A Reconfigurable Power-Conscious Core Wrapper and its Application to SOC Test Scheduling

Erik Larsson and Zebo Peng

Embedded Systems Laboratory

Linköpings Universitet, Sweden

Abstract[1]

This paper presents a novel reconfigurable power-conscious core test wrapper and discusses its application to optimal power-constrained SOC (system-on-chip) test scheduling. The advantage with the proposed wrapper is that at each core it allows (1) a flexible TAM (test access mechanism) bandwidths, and (2) a possibility to select the appropriate test power consumption. Our scheduling technique, an extension of a preemptive scheduling approach, produces optimal solutions in respect to test time, and selects wrapper configurations in a systematic way that implicitly minimizes the TAM routing and the wrapper logic. Experimental results show the efficiency of our approach.

1. Introduction

Test power consumption and test time minimization are becoming major challenges when developing test solutions for core-based designs. These problems can be tackled by:

- *design for low power testing* where the system is designed to minimize test power consumption, which allows consequently testing at a higher clock frequency [5, 21, 22, 24], and
- *test scheduling* where the tests are organized in such a way that the test time is minimized while considering test power limitations and test conflicts [1, 4, 6, 7, 8, 9, 11, 12, 13, 14, 15, 16, 17, 18].

A core-based design is composed of pre-defined cores, UDL (user-defined logic) blocks and interconnections. From a testing perspective, all testable units, that is core logic, UDL blocks and interconnections, are defined as cores. The TAM (Test Access Mechanism), the set of wires connecting the ATE (Automatic Test Equipment) and the cores under test, is responsible for the test data transportation. A wrapper is the interface between the TAM and a core and it can be in one of the following modes at a time: *normal operation*, *internal test* or *external test*. A core can be either wrapped or unwrapped, which means we have two types of tests: *wrapped core test* at *wrapped* cores and *cross-core test* (interconnection test) at *unwrapped* cores [19].

We have previously [17]:

- shown that the problem of scheduling *wrapped core tests* on the TAM is equivalent to independent job scheduling on identical machines,
- made use of a preemptive approach to produce a solution with optimal test time in linear time [2] using reconfigurable wrappers [15], and
- extended the scheduling algorithm to handle wrapped core tests and cross-core tests while preserving the production of an optimal solution in linear time.

The main advantages of our previous approach are that: (1) optimal test time is achieved in linear time, (2) the TAM routing is minimized by assigning a minimum of TAM wires per core, and (3) the reconfigurable wrappers are selected and inserted in a systematic manner to minimize the number of configurations, which minimizes the added logic.

In this paper, we propose a reconfigurable power-conscious (RPC) core test wrapper to regulate the test power at core-level, and we describe its application to optimal test scheduling. The main contributions are:

- the development of a RPC core test wrapper which combines the gated sub-chain scheme presented by Saxena *et al.* [24] and the reconfigurable core test wrapper introduced by Koranne [15],
- the integration of the RPC wrapper in the preemptive test scheduling approach, and
- the formulation of a power condition that, if satisfied, guarantees that our preemptive test scheduling scheme produces optimal test time.

The advantages of the proposed approach are that we can:

- regulate the power consumption at each individual core, which allows the test clock speed to increase,
- regulate the test power consumption at system level, which should be kept within a given value in order to reduce the risk of over-heating which might damage the system under test,
- select the best TAM size, and
- perform minimization of the TAM routing and the overhead when using the RPC wrapper.

The rest of the paper is organized as follows. Related work is reviewed in Section 2 and our reconfigurable power-conscious test wrapper is introduced in Section 3. In Section

[1] The research is partially supported by the Swedish National Program on Socware.

4 we present our optimal preemptive test scheduling technique, based on a known preemptive scheduling algorithm [2]. In the experiments we have made a comparison with previous approaches and we illustrate the advantages with our wrapper and its use in the proposed scheduling approach in Section 5. The paper is concluded with conclusions in Section 6.

2. Background and Related Work

Figure 1 shows an example of a core-based system consisting of five cores, a TAM and an externally placed ATE (serving as test source and test sink). The test source, where the test stimulus are stored, and the test sink, where the test responses are saved, are connected to the N_{tam} wire wide TAM. The test source feed test stimulus to the cores via the TAM and the produced test responses from the cores are also transported via the TAM to the test sink. The cores are equipped with a test method; core 1 is, for instance, scan tested. A wrapper is the interface between a core and the TAM. Its main task is to connect the scan chains and the wrapper cells at a core into a set of wrapper chains, which are connected to the TAM. Cores with a dedicated wrapper, such as core 1, 2, and 3, are wrapped while cores without a dedicated wrapper, such core 4 and 5, are unwrapped. A core test is a test at a wrapped core, while a cross-core test is a test at an unwrapped core. The execution of a core test and a cross-core test therefore differs from each other. A core test is performed by placing its wrapper cells in internal test mode, and the test stimulus are transported direct from the test source via the TAM to the core. The test responses are transported from the core via the TAM to the test sink. The testing of an unwrapped core (such as core 4 in Figure 1), cross-core test, requires that the wrapper at core 1 and the wrapper at core 2 are both placed in external test mode. Test stimulus are then transported from the test source on the TAM via the wrapper cells at core 1 to core 4. The test responses are transported from core 4 via the wrapper cells at core 2 and the TAM to the test sink.

In the particular example given in Figure 1, the testing of core 1 and core 4 cannot be performed concurrently due to that in both cases the wrapper at core 1 is needed, and a wrapper can only be in one mode at a time. In general, test conflicts such as this one must be considered in the scheduling process. There are mainly two types of test conflicts: (1) TAM wire conflicts and (2) core wrapper conflicts, respectively.

