2024
DESIGN AND VERIFICATION™
DVCON
CONFERENCE AND EXHIBITION
UNITED STATES
SAN JOSE, CA, USA
MARCH 4-7, 2024

New Innovative Way to Verify Package Connectivity

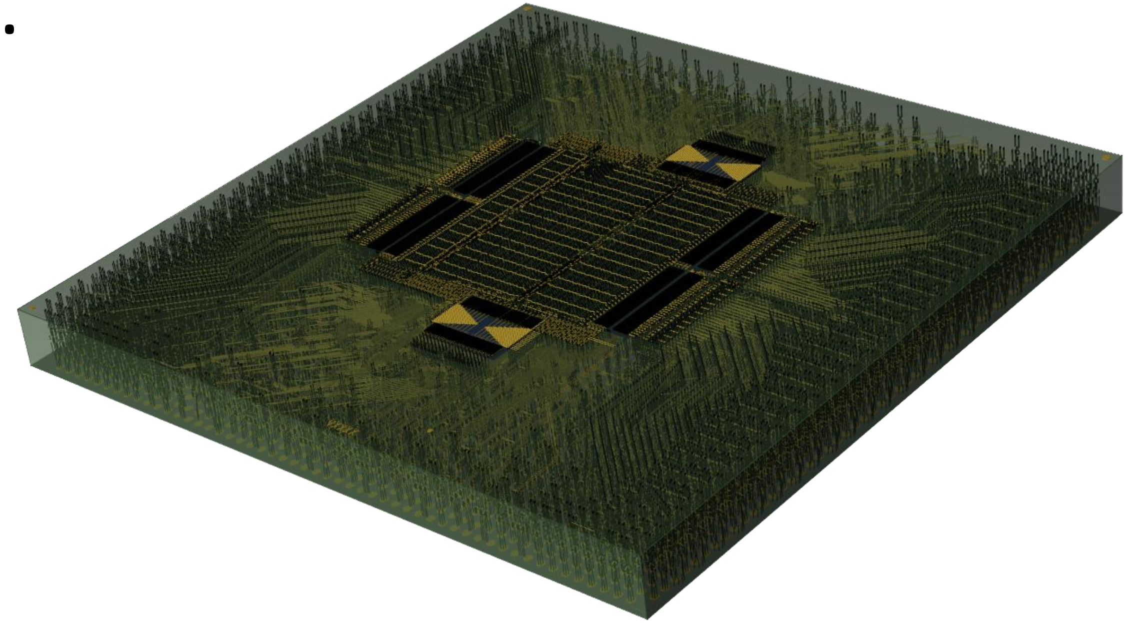
Mike Walsh (mike.walsh@siemens.com) – Siemens EDA Cary, NC

