E_g, g)$, $E_g = \mathcal{I} \times \mathcal{F} \cup \mathcal{F}^2 \cup \mathcal{F} \times O$ with edge weights $g: E_g \to \mathbb{Z}$,

$$g(v_i, v_j) = \begin{cases} 0 \text{ for } (v_i, v_j) \in E \\ 1, \text{ if } (v_i, v_j) \text{ is not contained in DG} \end{cases}.$$

Problem OSPSS (optimal scan path shift sequence)

Instance: A weighted dependence graph $DG_g = (V, E_g, g)$, $V = \mathcal{I} \cup \mathcal{F} \cup O$, of a circuit structure and a bound $K_{max} \in N$ are given.

Question: Does there exist a sequence of edges $(v_0, v_1), (v_1, v_2), \ldots, (v_r, v_{r+1})$ with $\Sigma\, g(v_i, v_{i+1}) < K_{max}$, $v_0 \in \mathcal{I}$, $v_{r+1} \in O$, such that $\{v_1, \ldots v_r\} = \mathcal{F}$?

It is easy to see that for $K_{max} = 1$ the problem SPSS is contained as a special case. If no Hamilton path exists, the sum of edge weights $\Sigma\, g(v_i, v_{i+1})$ has to be minimized in order to produce the smallest number of additional dependences for implementing the scan path. The problem of searching for a Hamilton path with minimal weight is well-known as traveling salesman problem and there exist standard algorithms to solve it [Chri 75].

3.3 Application to Two-Level Logic

The dependence graphs defined so far have to be somewhat modified to actually represent the shift sequences, which are realizable with minimal effort using a specific design style for the combinational logic (e.g. two-level or multi-level logic). This is illustrated in the following example for a PLA realization.

Example 1: The dependence of a state variable z_i on z_k is a necessary but not a sufficient condition for being able to hide a shift transition from z_k to z_i in the system logic. Although every state variable in the circuit of Fig. 3a depends on every other variable and we therefore have $2 \cdot 2! = 4$ Hamilton paths, only the shift sequence $i_2 \to z_1 \to z_2$ illustrated in Fig. 3b leads to a realization not requiring additional product terms, by merging the shift transition marked with ① in Fig. 3b with the corresponding product term in Fig. 3a. The Karnaugh maps for the next state variables z_1^+ and z_2^+ in Fig. 3c have been supplemented by an input variable Test, such that for Test = 0 the system function as shown in Fig. 3a is realized, whereas for Test = 1 the shift transitions listed in Fig. 3b are realized, provided that $i_1 \equiv 1$. Since in shift mode (Test = 1) the input i_1 is kept at constant 1, all combinations with Test = $1 \wedge i_1 = 0$ result in don't cares. The final structure with the additional input Test after minimization was already shown in Fig. 1. To simplify the example, output variables are not considered in Fig. 3 but they can be treated in a similar fashion.

