I. Introduction

A System-On-Chip design usually contains multiple cores glued together in a bus structure. The cores can be third party's IP's, legacy cores, memories and user-defined logic. Testing a SoC has become a bottleneck and can become a nightmare if not dealt with properly.

One of the major tasks in SoC testing is test scheduling. In test scheduling, tests are scheduled to minimize test time taking into account a number of constraints like power dissipation, precedence, and resource sharing.

II. Problem description

Considering a SoC design consisting of multiple IP cores. Core tests are divided into two categories: (1) external tests, and (2) BIST tests.

External tests are applied to the core under test by external ATE (automatic test equipment) via specialized test access mechanism (TAM). Here, we assume that the SoC under test uses a bus-based TAM structure as shown in Fig. 1. In this example, the uP core, Memory 1, and cores 1 to N are connected to the TAM bus. During an external test session, a portion of the TAM bus bandwidth is allocated to the core under test for test data transportation. Multiple external tests can be executed concurrently if there is no TAM bus usage confliction, i.e., if the same TAM bus wire is allocated to core a and b, then the external test sessions of core a and b cannot overlap.
BIST tests, on the other hand, do not rely on TAM for test data transportation. In Fig. 1, Memories 2, 3, and 4 are not connected to the TAM bus because they rely on BIST tests. (In practice, one may choose to also connect them to the TAM bus so that both BIST and external tests can be performed.) To realize BIST tests, Memories 2 and 3 share the same BIST engine. Memory 4, on the other hand, uses its own BIST circuitry. During a BIST test session, the ATE only has to supply the test control signals and receive a limited amount of test results (not shown in Fig. 1). Usually, a dedicated low-bandwidth TAM is allocated for this purpose, which is not considered as a constraint in this problem.

In this SoC test scheduling problem, you are required to find a test schedule and TAM bus assignments to minimize the total test time without violating any of the given constraints. For the test schedule part, you are to determine the start time of each core test. For the TAM bus assignment part, you must determine which TAM bus wires are connected to each core that has external tests. Note that a TAM bus wire may be connected to multiple cores.

The constraints that you have to consider are:
1. If two external core tests use the same TAM bus wire, then they cannot overlap.
2. For a core with multiple tests, only one test can be performed at a time.
3. A BIST engine (a test resource) can be shared by multiple cores. BIST tests that use the same BIST engine cannot overlap.
4. If a core has both BIST and external tests, the BIST tests must be performed prior to external tests. The user may also specify that one test must be executed before another. (The order of the BIST or external tests of the same core can be specified this way.)
5. Assuming a simple additive model, if the peak power of test Ti and Tj are Pi and Pj respectively, then the peak power when Ti and Tj overlap is Pi + Pj. The peak power of the test schedule may not exceed a given peak power Pmax.

III. Input format
The inputs include the core test descriptions and the system description. A line starting with # is a comment line.

1. System description
The system description starts with the keyword System. The characteristics of the system is enclosed in a block between the begin and end keywords. In the following, keywords are bold-faced, and they are case-insensitive.

System
begin
  TAM_width 48
  # The system TAM bus width is 48.
  # The TAM bus wires are numbered 0, 1, 2, ..., 47
  Power 60
  # The maximum allowable power is 60.
  Precedence test_1 > test_2 > test_3
  # The precedence relationships specified here are:
  # 1. test_1 must be executed before test_2
  # 2. test_2 must be executed before test_3
  Precedence test_1 > test_4
  # More than one precedence constraint can be specified.
  # That the BIST tests of a core must be performed before
  # its external tests is not explicitly specified.
  Resource rsc1 rsc2
  # This SoC system has two BIST resources: rsc1 and rsc2
end

2. Core test description
The core description describes the core and its test set characteristics.

Core core_a
begin
  TAM_width 16
  # The core will occupy 16 bits of the available TAM bus
  # during an external test.

  # The description of external test test_1 of core_a.
  # The names of tests must be unique.
  External test_1 length 500 power 12 preemption 2
  # Specifying the parameters of core_a's external test_1:
  # length: 500 test patterns.
# power: The peak power consumption of this test.  
# preemption: This test allows two preemptions.  
#     The default is 0.  
#     Inclusion of utilizing "preemption" in  
#     your program is optional (see "evaluation  
#     criteria).  
# The order that the parameters are specified is not  
# important.

External test_2 length 1000  
External test_2 power 12  
# A core may have multiple tests.  
# Specifying test parameters in multiple lines is allowed.

BIST test_3 length 5000 power 100 resource rsc_1  
# "test_3" is a BIST test which require resource rsc_1.

end

Core core_b  
begin  
    TAM_width 24  
    External test_4 length 200 power 20  
    External test_5 length 1000 power 40  
End

IV. Output format

You may dump the outpus to a file and name it as you like. (The examples shown below are irrelevant to the above input format examples.)

Schedule  
begin  
    Test_time 10000  
    # Total test time of the test scheduling.

    TAM_assignment core_a [16:14,10] [8,7] [2:5] [0]  
    # There are many ways to describe the TAM bus assignments.  
    # Here, TAM bus bits 16, 15, 14, 10, 8, 7, 5, 4, 3, 2, 0  
    # are assigned to core_a.

    TAM_assignment core_b [16:11]  
    TAM_assignment core_b [10:9]  
    # TAM assignments can spread over a few lines.
**External** core_a test_a (500,999) (2000,3999)
# The external test "test_a" of "core_a" starts at
# time 500. It is stopped at time 1000 and resumed
# at time 2000 due to pre-emption. "test_a" is finally
# finished at time 3999.

**BIST** core_a test_b (0,399)
# Note that (0,399) corresponds to a test length
# (or time) of 400.

end

V. Examples

In the following, a test schedule example is given to better understand the problem.

# Beginning of inputs
System
begin
  TAM_width 24
  Power 1000
  Resource rsc_1 rsc_2
end
Core core_1
begin
  TAM_width 16
  BIST bist_1 length 325 power 10 resource rsc_1
  External ext_1 length 200 power 12
end
Core core_2
begin
  TAM_width 8
  BIST bist_2 length 150 power 10 resource rsc_2
  External ext_2 length 300 power 10
end
Core core_3
begin
  TAM_width 8
  BIST bist_3 length 300 power 10
  External ext_3 length 150 power 10
end
Core core_4
Begin
One possible solution is shown in Fig. 2. This is not an optimal solution. See if you (or your program) can find a better one.

The outputs corresponding to the test schedule in Fig. 2 are listed below:

```plaintext
# beginning of the test schedule
Schedule
begin
  Test_time 650
  TAM_assignment core_1 [0:15]
  TAM_assignment core_2 [0:7]
  TAM_assignment core_3 [8:15]
  TAM_assignment core_4 [16:23]
```
VI. Evaluation criteria

The evaluation of the program quality is based on the following criteria:
1. Correctness and robustness of the program.
2. Minimal total test time of the test scheduling.
3. CPU time and memory usage.
4. Bonus point for preemption test scheduling.
5. Bonus point for visual presentations of the test scheduling results.

VII. References