Jin Hou (hou.jin@siemens.com) – Siemens EDA Fremont, CA

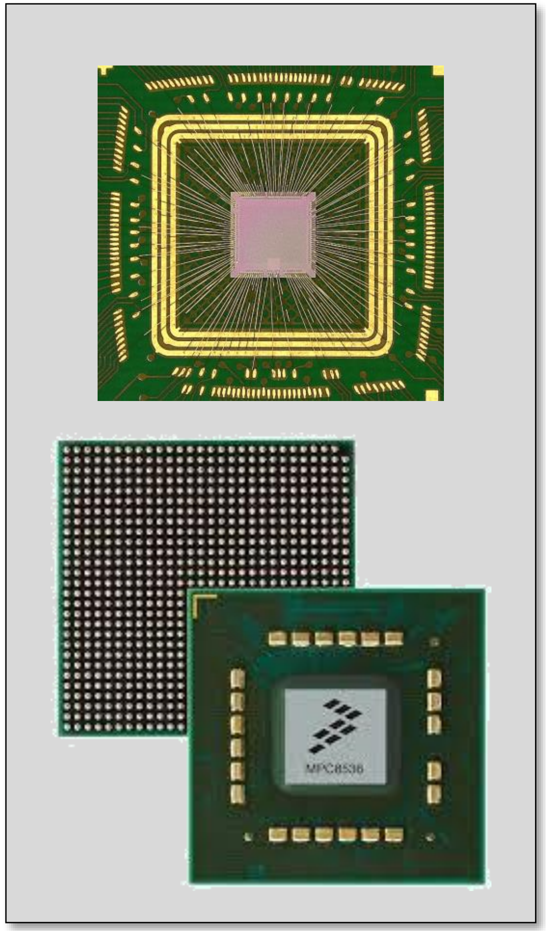
SIEMENS

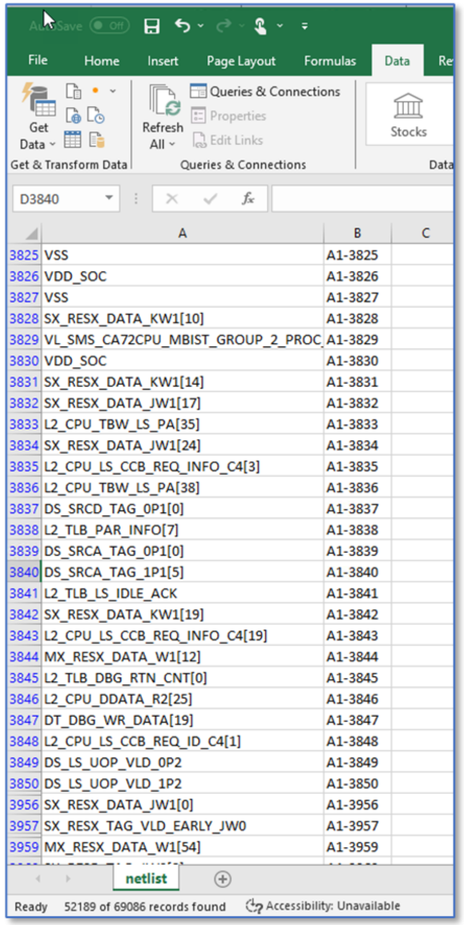
accellera
SYSTEMS INITIATIVE

Integration of multiple ICs (chiplets) on a single substrate is critical for high performance computing. Due to the large number of connections involved connecting the various ICs, it is hard to verify the correctness of the connections.



Historical vs Modern Package Design





	A	B	C
3825	VSS	A1-3825	
3826	VDD_SOC	A1-3826	
3827	VSS	A1-3827	
3828	SX_RESX_DATA_KW1[10]	A1-3828	
3829	VL_SMS_CA72CPU_MBIST_GROUP_2_PROC	A1-3829	
3830	VDD_SOC	A1-3830	
3831	SX_RESX_DATA_KW1[14]	A1-3831	
3832	SX_RESX_DATA_JW1[17]	A1-3832	
3833	L2_CPU_TBW_LS_PA[35]	A1-3833	
3834	SX_RESX_DATA_JW1[24]	A1-3834	
3835	L2_CPU_LS_CCB_REQ_INFO_C4[3]	A1-3835	
3836	L2_CPU_TBW_LS_PA[38]	A1-3836	
3837	DS_SRCD_TAG_0P1[0]	A1-3837	
3838	L2_TLB_PAR_INFO[7]	A1-3838	
3839	DS_SRCA_TAG_0P1[0]	A1-3839	
3840	DS_SRCA_TAG_1P1[5]	A1-3840	
3841	L2_TLB_LS_IDLE_ACK	A1-3841	
3842	SX_RESX_DATA_KW1[19]	A1-3842	
3843	L2_CPU_LS_CCB_REQ_INFO_C4[19]	A1-3843	
3844	MX_RESX_DATA_W1[12]	A1-3844	
3845	L2_TLB_DBG_RTN_CNT[0]	A1-3845	
3846	L2_CPU_DDATA_R2[25]	A1-3846	
3847	DT_DBG_WR_DATA[19]	A1-3847	
3848	L2_CPU_LS_CCB_REQ_ID_C4[1]	A1-3848	
3849	DS_LS_UOP_VLD_0P2	A1-3849	
3850	DS_LS_UOP_VLD_1P2	A1-3850	
3956	SX_RESX_DATA_JW1[0]	A1-3956	
3957	SX_RESX_TAG_VLD_EARLY_JW0	A1-3957	
3959	MX_RESX_DATA_W1[54]	A1-3959	

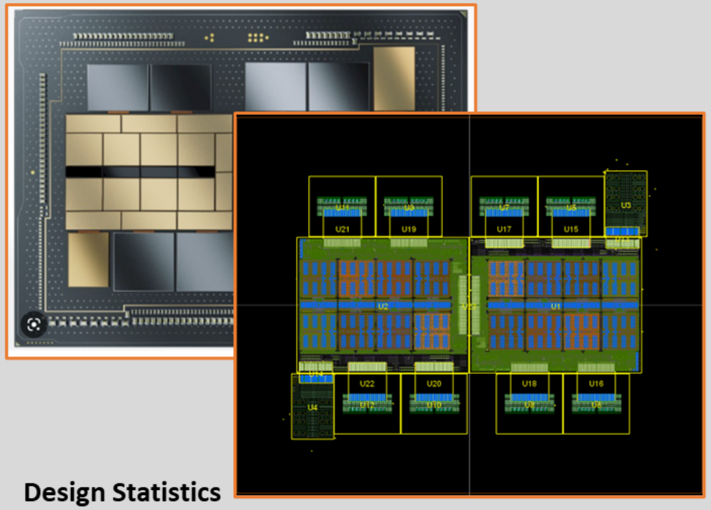
Ready 52189 of 69086 records found Accessibility: Unavailable

Traditional Package Netlist

- Captured in Excel
- Simple table of net names and pin assignments
- Created by package designer using input from silicon team
- Low risk for single die and low pin count package designs
- Netlist verification performed manually or through simple text compare scripts

Package Complexity has exploded

- Multi-die designs are common
- Device pin counts approaching 1m and trending higher
- 200k+ net connections
- Created from disparate data sources (GDS, LEF/DEF, CSV)
- Netlist verification is a challenge – does netlist match intent?