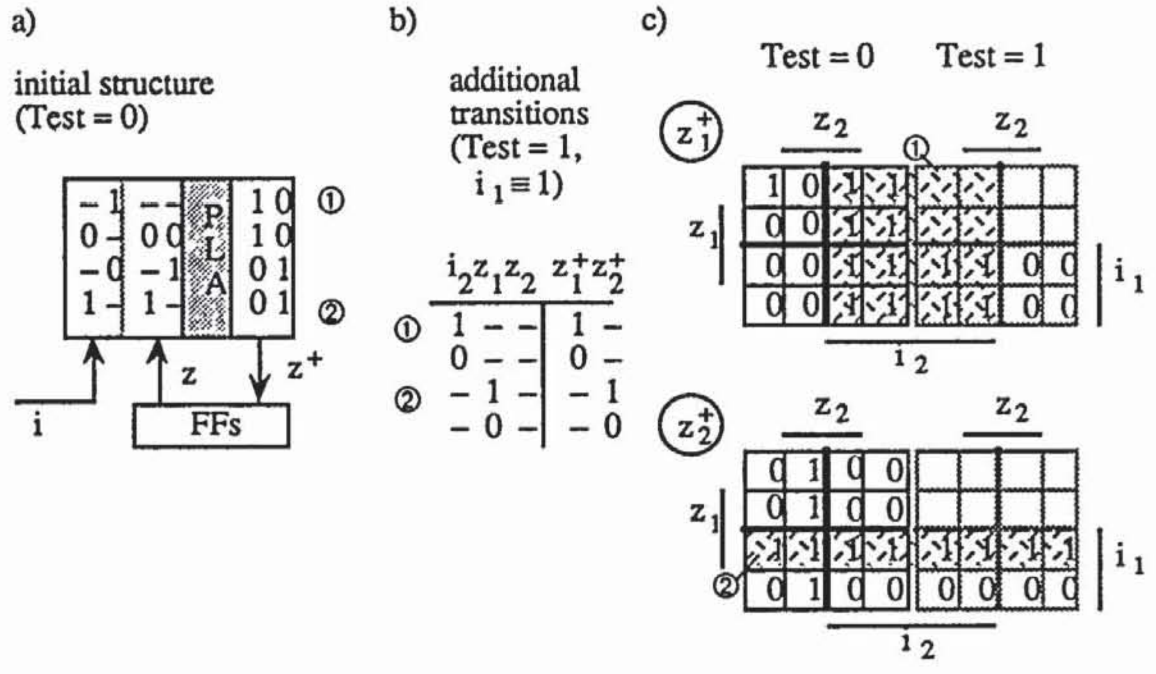


Fig. 3: Example for the integration of the shift logic into the system logic.

For a two-level sum-of-products (PLA-) realization of the combinational logic only product terms, in which zero or one state variables z_i are relevant, are candidates for a direct merging with product terms for shift transitions. If in such a product term a state variable z_i is 1, whenever it is activated $z_i = 1$ is shifted; if it is 0, this corresponds to shifting the complemented value of z_i, i. e. in the scan path model the complementation of a scan path stage. The possibility of complementing shift values increases the minimization potential for an emulated scan path by adapting the polarity of scan path stages to the circuit to be synthesized. If in a product term no state variable is relevant, it can be merged with the product term for the shift input of the emulated scan path. In the dependence graph basic to the optimization procedure only edges, which correspond to such minimization-relevant product terms, may be included.

Example 1 (cont'd.): The relevant dependence graph only contains the edges $e_2 \rightarrow z_1$, $z_2 \rightarrow z_2$ and $z_1 \rightarrow z_2$. Self-loops like $z_2 \rightarrow z_2$ can be discarded at once, since they are not usable for realizing helpful shift transitions.

Example 2: If in some circuit the shift transitions of Fig. 4a are realized (the complemented value of z_1 is shifted) this corresponds to the scan path structure illustrated in Fig. 4b.

a)

i	z_1	z_2	z_1^+	z_2^+
1	–	–	1	–
0	–	–	0	–
–	0	–	–	1
–	1	–	–	0

b)

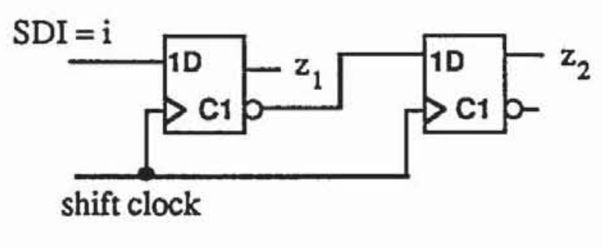


Fig. 4: Complementation of shift variables.

3.4 *Application to Multi-Level Logic*

For multi-level realizations the savings are more difficult to estimate, but the minimization potential is larger. Common multi-level synthesis algorithms, e. g. [BRSW 87], produce boolean networks, whose nodes represent subfunctions minimized in two-level form, which are connected by edges (intermediate variables). For every output variable of the net there exists an output node, which contains the last stage of logic to produce it. Since this node represents a two-level logic function of its input variables, it could be treated similarly as above. A difference to two-level logic is that it is not necessary for a possible minimization that in a certain cube only one state variable is relevant, as the global effect of a different factorization or decomposition might lead to a reduction of the logic independent of two-level minimizability. The provisional inclusion of a shift transition into the pertinent output node of the boolean network followed by a minimization pass provides an estimate of the realization complexity to be expected. Since the minimization of shift transitions belonging to different output nodes is relatively independent, this approach makes it possible to compare all possible shift transitions ending in a certain output node. The value obtained this way is used as edge weight $g(v_i,v_j)$ of the weighted dependence graph DG_g. The number of shift transitions, which have to be evaluated this way, is $|E_g| = r \cdot (p+q+r-1)$. After choosing the shift transitions to be implemented, a final multi-level minimization is performed to obtain a globally optimized implementation of the combinational logic.

