FPGA Prototyping For Large Multi-Core/Multi-Die Designs

Joseph Marceno, Executive Director, Engineering
Synopsys

Agenda

- Modular Hardware-Assisted Verification (HAV)
- Modular HAV Flow in HAPS ProtoCompiler
- Design Example
- Die-to-Die Design Example
- Summary

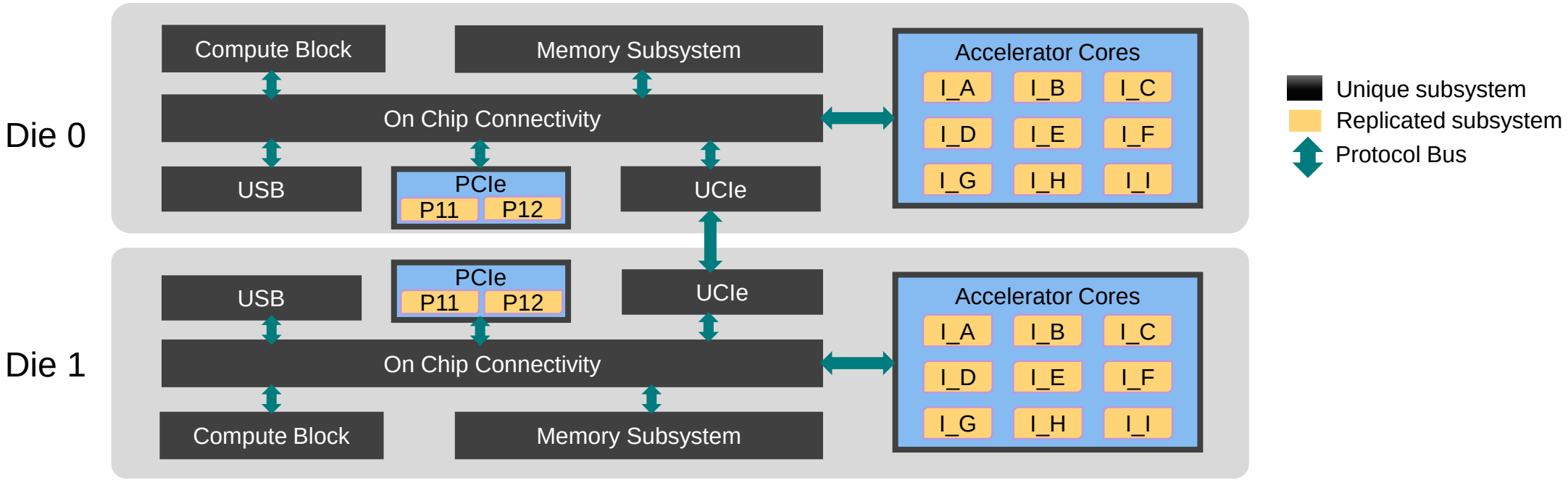
Introduction to Modular Hardware-Assisted Verification

Prototyping Trends and Challenges

Top Trends
Design size for prototypes growing significantly • 2019 : 100 to 200 Million gates • 2024 : More than 600 Million gates
Configurable and replicated computational units • Identify bugs and inefficiencies in SW
New Multi-die verification flows • Independent validation of each die • Validate inter-die connectivity

Top Challenges
Predictable prototype bring-up
Timing closure for high-performance prototype
Scale of compute resources to build large prototypes
Integrating sub-system prototypes for chip validation
Asynchronous clock domains for sub-systems

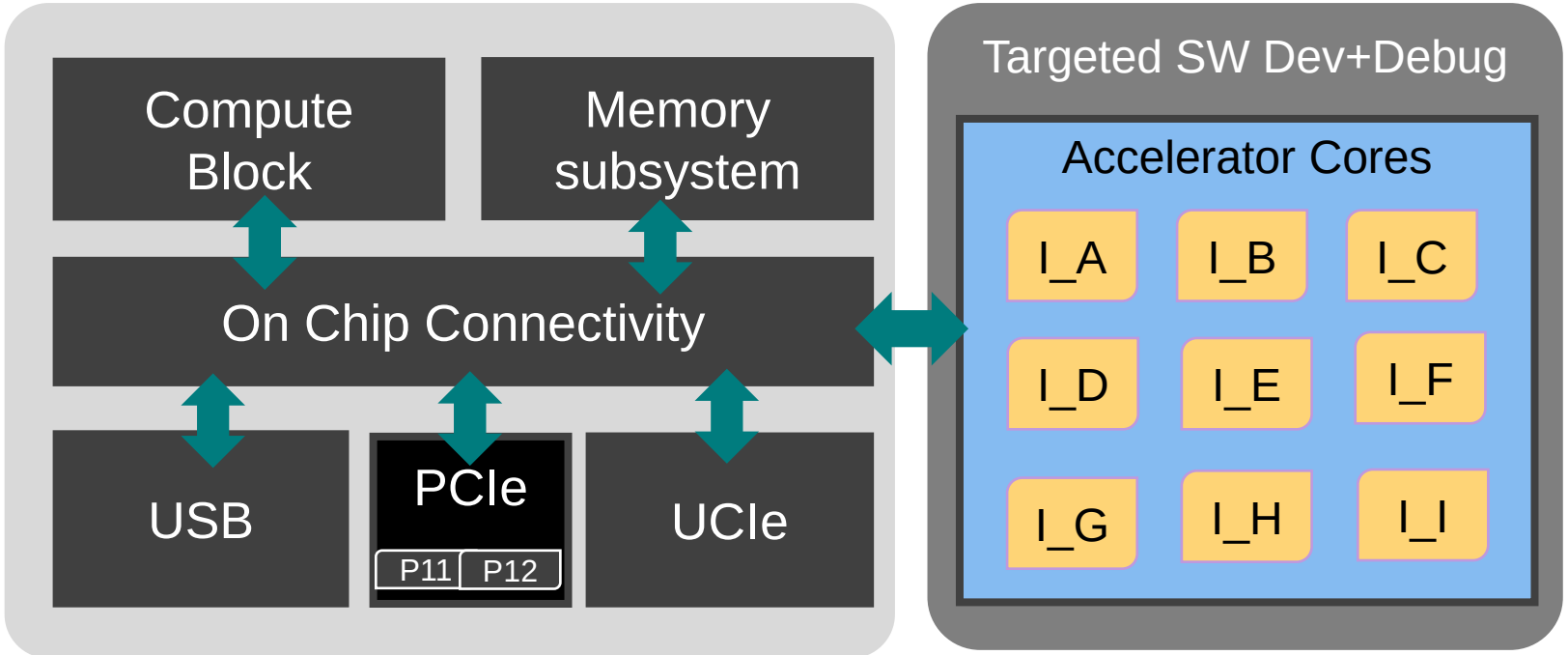
Large SoCs, Multi-Die Designs: Modular Architectures



- IP blocks connected via protocol busses
- Protocol busses across die-die interfaces
- Protocol busses enable independent clocking on either side of the bus

Multi Core Designs with Multi-Threaded SW

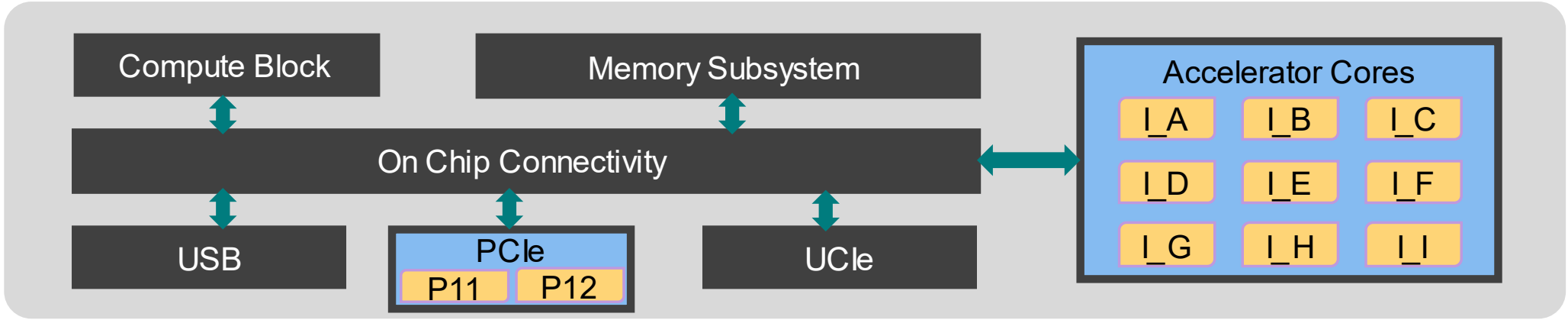
- Highly replicated execution units: CPU, GPU, Accelerators
- Complex multi-threaded software architecture
- Many complex SW bugs only show up on multi-core HW
- Need to build large multi-core prototypes for early debug of SW



