Test Bench for CCounterOprLevel

Core description

The core ***CCounterOprLevel_T*** is almost identical to the core ***CCounterLevel_T***. There is a template parameter to define the type of the counter, for example if TYPE = ubyte, an 8 bits unsigned counter is created. The counter is clocked at each step, i.e. at FPGA clock. Instead of individual signals, the input ***iOpr*** controls the operation. Its type is:

```
enum Opr_t { cOprNone, cOprReset, cOprUp, cOprDown, cOprLoad };
```

Therefore, if ***iOpr*** equals ***cOprUp*** then the counter counts up.

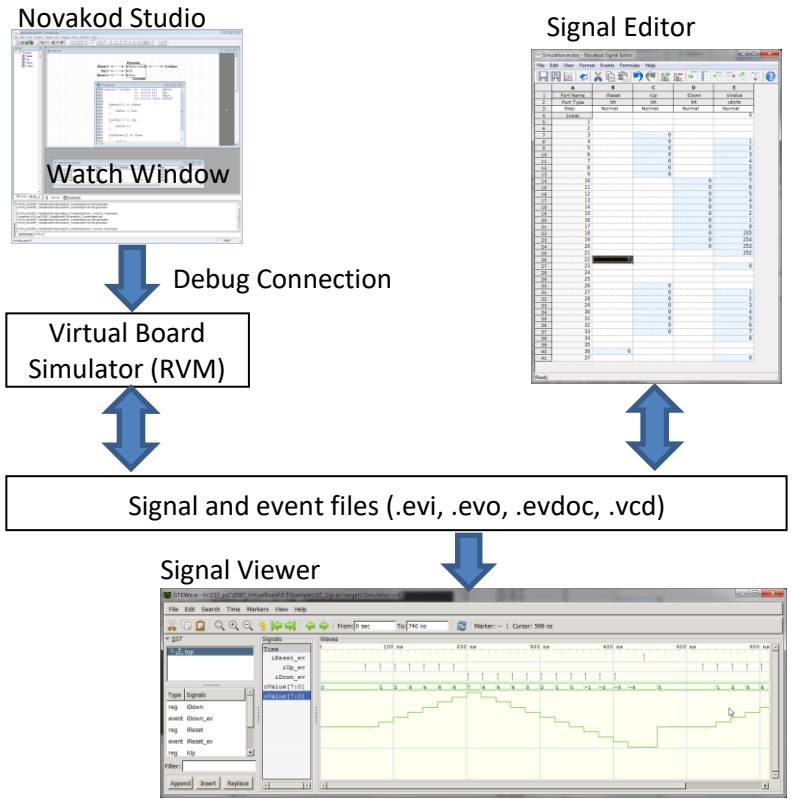
Test bench

This test bench uses the ***Signal Editor*** and the ***Signal Viewer*** to test the core. The test will be done in single step simulation. Follow the instructions to:

- Create signals in the ***Signal Editor.***
- Execute with the ***Single Step*** function (F8).
- Use the ***Watch Window*** in single step mode.
- View the result in the ***Signal Viewer.***

Description

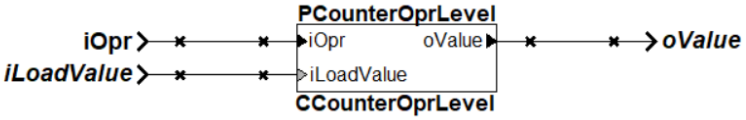
The ***Signal Editor*** is an Excel like spreadsheet to edit the input signals and view the output signals, including the events. The signal editor saves its information in a ***.evdoc*** file.



During simulation, the signals are read from an input event file (***.evi***) and written to an output event file (***.evo***). After simulation, the ***Signal Editor*** is used to view the resulting output signals. Then, the ***Signal Viewer*** will display the signal in the more common temporal view, with digital or analog representation. The signal viewer is ***GTK Wave*** and it uses ***.vcd*** (value change dump) files.