Example 3: For a multi-level realization of the boundary scan TAP controller [IEEE 90] the dependence graph of Fig. 5 is obtained. Depending on whether it is preferable for the shift value to be complemented or not (cf. Example 2), the edges in Fig. 5 are drawn in black or grey. The edge weights correspond to the additional number of literals necessary to realize that state transition. An optimal shift sequence is TMS $\xrightarrow{+} z_1 \xrightarrow{-} z_2 \xrightarrow{-} z_4 \xrightarrow{+} z_3$, where $\xrightarrow{+}$ denotes a shift transition without complementation of the state variable and $\xrightarrow{-}$ a transition with complementation. The variable z_3 is used as additional shift output, since the primary outputs of the TAP controller are not directly accessible. The sum of edge weights for the optimal shift sequence is 18, but after the final minimization step only 11 additional literals are needed, since a better decomposition of the boolean network with shift transitions can be found.

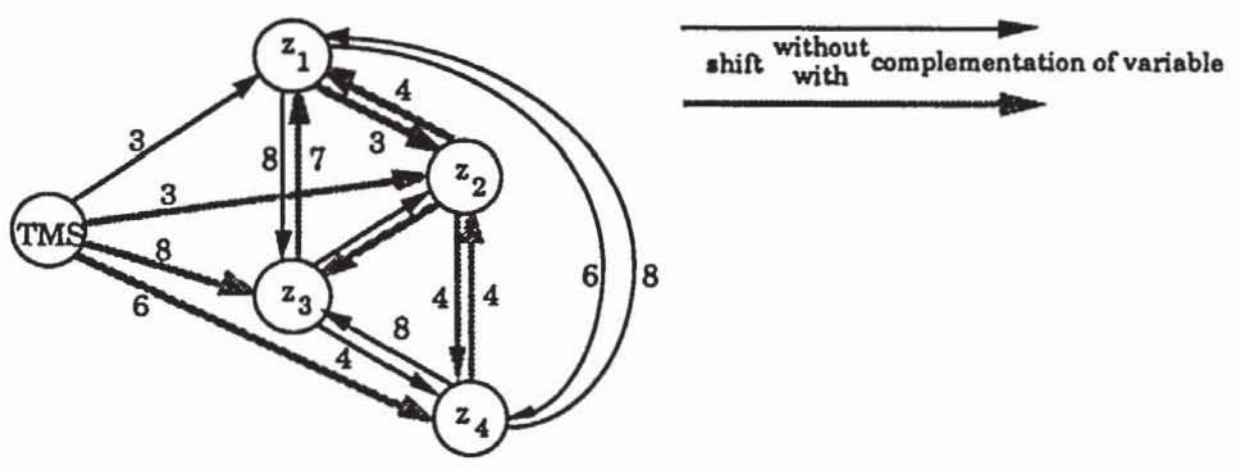


Fig.5: Multi-level dependence graph for the boundary scan TAP controller.

3.5 Fixing Primary Input Values

As arbitrary shift transitions have to be accessible in parallel, all shift product terms have to be usable in arbitrary combinations. One possibility is to activate the shift product terms no matter what the input values are, the other is to set the input signals to fixed values while shifting. The two possibilities can be mixed, e. g. when a subset of input variables is not controllable. Fixed input values while shifting imply that the next state and output become "don't care" for all other input values, thus increasing the minimization potential. While determining the shift transitions, for all scan path stages conditions are obtained, which the input values should satisfy such that the best optimization of the combinational logic is achievable. The input signal vector $\bar{i}$ satisfying the maximal number of input conditions should be used in scan mode.

Example 1 (cont'd.): In Fig. 3 for product term ② (implementing the transition $z_1 \rightarrow z_2$) it is necessary that $i_1 = 1$ in order to be able to merge it with the fourth product term of the original PLA. For shift transition ① ($i_2 \rightarrow z_1$) the value of i_1 is arbitrary. Let's consider the sequence (0,0,1,1,0) at the shift input i_2 in Fig. 1. It is easy to validate that by extending this sequence with the inputs Test = 1 and $i_1 \equiv 1$, i.e. by applying (110, 110, 111, 111, 110), as desired we get the flipflop contents 0x, 00, 10, 11 and 01.

