2014

Isolation Verification Tool (IVT) Software User Manual

John D. Corbett
Xilinx, Inc

XILINX®

Revision History

Version	Date	Notes
1.0	9/30/2010	Initial release.
2.0	2/18/2011	Updated for IVT 6.38.
2.1	2/22/2011	Incorporated feedback from Ted Ennis on document organization and wording.
2.2	3/3/2011	Added table of SCC-related application notes.
2.3	6/14/2011	Updated description of UCF area range checking, Spartan-6 DSP checking, and error messages. Made various minor improvements.
2.4	8/4/2011	Updated for IVT 6.48. Updated error messages.
3.0	12/9/2011	Updated for IVT 7.08.
3.1	7/18/2013	Updated for IVT 7.40.
3.2	12/11/2013	Updated for IVT 7.41. No functional change.
7.43	12/22/2014	Updated for IVT 7.43. No functional change.

Table of Contents

Table of Figures

Table of Tables

1 Introduction

The Xilinx® Isolation Verification Tool (IVT) software is part of the Xilinx Isolation Design Flow, (formerly Single Chip Crypto (SCC) technology). The Xilinx IDF enables unprecedented levels of security (1)(2), reliability, and performance to be achieved with Xilinx FPGAs. IVT aids trained evaluators in verifying that a suitably-prepared Xilinx FPGA design meets strict isolation requirements needed for high-reliability applications. This document describes the usage of the IVT application.

Cryptographic systems are a good starting point for developing technology for high reliability systems because they have the most extreme reliability requirements of any industry. Although cryptographic systems were the initial inspiration for the development of the Xilinx Isolation Design Flow, the benefits it offers apply to all domains that have extreme safety or reliability requirements. Such domains include,

- Aviation (2), (3), (4), (5)
- Industrial controls (6),
- Financial systems,
- Communications infrastructure,
- Medical devices (7),
- Scientific instruments, and
- Space systems (8).

The reason for this is that the most powerful technique for creating systems that are more reliable than their components is the elimination of single points of failure through careful design of redundant subsystems. Two subsystems are redundant if they perform a common function and no single component failure can result in both malfunctioning silently.

IVT has two operational modes. In UCF-mode, IVT evaluates the pin assignments and floorplanning constraints. In NCD-mode, IVT evaluates routing resources and logic placement, as well. IVT produces two output files: a plain text report, and a floorplan diagram.

This document presents the rationale for IVT, the tests IVT performs, the input required, and the output generated.

2 FPGA Architecture

A Field Programmable Gate Array (FPGA) is a chip that contains logic elements and routing both of which are controlled by configuration memory. Logic elements range in complexity from simple combinatorial logic functions up to complete embedded processors. Logic elements and routing are arrayed in a grid of tiles. The structure of an FPGA is extremely regular. Each tile contains one of a small variety of VLSI modules dedicated to logic or routing. Logic tiles include:

- *Configurable logic blocks* (CLBs), each containing a small amount of programmable logic and memory,
- Input/output pin circuitry (IOBs),
- Clocking (CMTs), and

- Other specialized circuitry, including Block RAMs (BRAMs), digital signal processing (DSPs), embedded processors, configuration access, etc.

A typical Xilinx FPGA might have thousands of tiles, but only dozens of tile types. CLBs and most other tile that occupy a single grid cell include a single interconnect block. Some tiles such as Block RAMs and DSPs occupy more than one grid cell and contain one interconnect block per grid cell. Some grid cells contain more than one tile. The I/O buffers are two per grid cell. The pair of I/O buffers in a grid cell share a single interconnect block.

The majority of user signals are routed using the global switch matrix (GSM). The GSM is composed of many instances of a single interconnect block design. The interconnect block offers a large set of programmable routing resources that enable any tile to talk to any other tile. Other kinds of signals including clocking and configuration are discussed in section The Xilinx Isolation Strategy below.

An FPGA is configured for a particular purpose by loading configuration memory with a particular bitstream. The bitstream specifies the exact function of each and every tile in the device; logic tiles are configured to perform a specific function and interconnect blocks within them are configured to provide the required routing between the tiles.

3 Reliability through Isolation

The most powerful technique for creating systems that are more reliable than their components is *the elimination of single points of failure through careful design of redundant subsystems*. Two subsystems are redundant if they perform a common function and no single component failure can result in both malfunctioning silently. To put it another way, redundant subsystems are isolated from one another.

For cryptographic systems, safeguarding sensitive information is paramount. It is assumed that equipment will occasionally fail. However, the equipment is rigorously designed to fail without exposing sensitive information. For other domains, other failure modes may be more desirable, such as to degrade to a reduced functionality mode.

Although the inspiration for the development of the Xilinx Isolation Design Flow was to provide a better way to develop cryptographic equipment for Information Assurance applications, the technology developed is fundamentally a domain-independent method for developing highly reliable systems using Xilinx FPGAs.

4 The Xilinx Isolation Strategy

Analysis of isolation requires understanding of:

- Device architecture,
- Device layout,
- Organization of the design into isolated functions,
- Floorplan of the design,
- Signal paths between isolated functions,
- Potential unintended signal paths, and
- Failure modes of all of the above.

An FPGA design may be partitioned into sections that that will occupy specific geometric regions of the chip. The IVT software displays and analyzes the floorplan, isolated functions, user signal paths, and unintended signal paths.

There are two approaches to showing that isolation requirements are met for user design signals:

- Analyze every path into and out of the isolated function; or
- Show that there is a barrier that blocks all unintended communication paths into and out of the isolated function.

The former approach is referred to as *fault-cost-based isolation analysis*. Cost refers to a conservative estimate of the number of faults required to enable a signal path that is not part of the design. IVT performs a cost-limited depth first search to efficiently identify potential isolation failures.

The barrier approach is referred to as *tile-based isolation analysis*. Analysis shows that a barrier one tile wide is sufficient to block user signals for all the Xilinx devices considered to date. The tiles separating isolated functions are referred to collectively as the *fence*.

A combination of fault-cost-based isolation analysis for the user design signals and tile-based isolation analysis for the I/O tiles was used for Virtex-4. Isolation analysis for Virtex-5 and subsequent devices is exclusively tile-based.