Modular Design Verification Environments

- Independent teams developing subsystems in parallel
- Individual teams responsible for validating subsystems
- System teams responsible for integrating and validation of full-chip
- Verification techniques like Formal and Simulation are modular
- Natural extension of modularity into prototyping and emulation platforms

Modular Prototypes: Benefits



Scalability and Predictability

- Independent build and validation
- Integrate prototypes into full chip
- Savings on time and resources

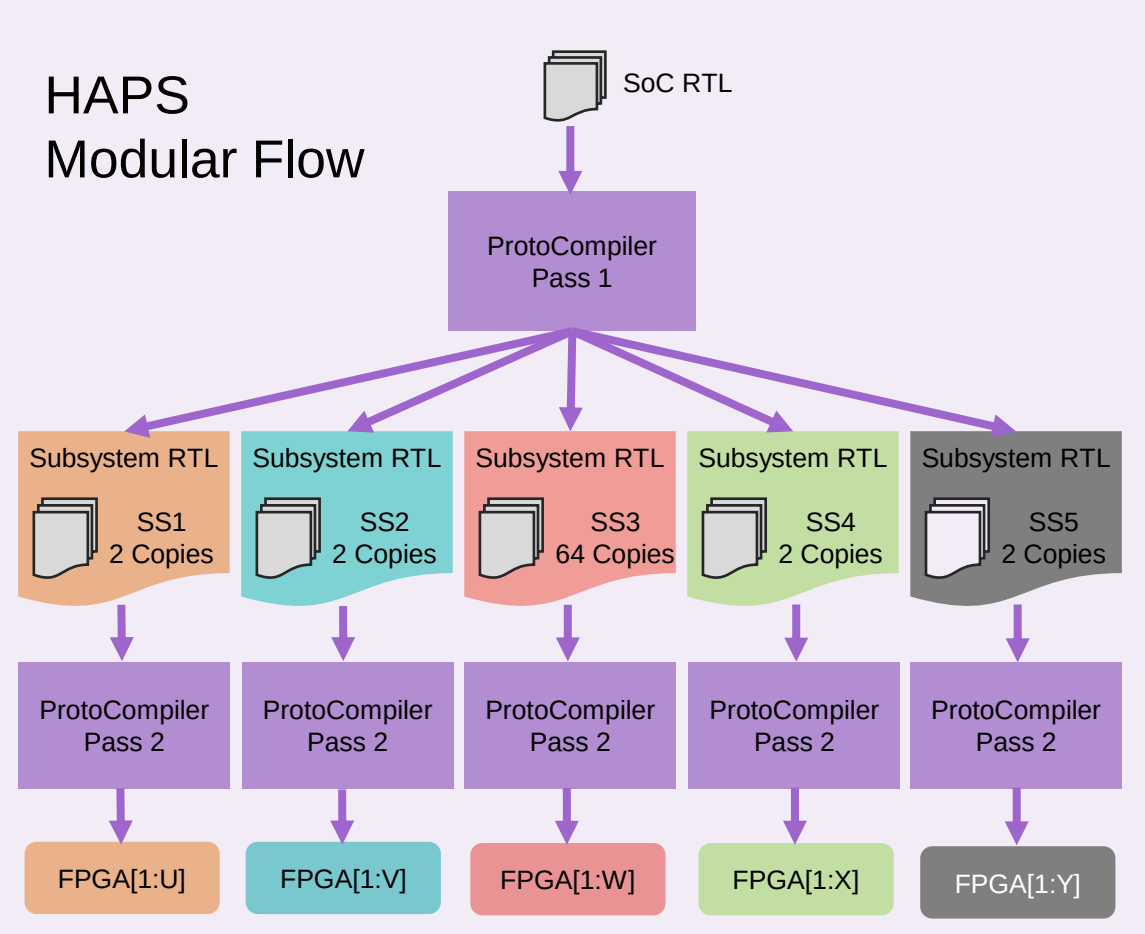
Incrementality

- Only rebuild modified subsystems
- Bit-file reuse for multi-core
- Build once, replicate N times