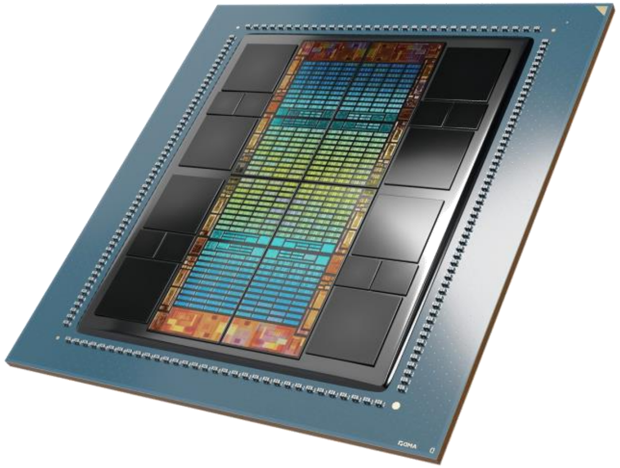


Design Statistics

- 47 die, 5 process nodes
- >100B transistors
- 77.5mm x 62.5mm
- 51,708 nets
- 214,657 connections
- 25 layers
- 250K device pins
- 606K vias

Modern substrate design has become a system integration task:

- Heterogeneous integration introduces new challenges for the substrate design engineer
- Source data is being aggregated from a myriad of data formats:
    - Ball map CSV files
    - OASIS, GDS, and LEF/DEF from P&R tools
    - Verilog RTL
    - Spreadsheet Data
    - Plain text files
    - 3Dblox

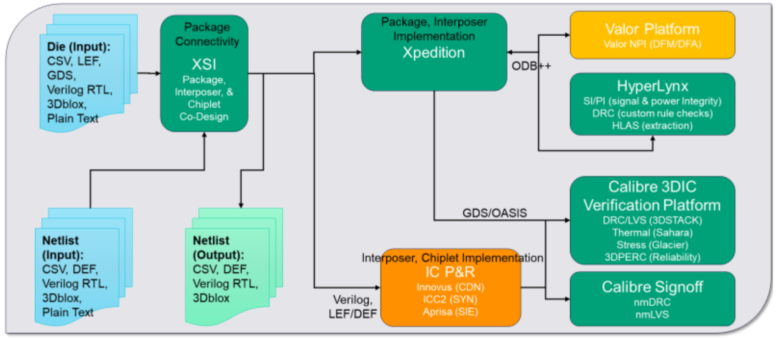


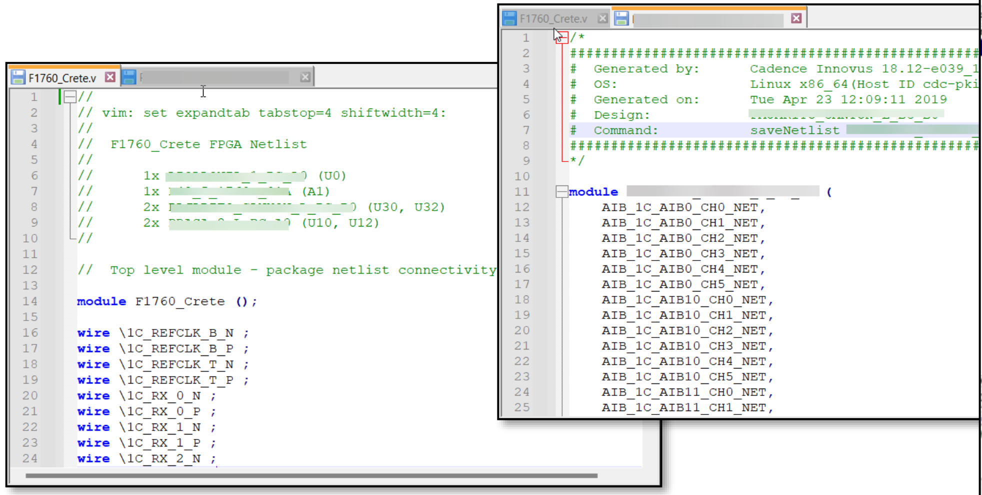
Legacy connectivity methods no longer viable