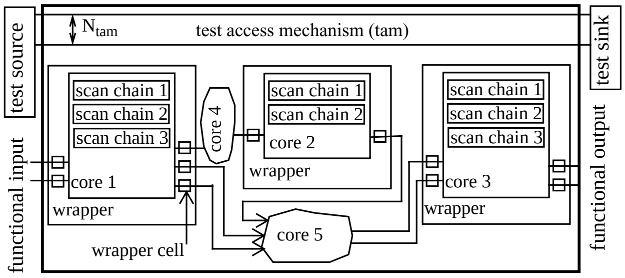


Figure 1. An example system with three wrapped cores (1,2,3) and two unwrapped cores (4,5).

Several test scheduling techniques have been proposed [1, 4, 6, 7, 8, 9, 11, 12, 13, 14, 15, 16, 17, 18]. Chou *et al.* proposed a technique for general systems where each test has a fixed test time and a fixed power consumption value [4]. The objective is to organize the tests in such a way that the total test application time is minimized while test conflicts and test power limitation are taken in to account [4]. The testing time for a test can often be modified. In scan testing, for example, the test time at a core can be modified by adjusting the number of wrapper chains that the scanned elements are configured into. A low number of wrapper chains at a core reduces the number of occupied TAM wires at the expense of higher testing time. And vice versa. Several wrapper chain configuration and TAM wire assignment algorithms to minimize the test time for *core tests* at *wrapped cores* have been proposed [6, 7, 8, 9, 11, 12, 13, 14, 15, 16]. For instance, Iyengar *et al.* [12] used an ILP (Integer-Linear Programming) approach. Koranne [15] introduced a reconfigurable wrapper with the advantage of allowing N_{tam} wrapper chain configurations per wrapped core. In order to minimize the added overhead due to the flexible wrapper, a limited number of cores are selected to have a reconfigurable wrapper prior to the scheduling. Iyengar *et al.* [11] and Huang *et al.* [9] proposed scheduling techniques for core tests under power constraints with fixed test power consumption for each test. Nicolai and Al-Hashimi [21] proposed a technique to minimize useless switches, and Gerstendörfer and Wunderlich [5] introduced a technique to disconnect the scan-chains from the combinational logic during the shift process, both leading to a lower power consumption during test, which reduces test time by allowing clocking at a higher frequency. For the same purpose, Saxena *et al.* [24] proposed an approach to gate sub-chains.

3. A Novel Reconfigurable Power-Conscious Core Test Wrapper

The RPC (reconfigurable power-conscious) test wrapper we propose combines the gated sub-chain approach proposed by Saxena *et al.* [24] and the reconfigurable wrapper introduced by Koranne [15]. The basic idea in the approach proposed by Saxena *et al.* [24] is to use a gating scheme to lower the test power dissipation during the shift process. Given a set of scan-chains as in Figure 2 where the three scan-chains are connected into a single chain. During the shift process, all scan flip-flops are active, leading to high switch activity in the system and therefore high power consumption. However, if a gated sub-chain scheme is introduced (Figure 3), only one of the three chains is active at a time during the shift process. The advantage is that the switch activity is reduced in the scan-chains and also in the clock tree distribution while the test time remains the same in the two cases [24].

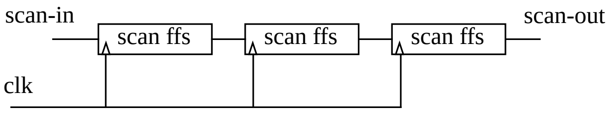


Figure 2. Original scan chain [24].

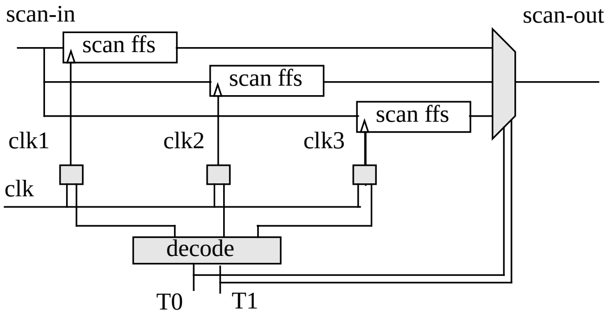


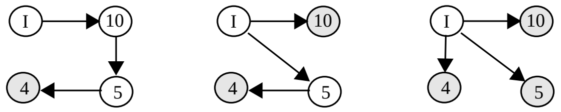
Figure 3. Scan chain with gated sub-chains[24].

The wrapper proposed by Koranne allows, in contrast to approaches such as Boundary Scan, TestShell and P1500, several wrapper chain configurations [15]. The main advantage is the increased flexibility in the scheduling process. We use a core with 3 scan chains of length {10, 5, 4} to illustrate the approach. The scan-chains and their partitioning into wrapper chains are specified in Table 1.

TAM width	Wrapper chain partitions	Max length
1	[10,5,4]	19
2	[(10),(5,4)]	10
3	[(10),(5),(4)]	10

Table 1. Scan chain partitions.

For each TAM widths (1, 2, and 3) a di-graph (directed graph) is generated where a node denotes a scan-chain and the input TAM, node I (Figure 4). An arc is added between two nodes (scan-chains) to indicate that the two scan-chains are connected, and the shaded nodes are to be connected to the output TAM. A combined di-graph is generated as the union of the di-graphs. Figure 5 shows the result of the generated combined di-graph from the three di-graphs in Figure 4. The indegree at each node (scan-chain) in the combined di-graph gives the number of signals to multiplex. For instance, the scan chain of length 5 has two input arcs, which in this example means that a multiplexer selecting between an input signal and the output of the scan chain of length 10 is needed. The multiplexing for the example is outlined in Figure 6.



(a) 1 wrapper-chain (b) 2 wrapper-chains (c) 3 wrapper-chains

Figure 4. Di-graph representations.

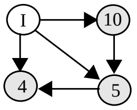


Figure 5. The union of di-graphs in Figure 4.