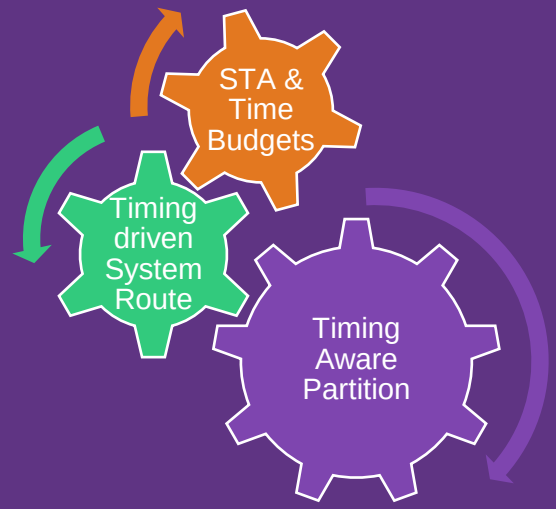
Modular HAV Flow in HAPS ProtoCompiler

Synopsys HAPS ProtoCompiler Modular Flow

Two-Pass, Timing-Driven Partition Flow

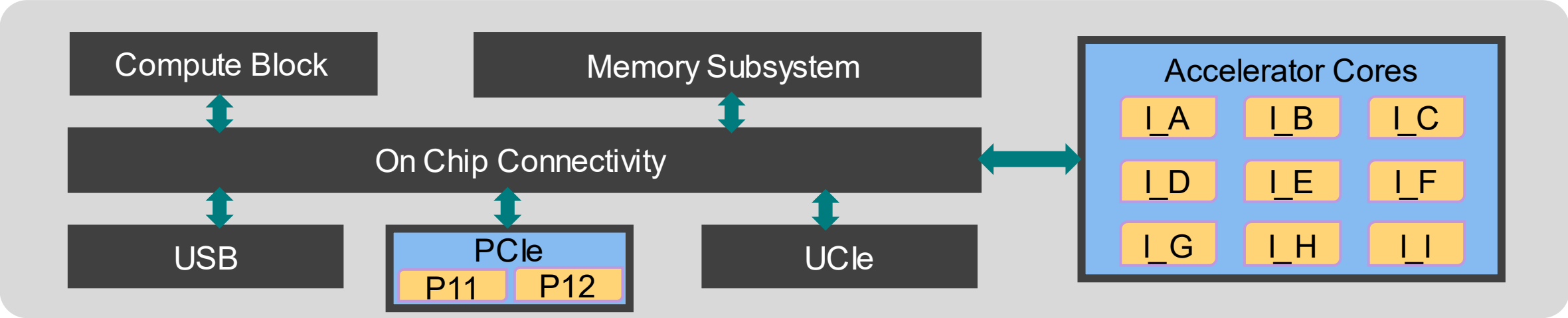


HAPS
Timing-Driven
Partitioning
Engine



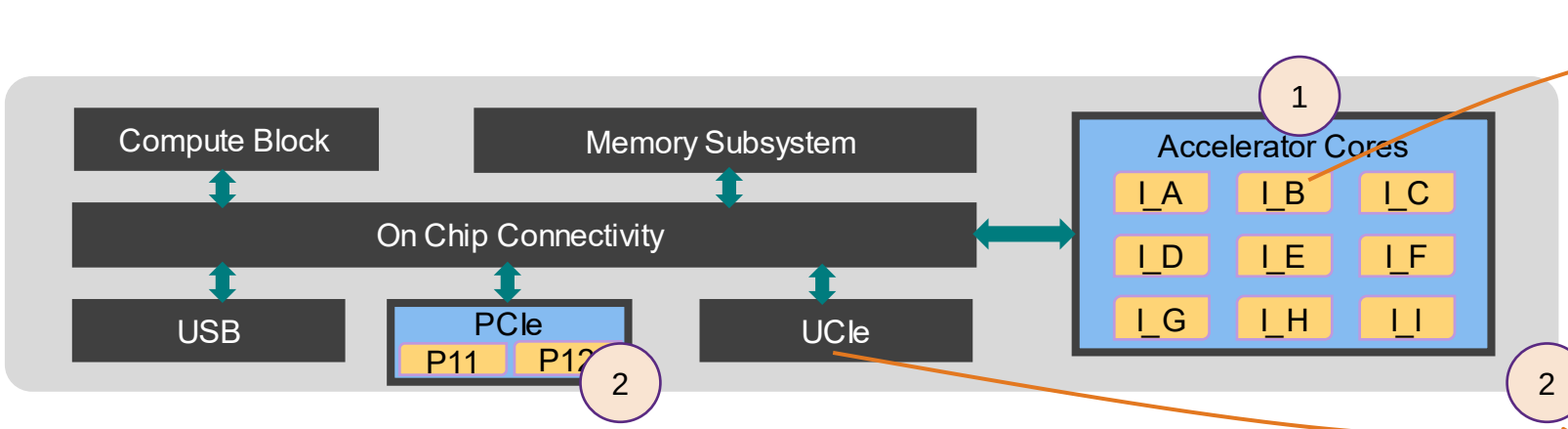
- User-guided, timing-aware partition into subsystems
- Self-contained, netlist-based subsystem projects
- Automated or guided partition of subsystems to FPGAs

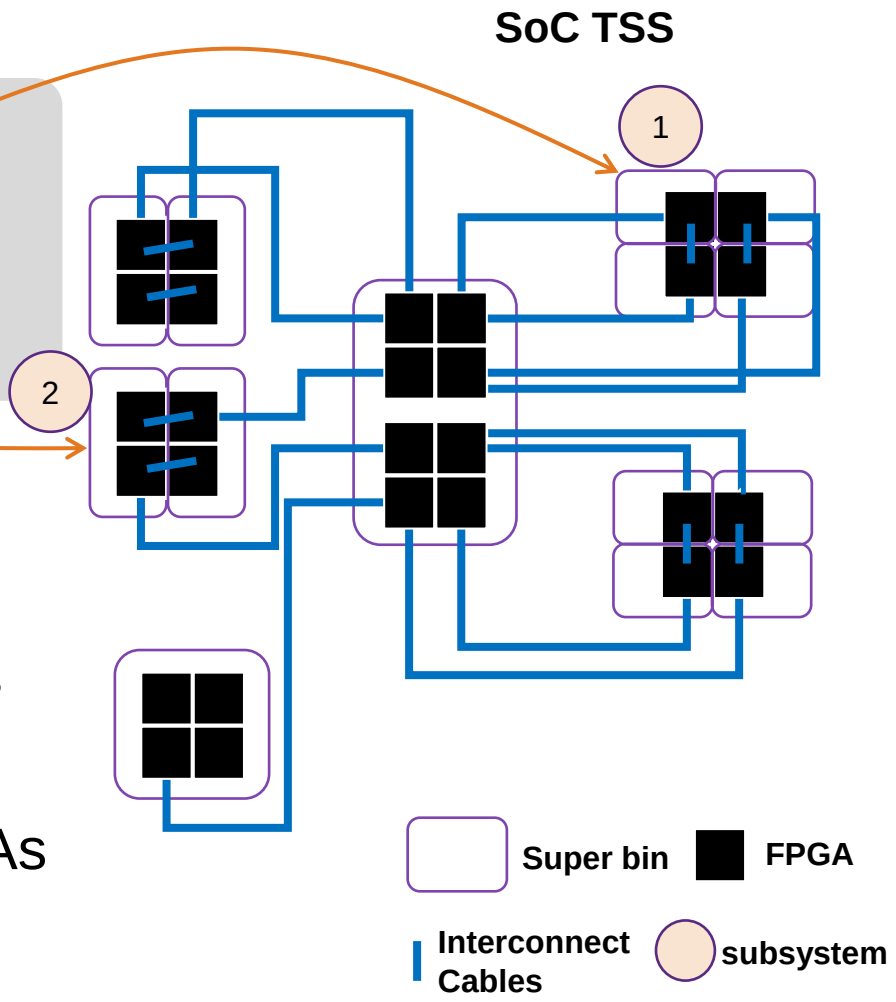
Modular First-Pass Guided Partition



- User-defined subsystems
- Well-defined interface between subsystems
- Identification of replicated subsystems
- Resolve cross boundary dependencies

First Pass Partition & System Route



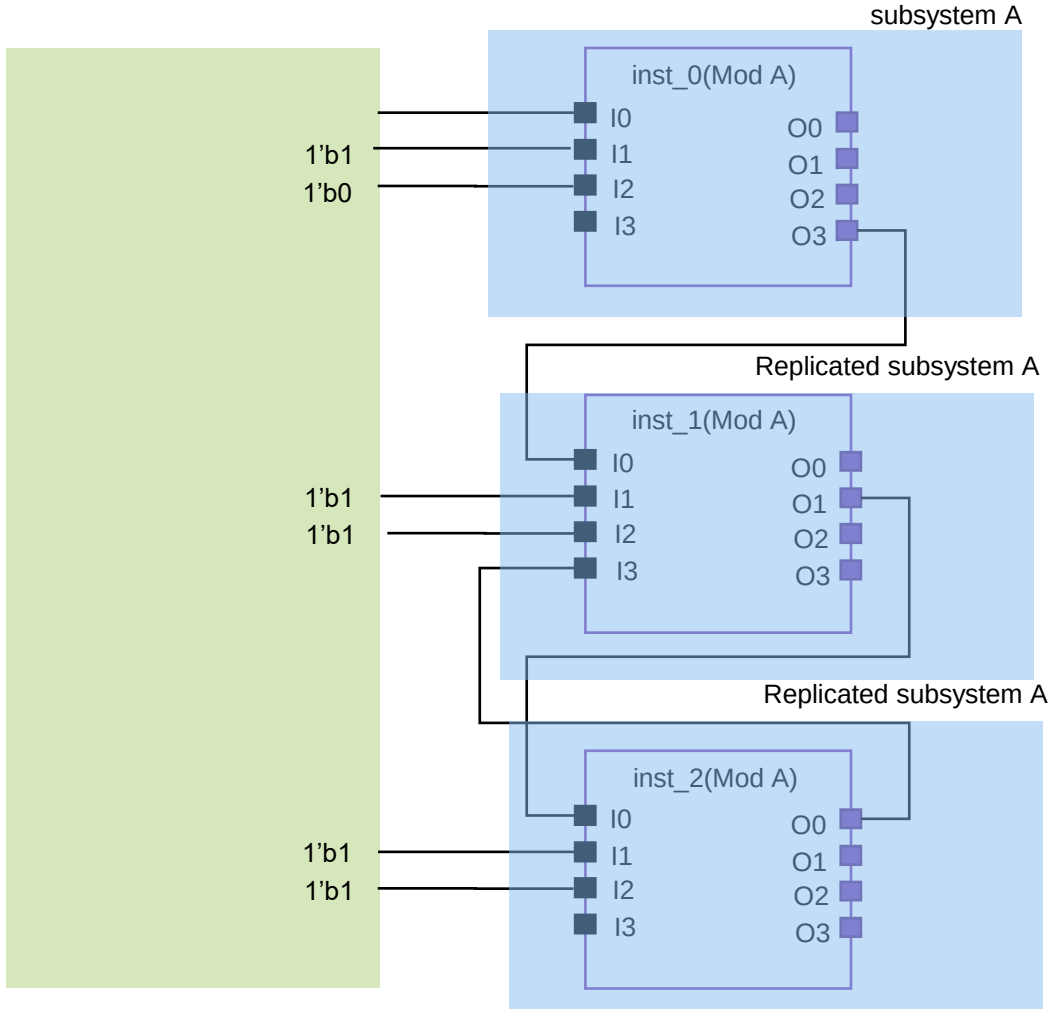


- Users define subsystem module contents
- Users specify unique or replicated subsystems
- Replicated subsystems must map to identical super bins
- Users define the HAPS HW model (Soc TSS)
- Each subsystem will be assigned to a collection of FPGAs (super bin)

