

Libero V11.2SP2 Implementing TMR (globally) in A3P and RT3P devices

Create your Libero Project as normal and when ready to implement the TMR'd version of the design follow these steps.

Open Synthesis Interactively to bring up the Synplify Pro GUI. Run the design first to get a base line utilization of the logic within your design.

The screenshot displays the Synplify Pro GUI interface. The main window shows the 'Run' tab with a 'Project Settings' table and a 'Run Status' table. The 'Area Summary' table is highlighted with a red border, showing Core Cells (903) and IO Cells (104). The 'Timing Summary' table is also visible, showing a clock name 'CNTADDCLK' with a required frequency of 100.0 MHz and an estimated frequency of 82.0 MHz. The bottom status bar shows the return code and run time.

Project Settings			
Project Name	CNTADD_syn	Device Name	synthesis: Microsemi ProASIC3L : RT3PE3000L
Implementation Name	synthesis	Top Module	[auto]
Retiming	0	Resource Sharing	1
Fanout Guide	24	Disable I/O Insertion	0
Disable Sequential Optimizations	0	FSM Compiler	1

Run Status								
Job Name	Status	U	A	W	CPU Time	Real Time	Memory	Date/Time
Compile Input (compiler)	Complete	10	3	0	-	00m:01s	-	10/29/2019 9:06:27 AM
Pre-mapping (premap)	Complete	3	1	0	0m:00s	0m:00s	109MB	10/29/2019 9:06:28 AM
Map & Optimize (fpga_mapper)	Complete	13	1	0	0m:05s	0m:05s	137MB	10/29/2019 9:06:34 AM

Area Summary			
Core Cells	903	IO Cells	104
Block RAMs (v_ram)	0		

Timing Summary			
Clock Name (clock_name)	Req Freq (req_freq)	Est Freq (est_freq)	Slack (slack)
CNTADDCLK	100.0 MHz	82.0 MHz	-2.196

```
Return Code: 1
Run Time:00h:00m:06s
Complete: Map on CNTADD_syn|synthesis
Complete: Logic Synthesis on CNTADD_syn|synthesis
0
```

You can further analyze your logic utilization and register usage by examining the Log file looking at the Core Cell Usage section as shown below:

Report: CNTADD_syn (synthesis)

Resource	Count	Used	Limit
AK1E	0	1.0	0.0
BUFF	1	1.0	1.0
CLKINT	2	0.0	0.0
MA73	36	1.0	36.0
MX2	72	1.0	72.0
MX2A	19	1.0	19.0
MX2B	4	1.0	4.0
MX2C	9	1.0	9.0
MOR2A	19	1.0	19.0
MOR2B	160	1.0	160.0
MOR3A	1	1.0	1.0
MOR3B	13	1.0	13.0
MOR3C	45	1.0	45.0
OAL	1	1.0	1.0
OALC	2	1.0	2.0
OAT1	4	1.0	4.0
OR2	7	1.0	7.0
OR3	2	1.0	2.0
VCC	1	0.0	0.0
XAL	30	1.0	30.0
XOR2	10	1.0	10.0
XOR3	4	1.0	4.0
XOR2	58	1.0	58.0
XOR3	26	1.0	26.0
DFN1	96	1.0	96.0
DFN1C0	33	1.0	33.0
DFN1C100	95	1.0	95.0
TOTAL	907		903.0

IO Cell usage:

Cell	Count
CLKBUF	2
INBUF	70
OUTBUF	32
TOTAL	104

Core Cells: 903 of 75364 (1%)

Now in the Synplify Pro GUI click on the SCOPE (Synplify Constraints Editor window) to bring up the constraints editor shown below. Select the Attributes Tab at the bottom

Current Scope: Top Level

Enable	Object	Type	Object	Attribute	Value	Value Type	Description	Comment
1								
2								
3								
4								
5								
6								
7								
8								
9								
10								
11								
12								
13								
14								
15								
16								
17								
18								
19								

Attributes

Return Code: 1
Run Time: 00h:00m:06s
Complete: Map on CNTADD_syn|synthesis
Complete: Logic Synthesis on CNTADD_syn|synthesis
0

Next in the Object Column you can either type <global> or go to the far right hand side of the cell (just left of the cell dividing line and double left mouse click to bring up the various "objects" in the design. In

this case, I'm wanting to TMR everything but you could individually select critical elements such as statemachines etc.

