Starting with a Timing Budget Flow, Partition the design, Running Top Level IPO, Perform Synthesis, Loading the Partitioned design, and Running Refine Timing Budget.

First Encounter Workshop 4

What you will learn

- Preparing the design for timing budget
- Partitioning with timing budgets
- IPO'ing the partitioned top level
- Performing synthesis (actually not taught in class)
- Loading physical synthesis placement
- Fine tune timing budgets

1. TESTCASE INFORMATION

The testcase is the same one used in Workshop 1 & 2. The hierarchical instance SH17 contains 27,551 instances. SH25 contains 7,293, and SH7 contains 2,232. This leaves the top level with about 22,070 instances, subtracting the 3 partitions. The netlist format is hierarchical Verilog, and the process is 3 layers of metal. It has one clock source, MCK.

2. PREPARING THE DESIGN FOR BUDGETING

Start First Encounter (`encounter`) in the appropriate directory.

Open the *Design Import* form and load in the configuration file `train_p.conf`.

As in workshop 2, load the top-level floor plan file, `train_p.fp0`.

The next few steps are to prepare the floor plan for a partition design and then run the partition program to derive the timing budget files.

In the floor plan, highlight the guide *SH7*, and then type ‘q’ and change the *Constraint Type* to *Fence*. Do, the same for guides *SH17* and *SH25*. These are the partitions you will be creating timing budget files for.

Run placement with *Medium Effort* (*Place -> Place…*).

Next, perform Trial Route with *Medium Effort* (*Route ->Trial Route…*).

Proceed with RC extraction and timing analysis (*Timing -> Extract RC…* and *Timing -> Timing Analysis*….). Or you can enter the following FE commands in the Encounter shell (commands can be all lower case):

```
amoebaPlace
trialRoute
extractRC
buildTimingGraph
```

3. PARTITIONING WITH TIMING BUDGETS

Open the *Partition -> Specify Partition…* form and specify the three partitions with the following information:

Partition Name	Hinstance	Core to all directions
SH17	SH17	5 Microns
SH25	SH25	2 Microns
SH7	SH7	2 Microns

Be sure to click the *Add* button for each partition. For now, leave all the default selections as they are in the form, especially saving the partition information to a file.

Click OK when ready to load in the partition information.

Next, you will actually create the partitions, which will also generate the pin assignment and deriving of timing constraints.

Open the *Partition -> Partition…* form. Make sure you choose *Perform Pin Assignment* **and** *Derive Timing Budget* (this is to create a timing constraint file for synthesis applications). Also, choose the option *With Trial IPO Estimates* which emulates an in-place optimization performed at the top level. This is necessary for the modules being partitioned, otherwise any undersized and under driven gates would cause submodule budgets to be over constrained.

Optionally, choose *Create timing budget verifier* option. The *Create timing budget verifier* option creates a directory, `<design>_verifier` and allows you analyze the timing budget results in another FE session.

Click *OK* when ready.

Open the *Partition -> Save Partition* form and enter a directory name of `PTN` in the *Partition Result Directory* field, choose Output Format: *PDEF*.

Exit First Encounter.

Now `cd` to the `PTN` subdirectory and perform an `ls` to view the generated/saved directories. Note there are 4 subdirectories (5 subdirectories if you chose the *Create timing budget verifier* option). There is a subdirectory for each partition specified, as well as a subdirectory for the top level. Change directory into that you named for partition, *SH7* and do an `ls`. Note the files that were generated by FE. The files <`SH7>.constr.pt` and <`SH7>.pdef` will be used by physical synthesis later.

Change directory (`cd`) to `TOPCHIP_SP`. Note the files generated by FE. The difference in this view of the top level is that each partition is now seen as a block (hard macro) and FE has generated a STAMP model for each block. Note that it is not advisable to run IPO the top level in the flat view if the design is going to be done hierarchical. This is because there is no reason to make changes in submodules before synthesis. Once IPO is done for top level, these changes can then be fed back to the design community.

4. IPO THE PARTITIONED TOP

Change directory (`cd`) to `TOPCHIP_SP` and start another First Encounter. Do a *Design Import* and load in the `TOPCHIP_SP.conf` (Note the values in the *Stamp Model Definitions* and *Model Data* fields).

Next, load in the floor plan `TOPCHIP_SP.fp` file. Note that *SH7*, *SH17* and *SH25* are now blocks.

Run placement (to place the standard cells at the top), trial route, RC extraction and timing analysis. Look at the timing (*Timing Analysis -> Slack Browser…* using the file `TOPCHIP_SP.slk`).

Perform IPO by executing *Timing -> In-Place Optimization…*, and in the IPO form, make sure to use the default settings, do NOT choose *Synthesis after IPO* (*Post-IPO Optimization*). After IPO finish running, examine the `TOPCHIP_SP_ipo.slk` report to verify that timing did improve.

Exit First Encounter.

5. PERFORM SYNTHESIS

In this workshop you do **NOT** actually perform synthesis, however, the process is that the `*.constr` (PrimeTime constraints) and the `*.pdef` files created during partitioning for SH7, SH17 and SH25 would be used to perform physical synthesis.

So, in this workshop the results of physical synthesis are provided in the `BUDGET` directory. In the directory, there are the partition DEF files and the netlist file for entire design with the top level IPO'd.

6. LOAD-IN THE DEF'S

Change directory back to the original run directory (`cd ../..`).

Start First Encounter and load in the configuration file `train_p_topipo.pc.conf`. This contains the netlist that has the top level ipo changes (from step #4), as well as the submodule physical synthesis netlists.

Load in the original floor plan file, `train_p.fp0`.

Now, *Partition -> Specify Partition* and load in the saved specify partition file from step #3. Next, create the partitions (*Partition -> Partition*) and you can deselect the *Perform Pin Assignment* option.

Now to load the placement data in DEF format for each of the partitions, type `setTopCell hnl_40` in the Encounter console (hnl_40 is the master name of the module SH7). Now, do a *Place -> Load Place -> DEF* and choose the file `BUDGET/SH7.def`. Switch to the Placement View and notice the partition has placement data.

Type `setTopCell hnl_35`. Do a *Place -> Load Place -> DEF* and load in the file `BUDGET/SH17.def`.

For the last partition, type `setTopCell hnl_21`. Do a *Place -> Load Place -> DEF* and load in the file `SH25.def`.

These next few steps load the physical synthesis placement results back into the top view and flatten (unpartition) the entire design.

Type `setTopCell TOPCHIP_SP` to return to the top chip. Load a *Place -> Load Place -> Place* and select the file `PTN/TOPCHIP_SP/TOPCHIP_SP_ipo/TOPCHIP_SP.place` that is from step #4.

To flatten the design, use the *Partition -> Unpartition* form. Now the entire design is flat as it was in the beginning.

Next, run, Trial Route, RC extraction, and build the timing graph to observe full chip timing analysis flat with physical placement data on the flat design.

7. FINE TUNE BUDGETS

The timing refine loops are not performed in the workshop, but in practice you would now repeat all of the steps to fine tune the timing budgets. This means that you would perform partitioning again, i.e., specify partition, partition, save partition and repeat IPO'ing the top, run synthesis, and then retrieve new physical synthesis placement.