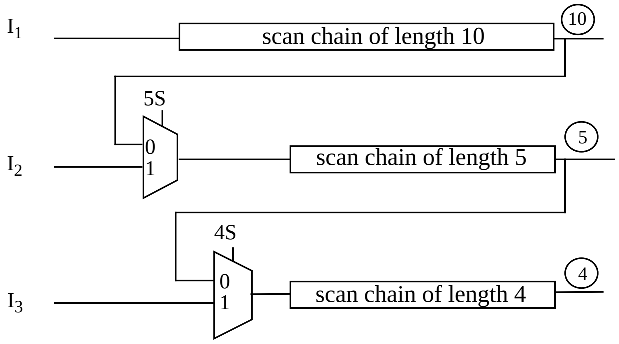


Figure 6. Multiplexing strategy [15].

Our combined approach works in two steps. First, we generate the flexible wrapper using Koranne's approach. Second, we add clock gating, which means we connect the *inputs* of each scan-chain to the multiplexers, which is to be compared to connecting the *outputs* of each scan-chain as in the approach by Koranne. We illustrate our approach using the scan chains specified in Table 1. The result is given in Figure 7, and the generated control signals are in Table 2.

The advantages with our approach are that we gain control of the test power consumption at each core, and we do not require the extra routing needed with Koranne's approach, as illustrated in Figure 8.

We could make use of the RPC wrapper at all cores, which would lead to a high flexibility since we could reconfigure the wrapper into any configuration. However, in

Wrapper chains	T0 T1 T2	5S 4S	S1 S2	clk10 clk5 clk4
3	0 0 0	1 1	0 0	1 1 1
2	0 0 1	1 x	0 0	1 0 0
	0 1 0	1 0	0 1	0 1 1
1	0 1 1	x x	0 x	1 0 0
	1 0 0	0 x	1 0	0 1 0
	1 0 1	0 0	1 1	0 0 1

Table 2. Control signals.

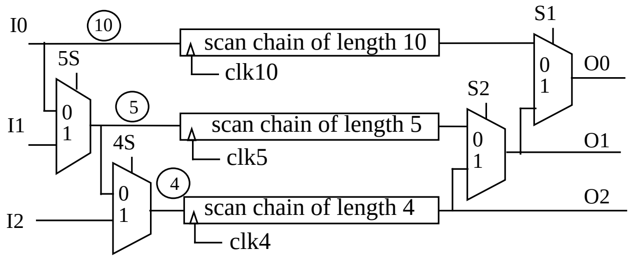
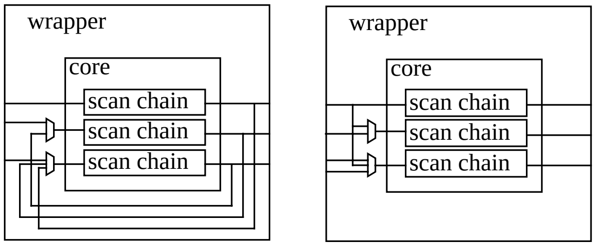


Figure 7. Our multiplexing and clocking strategy.



(a) Koranne's routing [15]. (b) Our approach.

Figure 8. Wrapper routing.

order to minimize the overhead, we will use a systematic approach to select cores and number of configurations at each core (described below).

4. Optimal Power-constrained Test Scheduling

In this section we describe our power-constrained test scheduling technique that produces optimal test time in linear time while scheduling both core tests and cross-core tests. The approach also selects the wrapper configurations.

4.1 Optimal Scheduling of Core Tests

The test scheduling problem of core tests is equal to the independent job scheduling on identical machines since each test t_i at a core c_i, (i=1, 2, ..., n) with testing time τ_i is independent on all other core tests, and each TAM wire w_j (j=1, 2, ..., N_{tam}) is an independent machine used to transport test data [17]. The LB (lower bound) of the test time for a given TAM width N_{tam} can be computed by [2]:

$$LB = max\left\{ max(\tau_i),\ \sum_{i=1}^{n} \tau_i / N_{tam} \right\} \qquad 1$$

The problem of independent job scheduling on identical machines can be solved in linear time ($O(n)$ for n tests) by using preemption [2]: assign tests to the TAM wires successively, assign the tests in any order and preempt tests into two parts whenever the LB is reached. Assign the second part of the preempted test on the next TAM wire starting from time point zero.

An example (Figure 9) illustrates the approach where the five cores and their test times are given. The LB is computed to 7 (Equation 1) and due to that $\tau_i \leq$LB for all tests; the two parts of any preempted test will not overlap. The scheduling proceeds as follows: The tests are considered one by one, for instance, starting with a test at c_1 scheduled at time point 0 on wire w_1. At time point 4, when the test at c_1 is finished, the test at c_2 is scheduled to start. At time point 7 when LB is reached, the test at c_2 is preempted and the rest of the test is scheduled to start at time 0 on wire w_2. Therefore the test at c_2 is partitioned into two parts. In execution of the test at c_2, the test starts at wire w_2 at time point zero. At time point 2, the test is preempted and resumed at time point 4. The test ends at time point 7. At the preemption of a test, another wire is assigned to the core and a multiplexer is added for wire selection. For the test of c_2, a multiplexer is added to select between w_1 and w_2.

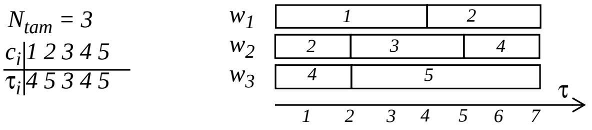


Figure 9. Optimal TAM assignment and preemptive scheduling.

In general preemptive scheduling, extra time is introduced at each preemption point due to the need to set up a job and also to save its state. In our case, the machines are the wires and no extra time is needed to set up and save the state. Also, in testing no other tasks are performed at the cores but testing, *i.e.* the core's state can be left as it is until the testing continues. The advantage is that the state of the core is already set and testing of it can start at once.

Assume that a core has a wrapper-chain of length l (l cycles are needed to perform a shift-in of a new vector and a shift-out of the previous test response). If the test is preempted when x% of the l cycles are shifted in it means that when the test restarts x% of the new test vector is already loaded and x% less cycles are needed in the first shift process, *i.e.* there is no time overhead due to setting up and saving the state of a core; all tests can be stopped at LB.