Modular Partition Considerations

All replicated copies must be identical

- Interface drivers/sinks can change
- Preserve subsystem IOs by default
- Enable cross-boundary optimizations (constants, GCC)
- Use worst-case timing for all IOs



Modular HAV Flow Example

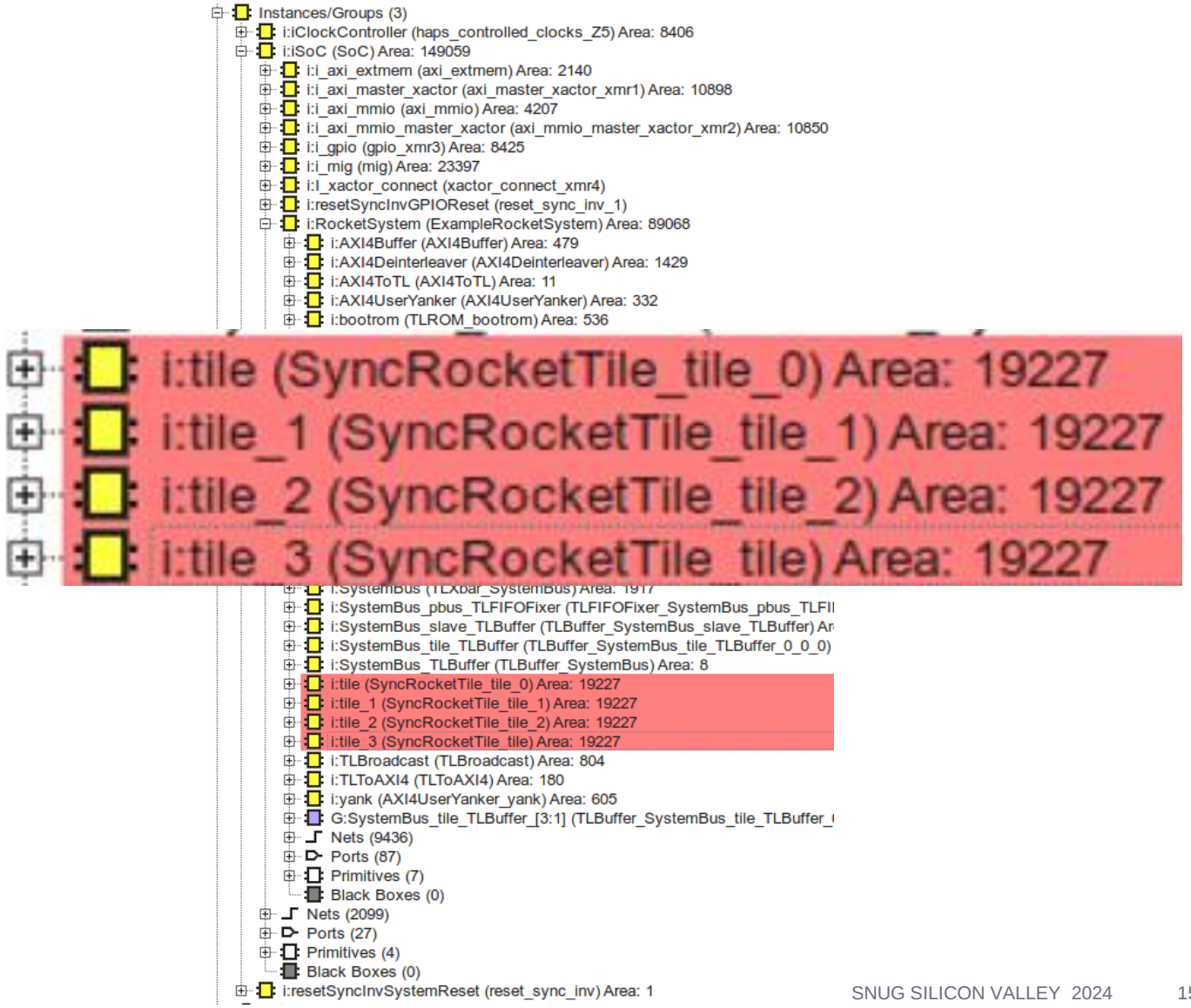
Design Overview

- RISC Processor Design
  - 4 identical **RocketTile** blocks
  - Design size

Resource	Utilization
RAM	582
DSP	46
LUT	125076
FF	83478
Top level ports	26

  - Timing Constraints
    - 3 top level design clocks
      - **Master clock**
      - **JTAG clock**
      - **DDR clock**

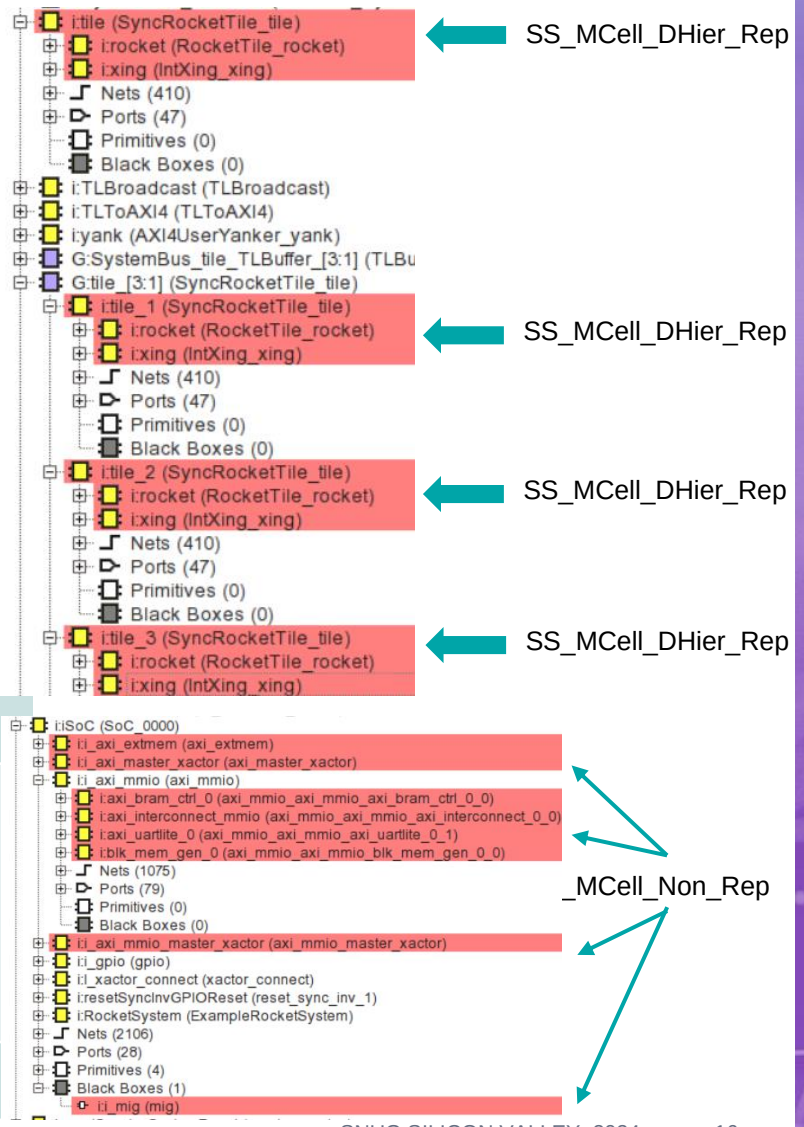
Design Hierarchy View



Modular Flow Setup

Subsystem Type Identification

Name of subsystem	Instance assigned to subsystem	Type of subsystem
SS_MCell_DHier_Rep	{i:iSoC.RocketSystem.tile.rocket} **i_SS_MCell_DHier_Rep_1** {i:iSoC.RocketSystem.tile.xing} **i_SS_MCell_DHier_Rep_1** {i:iSoC.RocketSystem.tile_1.rocket} **i_SS_MCell_DHier_Rep_2** {i:iSoC.RocketSystem.tile_1.xing} **i_SS_MCell_DHier_Rep_2** {i:iSoC.RocketSystem.tile_2.rocket} **i_SS_MCell_DHier_Rep_3** {i:iSoC.RocketSystem.tile_2.xing} **i_SS_MCell_DHier_Rep_3** {i:iSoC.RocketSystem.tile_3.rocket} **i_SS_MCell_DHier_Rep_4** {i:iSoC.RocketSystem.tile_3.xing} **i_SS_MCell_DHier_Rep_4**	Multi cell & replicated
SS_Mcell_Non_Rep	{{i:iSoC.i_axi_mmio.axi_uartlite_0} {i:iSoC.i_axi_mmio.axi_interconnect_mmio} {i:iSoC.i_axi_mmio.blk_mem_gen_0} {i:iSoC.i_axi_mmio.axi_bram_ctrl_0}} **{i_SS_Mcell_Non_Rep_1}** {{iSoC.i_mig} {i:iSoC.i_axi_mmio_master_xactor} {i:iSoC.i_axi_master_xactor} {i:iSoC.i_axi_extmem}} **{i_SS_Mcell_Non_Rep_1}**	Multi cell from different hierarchy and Non replicated subsystem
ROD	Cells auto assigned by tool	Rest of Design