- HBM[‡] is driving explosion of connectivity for AI and HPC applications
- Using Excel for netlist management is no longer practical
- Verilog RTL is becoming more prevalent due to chiplet designs
  - Far more efficient to connect die to die interfaces
  - Use of bus notation greatly decreases chance of error (d[63:0] vs d63, d62, …., d1, d0)
  - Ability to re-use RTL work done by ASIC design and verification teams
- Specialized tools such as Siemens Xpedition Substrate Integrator (XSI) address the challenges of managing large scale connectivity problems

[‡] HBM: https://en.wikipedia.org/wiki/High_Bandwidth_Memory

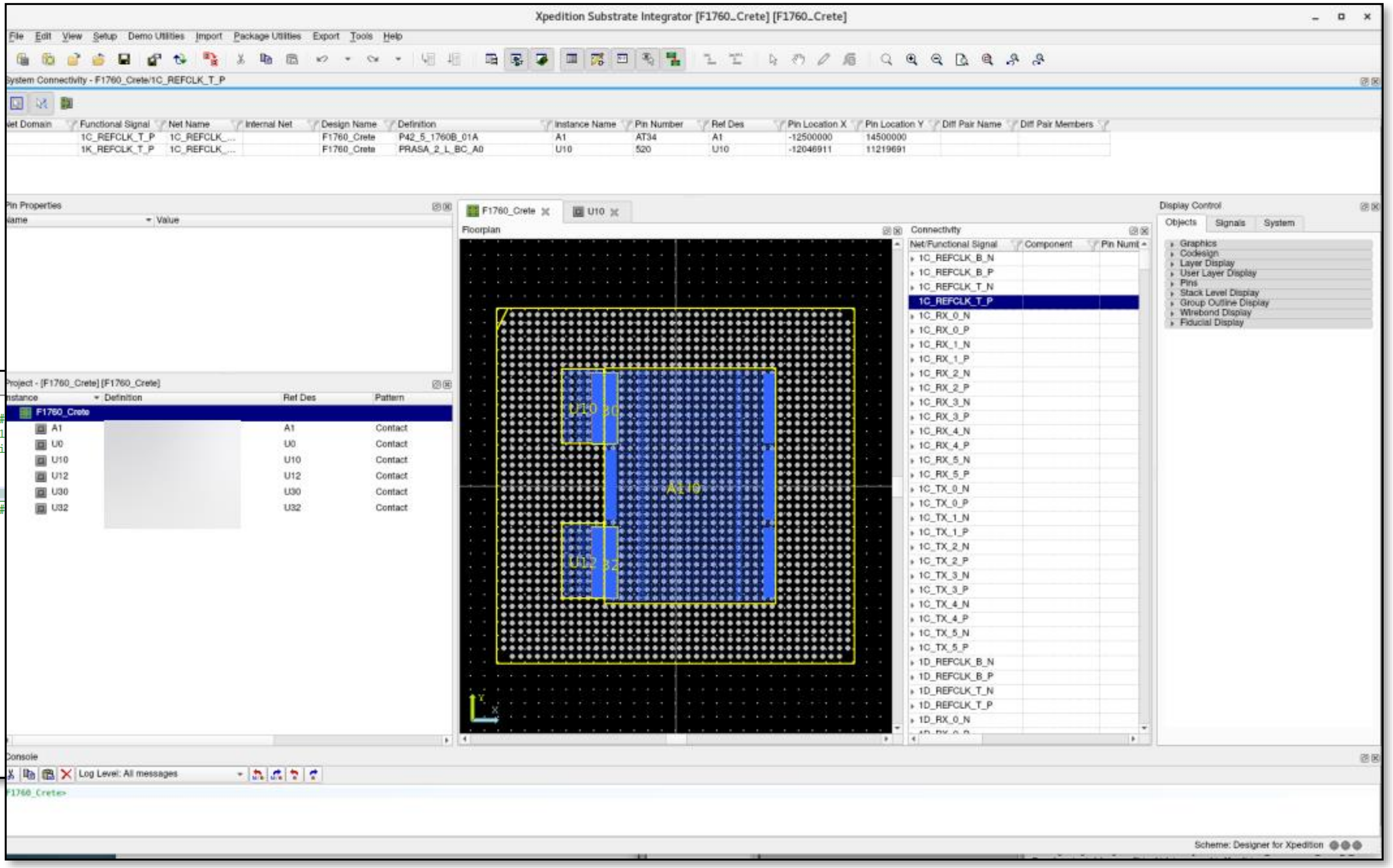
Import Verilog Connectivity into XSI





```
//
// vim: set expandtab tabstop=4 shiftwidth=4:
//
//  F1760_Crete FPGA Netlist
//
//      1x [redacted] (U0)
//      1x [redacted] (A1)
//      2x [redacted] (U30, U32)
//      2x [redacted] (U10, U12)
//

//  Top level module - package netlist connectivity

module F1760_Crete ();

wire \1C_REFCLK_B_N ;
wire \1C_REFCLK_B_P ;
wire \1C_REFCLK_T_N ;
wire \1C_REFCLK_T_P ;
wire \1C_RX_0_N ;
wire \1C_RX_0_P ;
wire \1C_RX_1_N ;
wire \1C_RX_1_P ;
wire \1C_RX_2_N ;
```

```
/*
########################################################
#   Generated by:       Cadence Innovus 18.12-e039_1
#   OS:                 Linux x86_64(Host ID cdc-pki
#   Generated on:       Tue Apr 23 12:09:11 2019
#   Design:             [redacted]
#   Command:            saveNetlist [redacted]
########################################################
*/

module [redacted] (
    AIB_1C_AIB0_CH0_NET,
    AIB_1C_AIB0_CH1_NET,
    AIB_1C_AIB0_CH2_NET,
    AIB_1C_AIB0_CH3_NET,
    AIB_1C_AIB0_CH4_NET,
    AIB_1C_AIB0_CH5_NET,
    AIB_1C_AIB10_CH0_NET,
    AIB_1C_AIB10_CH1_NET,
    AIB_1C_AIB10_CH2_NET,
    AIB_1C_AIB10_CH3_NET,
    AIB_1C_AIB10_CH4_NET,
    AIB_1C_AIB10_CH5_NET,
    AIB_1C_AIB11_CH0_NET,
    AIB_1C_AIB11_CH1_NET,
```