5 The FPGA Development Flow with Isolation Analysis

To facilitate isolation analysis, the usual FPGA flow has a new set of constraints to control routing and additional floorplanning requirements. First, the design must be manually floorplanned. Second, constraints must be applied to isolated regions of the floorplan to constraint routing to follow strict rules. Finally, IVT must be used to demonstrate that the design is correctly implemented.

Figure 1 below shows where isolation analysis fits in to the usual FPGA development flow. IVT is useful at two points:

- During floorplanning (upper right blue boxes), IVT helps to avoid costly circuit board layout mistakes and helps document the floorplan, a key part of the isolation approach of the design. The inputs during the floorplanning stage are the floorplan and isolation groups.

- Once the design is complete (lower right green boxes), IVT is used to help prove that all relevant security and reliability requirements have been met. The input for the final verification is the complete placed and routed NCD file.

The flowchart notation is as follows:

- Boxes represent processes,
- Parallelograms represent data,
- Trapezoids (quadrilaterals with one pair of parallel sides) represent manual input,
- Shapes with curved bottoms represent output,
- Arrows represent information flow, and
- Color is used only for grouping and emphasis.

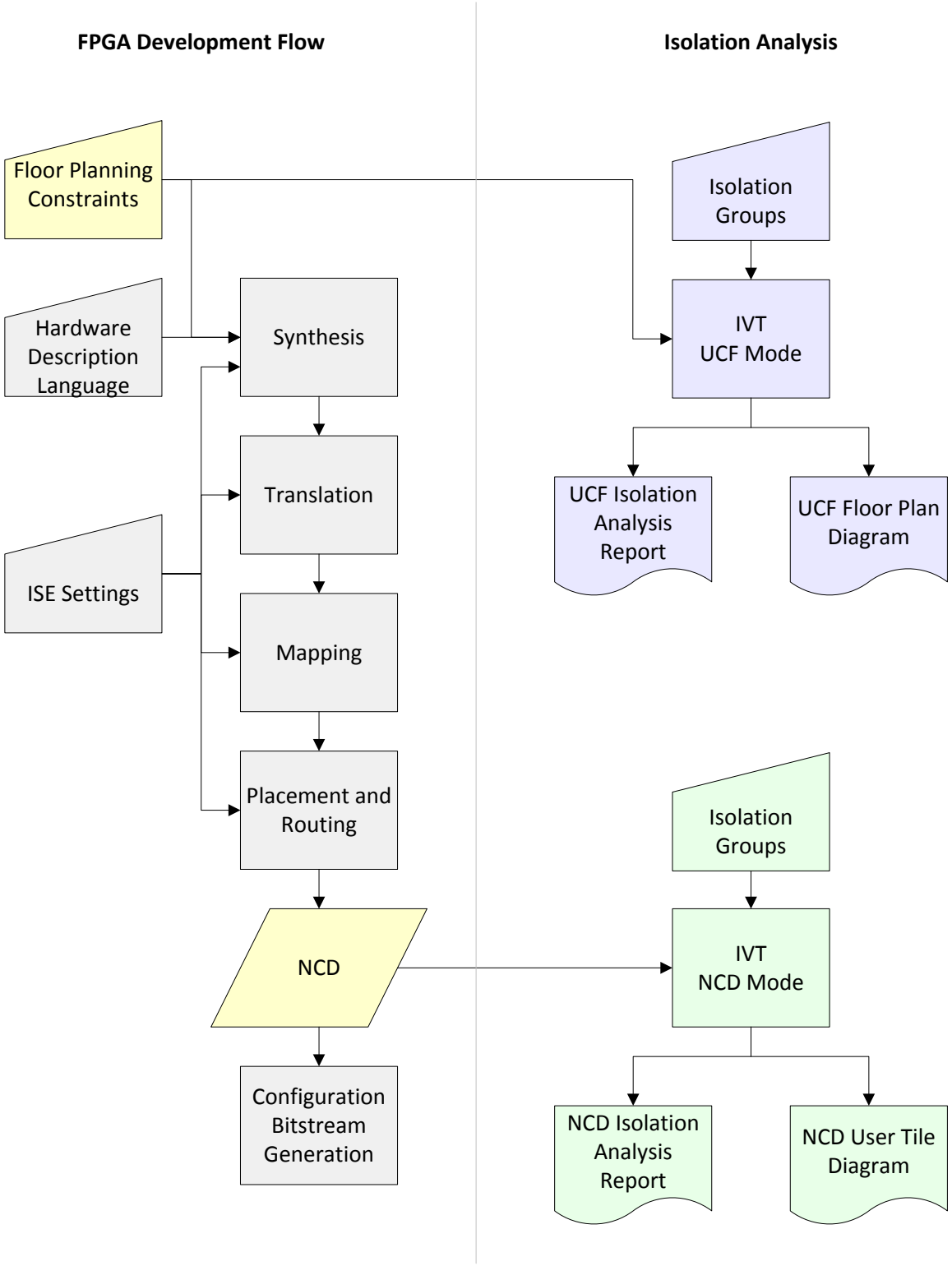

Figure 1: The FPGA development flow with isolation analysis

6 The IVT Software Overview

IVT requires one of the following operating systems:

- Red Hat Enterprise Workstation 5 (32-bit and 64-bit)
- Red Hat Enterprise Workstation 6 (32-bit and 64-bit)
- SUSE Linux Enterprise 11 (32-bit and 64-bit)
- Windows 7 Professional (32-bit and 64-bit)
- Windows Server 2008 (64-bit)
- Windows XP Professional (32-bit and 64-bit)

The directory containing the Xilinx executables and dynamic link libraries must be listed in the PATH environment variable for IVT to find them.

IVT is a report generator. It is a batch application with a command line- and file-based user interface. It takes design files as input and produces two files as output: a plain text report, and a floorplan diagram in SVG format.

IVT has four non-trivial modes:

- UCF Mode,
- Cost-based NCD Mode,
- Tile-based NCD Mode, and
- NCD Cost Debug Mode.

The IVT UCF Mode analyzes the layout and pin locations defined in the User Constraint File.

