OPNET 的 ODB 调试、动态展示以及 VC 联调

OPNET 的 ODB 调试是非常强大的一个调试方法,操作界面类似 DOS 窗口,或者 linux 窗 口,有很多指令可以选择,但是我觉得只用其中几个指令搭配上 VC 联调以及探测统计量, 我们在 OPNET 的调试中就可以无往而不利!

不过首先还是提供 ODB 调试的全过程链接吧:

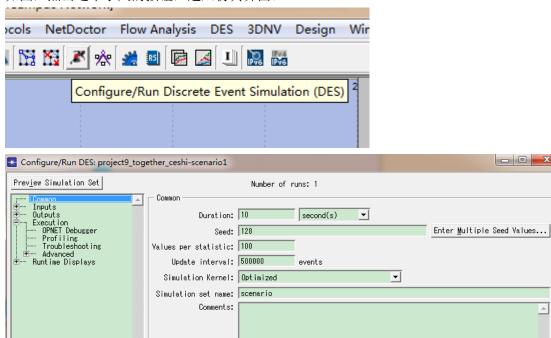
http://blog.163.com/zhoumhan 0351/blog/static/399542272009826105222389/

分三点介绍:

- 1、怎么进入 ODB 调试界面
- 2、ODB 动态展示
- 3、VC 联调

一、怎么进入 ODB 调试:

如图,点击这个小人的按钮,进入仿真界面:



_ D X

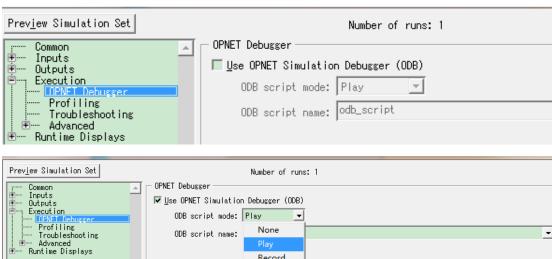
Run Cancel Apply Help

针对这个仿真界面,我们需要改动一些设计:

Simple... | Edit Simulation Sequence...

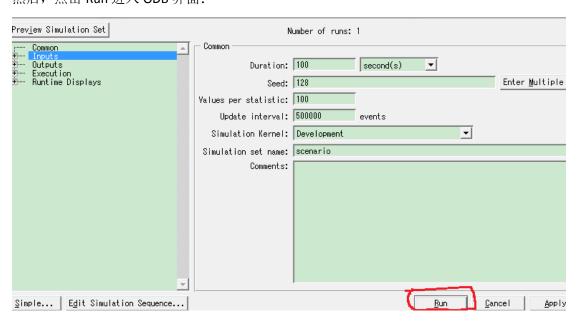
ation:	10	second(s) ▼		
Seed:		Jeconia(e)		Ente
istic:	100			
erval:	500000	events		
ernel:	Optimized		-	
name:	Based on 'kernel_type' preference			
ments:	Development			
	Optimized			
Descrip	Cilatiaa Cat	1		

ODB script name:

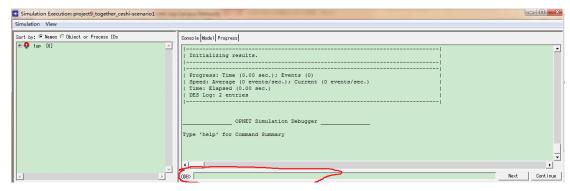


Record

然后,点击 Run 进入 ODB 界面:



ODB 操作界面:



红框标出的就是指令的输入界面。

在这里输入 help 你可以看到所有的指令分类,如果想看更详细的就按照介绍的这样输入 help <你想看的类或者指令>,如 help<fulltrace>

ODB> help				
OPNET Deb	ugger Help			
There are three levels of help:				
help	displays summary of help categories.			
help <category></category>	display help on specified category.			
help <command/>	display help on specified command.			
Command Categories [builtin]				
all				
basic				
action				
event				
memory				

基于假设我们是在第 13 个事件报错(实际上这个工程没有 bug,在这里主要是介绍一下流程),我们要输入一下指令,如图:

这个指令是让它在第13个事件停止,继续输入指令:

```
Type 'help' for Command Summary

ODB> evstop 13
Added breakpoint #0: stop at event (13)
```