Finally, in some cases, such as for some types of memories such as DRAMs, the testing cannot be preempted. For instance, assume that test t_2 cannot be preempted as in Figure 9. In such a case, when LB is met, the scheduling algorithm restarts at LB (and not at time 0) and moves towards zero. The resulting schedule is in Figure 10. Note that, test t_2 now makes use of one wire during time point 4 to 5 and two wires during time 5 to 7, which is possible using the reconfigurable wrapper. This overlapping is further discussed below.

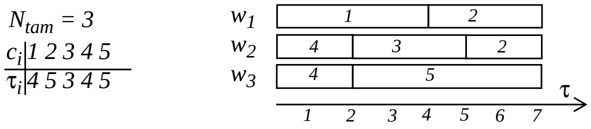


Figure 10. Optimal TAM assignment and preemptive scheduling where test t_2 cannot be interrupted.

4.2 Transformations for Optimal TAM Utilization

A long test time for one of the test in the system may limit the solution, *i.e.* LB is given by the test time of a test ($max(\tau_i)$ in Equation 1). In such a case, the test time can be reduced by assigning more TAM wires so that the length of the wrapper chains becomes shorter. Our approach is straight forward, we remove $max(\tau_i)$ from Equation 1:

$$LB = max \sum_{i=1}^{n} \tau_i / N_{tam}. \qquad 2$$

When LB is computed, we use the scheduling approach illustrated above (Figure 9). For illustration, we use the same example but with a wider TAM (N_{tam}=7). The scheduling result is presented in Figure 11. A test may now overlap in using the wires (machines). For instance, the test at c_1 uses wire w_1 and w_2 during time period 0 to 1 and only wire w_1 during period 1 to 3. A reconfigurable wrapper is required to solve this problem.

We solve the overlapping problem in two consecutive steps: partitioning of the tests, and inserting of reconfigurable wrappers. After assigning TAM wires to all tests, we determine the partitions, which is illustrated in Figure 11. For instance, in partition 1 of the test at c_2, w_3 is used during period τ_{21} and in partition 2 of the test at c_2, w_2 and w_3 are used during period τ_{22}. From this we can determine that two wrapper chains are initially needed and then a single wrapper chain is needed. In total, two configurations are needed for core c_2. The generic partitioning of a test's usage of wires over the testing time is given in Figure 12. For each test, a start time $start_i$ and an end end_i are assigned by the algorithm, respectively. The number of partitions, which will be the number of configurations, is computed for each test by the algorithm given in Figure 13. If the test time τ_i for a test t_i is below LB, only one configuration is needed. A multiplexer might be required for wire selection if $start_i$>end_i. From the algorithm, we find that the maximal number of partitions per test is three, which means we in the worst case have to use three configurations per core. The wrapper logic is then in range |C|×3×*technology parameter* (maximum 3 configurations per core). In the approach by Koranne [15] the added logic, if a reconfigurable wrapper is added at all cores, is given by |C|×N_{tam}×*technology parameter.*

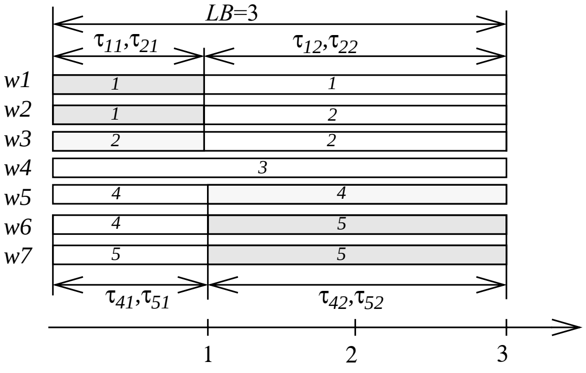


Figure 11. Partitioning of the schedule in Figure 11.

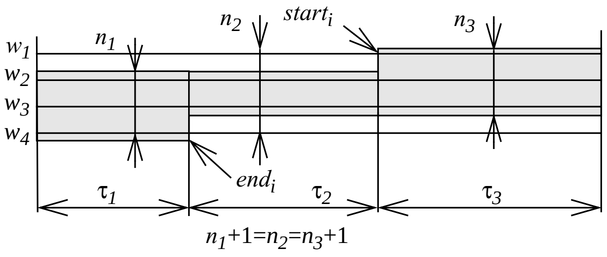


Figure 12. Bandwidth requirement for a general test.

```
for all $t_i$ begin
  if $\tau_i \leq$ LB then begin
    if $start_i \leq end_i$ then begin
      no pre-emption for $t_i \longrightarrow$ no wrapper logic is added.
    end else begin
      the test is split into two parts at different wires $\longrightarrow$
      1 multiplexer is inserted.
    end
  end else begin
    if $start_i \leq end_i$ then begin
      one configuration needed from time point 0 to $start_i$,
      one configuration needed from time point $start_i$ to $end_i$,
      one configuration needed from time point $end_i$ to LB,
    end else begin
      one configuration needed from time point 0 to $end_i$,
      one configuration needed from time point $end_i$ to $start$,
      one configuration needed from time point $start_i$ to LB,
    end
  end
end
```

Figure 13. Algorithm to determine wrapper logic.

4.3 Cross-Core Test Scheduling

There are no wrapper conflicts among *core tests* since each core has its dedicated interface to the TAM. For *cross-core tests,* on the other hand, there is not a dedicated interface to the TAM and wrapper conflicts must therefore be taken into account.