The IVT NCD Mode analyzes the complete placed and routed design. For Virtex-4, the analysis is fault-cost-based routing search. For Virtex-5 and beyond, the analysis is a tile-based scan for a fence between isolated functions.

The NCD Cost Debug Mode is used to debug Virtex-4 fault-cost-based isolation analysis. In addition to the usual NCD arguments, IVT accepts a signal path. IVT reports the fault cost of the path. This is used by the IVT developers to check that information added to the cost function is correct.

IVT has two trivial modes:

- Usage Message Mode, and
- License Agreement Mode.

7 Fault-Cost Based Isolation Analysis (Virtex-4 only)

For Virtex-4, IVT performs isolation analysis using a combination of fault-cost-based routing search and tile-based analysis. Internal signal isolation is verified with routing search; I/O isolation is verified with tile-based checks.

The purpose of fault-cost-based isolation analysis is to show that no signal path exists or could arise through a given number of failures that would compromise data separation or independence. Recall that fault cost is a conservative estimate of the number of faults required to enable a signal path that is not part of the design. For Virtex-4, IVT performs an exhaustive search to identify potential isolation failures.

A *fault* is a hypothetical component failure that could result in a short-circuit allowing a signal to propagate to wires in the FPGA inconsistent with the FPGA application designer's intentions. In general, the only active routing resources in a design are the ones needed to implement the design. Enabling unnecessary routing is considered a bug for many reasons, including wasted power, decreased reliability, and increased difficulty of troubleshooting.

IVT does not examine routes that might arise from breaking or disconnecting existing routes. This is safe because in order for such a failure to enable a signal to pass that could not pass before, there would also need to be an enabled route for that signal to pass and the existence of such a route would be a failure in and of itself. So IVT searches for routes that could conceivably be created by shorting a small number of gates.

The signals in the user design are called nets. Nets are composed of nodes. A node is the smallest programmable unit of routing. Each node has the potential for being connected to several other nodes. IVT uses all nets in the design as starting points for the search procedure. Each net is either global or associated with one isolation group. (A net associated with more than one isolation group would be an error.) The connection between a pair of nodes is called an arc. An arc always connects exactly two nodes. Many nodes have a branching structure enabling them to connect to multiple adjacent nodes.

7.1 Cost Function

For any pair of nodes that can be connected but are not connected in the user design, IVT uses a fault cost function to determine a conservative lower bound on number of failures required to short the nodes together. Not all adjacent pairs of nodes have been analyzed. For pairs that have not been analyzed, the fault cost is conservatively assumed to be zero, i.e. the nodes are considered shorted together before any faults are assumed.

If too few non-zero costs were assigned, IVT might discover failing paths composed on many zero cost connections between wires. This condition was observed during development and led to occasional elaboration on the initial cost function.

7.2 Categorizing Nets as Isolated or Global

IVT supports two ways of specifying groups of isolated nets:

- Via partial NCD files and
- Via block hierarchy.

The partial NCD file method was developed first. In this scheme, input was supplied as several partial NCD files, one per isolated module, and one complete NCD file with all the modules combined. Each partial NCD file included all common elements as well. This scheme led to the terminology "combined" nets for nets in the combined design and "shared" nets for nets in more than one isolation group. Shared nets are considered global.

Since the isolated regions can include global routing, IVT must examine all the isolated regions to determine which nets are unique to an isolated module and which are

shared. Furthermore, we expect the combined design to contain every net from all modules; however, nets with distinct names are occasionally merged. This does not appear to happen for nets in the general routing and therefore IVT need only recognize these nets and filter them.

IVT categorizes nets as follows. First, every net in the combined design is labeled as “combined”. This defines the universe of discourse (with one exception noted above). Next IVT processes each module assigning an isolation group to the nets. For each net in the module, if the previous label is “combined”, the label is updated to the name of the modules isolation group, e.g. “Red”, unless there is a reason to categorize the net based on its usage. If the previous label is the name of another isolation group, then IVT attempts to determine if there is a valid reason the net need not be confined to an isolated module. If IVT cannot find a reason, it reports the net for user analysis. In the verbose reporting format, IVT lists the first reason it finds for categorizing a net.

No matter why a net does not need to be isolated, there is no point in performing routing analysis on it. Further routing analysis would only confirm what has already been established and so, the net is labeled “ignored”.

Any net that appears in a module, but not in the combined design and cannot be safely ignored is reported as an error. This error could only arise if multiple NCD files are used.

Originally, isolation groups were defined by multiple partial NCD files. For isolation groups specified by block name, the nets are obtained from the combined design. For isolation groups defined by partial NCD files, all nets under the named block and all nets that do not belong to any other user block (‘shared’ or global user nets) are included. This approach of adding the shared nets and later removing them was implemented to minimize the changes to the output reports and thus minimize the risk that mistakes in the new method of defining isolation groups would go unnoticed.

IVT provides a list of uncategorized user global nets, however when IVT is unable to put a shared net into one of the permitted categories, it is the designer's responsibility to verify that the shared net is acceptable. This analysis requires understanding of the design and thus is outside the scope of IVT.

7.3 Net Categories Related to Isolation

IVT reports all global signals and classifies global signals to aid designers and independent evaluators in analyzing these signals.

The reporting of global signals starts with the list of shared signals described above in section Categorizing Nets. This list is further categorized as shown in Table 1 below. For each of the categories below there is a section or subsection in the NCD mode report.

Table 1: Net categories

Net Category Headings
Uncategorized User Global Nets
Categorized Nets
Nets Driven By Ground
Nets Driven By Vcc
Nets Driven By Constants or Unused Blocks
Nets Without General Routing (Including Clock Nets)
Nets Driven by Global Clock Sources (BUFG, DCM, PLL, and PMCD)
Nets Inside Trusted bus macros
Nets Attached to Trusted bus macros
Nets Driven by Regional Clock Buffers
Non-user Digital Clock Manager Nets
Non-user Nets

Uncategorized User Global Nets – lists nets (signals) in the design that are above the isolated modules in the design hierarchy and may connect isolated functions. All such nets must be examined for their impact on the data separation and independence requirements of the system. Ideally, the only nets listed in this section would be nets specifically intended to connect isolated functions. In practice some global clock signals appear here as well.