First Pass - Partition

Subsystem Definition

TCL command - option set subsystem_partition 1

PCF commands

Syntax – subsystem_create <SS name> <instance> -bins <FPGAs>

Replicated subsystem will have same subsystem name.

Create subsystem

subsystem_create SS_MCell_DHier_Rep i_SS_MCell_DHier_Rep_1 -bins {mb_c0.uA mb_c0.uB}

subsystem_create SS_MCell_DHier_Rep i_SS_MCell_DHier_Rep_2 -bins {mb_c1.uA mb_c1.uB}

subsystem_create SS_MCell_DHier_Rep i_SS_MCell_DHier_Rep_3 -bins {mb_c2.uA mb_c2.uB}

subsystem_create SS_MCell_DHier_Rep i_SS_MCell_DHier_Rep_4 -bins {mb_c3.uA mb_c3.uB}

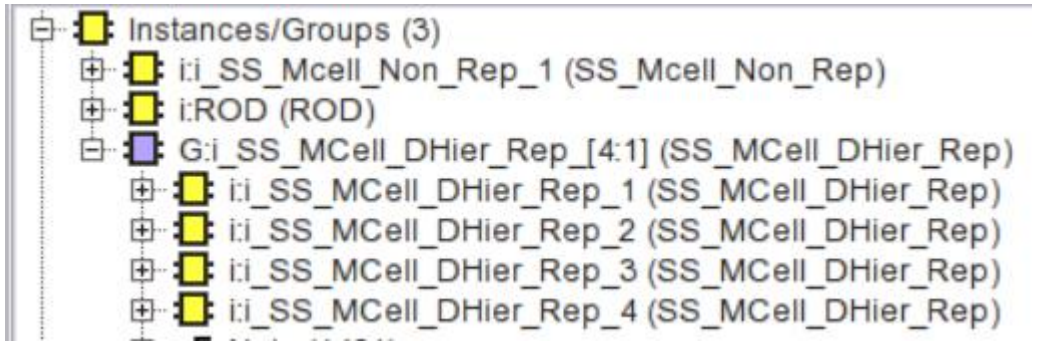
subsystem_create SS_Mcell_Non_Rep i_SS_Mcell_Non_Rep_1 -bins {mb_c0.uC mb_c0.uD}

ROD

Remaining unlocked FPGAs are considered for ROD.

bin_attribute -locked {mb_c1.uC mb_c1.uD mb_c2.uC mb_c2.uD mb_c3.uC mb_c3.uD mb_h.uC mb_h.uD}

Partition schematic




```
@N: AP238 |Target System Summary After Constraints:
Name: 'SUBSYSTEM' Bins: 61

FPGA_Bin               Pins                  LUT    LUTM      DFF BRAM URAM  DSP ZDPIMOD   IO
-------------------------------------------------------------------------------------------------
mb_h.uC                 108 LOCKED (MODULAR FLOW)
mb_h.uD                 108 LOCKED (MODULAR FLOW)
mb_c3.uC                158 LOCKED (MODULAR FLOW)
mb_c3.uD                158 LOCKED (MODULAR FLOW)
mb_c2.uC                108 LOCKED (MODULAR FLOW)
mb_c2.uD                108 LOCKED (MODULAR FLOW)
mb_c1.uC                108 LOCKED (MODULAR FLOW)
mb_c1.uD                108 LOCKED (MODULAR FLOW)
ROD                     588               8171520 4085760 16343040 4320  640 7680    1984 4144
i_SS_MCell_DHier_Rep_1  135                LOCKED
i_SS_MCell_DHier_Rep_2  135                LOCKED
i_SS_MCell_DHier_Rep_3  135                LOCKED
i_SS_MCell_DHier_Rep_4  135                LOCKED
i_SS_Mcell_Non_Rep_1    316               8171520 4085760 16343040 4320  640 7680    1984 4144
=================================================================================================
```

First Pass - Partition

Module Assignment to subsystems; Tool Outputs

PCF assignment of cells to subsystems

assign_cell {i:iSoC.RocketSystem.tile.rocket} i_SS_MCell_DHier_Rep_1

assign_cell {i:iSoC.RocketSystem.tile.xing} i_SS_MCell_DHier_Rep_1

assign_cell {i:iSoC.RocketSystem.tile_1.rocket} i_SS_MCell_DHier_Rep_2

assign_cell {i:iSoC.RocketSystem.tile_1.xing} i_SS_MCell_DHier_Rep_2

assign_cell {i:iSoC.RocketSystem.tile_2.rocket} i_SS_MCell_DHier_Rep_3

assign_cell {i:iSoC.RocketSystem.tile_2.xing} i_SS_MCell_DHier_Rep_3

assign_cell {i:iSoC.RocketSystem.tile_3.rocket} i_SS_MCell_DHier_Rep_4

assign_cell {i:iSoC.RocketSystem.tile_3.xing} i_SS_MCell_DHier_Rep_4

assign_cell {{i:iSoC.i_axi_mmio.axi_uartlite_0} {i:iSoC.i_axi_mmio.axi_interconnect_mmio} {i:iSoC.i_axi_mmio.blk_mem_gen_0} {i:iSoC.i_axi_mmio.axi_bram_ctrl_0}} {i_SS_Mcell_Non_Rep_1}

PCF assignment of cells to FPGA

assign_cell {{iSoC.i_mig} mb_c0.uA

Partition logs – same format as standard partition flow

Connectivity	Point-to-Point	Mult-Terminal
mb_h.uC<->mb_h.uD (FIXED)	81	0
mb_c3.uC<->mb_c3.uD (FIXED)	81	0
mb_c3.uC<->mb_c3.uD	50	0
mb_c2.uC<->mb_c2.uD (FIXED)	81	0
mb_c1.uC<->mb_c1.uD (FIXED)	81	0
ROD<->i_SS_MCell_DHier_Rep_1	100	0
ROD<->i_SS_MCell_DHier_Rep_2	100	0
ROD<->i_SS_MCell_DHier_Rep_3	100	0
ROD<->i_SS_MCell_DHier_Rep_4	100	0
ROD<->i_SS_Mcell_Non_Rep_1	100	0
ROD<->TOP_IO_HT3_mb_h_A24	52	0
i_SS_Mcell_Non_Rep_1<->TOP_IO_HT3_mb_c0_D24	52	0
i_SS_Mcell_Non_Rep_1<->ddr3_sodimm2r_ht3.dimm	124	0
i_SS_Mcell_Non_Rep_1<->ddr3_sodimm2r_ht3.osc	2	0
i_SS_Mcell_Non_Rep_1<->ddr3_sodimm2r_ht3.led	3	0

DRC checks for subsystem equivalence

```
Subsystem equivalence DRC has detected non-equivalent instances:
   Root bin i_SS_MCell_DHier_Rep_1:
      iSoC.RocketSystem.tile.xing (IntXing_xing)
      iSoC.RocketSystem.tile.rocket (RocketTile_rocket)
   Copy bin i_SS_MCell_DHier_Rep_3:
      iSoC.RocketSystem.tile_2.xing (IntXing_xing)
      iSoC.RocketSystem.tile_2.rocket (RocketTile_rocket)
      iSoC.RocketSystem.tile_3.xing (IntXing_xing)
      iSoC.RocketSystem.tile_3.rocket (RocketTile_rocket)
@E: AP692 |Subsystem equivalence DRC requires that all copy bins contain only equivalent instances.
```