3.6 Overall Synthesis Procedure

The compatibility of input vectors can already be considered while determining the minimal weight Hamilton path in the dependence graph. To simplify the presentation, this is neglected in Fig. 6, where the approach to embed a scan path in a given circuit structure is summarized. The shift sequence is $z_{k_1} \rightarrow z_{k_2} \rightarrow \ldots \rightarrow z_{k_r}$, where $k_1, k_2, \ldots k_r$ is a permutation of the state variable indices $\{1, 2, \ldots r\}$. The shift input is denoted z_{k_0}, the shift output is $z_{k_{r+1}}$. The variable $\dot{z}_{k_j} \in \{z_{k_j}, \overline{z_{k_j}}\}$ is to be entered positive or complemented depending on the polarity of the "scan path" stage.

```
SCAN_PATH_EMULATION:
    Perform an initial minimization of the system logic, keep the external DC-set(k) for all variables k;
    Compute the weighted dependence graph $DG_g$;
    Determine a minimal weight Hamilton path $\dot{z}_{k0} \rightarrow \dot{z}_{k1} \rightarrow \ldots \rightarrow \dot{z}_{kr+1}$ from an element of $\mathcal{J}$ to an
      element of O, which contains all the elements of $\mathcal{F}$;
    Compute a maximal compatibility class of input vectors $\bar{i}$ for the shift mode;
    For all next state and output variables $k \in \{1, \ldots r, r+1, \ldots r+q\}$:        {keep system logic}
        ON-set(k) := {Test = 0} × ON-set(k);
        OFF-set(k) := {Test = 0} × OFF-set(k);
    For all indices $k_j$, $j \in \{1, 2, \ldots r, r+1\}$:                                  {add scan logic}
        ON-set($k_j$) := ON-set($k_j$) ∪ {Test = 1 ∧ $\bar{i}$ ∧ $\dot{z}_{k\underline{j-1}}$ = 1};
        OFF-set($k_j$) := OFF-set($k_j$) ∪ {Test = 1 ∧ $\bar{i}$ ∧ $\dot{z}_{kj-1}$ = 0};
    For all next state and output variables $k \in \{1, \ldots r, r+1, \ldots r+q\}$:        {keep don't cares}
        DC-set(k) = {Test = x} × DC-set(k) ∪ $\{0,1\}^{p+r+1}$ \ (ON-set(k) ∪ OFF-set(k));
    Finally minimize the logic again;
```

Fig. 6: Inclusion of an emulated scan path in a circuit structure.

4 Test of Circuits with Emulated Scan Path

An emulated scan path guarantees that all states are controllable and observable in r steps. The problem with the integration of scan path transitions into the system logic is that the shift mode cannot be simply validated using a shift register test.

Example 4: A 2-bit shift register is tested by the state sequence xx→1x→11→01→00→10 →x1→ xx, because for both flipflops all the possible transitions are checked. If the shift logic is merged with the system logic, it might happen that nevertheless the shift transition 00→00 does not work, as the flipflops do not only depend on the preceding flipflop in the chain. To detect that fault requires to scan in 00 into the chain. If during the scan process the faulty transition 00→00 is used, the fault might be masked.

In this section we first give an upper bound on the test length for validating the scan functionality, then we show that in practice this bound is very conservative and much shorter test lengths may be obtained. It is possible to safely use transitions hidden in the system logic to reach and identify certain states by first performing a functional test of these transitions, i. e. a checking experiment [Henn 64], which gives us the desired upper bound. The values of the primary inputs when activating the "shift" transitions are arbitrary but fixed (see section 3).

Lemma 4: [Chri 75] A strongly connected digraph contains a Euler cycle[2] iff for all vertices v_i the input degree is equal to the output degree, $\delta^-(v_i) = \delta^+(v_i)$.

Lemma 5: The state transition graph of the emulated shift register contains a Euler cycle.

Proof: The state transition graph of the shift register is strongly connected. Every shift register state has exactly 2 predecessors and 2 successors, as all inputs except for the shift input have fixed values while shifting. Therefore the input degree of each state vertex v_i is equal to its output degree, $\delta^-(v_i) = \delta^+(v_i) = 2$. ■

Theorem 6: The correct function of all shift transitions can be checked with the help of a checking experiment needing $2\cdot(n+r)$ test patterns[3].

Proof: For a shift register with n states 2n state transitions have to be checked, as from every state two transitions are possible, in which either a 0 or a 1 is serially shifted into the storage elements. There exists a sequence of state transitions of length 2n, in which each of these transitions is contained exactly once (all transfer sequences happen to be of length 0), hence this sequence constitutes a Euler cycle in the state transition graph of the shift register. The r outputs following a certain state transition are the contents of the r storage elements in the goal state of the transition, therefore each input sequence of length r is a distinguishing sequence for the goal state. If one applies r additional arbitrary patterns after visiting the 2n state transitions, a distinguishing sequence for the last r goal states is obtained, the distinguishing sequence for the first 2n – r goal states was already produced automatically while visiting the 2n state transitions. To initialize the shift register to a known state at the be-

[2] In a strongly connected digraph there exists a directed path from each vertex to each other vertex. An Euler cycle in a digraph is a cycle which contains all edges of the digraph exactly once.

