Unused flop deletion algorithm leading to improved time to market and SoC cost savings

Arvind Kumar (arvind-dlh.kumar@st.com)
STMicroelectronics, Home Entertainment and Display, India

Our existing flow to generate FE netlist

SoC netlist integration
(Integration of various IPs, SoC specific glues and interconnect IP)

↓

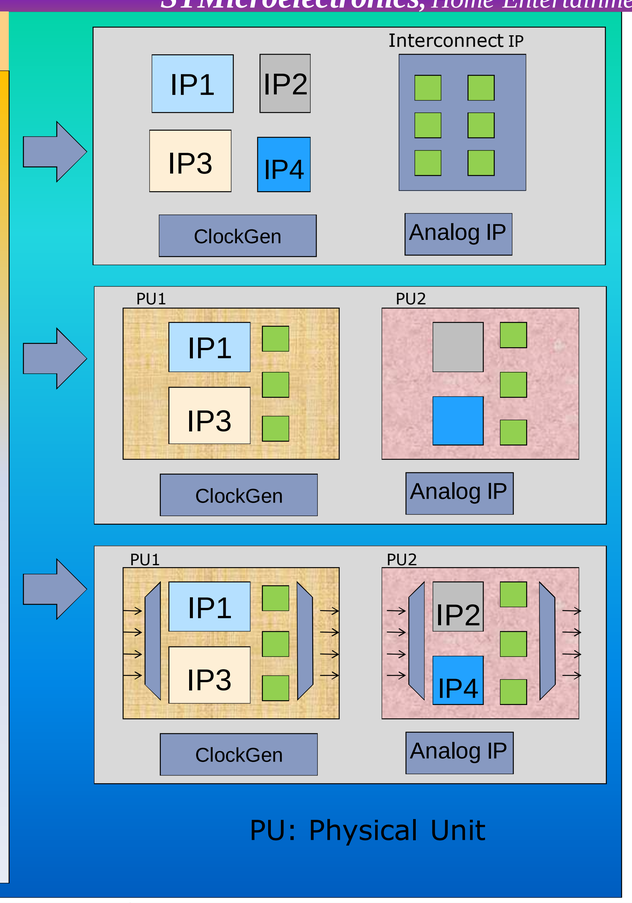
Hierarchy manipulation
(Creating physical hierarchy by grouping IPs and interconnect nodes)

↓

DFTMax insertion
(Stitch scan chains for SoC glues & interconnect nodes and insert DFT compression and decompression using IP test models)

↓

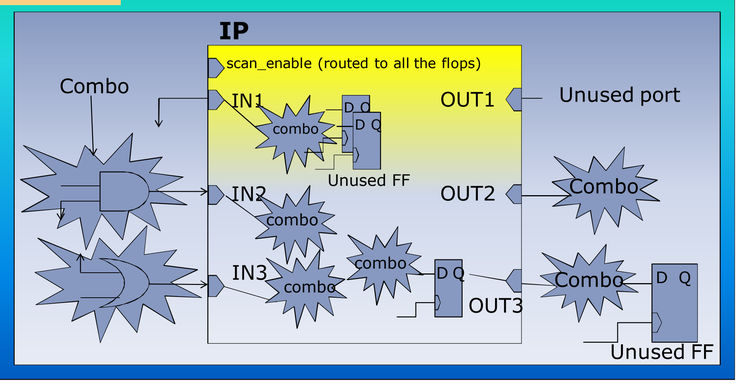
FE netlist
(Input for BE flow)



What are the issues with our existing flow?

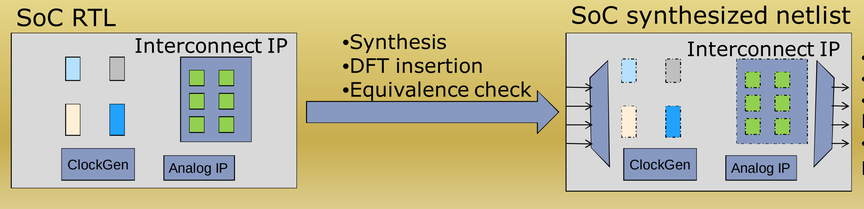
Design contains unused flops and associated gates, which leads to following issues:

- Increase chip area hence SoC cost
- Increases power consumption of the chip
- Causes congestion and hence increases flow run time down the physical implementation flow
- Need effort to close timing on these paths



Why we can not go for RTL synthesis at SoC level?

- Netlist will not be available right at the beginning of the flow as it has to go through synthesis and equivalence check flow
- For any bug fixed RTL of an IP, we need to re-spin the SoC FE flow
- IP swap will not be possible in FE netlist because of boundary optimization during full synthesis. Moreover, DFT flow at TOP will also create random ports at IP boundary
- Physical hierarchy creation will not be feasible and hence hierarchical implementation can not be used



- Netlist is available after 3 major task
- No way to create physical hierarchy
- Re-spin is mandatory to fix functional bug vs. IP swap in our flow
- Incremental synthesis run is not possible vs. per IP optimization

Proposed flow to generate FE netlist

SoC netlist integration
(Integration of various IPs, SoC specific glues and interconnect IP)

↓

Hierarchy manipulation
(Creating physical hierarchy by grouping IPs and interconnect nodes)

↓

Unused flop removal
(Area recovery by removing unused flops from scan chains and associated logic)