System Connectivity - F1760_Crete/1C_REFCLK_T_P

Net Domain	Functional Signal	Net Name	Internal Net	Design Name	Definition	Instance Name	Pin Number	Ref Des	Pin Location X	Pin Location Y	Diff Pair Name	Diff Pair Members
	1C_REFCLK_T_P	1C_REFCLK_...		F1760_Crete	P42_5_1760B_01A	A1	AT34	A1	-12500000	14500000		
	1K_REFCLK_T_P	1C_REFCLK_...		F1760_Crete	PRASA_2_L_BC_A0	U10	520	U10	-12046911	11219691		

Project - [F1760_Crete] [F1760_Crete]

Instance	Definition	Ref Des	Pattern
F1760_Crete			
A1	[redacted]	A1	Contact
U0	[redacted]	U0	Contact
U10	[redacted]	U10	Contact
U12	[redacted]	U12	Contact
U30	[redacted]	U30	Contact
U32	[redacted]	U32	Contact

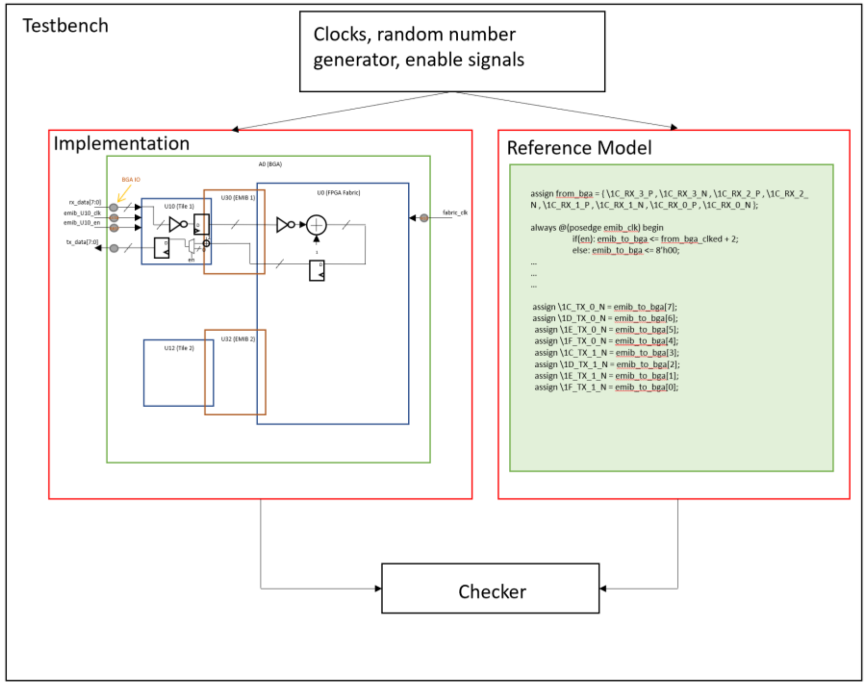
Connectivity

Net/Functional Signal	Component	Pin Numb
1C_REFCLK_B_N		
1C_REFCLK_B_P		
1C_REFCLK_T_N		
1C_REFCLK_T_P		
1C_RX_0_N		
1C_RX_0_P		
1C_RX_1_N		
1C_RX_1_P		
1C_RX_2_N		
1C_RX_2_P		
1C_RX_3_N		
1C_RX_3_P		
1C_RX_4_N		
1C_RX_4_P		
1C_RX_5_N		
1C_RX_5_P		
1C_TX_0_N		
1C_TX_0_P		
1C_TX_1_N		
1C_TX_1_P		
1C_TX_2_N		
1C_TX_2_P		
1C_TX_3_N		
1C_TX_3_P		
1C_TX_4_N		
1C_TX_4_P		
1C_TX_5_N		
1C_TX_5_P		
1D_REFCLK_B_N		
1D_REFCLK_B_P		
1D_REFCLK_T_N		
1D_REFCLK_T_P		
1D_RX_0_N		

```
F1760_Crete>
```