Nets Driven By Ground, Nets Driven By Vcc, Nets Driven By Constants or Unused Blocks – lists nets for which it is possible to determine that the signal value based on the nature of the driver of the signal.

Nets Without General Routing (Including Clock Nets) – lists nets either internal to a logic block (such as a CLB or an IOB) or part of a global clock net, therefore contained within an isolated region.

Nets Driven by Global Clock Sources (BUFG, DCM, PLL, and PMCD) – lists nets driven by clock-related resources. For most systems, providing independent clock sources to isolated functions is not necessary. In Information Assurance applications, clock-related failure typically would not offer appreciable risk of compromise. For fail-stop systems, the probability that a clock failure will result in silent malfunction are usually acceptable. When this risk is not acceptable, a multiple clock solution or external monitoring may be required.

Nets Inside Trusted Bus Macros, Nets Attached to Trusted Bus Macros – lists nets related to Trusted Bus Macros.

Nets Driven by Regional Clock Buffers – lists nets driven by BUFR elements.

Non-user Digital Clock Manager Nets – lists nets associated with the Digital Clock Manager (DCM) negative bias temperature instability (NBTI) mitigation.

Non-user Nets – lists nets that do not correspond directly to nets in the FPGA design source files. This category contains additional DCM nets that are not internally marked as NBTI-related.

8 Tile-Based Isolation Analysis

Tile-based isolation analysis is based on the idea of a barrier between isolated functions. An FPGA design may be partitioned into isolated functions that that will occupy specific geometric regions of the chip. The region in between the isolated functions forms a barrier call the fence.

Tiles comprising the fence consist interconnect blocks and logic blocks. Analysis of the interconnect blocks accounts for the majority of the user signals between tiles. It is important that the fence tiles be configured in a way that blocks signals and moreover that continue to block signals even in the presence of a small number of faults.

Analysis shows that an interconnect block that does not contain isolated signals will act as a barrier and prevent isolated signals of other blocks from passing through it. We call such an interconnect block *unused*. Note, an interconnect block is considered unused even if it contains global signals.

Analysis also shows that a logic block will act as a barrier to isolated signals (direct connects) if it is configured in its default (non-functioning) state.

Thus tiles containing a combination of interconnect blocks and logic blocks can be used as a barrier. The unused tiles separating isolated functions are referred to collectively as the fence. Tiles that can be used in the fence are called user tiles to distinguish them from internal tiles that lack the proper structure, are not visible to the user, or simply have not been sufficiently analyzed. The list of site types in Table 2 below have been analyzed and shown to be suitable constituents of the fence.

There are two ways of associating design elements with isolation groups:

- Through the logical hierarchy of blocks in the user design, and
- Through the geometry of area range constraints.

Whenever there are two distinct ways to define something inconsistencies are a possibility. The block hierarchy is the definitive statement of the logical organization of the design into isolated functions. The design at this level is not something that can be analyzed automatically by IVT; it is a matter for human experts. The mapping of that logical hierarchy onto the physical resources in a particular geometric arrangement on the FPGA is something that IVT can visualize and check. Figure 2 below illustrates the mapping of the logical block hierarchy (tree at left) to the geometric placement of that functionality in the FPGA (floorplan at right). The design is divided into three isolated

functions, colored red, yellow, and black. The white gaps in the floorplan represent the fence providing the physical isolation between the three functions.

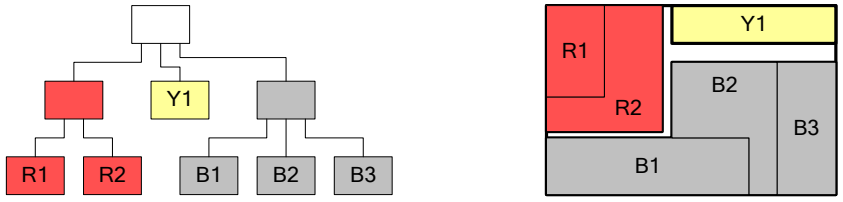

Figure 2: Mapping logical design hierarchy to geometric placement in the FPGA

To put it another way, IVT helps answer the question, "Does the physical organization of the design in the FPGA provide adequate isolation of the given logical structure?"

Figure 2 above illustrates a successful placement of isolated functions into the FPGA. However, there are a variety of ways things could go wrong and hence a variety of conditions for IVT to check. In the next several sections below, IVT engineering requirements related to security and reliability are discussed.

8.1 Area Range Constraints and "The Fence"

The implementation tools are required to isolate blocks according to constraints applied to the blocks. Included among these constraints are area ranges. Surprisingly, the implementation tools can guarantee a valid fence without strictly honoring the area range constraints. Thus the fence obtained is not necessarily exactly the fence specified. This extra liberty reduces the pressure on the router; however it complicates the interpretation of IVT results.

Earlier versions of tile-based isolation analysis treated any excursion beyond the area range constraints defining isolation regions to be a violation. IVT was repaired to treat only concrete instances of insufficient separation as violations.

8.2 Visualizing the Fence

IVT creates a diagram of the area range constraints and tiles that make up the floorplan of the design. Tile occupancy is only known for the placed and routed design (NCD Tile Mode). Area range constraints are available both from the User Constraints File (UCF Mode) and also from the NCD file for the completed design.

Area range constraints associate rectangles specified in site coordinates with area groups. The IVT command line parameters associate area groups with isolation groups. A site is a subdivision of a tile. For some tiles, there is only one site; for others there are multiple sites. For example a Virtex-5 CLB tile is divided into two sites called slices. (Other site types are not called slices, only CLBs.)

Each site type has its own coordinate system. To generate a floorplan, IVT converts site coordinates into a universal tile coordinate system. In the device model interconnect blocks are not represented as nested inside user tiles. They are typically drawn to the left of the logic block they are associated with in the schematics.

The floorplan IVT generates shows interconnect blocks, area range rectangles, and the entire chip boundary, but does not show other tile or site types. Information related to

the logic block of a tile is displayed in the interconnect block to the left of the logic block.

