

Petrify 实现四段握手协议

一、首先，再试验阶段实现了 c 单元，发现是可用的。

1. c 单元的时序图个 STG 图如下图 1 所示：

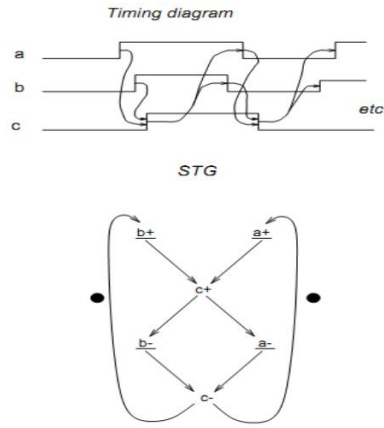


图 1 c 单元的时序图个 STG 图

2. c 单元的 c.g 文件如图 2 所示：

```
.outputs c
.inputs a b
.graph
b+ c+
a+ c+
c+ b-
c+ a-
b- c-
a- c-
c- b+
c- a+
.marking {<c-,b+> <c-,a+>}
.end
```

图 2 c 单元的 c.g 文件

3. 运行./petrify c.g -eqn c.eqn -cg -no 生成 c.eqn 文件如图 3 所示：

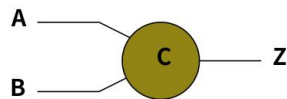
```
c.eqn
~/桌面/petrify-5.2-linux/petrify/bin
# EQN file for model c
# Generated by ./petrify 5.2 (compiled Tue  2 May 11:53:24 BST 2017)
# Outputs between brackets "[out]" indicate a feedback to input "out"
# Estimated area = 5.00

INORDER = a b c;
OUTORDER = [c];
[c] = (a b) + c (b + a);

# No set/reset pins required.
```

图 3 c.eqn 文件

4. 观察图 3 中 c.eqn 的结果，完全符合 c 单元图 4 中的真值表。可以替换之前代码中的 muller_C 模块。



真值表

A	B	Z
0	0	0
0	1	保持
1	0	保持
1	1	1

图 4 c 单元的真值表

二、因为两端握手信号的上升沿和下降沿表示同样的含义，因而信号的电平没有特定的含义。但是，数据通路内部的很多的电路需要电平敏感的控制，直接用四段握手协议会比较好。

1. 首先，根据四段握手协议 STG 图 5 所示，图中虚线箭头表示必须由环境保证的事件发生顺序，实现箭头表示必须由电路保证的事件发生顺序。“+”表示信号值由 0 到 1 的变化，“-”表示信号从 1 到 0 的变化。

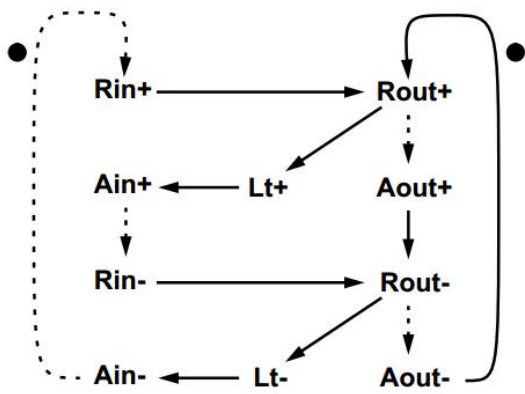


图 5 四段握手协议 STG 图

2.根据图 1 的 STG 图，写 phase_four.g 文件，如图 6 所示：



```
phase_four.g
~/桌面/petrify-5.2-linux/petrify/bin

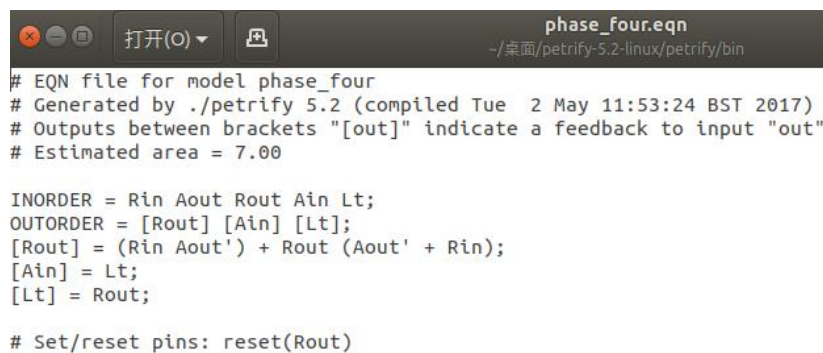
.outputs Rout Ain Lt
.inputs Rin Aout
.graph

Rin+ Rout+
Rout+ Lt+
Rout+ Aout+
Lt+ Ain+
Aout+ Rout-
Ain+ Rin-
Rin- Rout-
Rout- Lt-
Rout- Aout-
Lt- Ain-
Aout- Rout+
Ain- Rin+

.marking {<Aout-,Rout+> <Ain-,Rin+>}
.end
```

图 6 phase_four.g

3. 运行命令 `./petrify phase_four.g -eqn phase_four.eqn -cg -no` 生成 phase_four.eqn 文件，如图 7 所示：



```
phase_four.eqn
~/桌面/petrify-5.2-linux/petrify/bin

# EQN file for model phase_four
# Generated by ./petrify 5.2 (compiled Tue  2 May 11:53:24 BST 2017)
# Outputs between brackets "[out]" indicate a feedback to input "out"
# Estimated area = 7.00

INORDER = Rin Aout Rout Ain Lt;
OUTORDER = [Rout] [Ain] [Lt];
[Rout] = (Rin Aout') + Rout (Aout' + Rin);
[Ain] = Lt;
[Lt] = Rout;

# Set/reset pins: reset(Rout)
```

图 7 phase_four.eqn

4.此时应该根据图 7 的 phase_four.eqn 文件的逻辑来编写 verilog 文件来替换前面手写的握手协议，但是我之前用两端握手实现的流水线。 目前还没有对此进行替换。