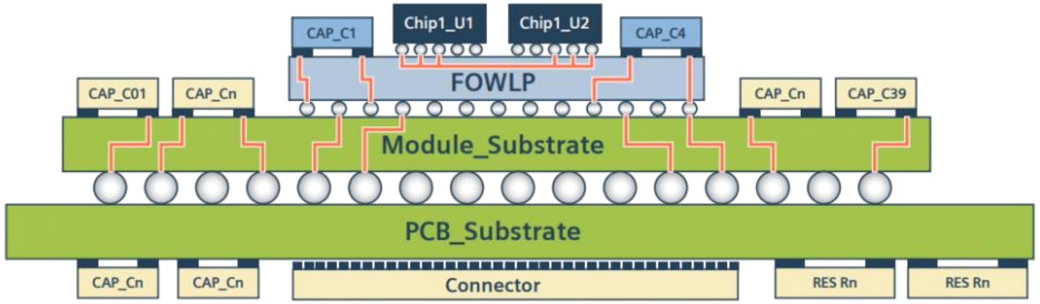
Challenges of Functional Verification

- Functional models for each die
- Models or provisions for discrete components
- Testbench to exercise the functionality and verify correctness
- Expertise to develop and run simulation(s)
- While it is technically possible to perform functional verification of a complex substate assembly, it is often impractical to do so for a number of reasons.



```
assign from_bga = { \1C_RX_3_P , \1C_RX_3_N , \1C_RX_2_P , \1C_RX_2_
N , \1C_RX_1_P , \1C_RX_1_N , \1C_RX_0_P , \1C_RX_0_N };

always @(posedge emib_clk) begin
          if(en): emib_to_bga <= from_bga_clked + 2;
          else: emib_to_bga <= 8'h00;
...
...
...

assign \1C_TX_0_N = emib_to_bga[7];
assign \1D_TX_0_N = emib_to_bga[6];
assign \1E_TX_0_N = emib_to_bga[5];
assign \1F_TX_0_N = emib_to_bga[4];
assign \1C_TX_1_N = emib_to_bga[3];
assign \1D_TX_1_N = emib_to_bga[2];
assign \1E_TX_1_N = emib_to_bga[1];
assign \1F_TX_1_N = emib_to_bga[0];
```

Layout vs Schematic



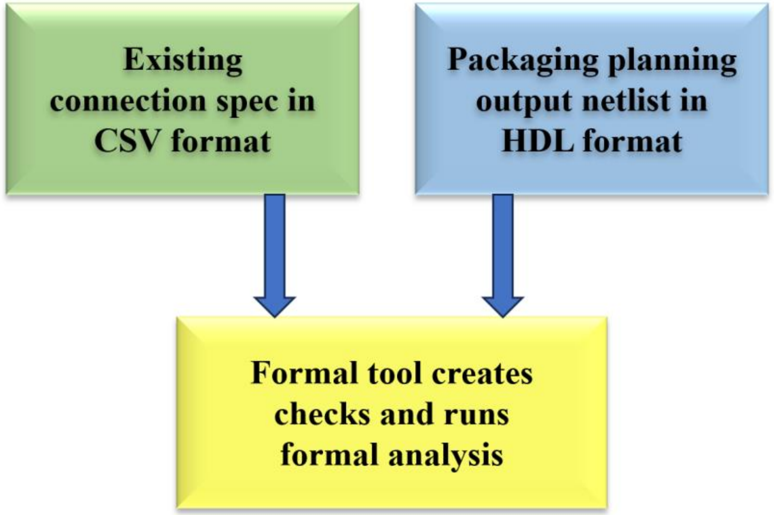
- What about LVS? Can’t LVS verify our system? Yes and No
- LVS at the system assembly level is gaining popularity thanks to tools such as Siemens Calibre 3DSTACK
- LVS can only tell us if the physical implementation of the design matches the source netlist
- LVS can identify shorts and opens and other similar physical issues but it cannot tell us if the design is actually “correct”

Automatic Formal-based Approach

- Benefits of formal connectivity solution for package designs:
  - Easy setup
    - No lengthy simulation testbenches
  - Detect connection errors
    - Specified connections don’t exist.
    - Unintended connections, e.g. short circuits.
  - Ensure correctness of connections
    - Exhaustive analysis and proof capability
  - Complex systems with millions of connections
  - Early detection
  - Compliance
    - Siemens Check Connect is ISO 26262 certified