Test conflicts can be modelled using a resource graph [4]. The system in Figure 1 is modelled as a resource graph as in Figure 14, where the nodes represents the tests and the resources. A resource may consist of cores and wrapper cells, which are explicitly captured as shown in Figure 14. An edge between a test and a resource (core or a wrapper cell) indicates that the test requires the resource during testing. The test conflicts are due to that the wrapper cells can only be in one mode at a time: In *core testing* the wrapper cells are in internal test mode while in *cross-core test* they are in external test mode. In Figure 14, we denote arcs from core tests with (*i*) - *internal mode* and arcs from

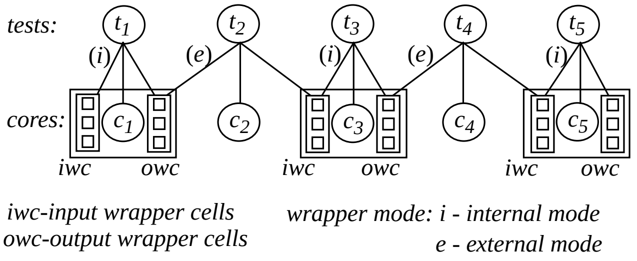


Figure 14. A resource graph of the system in Figure 1

cross-core tests with (*e*) - *external mode*. Breaking the graph into two resource graphs, one for core tests and one for cross-core tests makes it possible to schedule all core tests with the algorithm in section 4.1.

For a cross-core test, test vectors are transported from the TAM to a wrapped core placed in external test mode, which is feeding the test vectors to the unwrapped core (the target for the cross-core test). The test responses are captured at a wrapped core also in external test mode and then the test responses are fed to the TAM. A cross-core test involves therefore three cores. In Figure 1 a cross-core test at c_4 is performed by setting the wrappers at c_1 and c_2 in external test mode, and then test vectors are transported to c_4 through the wrapper at c_1 and the test response from c_4 is transported to the TAM using the wrapper at c_2. This demonstrates a cross-core test with a *one-to-one* mapping where the wrapper cells at the functional outputs at c_1 are connected via c_4 to the functional input wrapper cells at c_2.

Several other mapping combinations are possible for the wrapper input and wrapper output cells, including *one-to-many*, *many-to-one* and *many-to-many.* These mappings cover all combinations and we assume that each functional input and output can be in only one such mapping and in only one test set. In Figure 1, for instance, a functional output wrapper cell at c_1 cannot be in one test set with an input wrapper cell at c_5 and in another test set with an input cell at c_3. However, a wrapper cell at c_1 can be in the same test set as a wrapper cell at c_3 and at c_5. In some cases, the functional inputs and outputs at a wrapped core may be connected to different cores. Figure 1 shows such an example where the outputs at c_1 are partitioned into two sets, one set used by c_2 and c_4 and another set used by c_3 and c_5. However, these partitions operates independently when the wrapper is in external test mode. Therefore there is no conflict.

We have above shown that partitioning the tests into two distinct test phases, *core tests* and *cross-core tests,* eliminates the wrapper conflicts between the set of core tests and the set of cross-core tests. We make use of this property and divide the test scheduling into two separate parts, core testing followed by cross-core testing. The partition of the tests means we divide the tests into a core test part given by LB_{ct} and a cross-core test part given by LB_{ict}. To illustrate, we take the example in Figure 9 assuming that the test at c_2 and c_4 are cross-core tests, which means that executing the test at c_2 entails concurrent testing at c_1 and at c_3, and testing t_4 entails concurrent testing at c_3 and at c_5. The core tests are performed at core c_1, c_3 and c_5 and the cross-core tests are at core c_2 and c_4. The lower bound for the core tests: LB_{ct}=(4+3+5)/3=4 and the lower bound for the cross-core tests: LB_{ict}=(5+4)/3=3, *i.e.* $LB=LB_{ct}+LB_{ict}$. One test schedule is presented in Figure 15.

The test scheduling algorithm consists of four steps:

1. Compute LB_{ct} (lower bound) for the core tests,
2. Schedule all core tests,
3. Compute LB_{ict} for the cross-core tests, and
4. Schedule all cross-core tests.

The algorithm starts by selecting a core and assigning it to wire zero at time zero. If the core's test time is higher than LB, a new wire is used. The test time is reduced until it reaches zero and each time LB is reached, a new wire is added to the test. The start time and the end time of the tests are used when creating the partitions. We observe that the LB defines the test application time and also that all TAM wires are fully utilized, all tests ends at the same time (Figure 11). It means that partitioning the tests into two partitions (core tests and cross-core tests) will still produce an optimal solution.

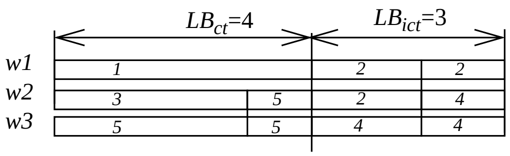


Figure 15. Partitioning of the schedule in Figure 11.

4.4 Optimal Power-Constrained Scheduling

Chou *et al*. introduced a power model where each test is denoted with a fixed power consumption value [4]. Recently a more elaborate model was presented by Rosinger *et al.* [23]. The power consumption depends on the switching activity in the circuit, and by reducing the switching activity, the power consumption is reduced. Saxena *et al*. [24] showed by experiments that sub-gating a single scan-chain into three sub-chains reduces the test power to approximately a third. Rosinger *et al*. proposed a technique to reduce both shift and capture power consumption. The experimental results indicate, in most cases, that the power consumption is lower than the intuitive approximation of dividing the consumption at a single chain with the number of partitions [22]. In this paper, we use a power model based on the results by Saxena *et al*, which means the power depends on the number and the length of the wrapper chain partitions. However, a more elaborate power model can easily be adopted in our approach.

We use an example to illustrate the test power modelling at scan-chain level (Figure 16). In Figure 16 (a) a single *wire* is assigned to the core where the three *scan chains* form a single *wrapper chain*. The result is that the wire usage is minimized but both the test time and the test power are relatively high. In Figure 16 (b) three TAM wires (one per wrapper chain) are used resulting in a lower test time while the test power consumption remain the same as in Figure 16(a). However, in Figure 16(c), our approach, the same test time is achieved as in Figure 16(a) but at a lower test power consumption. The reduction in test power in this

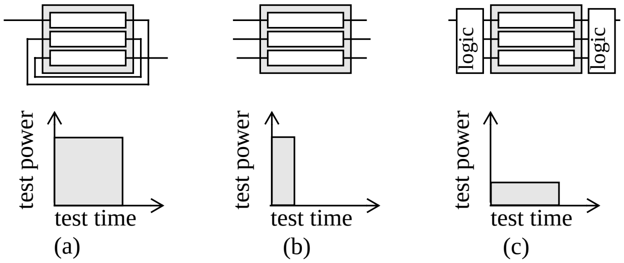


Figure 16. Core design alternatives.

example is due to that each scan-chain is loaded in a sequence, and not more than one scan-chain is activated at a time.

