Counter Library

Novakod Studio is the integrated development environment (IDE) for the ***psC language***.

- Cores in this library use the psC language.
- Compiling the core with the IDE generates Quality VHDL code.
- The generated VHDL code can be used in any FPGA project.
- You can use the cores as is or create your own customized core.
- You can also use Novakod Studio for any FPGA applications, then generate VHDL.

Materials

You need ***Novakod Studio*** and the ***DE1SoC BSP*** (Board Support Package)

- Download at: https://icitechno.com/download
- Licenses at: https://icitechno.com/licenses

If you want to experiment with a real board, you need the DE1SoC board. This board is fully integrated into Novakod Studio. ***psC programs run without modification on the real board***.

- Buy at: Terasic

Very important

Folder paths and file names you create must not contain spaces or special characters.

To begin with…

1. First double-click ***CopyLib.bat***
2. Open the library:
   `C:\Novakod_Studio\OpenCoresLib\simple_customized_counter\CounterLib.psC`
3. Read the first part of the counter library.
4. Have a look at the code of the four counter templates.
5. Select the test bench for the desired core.

Core	Test bench
`CCounterEvent_T`	`TestCounterEvent`
`CCounterLevel_T`	`TestCounterLevel`
`CCounterOprLevel_T`	`TestCounterOprLevel`
`CCounterOprEvent_T`	`TestCounterOprEventBoard` `TestCounterOprEventAPI`

6. Follow ***ReadMe.pdf*** in the selected folder.

Have fun!

Basic coding rules

Print this page...

The psC language is based on C++ syntax and everything you learned about designing, coding and documenting C++ programs can be used with psC. You should look at the examples to get a good feeling for the coding style.

— Naming convention

As in C++, carefully choose names for variables, ports, functions, components, and so on, to reflect their usage. This must be done as early as possible as it greatly improves readability. In psC, the recommended naming convention is capitalized first word letter, like ExeOpr.

— Indentation

Indentation of 4 spaces, no tabulation, is recommended for compatibility between editors.

— Suffixes and prefixes

Here is a list of prefixes and suffixes specific to psC. You should use them systematically.

Prefix	Suffix	Apply to	Example

C		Component	`// Component Ports`
			`component CTest (in int iP,`
			`out int oP)`
			`{ };`
			
			`CTest PTest; // Process`

P		Process	
i		Input port	
o		Output port	
	_t	Type	`typedef int:3 int3_t;`

c		Constant	`const int cLines[] = 1 to 2;`
			`const int cCols [] = 1 to 3;`
			`const identifier cId[] = { A, B, C };`
			`enum Color_t { cRed, cGreen, cBlue };`

t		Temp variable	`temp tAdd = V1 + V2;` `temp fix8 tAdd(fix8 pF0, fix8 pF1) = pF0 + pF1;`
p		Parameters	`function fct(int pVal, ubyte pTyp) { };`

	_T	Template	`template< int NVAL, identifier NAME >`
			`function Add_T( ) { };`
			
			`function Add_T<8, oPort> Add_I;`

	_I	Template instance	
All Caps		Template parameters	

One to three capital letters		For…end parameters	`for I in <cRange>`
			`CInc PInc##I;`
			`end`

s_ g_		Reserved, do not use	