Two Flows of Verifying Package Connectivity

- Using existing connection spec in CSV

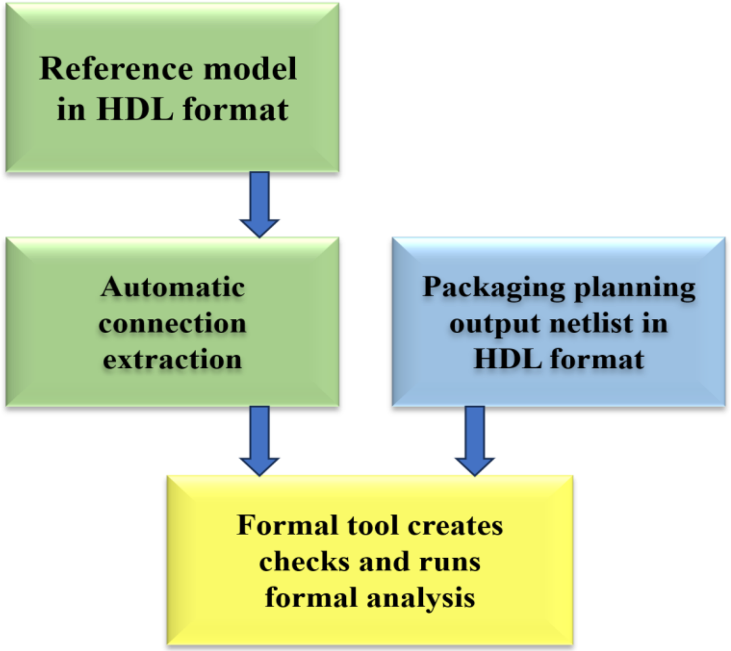


Existing connection spec in CSV format

Packaging planning output netlist in HDL format

Formal tool creates checks and runs formal analysis

- Using reference model to extract spec



Reference model in HDL format

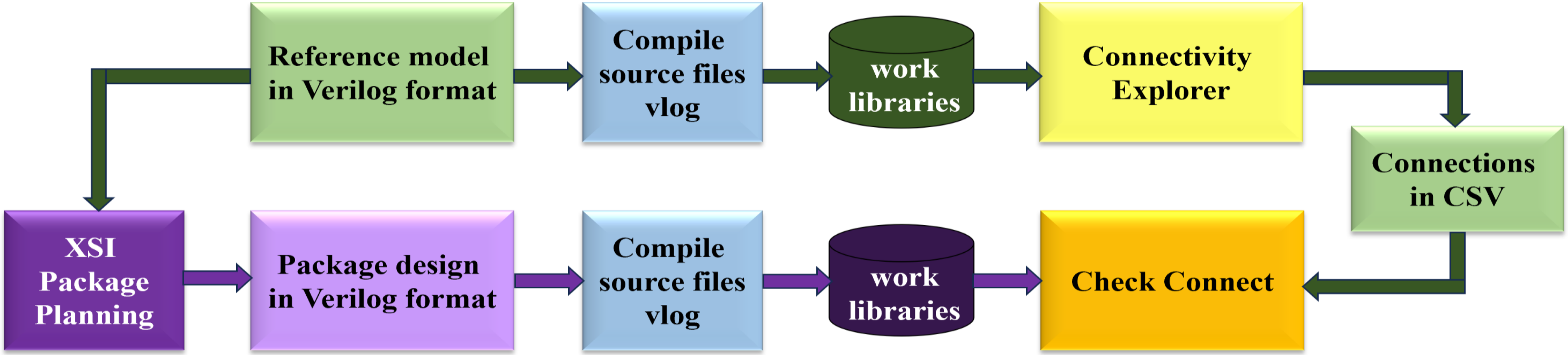
Automatic connection extraction

Packaging planning output netlist in HDL format

Formal tool creates checks and runs formal analysis

The Detail Flow of the Examples

- Package design used tool XSI, and XSI generated Verilog netlist.
- Connectivity Explorer generated connection spec in CSV file.
- Check Connect verified the connections of XSI output against the spec.


Reference model in Verilog format
Compile source files vlog
work libraries
Connectivity Explorer
Connections in CSV
XSI Package Planning
Package design in Verilog format
Compile source files vlog
work libraries
Check Connect

The Script to Verify Package Connectivity

- **Makefile:**

```
#### Compile designs
 Compile_vl:
        vlog -sv -f flist_golden.txt -work lib_golden
        vlog -sv -f flist_package.txt -work lib_package
#### Generate Connectivity Spec
 Generate_conn_csv:
        qconnect_check -explore -od log_csv \
        -infile config.txt \
        -dut F1760_Crete -work ./lib_golden
#### Run Formal Analysis
 Check_connect:
        qverify -od log_cc -do "\
        connectcheck compile -d F1760_Crete -work lib_package;\
        connectcheck load csv log_csv/qconnect_explore_F1760_Crete.csv;\
        connectcheck verify "
```

Understand Formal Verification

- Black box function logic of blocks help formal performance
- If formal finds a violation, it provides waveforms to show the violation
- If formal proves a connection, no stimulus can violate it.
- When all spec items are proven, the package design has all expected connections.
- What if the connectivity specification misses some connections?
  - Check Connect can detect them.

