华为海思——故障仿真并行加速赛题指南

Department name: 华为海思 + 诺亚香港
Author’s name: 叶仕杰，甄慧玲，黄宇
Date: 2023.09.11

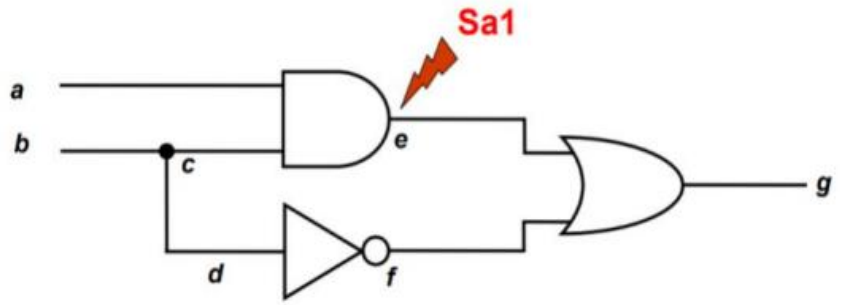
Security Level:

HUAWEI

Contents

赛题背景

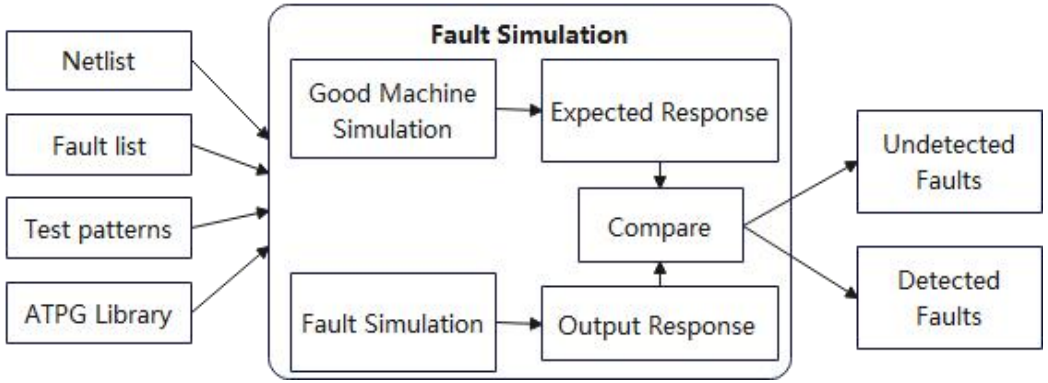
- 故障仿真（Fault Simulation）是电路测试和诊断的重要步骤，是DFT（Design for Test）的重要组成部分。
- 主要用于评估电路运行过程中可能出现的故障。故障仿真在测试向量生产、故障诊断、生产故障分析等环节中都发挥了至关重要的作用，确保芯片设计稳定可靠的同时，为芯片制造过程提供了必要的支持
- 如下图所示，如果电路的e点对电源短路，我们可以将这一电路缺陷抽象为名为Stuck-At-1的故障模型，也就是说：电路表现为不论a，b的输入如何变化，e点电平始终为1（即高电平）。



- 在芯片量产测试的ATE机台上，为了检测到这个芯片内部的缺陷，我们只能通过对输入a，b施加特定的激励，捕获并检测输出g的电平与预期的电平是否一致，以推断电路是否存在缺陷（可能是由于e点的Stuck-At-1故障导致）。
- 举个例子，当电路的e点出现对电源短路故障时，如果对a，b施加激励（a = 0, b = 1），此时，g的预期输出为0。但是，由于e点存在类型为stuck-at-1的fault，因此实际输出为1。因此，我们称e点的Stuck-At-1 fault能被pattern（a = 0, b = 1）检测到。

赛题内容解析

- 工业故障仿真的一般流程一般按照下图所示。



- 在本赛题中，我们的主要考察内容是故障仿真的效率（在保证正确性的前提下），其他内容则由出题方提供的的输入和运行脚本可直接运行用例。需要说明的是：
  - ATPG Library会由打包Case中提供；
  - Fault list为FAN_ATPG自动插入；
  - Pattern的格式为FAN_ATPG私有自定义.pat文件格式

```
G0 G1 G2 G3  |
U_G5 U_G6 U_G7  |
G17
BASIC_SCAN
_num_of_pattern_5
_pattern_1 0000 |  | 011 |  | 0 |  | 011
_pattern_2 0111 |  | 000 |  | 1 |  | 000
_pattern_3 1010 |  | 010 |  | 1 |  | 100
_pattern_4 1011 |  | 000 |  | 0 |  | 010
_pattern_5 0001 |  | 110 |  | 1 |  | 000
```

赛题内容解析

- 工业中大规模故障仿真主要围绕两部分：
- **内存挑战：**
  - ✓ 芯片工艺在微米级、纳米级集成之后，设计中的逻辑门数量、Scan Chain数量、Scan Cell数量、测试点连接数量等都呈现出大幅增长、庞杂化的趋势。
  - ✓ 这就使得传统的故障仿真方法所需内存随之增加，需要采用新的仿真平台和算法来满足内存需求。
- **Run Time挑战：**
  - ✓ 在硬件实现中，复杂电路的大小远远超过计算机的容量，软件实现的仿真性能比硬件还要低。
  - ✓ 当今电路设计中需要进行的Test Patterns数量较多，因此仿真过程需要大量的计算资源。