这个指令是让 OPNET 执行下去,然后继续输入指令:

```
ODB> evstop 13
Added breakpoint #0: stop at event (13)

ODB> continue

(ODB 14.5.A: Event)

* Time : 0.0 sec, [0s]

* Event : execution ID (13), schedule ID (#13), type (begin sim intrpt)

* Source : Simulation Kernel

* Data : none

> Module : top.Campus Network.node_13.source [Objid=124] (processor)

breakpoint #0 trapped: "stop at event (13)"
```

这个指令是让 OPENT 在第 13 个事件的仿真中,显示出所有的仿真过程:

```
Console | Model | Progress |

ODB> continue

(ODB 14.5.A: Event)

* Time : 0.0 sec, [0s]

* Event : execution ID (13), schedule ID (#13), type (begin sim intrpt)

* Source : Simulation Kernel

* Data : none

> Module : top.Campus Network.node_13.source [Objid=124] (processor)

breakpoint #0 trapped: "stop at event (13)"

ODB> fulltrace
Full trace is now enabled
```

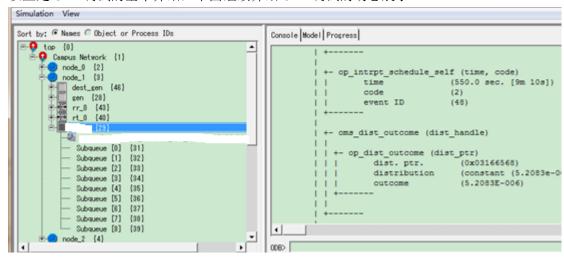
然后输入 next/continue 让仿真继续(因为我这里是没有错的,所以我只能输入 next)。 输入了 next 之后,可以看到从这个事件开始,每个进程、函数、中断直行的过程:

```
| | object ID (124)
| +- op_ima_obj_attr_get (objid, attr_name, value ptr)
    objid (124)
1.1
       attr name
                     (Packet Interarrival Time)
                      (constant (0.0000052083))
       value
       completion code (success)
| +- op_ima_obj_attr_get (objid, attr_name, value_ptr)
| | objid (124)
       attr name
                     (Packet Size)
1.1
       value
                     (constant (1024))
       completion code (success)
```

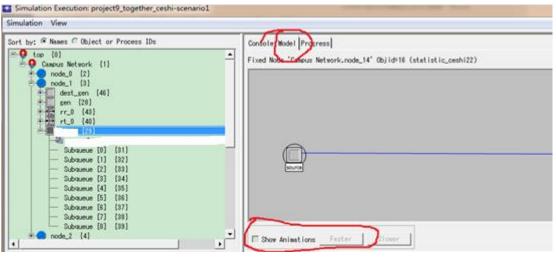
内容很多,不详细截图了。

二、动态展示:

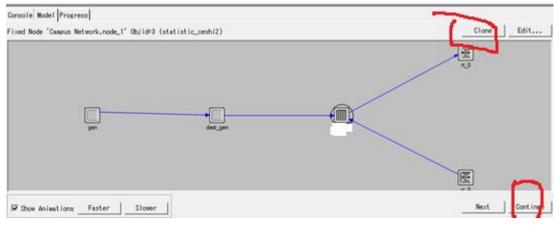
以上是 ODB 调试的基本介绍,下面继续介绍 ODB 调试的动态展示。



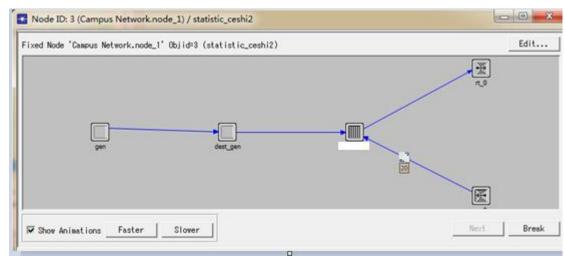
在左边窗口选择要观看的节点、或者进程



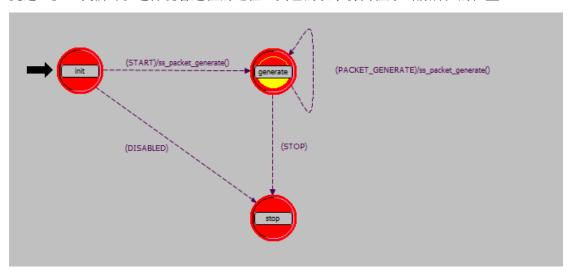
选择 model,下面的演示要勾上!如图:

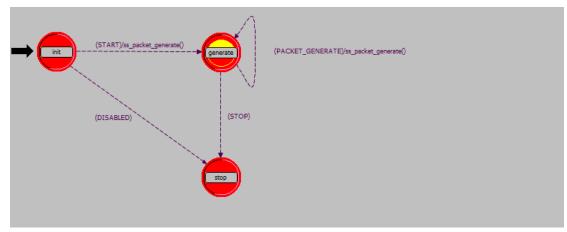


单击 CLONE,将窗口复制出来,便于最大化观看!点击 continue 之后,我们可以看到,包的流动,如图:



更进一步,我们可以选择观看进程的过程,黄色的表示仿真程序当前所在的位置:

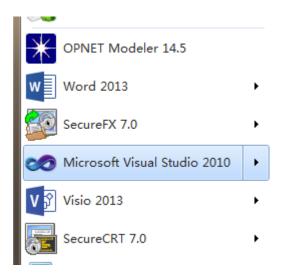




好了,动态展示到这里所有的步奏都讲完了,下面讲一下怎么 VC 联调。

三、VC 联调

和前面描述的一样,点击仿真,进入 ODB 界面,然后打开 VS2010(或者其它版本的 VS):



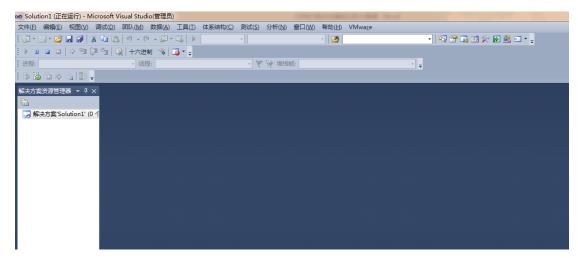
打开之后,点击调试:



选择附加到进程,点击之后再选择如图:



点击附加,然后进入如下界面:

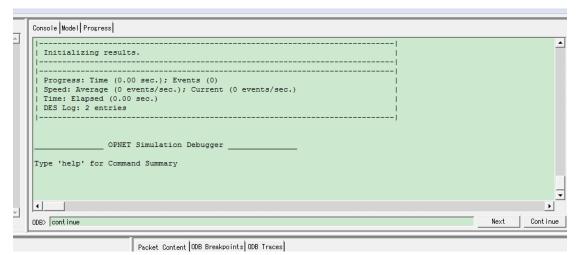


点击文件,打开,选择要调试的 OPNET 进程,确定,进程显示如下:

在进程中找到刚才 ODB 调试出 bug 的函数,或者选择觉得有问题的地方,像调试 C 一样,F9 设置断点:

```
(未知范围)
    979
    980
                    num_pulse_flag++;
    981
    982
                else if(((num_pulse_dettime1-num_pulse_dettime2)/(num_pulse_dettime2-num_pulse_dettime3))<=1 && num_pulse_flag>0)
    983
                    num_pulse_flag--;
    985
    986
    987
                if(num_pulse_flag==0)
    988
    989
990
                    ...
num_pulse = (num_pulse_dettime1+num_pulse_dettime2+num_pulse_dettime3+num_pulse_dettime4+num_pulse_dettime5+:
```

然后回到 ODB 界面,输入 continue 指令:



然后进程就会在设置断点处停住:

```
(未知范围)
   976
                            if(num_pulse_dettime2=num_pulse_dettime3!=0)
   977
                            if(((num_pulse_dettime1-num_pulse_dettime2)/(num_pulse_dettime2)/
   978
   979
   980
                                num_pulse_flag++;
   981
   982
                            else if(((num_pulse_dettime1-num_pulse_dettime2)/(nu
   983
   984
                                num_pulse_flag--;
   985
   986
   987
   988
                            if (num_pulse_flag==0)
   989
                                num_pulse = (num_pulse_dettime1+num_pulse_dettim
   990
   991
   992
                            else if(num pulse flag==1)
```

然后,你就可以像调试 C一样调试 OPNET 的进程了!

有一点需要注意的地方,就是 OPNET 上的部分参数在 VS2010 上无法查看,这时候就得在头文件中单独设置一个同类的参数,然后将那个参数的值赋给头文件中新声明的参数,然后就可以观察了!