Our test scheduling technique [17] minimizes the number of TAM wires at each core by assigning as few wires as possible to each core. It means that each wrapper chain includes a high number of scanned elements. This is an advantage since it maximizes the possibility to gate scan-chains at each wrapper chain and hence control the test power consumption at the cores.

We assume that the test power at a core is evenly distributed over the scanned elements. The algorithm to compute the power limit (P_{limit}) for a system is in Figure 17. At step 2, the LB is computed, and at step 3, the maximal number of required TAM wires are computed. At step 4, the amount of test power consumed by each scan chain, and wrapper cell is computed. At steps 5 and 6, the N_{tam} values with highest test power are summarized which is the P_{limit}. If P_{limit} is below P_{max} ($P_{limit} \leq P_{max}$), optimal test time can be achieved.

1. Given: a system with *i* cores, where each core *i* consists of ff_i scanned flip-flops and wrapper cells partitioned into *j* partitions each of length sc_{ij} (including wrapper cells). The test time τ_i is computed as if all ff_i elements are connected to a single wrapper chain. The test power when all ff_i elements are active is given by p_i. N_{tam} is the TAM bandwidth.
2. Compute *LB* (*lower bound*) (algorithm in Section 4.3)
3. **For each** core *i* compute w_i as the maximal number of TAM wires that can be assigned to it assuming preemptive scheduling:
$$w_i = \left\lceil \frac{\tau_i}{LB} \right\rceil$$
4. **For each** scan-chain partition s_{ij} compute its power:
$$p_{ij} = \frac{p_i}{ff_i} \times sc_{ij}$$
5. **For all** cores: select the w_i scan elements with highest power value and sort them descending in a list *L*.
6. **For all** scan elements in L select the N_{tam} first and the P_{limit} is equal to the summation of the N_{tam} values.

Figure 17. Algorithm to compute the power limit.

We have now a relationship between the TAM bandwidth and the test power. The advantage is that we can use it when we determine the TAM bandwidth; N_{tam} can be increased as long as $P_{limit} \leq P_{max}$. It is also possible to increase the frequency of the test clock in order to minimize the test time as long as $P_{limit} \leq P_{max}$.

4.5 Minimization of TAM Wiring

The test scheduling approach above minimizes the number of TAM wires assigned to each core. The advantage is that, even if the floor-plan is unknown, the TAM routing cost is minimized since as few TAM wires as possible are assigned to each core. If the floor-plan is known, we can further minimize the TAM routing since the scheduling approach does not require any particular sorting of the tests. We take the system in Figure 9 with N_{tam}=7 resulting in a test schedule as in Figure 11 where the cores are sorted (and numbered) clock-wise as in Figure 18. The advantage is that neighbouring cores share TAM wires. For instance core 2, which makes use of TAM wire w_2 as soon as core 1 finish its use of w_2. Cores placed far away from each other are not sharing TAM wires, such as core 5 and core 3.

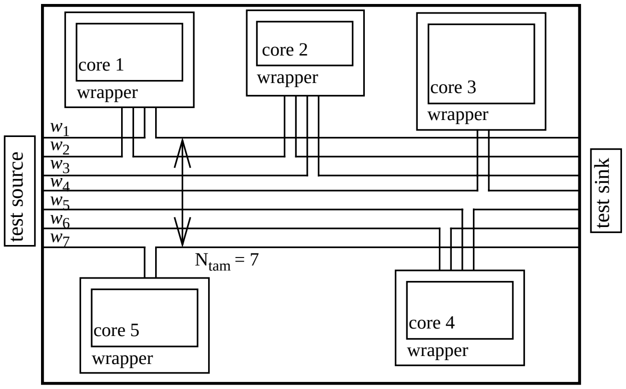


Figure 18. The example system assuming the five wrapped cores to be floor-planned.

5. Experimental Results

We have above shown that the test scheduling problem can be solved in linear time by using our proposed RPC wrapper. For illustration, we have made experiments using the P93791 design, one of the largest ITC'02 benchmarks [20]. We have made a test time comparison between our approach and techniques proposed by Goel and Marinissen [6, 7, 8], Huang *et al*. [9], Iyengar *et al.* [12, 13, 14], Koranne [15], and Koranne and Iyengar [16]. In our implementation we made use of the wrapper chain algorithm proposed by Iyengar *et al.* [12]. Similar to all previous approaches, we assume that all tests are core tests. The results are collected in Table 3. In some cases the previously reported results are below the lower bound computed by Goel and Marinissen [6]. These results are

Approach	**Test application time: *T***						
	TAM=16	**TAM=24**	**TAM=32**	**TAM=40**	**TAM=48**	**TAM=56**	**TAM=64**
Lower bound [6]	1746657	1164442	873334	698670	582227	499053	436673
Enumerate [12]	1883150	1288380	944881	929848	835526	537891	551111
ILP [12]	1771720	1187990	887751	(698583)	599373	514688	460328
Par eval [13]	1786200	1209420	894342	741965	599373	514688	473997
GRP[14]	1932331	131084	988039	794027	669196	568436	517958
Cluster [8]	-	-	947111	816972	677707	542445	467680
TRA [7]	1809815	1212009	927734	738352	607366	529405	461715
Binpack[10]	1791860	1200157	900798	719880	607955	521168	459233
CPLEX[15]	1818466	(1164023)	919354	707812	645540	517707	453868
ECTSP[15]	1755886	(1164023)	919354	707812	**585771**	517707	453868
ECTSP1[15]	1807200	1228766	967274	890768	631115	562376	498763
TB-serial [6]	1791638	1185434	912233	718005	601450	528925	455738
TR-serial [6]	1853402	1240305	940745	786608	628977	530059	461128
TR-parallel [6]	1975485	1264236	962856	800513	646610	540693	477648
K-tuple [16]	2404341	1598829	1179795	1060369	717602	625506	491496
Our approach	**1752336**	**1174252**	**877977**	**703219**	592214	**511925**	**442478**

Table 3. Test time comparison on P93791.

excluded and placed within parentheses. In six out of the seven TAM bandwidths our approach finds the solution with the lowest test time. In the other case, our approach is the second best. It should be noted that the lower bound of the test time is computed assuming that the last stimuli from the previous core can be shifted out on wires concurrently as the first vector of the following core is loaded on the same wires [6]. We have in our approach not considered such optimization. Hence, our approach can be further optimized.