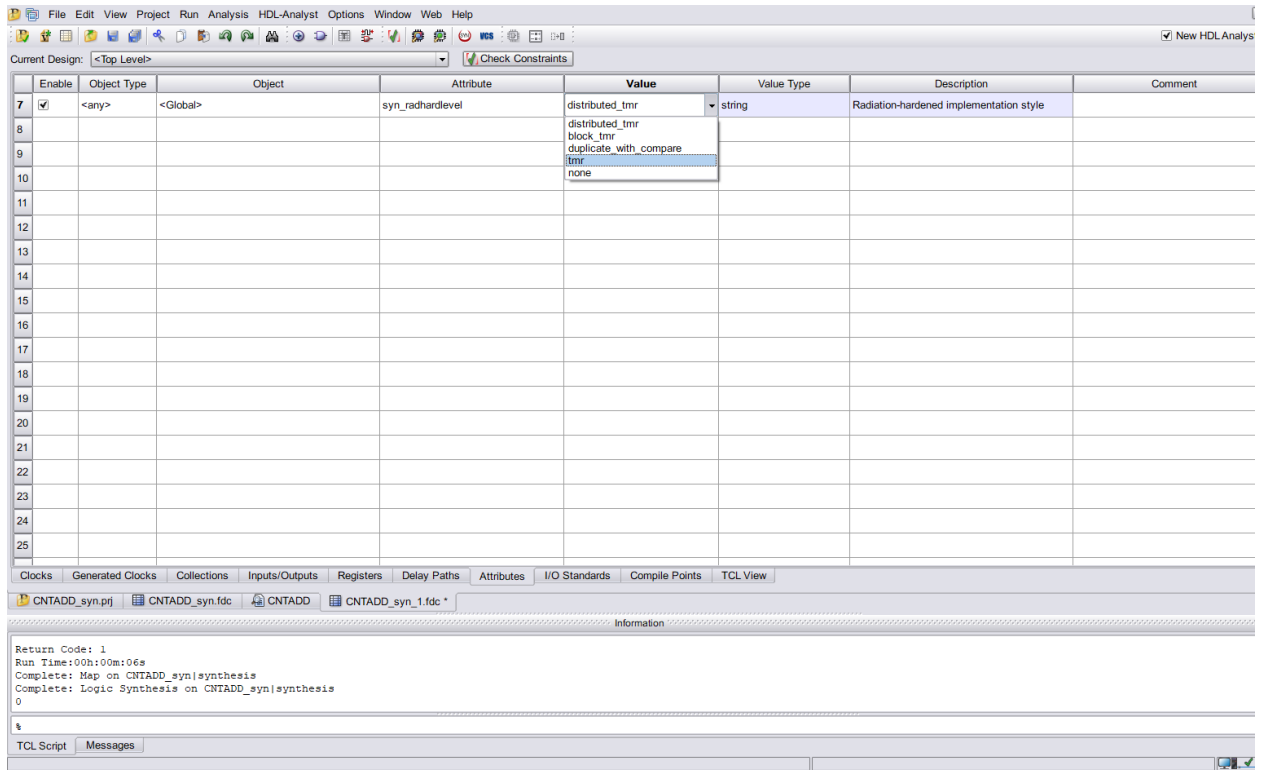
Now here is the magic 😊

In the Attribute Cell in the row you selected global type "syn_radhardlevel" and hit return. The line should populate the Value, Value Type and Description cells as shown below.