赛题内容解析

- 一般提升性能往往考虑以下几种途径：
  - **GPU并行加速：**GPU具有高并行计算能力，可以同时处理大量数据，可有效加快故障仿真的计算速度。
    - GPU的价格昂贵且功耗较高，需要使用特殊的编程模型（如CUDA），存在编程复杂性高和移植性差的问题；
  - **分布式并行**：相对于传统的集中式仿真，分布式故障仿真允许仿真任务在多个计算机或服务器上分配和并行运行，以提高仿真效率和性能。
    - 但需要考虑到多个节点之间的数据通信和同步开销等问题。
  - **多核CPU并行**：利用现代多核处理器的强大计算能力，将故障仿真分割为多个独立的任务并分配给不同的CPU内核。
  - 多核并行计算具有效率高、成本低等优点，且易于实现，是目前大规模设计测试中应用最广泛的手段
- 在本赛题中，我们主要考察依托华为的鲲鹏服务器，实现“高效的多核CPU并行故障仿真”。
  - 本赛题将采用多核CPU并行的方式，并在**鲲鹏ARM多核CPU架构服务器**进行打榜。
  - 打榜服务器CPU为Huawei Kunpeng920,主频2.6GHz,60vCPUs，内存120GiB。
  - 参赛队伍需要设计合理的并行算法方案，以减少故障仿真的时间和提高多核并行的Scalability。
- 参赛作品**必须保证Fault Simulation的正确性。**用例会通过命令接口导出Undetected Faults和Detected Faults，并检查其正确性。
  - Fault Simulation基础源代码及用例等相关资源可参见下列代码仓库。关于算法和系统的优化思路，可参考 Q&A。
  - https://github.com/NTU-LaDS-II/FAN_ATPG
  - 请仔细阅读开源代码的README，请勿随意修改系统的接口。
  - 整个基础代码里面接口有两类：一类是源代码算法接口；另一类是系统环境搭建脚本；README里面有详述。

提交要求

- 本赛题需要参赛队伍在开源项目（FAN_ATPG）提供的一系列的命令接口基础上开发多核并行故障仿真软件
- 本赛题打榜**只关注run_fault_sim命令的运行时间**，因此参赛队伍应关注故障仿真阶段的效率。
- 提交的作品需保证命令接口与开源项目保持一致，以确保用例能正常运行。
- 参赛队伍可以直接在FAN_ATPG项目上进行并行版本的开发，也可以接入自有的其他故障仿真模型，但接口需要与FAN_ATPG项目保持一致。
- 要求：参赛队伍需要分别提供一个单线程版本和一个多线程版本的故障仿真软件，所以需要**关注并行化算法以获得更高的多核并行加速比。**
- 本赛题的运行脚本如下：

```
read_lib techlib/mod_nangate45.mdt
read_netlist mod_netlist/s5378.v
report_netlist
build_circuit --frame 1
report_circuit
read_pattern pat/FAN_s5378.pat
report_pattern
set_fault_type saf
add_fault -a
```

用例初始化

```
run_fault_sim
```

故障仿真运行

```
report_fault -s UD > rpt/s5378_fsim_ud.rpt
report_fault -s DT > rpt/s5378_fsim_dt.rpt
exit
```

运行结果导出

评分机制

- 本赛题关注多核并行的加速比，按加速比进行打榜比赛。
  - 参赛队伍提交的作品包含一个单线程版本和一个多线程版本；
  - 这两个版本将运行同一组打榜用例，系统将记录每个用例的故障仿真阶段Run Time时间和内存占用峰值。
- 具体打榜分数的计算和排名规则如下：
  ① 单个用例的加速比 = （单线程版本Run Time / 多线程版本Run Time） * 因子，因子大小与用例Fault数量级正相关；
  ② 累加所有用例的加速比作为打榜分数，分数从大到小排序，分数相同的情况下，所有多线程版本用例的内存占用峰值累加值小的排前面；
  ③ FAN_ATPG原始代码的Run Time作为基线，如果提交作品Run Time大于基线的则按FAN_ATPG原始代码的Run Time作为计算。反之，则按提交作品的实际Run Time计算。
  ④ 对于某个用例的Fault输出不正确的情况，按FAN_ATPG原始代码的基线Run Time计算

评分举例

队伍1						
	Run time 单位(s)			因子	加速比	多线程 内存峰值 单位(M)
	基线	单线程	多线程			
用例1（Fault数量10万）	150	120	30	1	4	400
用例2（Fault数量20万）	320	300	100	2	6	800
用例3（Fault数量50万）	1000	800	320	5	12.5	2000
总计	-	-	-	-	22.5	3200

队伍1单线程版本Run Time小于基线的Run Time，则按单线程版本Run Time计算加速比，加速比总计22.5，内存总计3200M；

队伍2						
	Run time 单位(s)			因子	加速比	多线程 内存峰值 单位(M)
	基线	单线程	多线程			
用例1（Fault数量10万）	150	140	40	1	3.5	500
用例2（Fault数量20万）	320	315	84	2	7.5	1000
用例3（Fault数量50万）	1000	759	330	5	11.5	2600
总计	-	-	-	-	22.5	4100

队伍2单线程版本Run Time小于基线的Run Time，则按单线程版本Run Time计算加速比，加速比总计22.5，内存总计4100M；

队伍3						
	Run time 单位(s)			因子	加速比	多线程 内存峰值 单位(M)
	基线	单线程	多线程			
用例1（Fault数量10万）	150	180	50	1	3	380
用例2（Fault数量20万）	320	290	145	2	4	720
用例3（Fault数量50万）	1000	1200	666	5	7.5	1900
总计	-	-	-	-	14.5	3000

队伍3部分单线程版本Run Time大于基线的Run Time，则按基线的Run Time计算加速比，加速比总计15.5，内存总计3000M；

第一名：队伍1

第二名：队伍2

第三名：队伍3

Thank you.

Bring digital to every person, home and organization for a fully connected, intelligent world.