[3] It has to be assumed that the number of states n in the circuit is not increased by any of the faults.

ginning, r additional state transitions are necessary. The sum of lengths for initialization sequence, Euler cycle and distinguishing sequence for the last r goal states is $r+2n+r$. ■

These functional test patterns are easy to produce and identical for all circuits with the same number of flipflops. $2\cdot(n+r)$ also gives an upper bound on the number of test patterns needed to test the scan path independent of its structure. Even though this number grows linearly with the number of states, it guarantees acceptable test lengths for circuits typically modelled as FSMs.

Example 4 (cont'd): One possible checking sequence for a 2-bit shift register is xx→1x→11→11→01→00→00→10→01→10→11→x1→xx, $2\cdot(4+2) = 12$ input patterns are necessary to obtain this sequence.

This upper bound is, however, quite conservative. In practice a shorter structural test of the scan path will be preferred. It can be generated with a sequential test generator and the process is simplified by several characteristics of an emulated scan path: All the inputs except for the shift input can be set to their constant shift mode values. Since all parts of the circuit, that only depend on these constant inputs, do not contribute to the shift transitions, they can be discarded during fault injection and the fault set for test generation can be reduced. (These faults only have to be treated by the combinational test generator for the non-scan logic). Thus sequential test generation is speeded up, all the more so because the controllability and observability of all the states has been increased by the additional state transitions. To further shorten the time for sequential test generation, a shift register test sequence (cf. example 4) can be applied beforehand. It uncovers all the faults not involving complex dependences between state variables. Experimental evidence shows that most or all of the scan path faults can already be detected this way.

5 Extension to Partial Emulated Scan Paths

The extension of the approach to partial scan paths is straightforward. After state assignment (e. g. using the approach in [ChAg 89]) and logic minimization an initial dependence graph DG is obtained. The minimal number of vertices is removed from $\mathcal{F}$ to make the dependence graph cycle free [ChAg 89a, KuWu 90] and thus get a bound on the length of test sequences increasing linearly with the circuit size. The vertices removed are the flipflops, which have to be included in the scan path. Since only for these vertices new dependences can be created by adding the state transitions of the emulated scan path, the dependence graph without these vertices stays cycle free after including the additional state transitions. Therefore the length of the test sequence for a fault is still linearly bounded by the circuit size. To determine the shift transitions, all other vertices corresponding to flipflops outside of the partial "scan path" are removed from the original dependence graph DG. The subsequent steps correspond to those in section 3.

6 Results

Compared with a conventional scan path, by using the proposed method the two connections for shift input and output can be saved. Switching speed is not decreased by adding gates to multiplex the different flipflop inputs, a two-level realization is not transformed into a multi-level realization by the additional test hardware. For a multi-level design style the conventional scan path is a special case of the realization of the combinational logic. The conventional scan path is also the worst possible result obtainable if none of the shift transitions can be minimized with the state transitions in system mode. In a two-level realization the worst case results in a number of additional product terms equal to the number of storage elements plus 1. To utilize the optimization potential, additional design effort is necessary, which mainly seems to be justified for smaller circuits with complex structural dependences. Since the behavior of both scan path variants cannot be distinguished functionally, it is possible to mix them in one circuit, e.g. to equip the data path with a conventional scan path and to optimize the controller with an emulated scan path.

A collection of sequential circuits (cf. [MCNC 88]) was synthesized with an emulated scan path*. The number of transistors needed for the circuit realization is compared with a conventional LSSD solution in Table 1. For control circuits modelled as finite state machines like those in Table 1 LSSD overheads are usually high and can be reduced by using an emulated scan path. For such circuits the additional synthesis effort is small and a shift register test can be performed very efficiently, hence the solution with emulated scan path is generally preferable. In few cases, however, circuits synthesized this way might need more transistors than with a conventional scan path (cf. example *dk16*), mainly because of weaknesses in current multi-level synthesis programs not giving an optimal logic decomposition.

example	number of inputs/ outputs/states	transistor count without scan path	increase with ... emulated s. p.	LSSD s. p.
agch	2 / 4 / 4	106	11.3 %	15.1 %
s27	4 / 1 / 8	106	5.7 %	28.3 %
bbara*	4 / 2 / 10	156	14.1 %	20.5 %
dk512	1 / 3 / 15	224	8.9 %	14.3 %
tap*	1 / 9 / 16	208	10.6 %	15.4 %
keyb*	7 / 2 / 19	466	5.6 %	8.6 %
dk16	2 / 3 / 27	642	7.5 %	6.2 %

Table 1: Comparison of emulated and conventional scan path (s. p.).