The diagrams IVT generates are in Scalable Vector Graphics (SVG) format (9). SVG is a human-readable text-based format using XML notation. Like HTML, SVG can be rendered by modern web browser software. SVG can also be edited in various drawing programs such as Microsoft Visio, Adobe Illustrator, and Inkscape. SVG can also be embedded in Microsoft Office documents.

In UCF Mode, IVT displays area range constraints as color-coded rectangles and the device boundary but no tile occupancy information since it is not available. IVT identifies areas where there is insufficient separation between isolation groups by assuming that every tile within the area ranges contains isolated routing and logic.

In NCD Tile Mode, IVT adds to the UCF mode diagram tile information. Tiles are color-coded according to their block hierarchy. Isolation violations are highlighted by drawing an × through the square where the violation occurs.

Figure 3 below is actual IVT output with some annotations and small labels removed. The figure is a basic user tile diagram for a design with a black isolated function (left) and a red isolated function (right). The thin vertical gap between the black and red functions is the fence isolating the red function from the black function. In this design, red and black do not communicate. Each isolated function is composed of several overlapping area ranges. By allowing the ranges to overlap, the designer can work with fewer ranges.

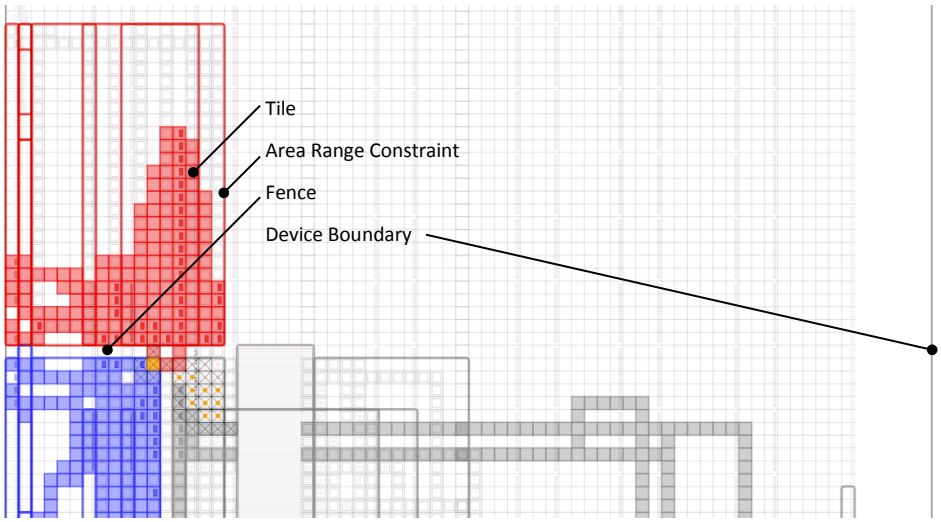

Figure 3: The user tile diagram

8.3 Checking the Floorplan Boundary Implementation

IVT checks that no tile contains nets from multiple isolated functions.

IVT checks that no pair of adjacent tiles contains nets from distinct isolation groups. E.g. A tile containing a “red” net cannot be adjacent to a tile containing a “blue” net and a single tile cannot contain nets from distinct isolation groups.

IVT checks that no wire (node) connecting a pair of isolation groups can contain a Programmable Interconnect Point (PIP) in any tile needed as a fence tile. That is to say,

the router must choose routes that provide no possibility of branching within tiles immediately between tiles containing isolated logic or routing. Nodes come in many shapes and sizes. Nodes often span multiple tiles and often have more than two points at which another node can be connected to them. Nodes that span multiple tiles also often pass straight through tiles without any switches for connecting to other nodes.

Since area range constraints are not strictly honored by the implementation tools, IVT must determine available fence tiles by their utilization. A fence tile is one that is all of the following:

- The type is one of the “User Tile” types,
- No isolated routing exists in the tile,
- No isolated logic exists in the tile, and
- No (global) route in the tile uses a routing resource that could be configured to branch or terminate in the tile, except for clocking routes.

This definition of a fence tile requires IVT to search for a fence between isolated regions. When IVT discovers a breach in the fence, it propagates that breach information through all tiles that are not isolated from the breach, since the breach might be represented in many equivalent ways.

8.4 The Tile Checking Algorithm

Although it is conceptually simple, there are some practical complexities involved in checking isolation based on tile occupancy and collecting the information needed to produce an informative diagram. The idea is to mark every interconnect block with all the nets that use it. Note that every net is either global or associated with a single isolation group based on the logical block hierarchy of the design.

The tile checking process proceeds as follows:

Each net is decomposed into routing resources each of which occupies exactly one tile. A data structure for each tile records the fact that the net from a particular isolation group or global net occupies the tile.

When a routing resource is in its default state, the switches that potentially connect it to other routing resources or logic elements are open.

All tiles configured to implement logical functions are recorded. Later this information will be filtered to obtain interconnect blocks in the fence adjacent to logic blocks that are configured. This makes it possible to mark direct connections in the fence that could conceivably bypass the fence. This is only ever expected to occur in purpose-built test designs.

For reporting purposes, all tiles outside all area ranges associated with isolation groups are documented in the output report.

Every tile is in one of two states:

- Contains isolated logic or routing (isolated tiles)
- Contains no isolated logic or routing (fence tiles)

Since inter-region signals (IRSs) are not permitted to use routing resources that have programmable interconnect points (PIPs) in fence tiles, there is no reasonable

possibility that a single error will cause data from an IRS to be connected to any global routing resource including another IRS.

All occupied (configured) tiles in the fence are collected. A tile is considered a fence tile if it is adjacent to two or more tiles containing isolated logic or routing from distinct isolation groups.

Colors are assigned to tile occupancy group sets. Singleton sets are assigned colors in alphabetical order by isolation group name until the sequence of predefined colors has been exhausted. Additional groups are assigned the “overflow” color.

Isolation errors involving adjacent tiles are collected for any tile that contains isolated routing and has a neighbor containing isolated routing from a different isolation group.

The color sequence is illustrated in Table 1 below.