	Enable	Object Type	Object	Attribute	Value	Value Type	Description	Comment
7	<input checked="" type="checkbox"/>	<any>	<Global>					
8				syn_netlist_hierarchy				
9				syn_noarrayports				
10				syn_noclockbuf				
11				syn_probe				
12				syn_radhardlevel				
13				syn_ramstyle				
14				syn_reference_clock				
15				syn_replicate				
16				syn_safefsm_pipe				
17				syn_shift_resetphase				
18								
19								
20								
21								
22								
23								
24								
25								

```
Return Code: 1
Run Time:00h:00m:06s
Complete: Map on CNTADD_syn|synthesis
Complete: Logic Synthesis on CNTADD_syn|synthesis
0
$
TCL Script Messages
```

Note: For some families the syn_radhardlevel does not appear in the drop down. You can simply type "syn_radhardlevel" hit the return and the line should populate and give you options for TMR in the value window.



When finished with the TMR selections select save and when prompted to add to the project select yes.

Return to the main project page and select run. Now you can see the registers show 3x usage and the LUTs (MAJ3 voter) show a dramatic increase as well.

Synplify Pro
Done: 0 errors, 5 warnings, 27 notes

Project Name: CNTADD_syn Device Name: synthesis: Microsemi ProASIC3L: RT3PE3000L
 Implementation Name: synthesis Top Module: [auto]
 Retiming: 0 Resource Sharing: 1
 Fanout Guide: 24 Disable I/O Insertion: 0
 Disable Sequential Optimizations: 0 FSM Compiler: 1

Run Status

Job Name	Status	CPU Time	Real Time	Memory	Date/Time
Compile Input (compiler)	Complete	8 3 0	00m:00s	-	10/29/2019 9:54:02 AM
Pre-mapping (premap)	Complete	3 1 0	0m:00s	109MB	10/29/2019 9:54:03 AM
Map & Optimize (pga_mapper)	Complete	13 1 0	0m:06s	137MB	10/29/2019 9:54:10 AM

Area Summary

Core Cells	IO Cells
1666	104
Block RAMs (v_ram)	0

Timing Summary

Clock Name (clock_name)	Req Freq (req_freq)	Est Freq (est_freq)	Slack (slack)
CNTADDCLK	100.0 MHz	75.4 MHz	-3.257

Return Code: 1
 Run Time: 00h:00m:07s
 Complete: Map on CNTADD_syn|synthesis
 Complete: Logic Synthesis on CNTADD_syn|synthesis
 0

Looking again at the synthesis report for Core Cell Usage

Report: CNTADD_syn (synthesis)

cell	count	area	count*area
AO1	53	1.0	53.0
AO1B	4	1.0	4.0
AO1A	2	1.0	2.0
AX1	45	1.0	45.0
AX1C	14	1.0	14.0
AX1D	11	1.0	11.0
AX1E	8	1.0	8.0
BUFF	1	1.0	1.0
CLK3INT	2	0.0	0.0
HA73	289	1.0	289.0
MD2	140	1.0	140.0
MD2B	50	1.0	50.0
MD2C	9	1.0	9.0
NOR2A	18	1.0	18.0
NOR3B	126	1.0	126.0
NOR3A	1	1.0	1.0
NOR3B	26	1.0	26.0
NOR3C	60	1.0	60.0
OA1C	2	1.0	2.0
OA1I	4	1.0	4.0
OS2	6	1.0	6.0
OS3	2	1.0	2.0
VCC	1	0.0	0.0
XA1	36	1.0	36.0
XNOR2	16	1.0	16.0
XNOR3	6	1.0	6.0
XOR2	67	1.0	67.0
XOR3	27	1.0	27.0
DFN1	289	1.0	289.0
DFN1CO	384	1.0	384.0
TOTAL	1670		1666.0

IO

cell	count
CLK3BUF	2
INBUF	70
OUTBUF	32

Return Code: 1
 Run Time: 00h:00m:07s
 Complete: Map on CNTADD_syn|synthesis
 Complete: Logic Synthesis on CNTADD_syn|synthesis
 0

Some additional information.

It appears that the Block_TMR and Distributed_TMR are Synplify Premier options and did not seem to do any mitigation within this particular design so possibly I did not set it up correctly for those options or they are not allowed in Synplify Pro.

Distributed TMR is used to triplicate blocks of logic with internal voters and (TMR) vote the output of those 3 logic blocks.

Block_TMR is used when you have a block of logic that cannot be internally modified (think maybe IP etc) so that block is then triplicated and the output of those 3 blocks is voted.