Pin compatibility checks for replicated subsystems

```
@N: AP636 |Subsystem TSS Summary

SubSystem           Instances                Bins                 Inner-connection  Outer-connecti
-------------------------------------------------------------------------------------------------
ROD                 ROD                      {mb_h.uA mb_h.uB}    139/139           588/588
SS_MCell_DHier_Rep  i_SS_MCell_DHier_Rep_1   {mb_c0.uA mb_c0.uB}  139/139           135/135
                    i_SS_MCell_DHier_Rep_2   {mb_c1.uA mb_c1.uB}  139/139           135/135
                    i_SS_MCell_DHier_Rep_3   {mb_c2.uA mb_c2.uB}  139/139           135/135
                    i_SS_MCell_DHier_Rep_4   {mb_c3.uA mb_c3.uB}  139/139           135/135
SS_Mcell_Non_Rep    i_SS_Mcell_Non_Rep_1     {mb_c0.uC mb_c0.uD}  89/89             316/316
=================================================================================================
```

First Pass – System Route

- Same report as seen in non-modular flow.
- Connections are shown between ROD and subsystems instead of FPGAs.
- Pass 1 reports inter subsystem metrics.
- Positive slack indicates timing is met post TDM insertion.

@S5.1.4 AP655 |Post-Optimization Routing Summary

Connection	Trace Usage	No. of Net failed timing	TDM Module	Available Slack Range (Post TDM)	Mean Slack	Untimed nets
ROD<->i_SS_Mcell_Non_Rep_1	100/100					
		0/46	NONE (11 ns)	84 to 989 ns	206 ns	2
		0/349	HSTDM_8 (40 ns)	152 to 159 ns	155 ns	0
		0/2	HSTDM_16 (50 ns)	147 to 147 ns	147 ns	0
ROD<->i_SS_MCell_DHier_Rep_1	100/100					
		0/50	NONE (11 ns)	87 to 989 ns	207 ns	0
		0/311	HSTDM_8 (40 ns)	152 to 159 ns	155 ns	0
		0/15	HSTDM_256 (226 ns)			15
ROD<->i_SS_MCell_DHier_Rep_2	100/100					
		0/50	NONE (11 ns)	87 to 989 ns	207 ns	0
		0/311	HSTDM_8 (40 ns)	152 to 159 ns	155 ns	0
		0/15	HSTDM_256 (226 ns)			15
ROD<->i_SS_MCell_DHier_Rep_3	100/100					
		0/50	NONE (11 ns)	87 to 989 ns	207 ns	0
		0/311	HSTDM_8 (40 ns)	152 to 159 ns	155 ns	0
		0/15	HSTDM_256 (226 ns)			15
ROD<->i_SS_MCell_DHier_Rep_4	100/100					
		0/50	NONE (11 ns)	87 to 989 ns	207 ns	0
		0/311	HSTDM_8 (40 ns)	152 to 159 ns	155 ns	0
		0/15	HSTDM_256 (226 ns)			15

First Pass – Output Files

- Subsystem - Replicated

```
shibin@fpga512comp1:/remote/sbg_qaresults/regression/77963/slowfs/sbg_builds20/jensine/haps_soc_demo_v0/SLP_FPGA_kb/i_SS_MCell_DHier_Rep_1 % ls
board.pcf       i_SS_MCell_DHier_Rep_1_attr.fdc         i_SS_MCell_DHier_Rep_1_srs          i_SS_MCell_DHier_Rep_1_srs.tcl   protocompiler_run_i_SS_MCell_DHier_Rep_1_srs_job.log  syn_dics_i_SS_MCell_DHier_Rep_1.fdc
board.tss       i_SS_MCell_DHier_Rep_1_haps_timing.fdc  i_SS_MCell_DHier_Rep_1.srs          i_SS_MCell_DHier_Rep_1_timing.fdc protocompiler_run_i_SS_MCell_DHier_Rep_1_srs.log      synlog.tcl
cdc_files.txt   i_SS_MCell_DHier_Rep_1_iostd.fdc        i_SS_MCell_DHier_Rep_1_srs.html     options.tcl                      sg0_fpga_path                                        testdata
fdc_files.txt   i_SS_MCell_DHier_Rep_1_pinloc.fdc       i_SS_MCell_DHier_Rep_1_srs_reports  pcffiles.txt                     src_srs.txt                                          version_info.txt
```

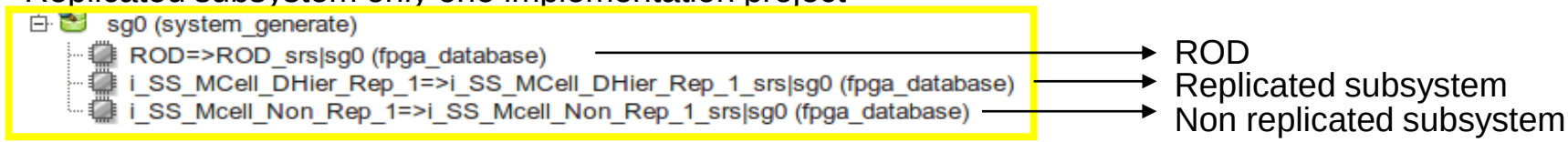
- Subsystem – Non-Replicated

```
shibin@fpga512comp1:/remote/sbg_qaresults/regression/77963/slowfs/sbg_builds20/jensine/haps_soc_demo_v0/SLP_FPGA_kb/i_SS_Mcell_Non_Rep_1 % ls
board.pcf       i_SS_Mcell_Non_Rep_1_attr.fdc         i_SS_Mcell_Non_Rep_1_srs          i_SS_Mcell_Non_Rep_1_srs.tcl   protocompiler_run_i_SS_Mcell_Non_Rep_1_srs_job.log  syn_dics_i_SS_Mcell_Non_Rep_1.fdc
board.tss       i_SS_Mcell_Non_Rep_1_haps_timing.fdc  i_SS_Mcell_Non_Rep_1.srs          i_SS_Mcell_Non_Rep_1_timing.fdc protocompiler_run_i_SS_Mcell_Non_Rep_1_srs.log      synlog.tcl
cdc_files.txt   i_SS_Mcell_Non_Rep_1_iostd.fdc        i_SS_Mcell_Non_Rep_1_srs.html     options.tcl                    sg0_fpga_path                                       testdata
fdc_files.txt   i_SS_Mcell_Non_Rep_1_pinloc.fdc       i_SS_Mcell_Non_Rep_1_srs_reports  pcffiles.txt                   src_srs.txt                                         version_info.txt
```