Table 2: Color table for the user tile diagram

IVT Color Name	Color Code	Color Swatch
White[0]	#FFFFFF	Blank
Orange[1]	#FFA102	Violation
Red[2]	#FF0000	Unused
Green[3]	#08D308	Unused
Gray	#888888	1st Module
Red	#E10000	2nd Module
Blue	#3030F0	3rd Module
Yellow	#FFDA44	4th Module
Blue Green	#008888	5th Module
Violet	#5B008E	6th Module
Light Gray	#BEBEBE	7th Module
Pink	#FE7777	⋮
Light Blue	#8080FE	⋮
Light Yellow	#FEEDA8	
Light Orange	#FEC565	
Cyan	#00ECEC	
Light Green	#40FC40	
Purple	#9A00F2	
Dark Gray	#685555	
Dark Red	#A50000	
Dark Blue	#000097	
Dark Yellow	#C4A426	
Dark Blue Green	#005959	
Dark Orange	#BB7600	
Dark Green	#009E00	⋮
Dark Violet[4]	#420068	22nd, 23rd,…

Notes:

0. Blank. Configured to be inactive.

1. Unused.
2. Violation. Occupied by isolated routing or logic from multiple isolation groups.
3. Unused.
4. This color is used for the 22nd and subsequent isolation groups. Although the colors are selected to be easily distinguished, the effectiveness of this approach rapidly diminishes after the first dozen or so. While the use of transparency reduces the possibility that valuable information will be occluded, it also decreases the color contrast and thus decreases the effectiveness of color coding. Viewing conditions are also significant. Projectors tend to have lower contrast than monitors do. For large designs, careful attention to labels and the report text may be required.

The colors above are displayed at full opacity. In the diagrams generated by IVT the colors are translucent so that when one shape is drawn over another, the occluded area is darker. The area ranges are also drawn with rounded corners to decrease ambiguity.

Figure 4 below shows how multiple configurations can give rise to an ambiguous appearance. A small area range will not be entirely occluded by a larger area range drawn over it. However, since rectangles are likely to share common edges, ambiguities still arise. The drawings show that rounding the corners of the rectangles eliminates some but not all ambiguities. The report text completes the picture.

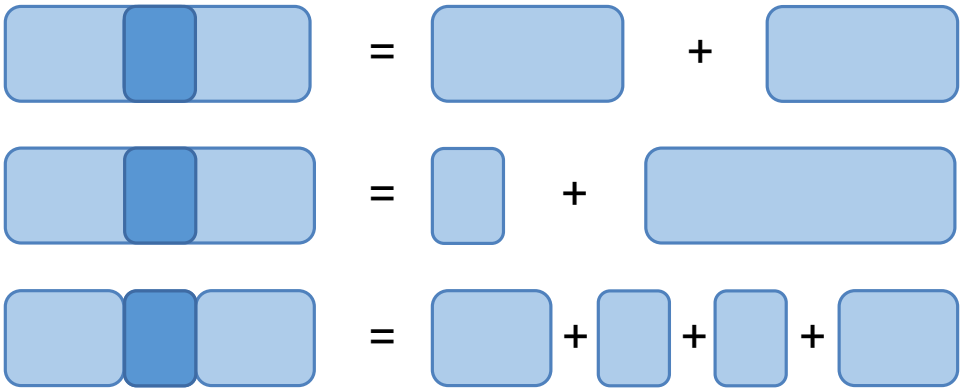

Figure 4: Transparency and rounded corners reduce ambiguity

8.5 Kinds of Tiles

Most of the site types that have an associated interconnect block are suitable fence tiles. Table 3 below shows the correspondence between sites names used to delineate floorplan boundaries in the user design and generic user tile types. Site types not listed below are not considered valid fence tiles. That is to say, all pairs of tiles containing isolated logic or routing from distinct isolation groups, shall be separated by at least one tile of the one of the site types below. For Virtex-4 and Virtex-5, this condition is necessary and sufficient to achieve isolation.

Table 3: The Mapping of Virtex-4 and Virtex-5 sites to user tile types

Site	User Tile
DCM_ADV_X*n*Y*m*	Clock Management Tile (CMT)
DCM_X*n*Y*m*	Clock Management Tile (CMT)
DSP48_X*n*Y*m*	Digital Signal Processor (DSP)
EMAC_X*n*Y*m*	Ethernet Media Access Controller (EMAC)
GT11_X*n*Y*m*	Multi-Gigabit Transceiver (MGT)
GTPA1_DUAL_X*n*Y*m*	3.75 Gb/s Gigabit Transceiver (GTP)
GTP_DUAL_X*n*Y*m*	3.75 Gb/s Gigabit Transceiver (GTP)
GTXE1_X*n*Y*m*	6.5 Gb/s Gigabit Transceiver (GTX)
GTX_DUAL_X*n*Y*m*	6.5 Gb/s Gigabit Transceiver (GTX)
IOB_X*n*Y*m*	Input/Output Buffer (IOB)
PCIE_X*n*Y*m*	Peripheral Component Interconnect Express (PCIE)
PLL_ADV_X*n*Y*m*	Clock Management Tile (CMT)
PPC405_ADV_X*n*Y*m*	IBM® PowerPC® 405 (PPC)
PPC440_X*n*Y*m*	IBM® PowerPC® 440 (PPC)
RAMB16_X*n*Y*m*	Block RAM (BRAM)
RAMB18_X*n*Y*m*	Block RAM (BRAM)
RAMB36_X*n*Y*m*	Block RAM (BRAM)
RAMB8_X*n*Y*m*	Block RAM (BRAM)
SLICE_X*n*Y*m*	Configurable Logic Block (CLB)

For Spartan-6, there are additional rules for the DCM and DSP sites.

Table 4: The Mapping of Spartan-6 sites to user tile types

Site	User Tile
DCM_CLKGEN_X*n*Y*m*	Clock Management Tile (CMT)
DCM_SP_X*n*Y*m*	Clock Management Tile (CMT)
DSP48A1_X*n*Y*m*	Digital Signal Processor (DSP)
MCB_X*n*Y*m*	Memory Controller Block
IOB_X*n*Y*m*	Input/Output Buffer (IOB)
IOI_X*n*Y*m*	Input/Output Buffer (IOI)
PLL_ADV_X*n*Y*m*	Clock Management Tile (CMT)
RAMB16BWER_X*n*Y*m*	Block RAM (BRAM)
RAMB18_X*n*Y*m*	Block RAM (BRAM)
RAMB36_X*n*Y*m*	Block RAM (BRAM)
RAMB8BWER_X*n*Y*m*	Block RAM (BRAM)
SLICE_X*n*Y*m*	Configurable Logic Block (CLB)

8.6 The User Tile Diagram

Since tile information is inherently geometric, it is helpful to have a visual representation of the design. The diagrams are extremely helpful in the development process and are expected to be helpful for customer documentation and for independent evaluation.