For the TAP controller from [IEEE 90] already used in example 3, layouts for the two solutions were created using a standard cell design system. The results are shown in Table 2. The delay on the critical path of the solution with emulated scan path is 6.7 % higher than without scan path, for a conventional scan path, however, the increase is 23.3 %. By performing a shift register test followed by a test of the combinational logic using the emulated scan path to shift patterns in and out, in this example all the shift transitions of the emulated scan path can be validated. With the shift register test already 53 % of the overall number of faults in the next state logic are detected, since contrary to a conventional scan path the system logic is exercised during the scan path test as well. This leads to a reduction of test application time from 64 clock cycles (conventional scan path) to 40 clock cycles (emulated scan path). The results show that an emulated scan path can result in significant improvements even for small examples.

* The examples marked with a star were realized with a separate shift output.

solution alternative	area in λ^2		crit. path next state / output logic	
scan path (cf. [IEEE 90])	560×691	386960	37	21
emulated scan path	520×667	346840	32	21

Table 2: Layout results for boundary scan TAP controller.

7 Conclusions

There exists a number of possibilities to incorporate a scan path into a circuit, realizing it as a separate structural entity is not the only way. By adapting the scan path to the circuit to be tested and by using the degrees of freedom inherent in a typical circuit description, an "emulated scan path" can be implemented with less overheads in terms of area and / or speed. The presented method of synthesizing sequential circuits makes use of this flexibility. Emulated scan paths are particularly useful for circuits with complex structural dependences like controllers. If area is the major goal, emulated *partial* scan paths can provide even better solutions. It was also shown how to test the emulated scan transitions by devising a short checking experiment or generating an efficient structural test.

References

AgCh 90 V. Agrawal, K. Cheng: FSM Synthesis with Embedded Test Function; JETTA, pp. 212-228, 1990.

BRSW 87 R. Brayton, R. Rudell, A. Sangiovanni-Vincentelli, Albert R. Wang: MIS: A Multiple-Level Logic Optimization System; IEEE Trans. on CAD, vol. 6, pp. 1062-1081, 1987.

ChAg 89 K. Cheng, V. Agrawal: Design of Sequential Machines for Efficient Test Generation; Dig. Tech. Papers ICCAD, pp. 358-361, 1989.

ChAg 89a K. Cheng, V. Agrawal: An Economical Scan Design for Sequential Logic Test Generation; Proc. FTCS 19, pp. 28-35, 1989.

Chri 75 N. Christofides: Graph Theory: An Algorithmic Approach; Academic Press, London, 1975.

DMNS 90 S. Devadas, H. T. Ma, A. R. Newton, A. Sangiovanni-Vincentelli: Irredundant Sequential Machines Via Optimal Logic Synthesis; IEEE Trans. on CAD, vol. 9, pp. 8-18, 1990.

EsWu 90 B. Eschermann, H.-J. Wunderlich: A Synthesis Approach to Reduce Scan Design Overhead; Proc. 1st EDAC, p. 671, 1990.

GaJo 79 M. Garey, D. Johnson: Computers and Intractability; Freeman, New York 1979.

Goel 80 P. Goel: Test Generation Costs Analysis and Projections; Proc. 17th DAC, pp. 77-84, 1980

Henn 64 F. C. Hennie: Fault Detecting Experiments for Sequential Circuits; Proc. 5th Annual Symp. Switching Circuit Theory and Logical Design, pp. 95-110, 1964.

IEEE 90 IEEE Std. P1149.1: Standard Test Access Port and Boundary-Scan Architecture, 1990.

KuWu 90 A. Kunzmann, H.-J. Wunderlich: An Analytical Approach to the Partial Scan Problem; JETTA, vol. 1, pp. 163-174, 1990.

Lawl 76 E. Lawler: Combinatorial Optimization: Networks and Matroids; Holt, New York, 1976.

MCNC 88 Logic Synthesis and Optimization Benchmarks, Version 2.0; MCNC, December 1988.