The psC test program

The test program consists of a single component, the core to be tested, and IO connections. The counter operation is controlled by the value of the input ***iOpr***.

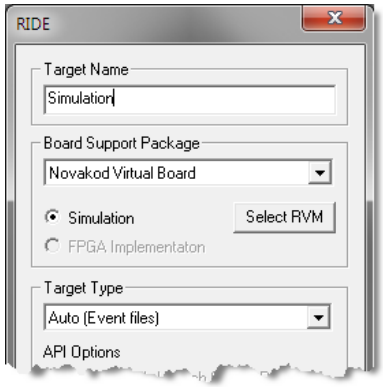


1) Double-click on ***Counter.rpj*** to start Novakod Studio.
2) Double-click on the ***main*** component to view the schematic.
3) Double-click on the ***CCounterOprLevel_T*** component to view the counter code.

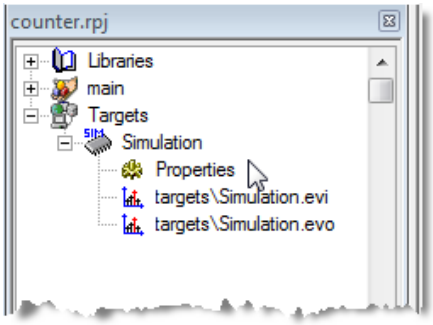
Compiling the test program

You will now compile the program, start execution, and use the editor signals to test it.

4) Under ***Targets*** open the ***Simulation*** properties and verify that the target is the Novakod Virtual Board.



5) Expand ***Targets*** then ***Simulation***, you should see the two event files:

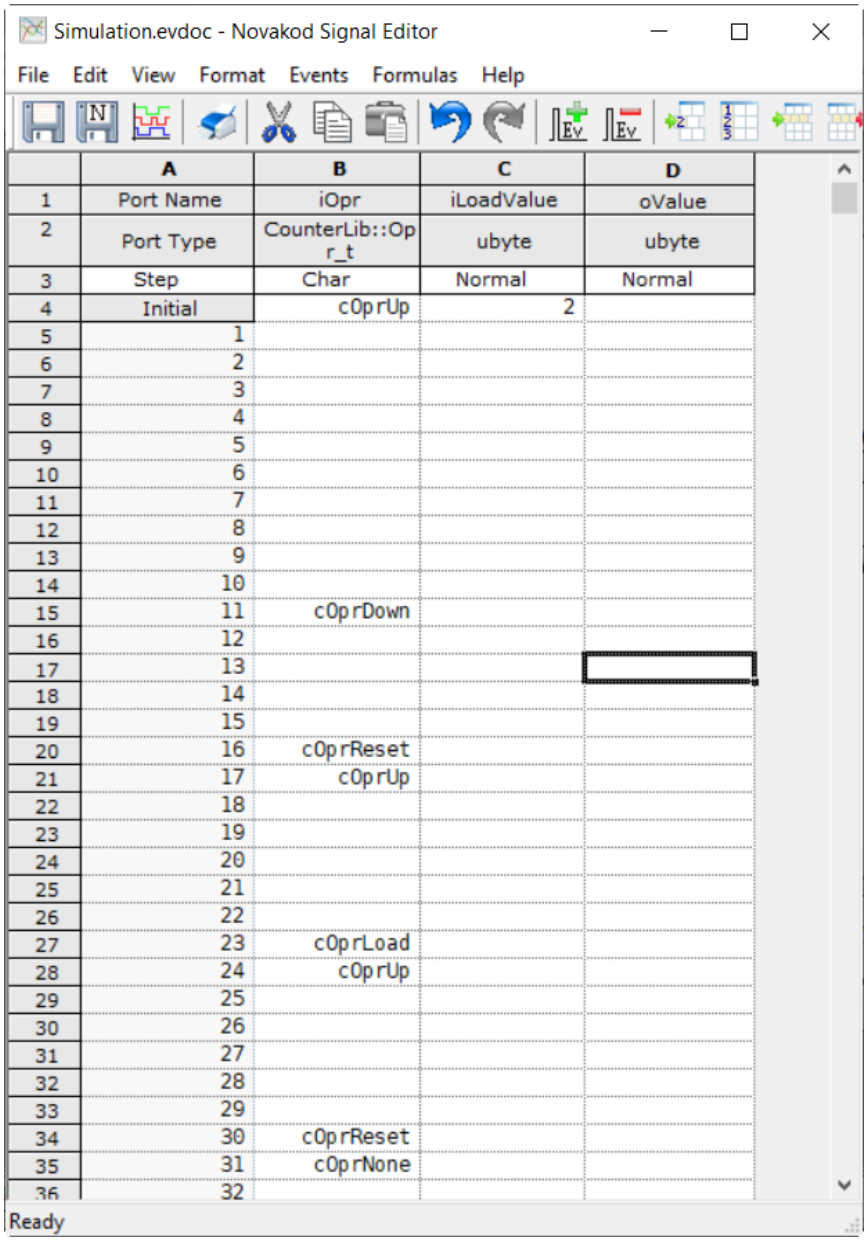


Viewing and editing input signals

The signals are already set, but you can change them as you wish.

6) Double-click on ***targets\Simulation.evi*** to start the ***Signal Editor***.

As you can see, the signal ***iOpr*** takes different values to control the counter. The values are displayed as their ***enum*** names. You can easily interpret the signal to predict the counter value. If you change the signals, don't forget to save.

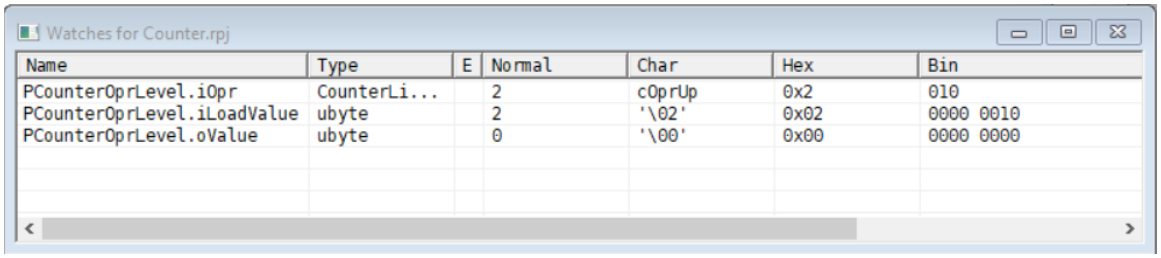


	A	B	C	D
1	Port Name	iOpr	iLoadValue	oValue
2	Port Type	CounterLib::Opr_t	ubyte	ubyte
3	Step	Char	Normal	Normal
4	Initial	cOprUp	2	
5	1			
6	2			
7	3			
8	4			
9	5			
10	6			
11	7			
12	8			
13	9			
14	10			
15	11	cOprDown		
16	12			
17	13			
18	14			
19	15			
20	16	cOprReset		
21	17	cOprUp		
22	18			
23	19			
24	20			
25	21			
26	22			
27	23	cOprLoad		
28	24	cOprUp		
29	25			
30	26			
31	27			
32	28			
33	29			
34	30	cOprReset		
35	31	cOprNone		
36	32			

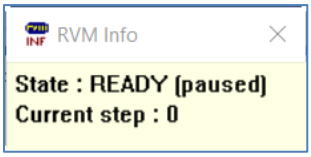
Running the test program

You are now ready to run and use the ***Single Step*** execution mode.

7) Select the menu ***Run → Start Paused*** to begin simulation, accept the build confirmation.
8) Right click on each port in the project ***Inspect*** tab to add them to the ***Watch window***.
9) You should see the initial values.



10) The RVM is in pause, and you can execute step by step.