The Scalable Vector Graphics format was chosen because:

- It is simple to generate;
- It is vector-based, and therefore not tied to a particular image resolution (hence 'scalable');
- It can be rendered in an ordinary web browser;
- It is a text-based format and can easily be inspected in a text editor; and
- It can easily be modified in various drawing programs for presentations and publications.

The notation used in the user tile diagram consists of small squares corresponding to interconnect blocks, larger rectangles corresponding to area ranges, and an outline representing the entire device. User tiles usually correspond directly to interconnect blocks, but larger tiles such as Block RAMs may span several squares and smaller tiles such as I/O buffers may share a tile.

Each interconnect block that is used in the design is labeled with its coordinates. If the tile contains one or more global nets, the tile is marked with an inset square. If the isolated nets in the tile come from one group, the tile is color coded for that isolation group.

Several interconnect blocks are shown in Figure 5 below. Each square represents one interconnect block. The coordinates are cascaded to reduce overlap. The coordinates

are an abbreviation for INT_X*a*Y*b*. For example “64, 163” corresponds to “INT_X64Y163” in the IVT report.

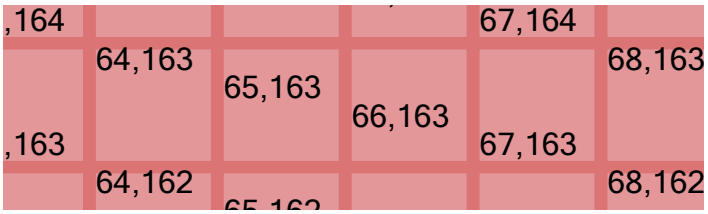

Figure 5: Interconnect blocks occupied by red isolated routing

Global routing in a tile is indicated by an inset square as shown in Figure 6 below. It is possible for a tile to contain both global routing and isolated routing. In this case, the color of the square indicates the isolation group of the isolated routing.

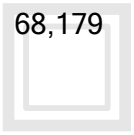

Figure 6: A tile containing only global routing

Global routing in a tile is indicated by an inset square as shown in Figure 7 below. It is possible for a tile to contain both global routing and isolated routing. In this case, the color of the square indicates the isolation group of the isolated routing and the rectangle in the right half of the tile indicates a logic function is configured.

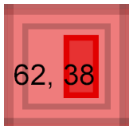

Figure 7: A tile containing isolated logic and isolated routing as well as global routing

Interconnect blocks that either violate isolation rules or are adjacent to logic tiles (not shown) that violate isolation rules are marked with a large × and a thin black outline as shown in Figure 8 below.

Figure 8: An interconnect block containing an isolation violation and global routing

Figure 9 below shows an invalid tile tainted by multiple isolation groups. The orange diamond indicates the taints. The inset outline indicates the presence of global routing. The small square in the center indicates that the tile is invalid as a fence because it uses a PIP to route a signal through it. The large × indicates an isolation violation.

Figure 9: An invalid tile tainted by multiple isolation groups

Area range constraints enclose one or more tiles and are depicted as transparent rectangles with rounded corners. In UCF mode, there is no information about the contents of specific tiles displayed. Only ranges for sites that are allowed as fence tiles are displayed. Figure 9 below shows a complicated Virtex-5 floorplan (IVT UCF mode) composed of nine isolation groups. An area range for a given site type may span other site types. For example, Block RAM ranges in the figure are many columns wide, whereas Block RAM columns in the hardware are one tile wide and interspersed with many columns of CLB slices, DSP columns, clock columns, etc. It would also have been possible to define the Block RAM area ranges to include only the Block RAM sites. Labels have been enlarged and moved for clarity.