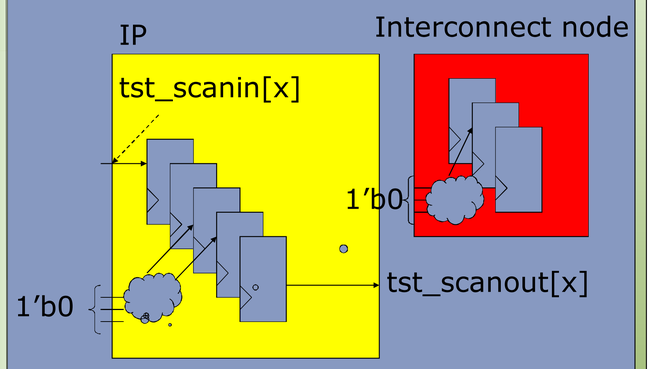
↓

DFTMax insertion
(Stitch scan chains for SoC glues & interconnect nodes and insert DFT compression and decompression using IP test models)

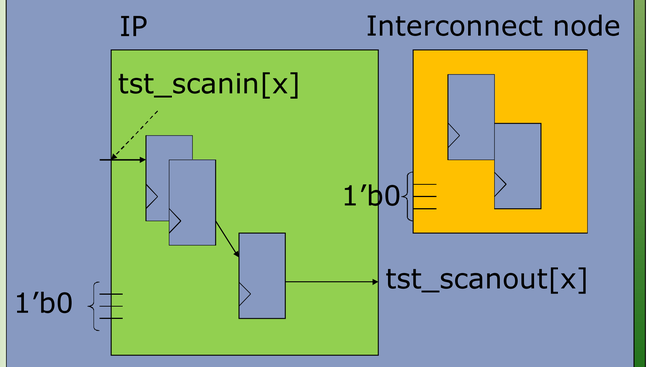
↓

FE netlist
(Input for BE flow)

Restructured netlist as input to the flow



Optimized netlist as output from the flow



How unused flops will be optimized away

Algorithm for stage-1: (at SoC level)

- After setting '*dont_touch*' attribute on IPs, simplify the constant in the synthesis tool (Design Compiler from Synopsys)
- Constant input ports and unconnected output ports is recorded in EDA tool language:
  - Constant 1 on input port: *set_logic_one [get_ports <port>]*
  - Constant 0 on input port: *set_logic_zero [get_ports <port>]*
  - Unconnected output port: *set_unconnected [get_ports <port>]*

Algorithm for stage-2: (@each IP)

- Constants at input ports will be propagated
- Unused (said unconnected in earlier case) port will be disconnected automatically from internal logic of the IP
- Constant '0' will be set for scan enable signals in the design to hide the scan chain connections during optimization
- Design will be optimized for area. Timing constraints for IP is optional
- Constant '0' from scan enable signals will be removed and scan chain will be re-build

Notes:

- Number of scan chains will be forced the same, as were in the original design. May be few chains will have less flops than the specified max length value
- During re-spin of the flow, only those IPs needs to be optimized for which either boundary constraints and/or IP functionality have changed

At SOC level (Stage-1):

- Dump the IP netlist from SoC netlist
- Derive boundary logic constraint for each IP from SoC netlist

For each IP (Stage-2):

- Break the scan chains
- Optimize the design for area
- Re-build the scan chains

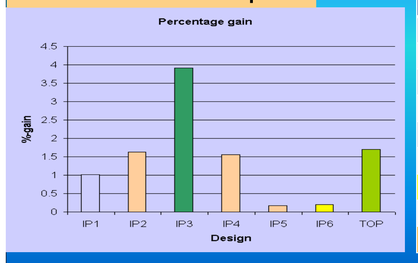
At SOC level (Stage-3):

- Check and swap the IPs
- Disable boundary optimization for IPs
- Run full netlist optimization

Algorithm for stage-3: (at SoC level)

- Swap the IPs, if all the following checks passes:
  - IPs are optimized successfully
  - Still have same number of scan chains and none of the scan chains are empty
- Boundary optimization of the IPs will be disabled. This ensures that:
  - Functionality to input port of the IP will not change
  - Functionality from output port of the IP will not change
  - IP swap is possible to implement functional ECO, in case we found a bug inside the IP
- If DFT compression is not inserted yet, then optimize the design at SoC level to remove unused logic from interconnect nodes and the glues
- In case of physical synthesis run, congestion and timing aware optimization can be performed and placement information can be passed to P&R tool to speed up the flow.

Real case example:



Design	Pre-Opt (flop count)	Pre-Opt (area-mm^2)	Gain (flop count)	Gain (%-age)	Run time (hours)
IP1	24,569	0.948	248	1.01	0.53
IP2	62,595	3.242	1,019	1.63	3:40
IP3	4,421	0.151	173	3.91	0:05
IP4	128	0.002	2	1.56	0:01
IP5	7,710	0.319	13	0.17	0:34
IP6	9,180	0.19	18	0.20	0:07
Total	108,603	4.852	1,465	1.70	5:20
TOP(excl. IPs)	156,967	9.172	1,042	0.66	6:45
TOP (overall)	158,432	9.28	2,507	1.58	12:05

Equivalence run time for this= ~3:00 hrs

Limitations (Future plan):

- A smart kit is required to automatically handle the netlist optimization flow, otherwise:
  - Manual setup for the flow will take time
  - During re-spin, boundary logic constraints has to be manually checked to identify IPs that needs re-spin
- RTL synthesis of IPs should happen with same clock frequencies as required for the SoC
- This flow needs same input specification as used at IP level synthesis:
  - DFT constraints like test clocks, set/reset, scan enable, constant signals etc.
  - Synthesis constraints like *dont_touch*, *size_only* etc.
  - Timing constraints [Optional]

Summary: (1) Run time is 1/3x w.r.t. SoC RTL synthesis flow (time to market)
(2) Absolute area gain is 3% w.r.t. to our older flow (cost saving)

STMicroelectronics