- ROD

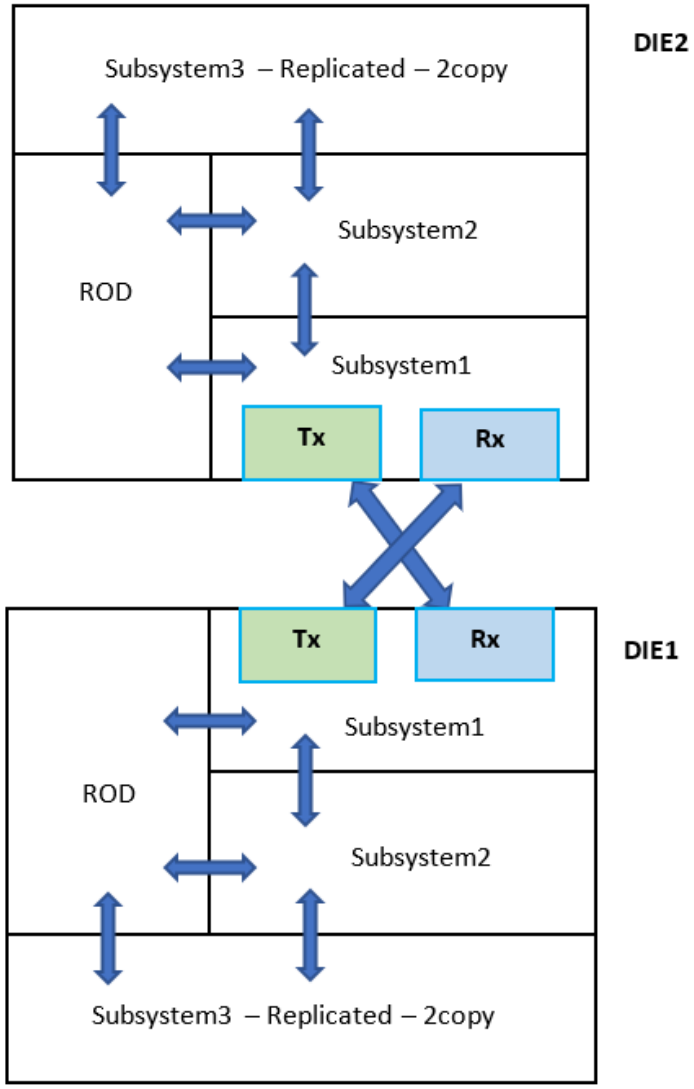
```
shibin@fpga512comp1:/remote/sbg_qaresults/regression/77963/slowfs/sbg_builds20/jensine/haps_soc_demo_v0/SLP_FPGA_kb/ROD % ls
board.pcf      fdc_files.txt  protocompiler_job.log  protocompiler.log.bak.1  protocompiler_run_ROD_srs_job.log  ROD_haps_timing.fdc  ROD_srs          ROD_srs_reports  sg0_fpga_path      syn_dics_ROD.fdc  version_info.txt
board.tss      options.tcl    protocompiler.log      protocompiler.log.bak.2  protocompiler_run_ROD_srs.log      ROD_iostd.fdc        ROD.srs          ROD_srs.tcl      sg0_fpga_path_exp  synlog.tcl
cdc_files.txt  pcffiles.txt   protocompiler.log.bak  protocompiler.log.bak.3  ROD_attr.fdc                       ROD_pinloc.fdc       ROD_srs.html     ROD_timing.fdc   src_srs.txt        testdata
```

- Subsystem/RoD Wrapper TCL scripts
  - *ROD_srs.tcl*
  - *i_SS_MCell_DHier_Rep_1_srs.tcl*
  - *i_SS_Mcell_Non_Rep_1_srs.tcl*
- Replicated subsystem only one implementation project



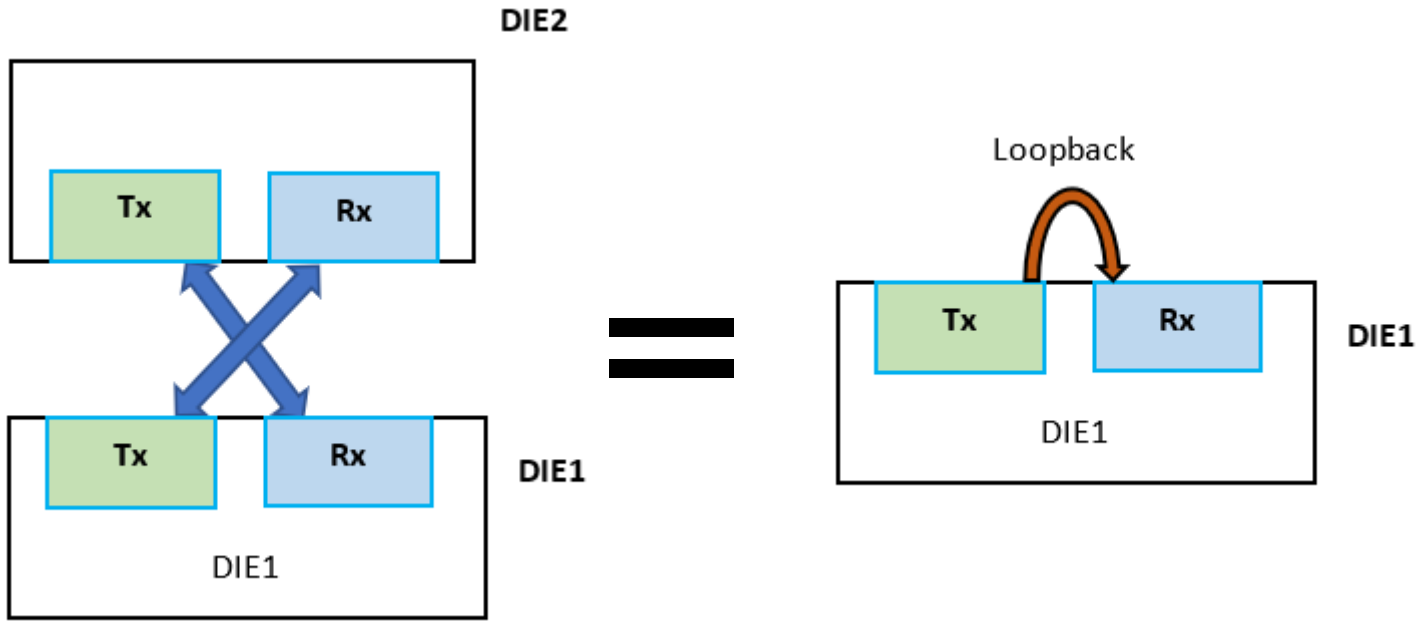
Die-To-Die Example

Die-To-Die Use Case



- Multiple prototype models to satisfy requirements of different end users
- Small model (e.g a single die)
- Big model (multi-die or die-to-die)

Loopback Concept

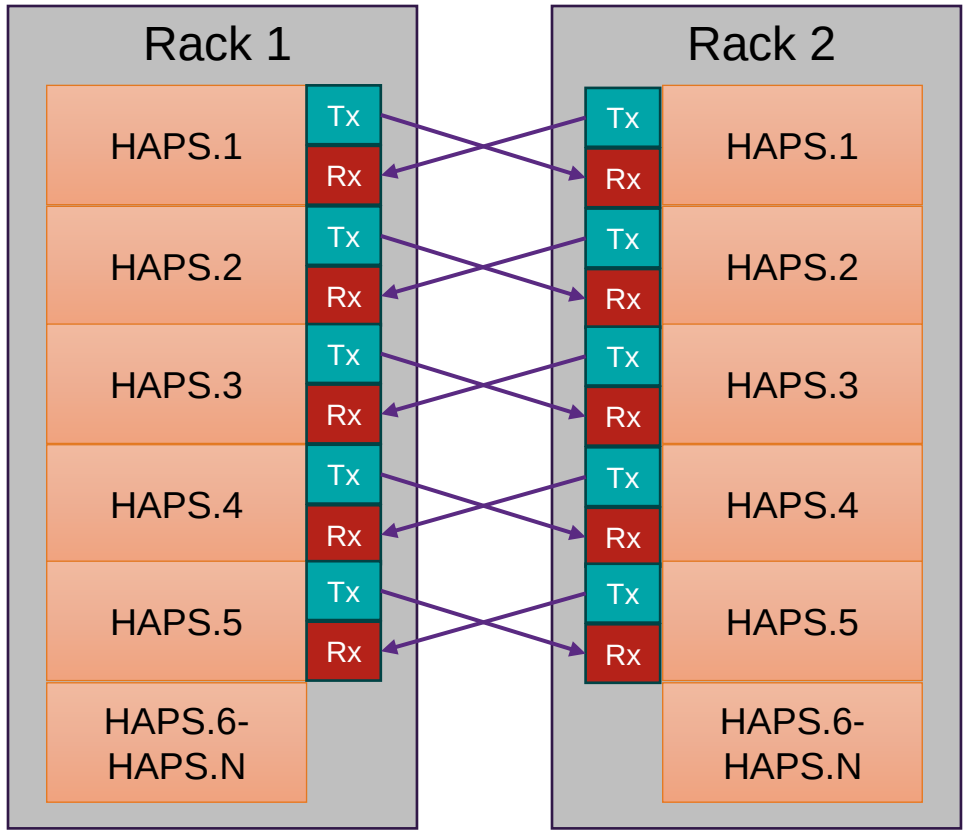


- Above models are functionally and timing equivalent!
- Mitigate design scope, resource requirements by only processing a single die copy

Die-To-Die Case Study

- Single-die design is 60 FPGAs, requires a modular implementation flow for fast, predictable build times.
- Need to contain compute, storage resources.
- Many copies of the single die design to be deployed.
- Very few dual die designs to be deployed.
- Want a single implementation project/flow for both scenarios.
- Hardware to be configured for either single die or dual die usage.