We have in Table 4 collected the overhead due to the use of our reconfigurable wrapper. The overhead is computed as follows. For cores with a single TAM bandwidth assigned to it, only one bandwidth is required and the cost is assumed to be zero. In some cases, only a multiplexer is to be added for the selection between wires and we assumed such cost to be equal to 1. For cores with three configurations, we assumed the cost to be equal to 3. We have collected the overhead at each scanned core and for the rest of the cores. The test times for the cores without scan-chains are in general shorter and in only a few cases additional logic is required.

We have also made experiments considering test power consumption. First, we illustrate the use of the RPC wrapper at core 12 assuming a fixed test time at a single TAM wire (Table 5). The test time remains the same while the test power consumption can be adjusted depending on the number of gated wrapper-chains. The added wrapper logic depends on the number of wrapper chains which indicate how many partitions are to be gated. An advantage with our scheduling approach is that we assign as few TAM wires as possible to each core, which makes it possible to have a high ratio between the number of gated wrapper chains and the number of TAM wires at each core. In other words, we have a high possibility to regulate the test power at each core.

We have compared our approach with the multiplexing and the distribution architecture [1]. In the multiplexing approach all cores are tested in a sequence where the full bandwidth is given to each core at a time. In the distribution architecture, every core is given its dedicated TAM wires. The distribution architecture is sensitive to test power consumption since the testing of all cores are started at the same time. All results are collected in Table 6. The distribution architect is not applicable when the TAM bandwidth is below the number of cores (32 in P93791). At the 50000 power limit the distribution architecture can not be used since activating all cores exceeds the power limit. At the limit 20000, the multiplexing approach is not applicable since core 6 limits the solution with its consumption of 24674. Our approach results in the same test time, however, the wrapper logic is increased in order to gate the wrapper chains.

TAM band-width	Test time	Wrapper logic at core c_i													
		c_1	c_6	c_{11}	c_{12}	c_{13}	c_{14}	c_{17}	c_{19}	c_{20}	c_{23}	c_{27}	c_{29}	Cores with no scan chains	Total
4	6997584	0	1	0	0	0	1	0	0	1	0	0	0	0	3
8	3498611	0	3	0	1	0	1	0	1	1	0	1	0	1	9
16	1752336	1	3	0	1	1	1	1	1	3	1	1	1	1	16
24	1174252	2	3	0	3	3	3	3	1	3	3	3	1	1	29
32	877977	2	3	1	3	3	3	3	3	3	3	3	3	1+1	35
40	703219	2	3	0	3	3	3	3	3	3	3	3	3	3+1	36
48	592214	2	3	0	3	3	3	3	3	3	3	3	3	3+2	37
56	511925	2	3	0	3	3	3	3	3	3	3	3	3	3+2+3	40
64	442478	2	3	1	3	3	3	3	3	3	3	3	3	3+3+3	42

Table 4. Number of configurations per core for our scheduling approach on P93791 (Table 3).

6. Conclusions

In this paper we have proposed a reconfigurable power-conscious core test wrapper, and described its application to preemptive test scheduling. The main advantages with the wrapper is the possibility to (1) control the test power consumption at each individual core and (2) the possibility to achieve several TAM bandwidths for a given core test. Test power control at each core is important since it allows testing at a higher clock frequency, which can be used to further decrease the test time. Test power control also allows a higher number of cores to be tested simultaneously. Flexible bandwidth is important since it increases the flexibility during the scheduling process. The advantage of the proposed test scheduling scheme, besides the production of an optimal solution in respect to test time, is that it also considers cross-core testing, which are used to test unwrapped testable units such as interconnections and used-defined logic.

Wrapper chains	Power consumption	Wrapper logic
1	4634	0
2	2317	1
3	1545	3
4	1159	4
5	927	5
6	773	6

Table 5. Test power consumption options at core 12 (P93791) with the RPC wrapper at a fixed test time (=1813502) and fixed TAM bandwidth (=1).

References

[1] J. Aerts and E. J. Marinissen, “Scan Chain Design for Test Time Reduction in Core-Based ICs”, *Proceedings of International Test Conference (ITC),* pp. 448-457, October 1998.

[2] P. Brucker, “Scheduling Algorithms”, *Springer-Verlag*, ISBN 3-540-64105-X, 1998.

[3] M. L. Bushnell and V. D. Agrawal, “Essentials of Electronic Testing for Digital, Memory, and Mixed-Signal VLSI Circuits”, *Kluwer Academic Publisher,* ISBN 0-7923-7991-8.

[4] R. Chou, K. Saluja, and V. Agrawal, “Scheduling Tests for VLSI Systems under Power Constraints”, *Transactions on VLSI Systems*, Vol. 5, No. 2, pp. 175-185, June 1997.

[5] S. Gerstendörfer and H-J Wunderlich, “Minimized Power Consumption for Scan-Based BIST”, *Proceedings of International Test Conference (ITC)*, pp. 77-84, Atlantic City, NJ, September 1999.

[6] S. K. Goel and E. J. Marinissen, “Effective and Efficient Test Architecture Design for SOCs”, *Proceedings of International Test Conference (ITC),* pp 529-538, October 2002.

[7] S. K. Goel and E.J. Marinissen, “A Novel Test Time Reduction Algorithm for Test Architecture Design for core-Based System Chips”, *Digest of papers European Test Workshop (ETW)*, pp 41-46, Corfu, Greece, May 2002.