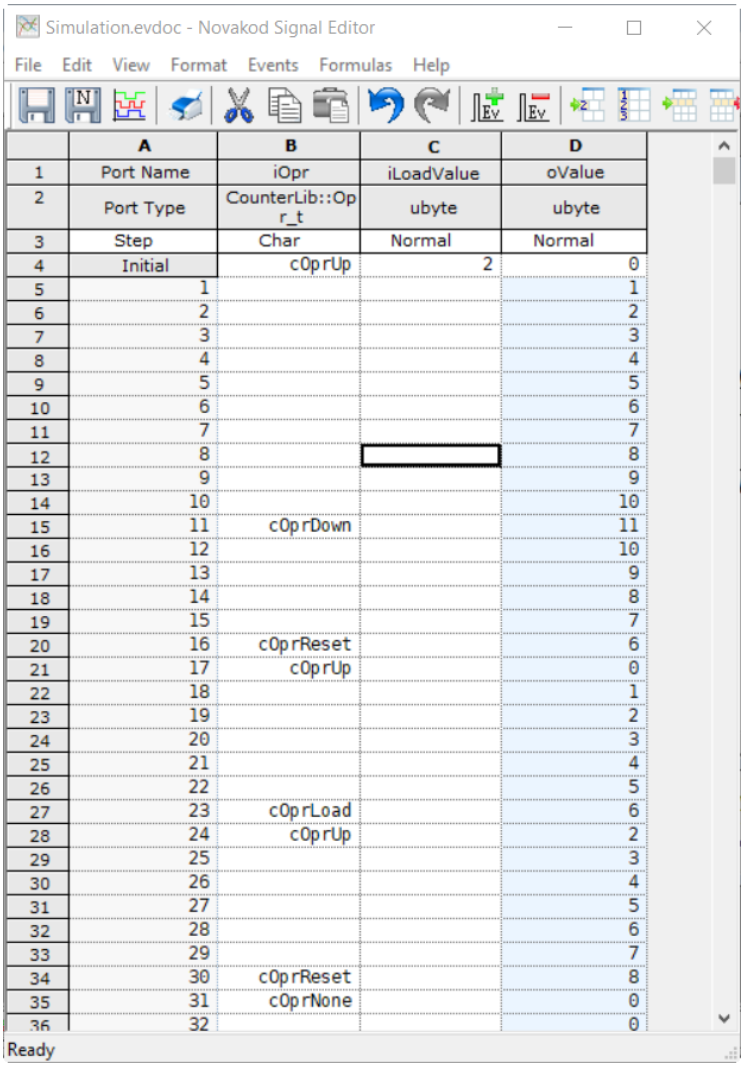


11) ***For single step, simply type F8.*** Repeat until you reach step 30. You will see all the input signals being applied at each step.
12) After 30 or more steps, stop the simulation by selecting menu ***Run → Stop***.

View results in signal editor

You can now view the results.

13) Double-click on ***targets\Simulation.evo***, the ***Signal Editor*** window starts with the values in ***oValue*** column.
14) Verify the counter operation.



	A	B	C	D
1	Port Name	iOpr	iLoadValue	oValue
2	Port Type	CounterLib::Opr_t	ubyte	ubyte
3	Step	Char	Normal	Normal
4	Initial	cOprUp	2	0
5	1			1
6	2			2
7	3			3
8	4			4
9	5			5
10	6			6
11	7			7
12	8			8
13	9			9
14	10			10
15	11	cOprDown		11
16	12			10
17	13			9
18	14			8
19	15			7
20	16	cOprReset		6
21	17	cOprUp		0
22	18			1
23	19			2
24	20			3
25	21			4
26	22			5
27	23	cOprLoad		6
28	24	cOprUp		2
29	25			3
30	26			4
31	27			5
32	28			6
33	29			7
34	30	cOprReset		8
35	31	cOprNone		0
36	32			0

View results in signal viewer

Novakod Studio integrates the well-known signal viewer called ***GTK Wave***.

15) Select menu ***File→ View Signals*** or click the shortcut  . The GTK Wave windows appears. GTK Wave has been pre-configured to show the desired signals. You can see the inputs and the output in hexadecimal or as an analog signal.