Die-To-Die Case Study: Physical Setup



- Single die = single rack configuration
- Physically cable between 2 racks to realize the dual die setup
- Rack 1, Rack 2 can be run as 2-single die designs OR 1-dual die design
- No need for dedicated dual die implementation & bring up

Setup Requirements

- Model the loopback connection (Tx::Rx) in the Target System Specification (TSS) for the SW flow

board_system_create -interconnect -manual HT3C_CABLE_200 -name conn_ss1_lpbk_1 -connector {FB1.F6J2 FB1.F6J3}

- For the runtime configuration, model the TSS connectivity as per the actual hardware

board_system_create -interconnect -manual HT3C_CABLE_200 -name conn_ss1_lpbk_1 -connector {FB1.F6J2 FB2.F6J3}

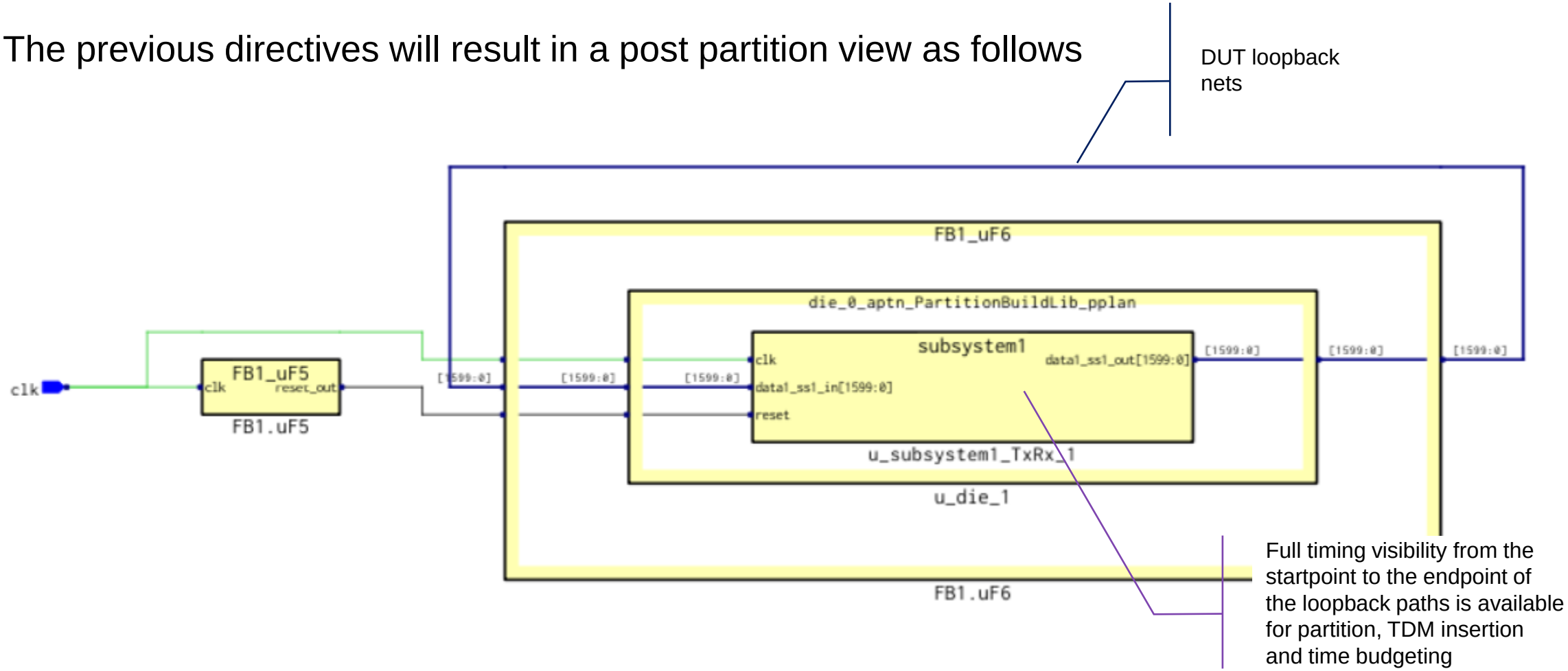
board_system_create -interconnect -manual HT3C_CABLE_200 -name conn_ss1_lpbk_1 -connector {FB1.F6J3 FB2.F6J2}

Partition Setup

- The following partition constraint tells the software which design nets comprise the loopback bus and which connectors are allocated for loopback use:
  - PCF command: `loopback_nets`
    `<list of nets>`
    `-trace_group <list of trace groups>`
    `-function <function group name of the loopback connections>`
  - Example:
    - TSS command: `board_system_create -interconnect -manual HT3C_CABLE_200 -name conn_ss1_lpbk_1 -connector {FB1.F6J2 FB1.F6J3}`
    - PCF command: `loopback_nets {n:u_die_1.data1_ss1_lpbk[1599:0]} -trace_group conn_ss1_lpbk_1 -function group1`

Partition Schematic

- The previous directives will result in a post partition view as follows



Software Reports/Outputs: system_route.log

Reference directives:

TSS command: `board_system_create -interconnect -manual HT3C_CABLE_200 -name conn_ss1_lpbk_1 -connector {FB1.F6J2 FB1.F6J3}`

PCF command: `loopback_nets {n:u_die_1.data1_ss1_lpbk[1599:0]} -trace_group conn_ss1_lpbk_1 -function group1`

```
@S5.1.2 AP265 |Global Route Trace Usage

Trace  Function  Trace_Usage   Clock_Usage  Net_Usage  Module    Trace_Name/Connection
-------------------------------------------------------------------------------------
   1)  group1          48/51                    1600            Trace Group: conn_ss1_lpbk_1
                          0            0          0  DIRECT    FB1.uF6.F6J3-> {FB1.uF6.F6J3}
                         46                    1584  HSTDM_72
                          2*                     16  HSTDM_16
```

- Group and trace connection overview

Software Outputs/Reports: system_route.rpt

Reference directives:

TSS command: `board_system_create -interconnect -manual HT3C_CABLE_200 -name conn_ss1_lpbk_1 -connector {FB1.F6J2 FB1.F6J3}`

PCF command: `loopback_nets {n:u_die_1.data1_ss1_lpbk[1599:0]} -trace_group conn_ss1_lpbk_1 -function group1`

```
@S3.1 AP346 |Routed Nets

net_attribute -function group1 u_die_1.data1_ss1_lpbk[1598]
global_route u_die_1.data1_ss1_lpbk[1598] -slack_pre_tdm 200 -slack_post_tdm 133.598 -source_clock clk:r \
   -from FB1.uF6 -to {FB1.uF6} -using conn_ss1_lpbk_1 -tdm HSTDM -ratio 72

net_attribute -function group1 u_die_1.data1_ss1_lpbk[1597]
global_route u_die_1.data1_ss1_lpbk[1597] -slack_pre_tdm 200 -slack_post_tdm 133.598 -source_clock clk:r \
   -from FB1.uF6 -to {FB1.uF6} -using conn_ss1_lpbk_1 -tdm HSTDM -ratio 72

net_attribute -function group1 u_die_1.data1_ss1_lpbk[1594]
global_route u_die_1.data1_ss1_lpbk[1594] -slack_pre_tdm 200 -slack_post_tdm 133.598 -source_clock clk:r \
   -from FB1.uF6 -to {FB1.uF6} -using conn_ss1_lpbk_1 -tdm HSTDM -ratio 72

net_attribute -function group1 u_die_1.data1_ss1_lpbk[1593]
global_route u_die_1.data1_ss1_lpbk[1593] -slack_pre_tdm 200 -slack_post_tdm 133.598 -source_clock clk:r \
   -from FB1.uF6 -to {FB1.uF6} -using conn_ss1_lpbk_1 -tdm HSTDM -ratio 72
```

Summary

Summary

- Large Multi-core and Multi-die designs present new validation challenges
- Modular HAV provides a scalable approach for prototyping large designs
- IP teams can build high-performance prototypes optimized for subsystem
- System integrators can combine multiple prototypes for chip validation
- SW developers can find and resolve complex bugs on multi-core prototypes


THANK YOU
YOUR
INNOVATION
YOUR
COMMUNITY