[8] S. K. Goel and E. J. Marinissen, “Cluster-Based Test Architecture Design for System-On-Chip, *Proceedings of VLSI Test Symposium (VTS),* pp. 259-264, Monterey, California, April 2002.

[9] Y. Huang, S. M. Reddy, W.-T. Cheng, P. Reuter, N. Mukherjee, C.-C. Tsai, O. Samman, and Y. Zaidan, “Optimal Core Wrapper Width Selection and SOC Test Scheduling Based on 3-D Bin Packing Algorithm”, *Proceedings of International Test Conference (ITC)*, pp. 74-82, Baltimore, Maryland, USA, October 2002.

[10] Y. Huang, W.-T. Cheng, C.-C. Tsai, N. Mukherjee, O. Samman, Y. Zaidan and S. M. Reddy, "Resource Allocation and Test Scheduling for Concurrent Test of Core-based SOC Design", *Proceedings of IEEE Asian Test Symposium (ATS)*, pp 265-270, Kyoto, Japan, November 2001.

[11] V. Iyengar, K. Chakrabarty, and E. J. Marinissen, "Wrapper/TAM Co-Optimization, Constraint-Driven Test Scheduling, and Test Data Volume Reduction for SOCs", *Proceedings of IEEE/ACM Design Automation Conference (DAC)*, pp. 685-690, New Orleans, Louisiana, June 2002.

[12] V. Iyengar, K. Chakrabarty, and E. J. Marinissen, "Co-Optimization of Test Wrapper and Test Access Architecture for Embedded Cores", *Journal of Electronic Testing; Theory and Applications (JETTA)*, pp 213-230, April 2002.

[13] V. Iyengar K. Chakrabarty, and E. J. Marinissen, "Efficient Wrapper/TAM Co-Optimization for Large SOCs", *Proceedings of Design and Test in Europe (DATE)*, pp. 491-498, Paris, France, March 2002.

[14] V. Iyengar, K. Chakrabarty, and E. J. Marinissen, "On Using Rectangle Packing for SOC Wrapper/TAM Co-Optimization", *Proceedings of VLSI Test Symposium (VTS)*, pp. 253-258, Monterey, California, April 2002.

[15] S. Koranne, "A Novel Reconfigurable Wrapper for Testing Embedded Core-Based and its Associated Scheduling", *Journal of Electronic Testing; Theory and Applications (JETTA)*, pp. 415-434, August 2002.

[16] S. Koranne and V. Iyengar, "On the Use of k-tuples for SoC Test Schedule Representation", *Proceedings of International Test Conference (ITC),* pp. 539-548, October 2002.

[17] E. Larsson and H. Fujiwara, "Optimal Test Access Mechanism Scheduling using Preemption and Reconfigurable Wrappers", *Proceedings of IEEE Workshop on RTL and High Level Testing (WRTLT)*, Guam, USA, November 21-22, 2002.

[18] E. Larsson and Z. Peng, "An Integrated Framework for the Design and Optimization of SOC Test Solutions", *Journal of Electronic Testing; Theory and Applications (JETTA)*, pp. 385-400, August 2002.

[19] E. J. Marinissen, R. Kapur, M. Lousberg, T. McLaurin, M. Ricchetti, and Y. Zorian, "On IEEE P1500's Standard for Embedded Core Test", *Journal of Electronic Testing: Theory and Applications, (JETTA)*, vol. 18, pp 365-383, August 2002.

[20] E. J. Marinissen, V. Iyengar, and K. Chakrabarty, "ITC'02 SOC Test Benchmarks Web Site", http://www.extra.philips.com/itc02socbench/.

[21] N. Nicolici and B. M. Al-Hashimi, "Power Conscious Test Synthesis and Scheduling for BIST RTL Data Paths", *Proceedings of International Test Conference (ITC)*, pp. 662-671, Atlantic City, NJ, Oct. 2000.

[22] P. M. Rosinger, B. M. Al-Hashimi, and N. Nicolici, "Scan Architecture for Shift and Capture Cycle Power Reduction", *Proceedings of IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems*, pp. 129-137, 2002.

[23] P. M. Rosinger, B. M. Al-Hashimi, and N. Nicolici, "Power Profile Manipulation: A New Approach for Reducing Test Application Time under Power Constraints", Transactions on Computer-Aided Design of Integrated Circuits and Systems, pp. 1217-1225, October 2002.

[24] J. Saxena, K. M. Butler, and L. Whetsel, "An Analysis of Power Reduction Techniques in Scan Testing", *Proceedings of International Test Conference (ITC)*, pp. 670-677, Baltimore, MD, October 2001.

P_{max}	TAM	Multiplexing architecture	Distribution architecture	Our approach
		Test time	Test time	Test time
100000	4	7113317	Not applicable	6997584
	8	3625510	Not applicable	3498611
	16	1862427	Not applicable	1752336
	24	1262427	Not applicable	1174252
	32	1210398	5317007	877977
	40	1119393	1813502	703219
	48	660143	1126316	592214
	56	645698	907097	511925
	64	645682	639989	442478
50000	4	7113317	Not applicable	6997584
	8	3625510	Not applicable	3498611
	16	1862427	Not applicable	1752336
	24	1262427	Not applicable	1174252
	32	1210398	Not applicable	877977
	40	1119393	Not applicable	703219
	48	660143	Not applicable	592214
	56	645698	Not applicable	511925
	64	645682	Not applicable	442478
20000	4	Not applicable	Not applicable	6997584
	8	Not applicable	Not applicable	3498611
	16	Not applicable	Not applicable	1752336
	24	Not applicable	Not applicable	1174252
	32	Not applicable	Not applicable	877977
	40	Not applicable	Not applicable	703219
	48	Not applicable	Not applicable	592214
	56	Not applicable	Not applicable	511925
	64	Not applicable	Not applicable	442478

Table 6. Test time on P93791 for the multiplexing architecture [1], the distribution architecture [1], and our approach at different power limitations.