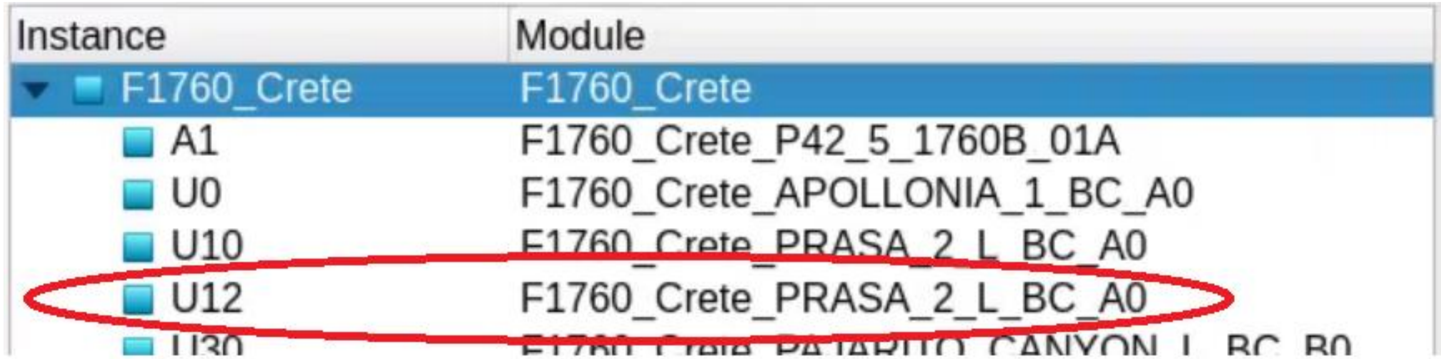
The Results of Two Testcases

- The results of two designs:

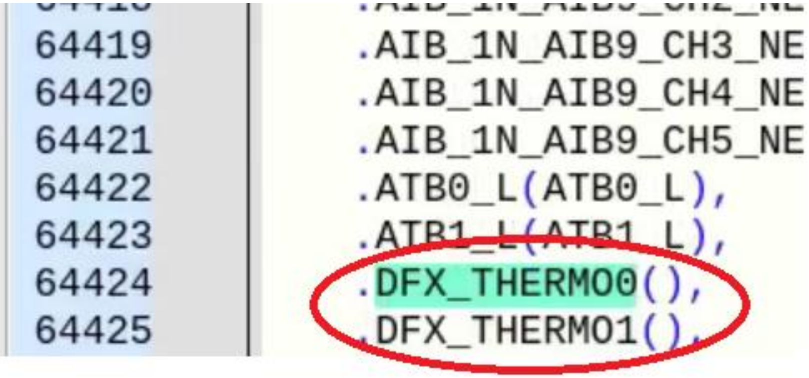
	Time for extracting connection spec	The number of connections	Verification results	Time for verifying the package design
Design 1	15 seconds	21367	All proven	30 seconds
Design 2	35 seconds	43440	All proven	56 seconds

Missing Connections Found

- The tool found two missing connections.
  - The designer has purposely left two thermal sensor pins unconnected
  - The designer was impressed they were identified.



Instance	Module
F1760_Crete	F1760_Crete
A1	F1760_Crete_P42_5_1760B_01A
U0	F1760_Crete_APOLLONIA_1_BC_A0
U10	F1760_Crete_PRASA_2_L_BC_A0
U12	F1760_Crete_PRASA_2_L_BC_A0
U30	F1760_Crete_PAJARITO_CANYON_L_BC_B0



```
.AIB_1N_AIB9_CH3_NE
.AIB_1N_AIB9_CH4_NE
.AIB_1N_AIB9_CH5_NE
.ATB0_L(ATB0_L),
.ATB1_L(ATB1_L),
.DFX_THERMO0(),
.DFX_THERMO1(),
```

Conclusion

- Verifying package connectivity is challenging.
- Automatic formal-based approach is the solution.
- The setup for running formal tool is simple and reusable.
- Formal can verify large number of connections for big package designs.
- Using formal method right after package planning can improve the quality of physical implementation dramatically and shorten the time to market.

Questions?

2024
DESIGN AND VERIFICATION™
DVCON
CONFERENCE AND EXHIBITION
UNITED STATES
SAN JOSE, CA, USA
MARCH 4-7, 2024

Thank you!

accellera
SYSTEMS INITIATIVE