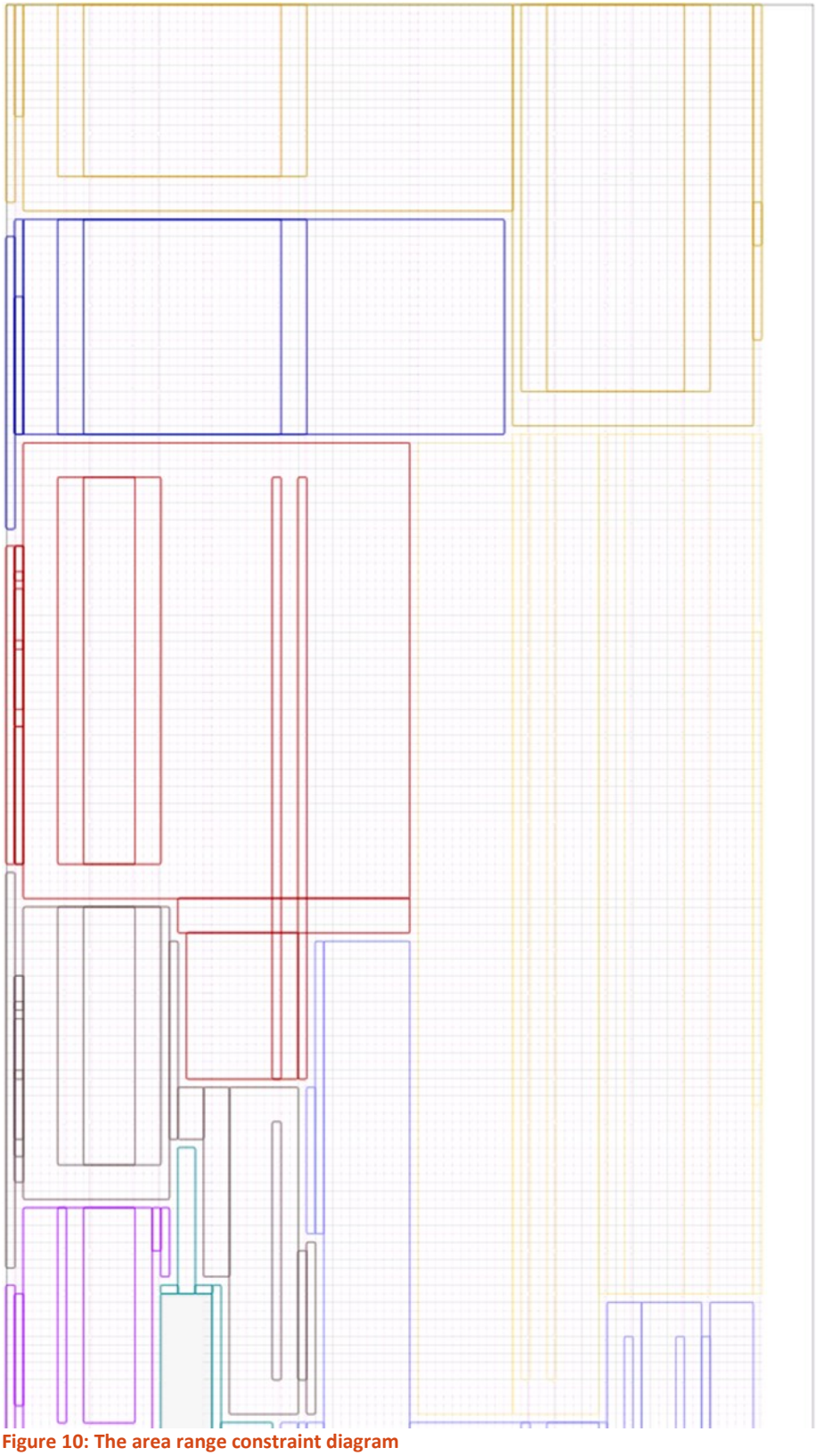

Figure 10: The area range constraint diagram

The effects of missing isolation constraints can be observed below in Figure 10. Note the resources from the blue region on the left intrude into the gray region on the right.

In tiles where resources from both regions are present, the tile is colored bright orange in addition to being marked with an ×.

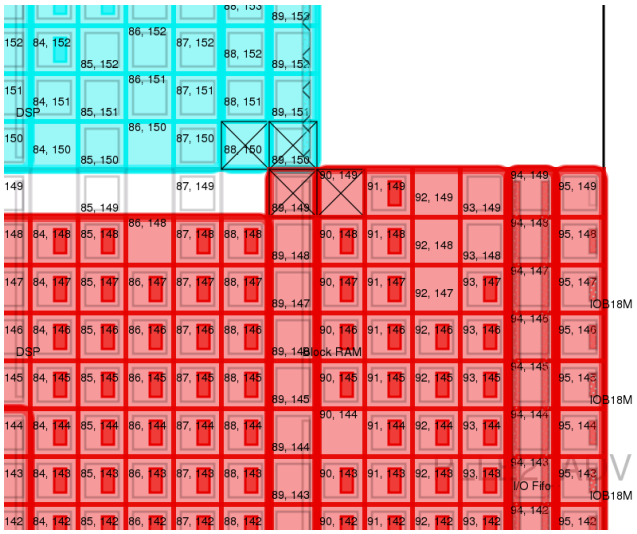

Figure 11: A portion of a design prepared with incorrect isolation constraints

Figure 11 below shows tiles containing TBM routing are outlined in black. The overall shapes of the TBMs are not simple rectangular strips from source to sync. The shapes reflect the layout of routing resources used to construct the TBMs. The shapes of these TBMs prevent them from being placed against the right edge of the blue region above. When the TBM is placed against the edge of a region antennas extending into the fence and possibly extending into an isolated region other than the red and blue regions the trusted bus macros are intended to connect.

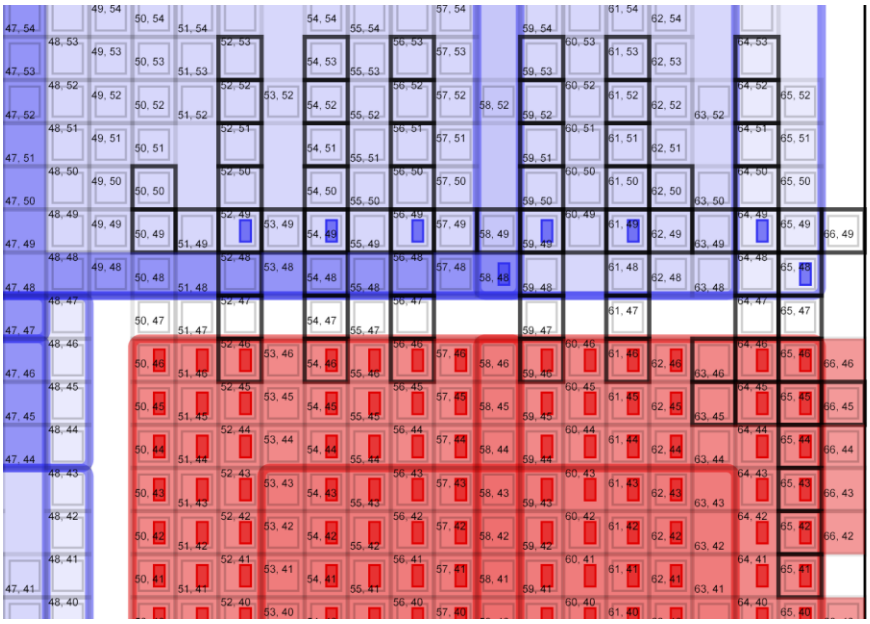

Figure 12: A portion of a Virtex-4 design containing Trusted Bus Macros

8.7 Fence Rules

The grid cells in the User Tile Diagram represent interconnect tiles and related logic. Most sites contain an interconnect block and a logic block. Most sites occupy a single grid cell, however some sites occupy more than one grid cell.

A site used as a fence tile must be entirely free of isolated logic, however interconnect tiles of the site may be used provided they are not needed to separate other sites containing isolated logic or routing.

This rule applies to all site types, including sites that occupy more than one grid cell, such as Block RAM, DSP, and PPC. Some site types have additional restrictions described below.

In the figures below, interconnect blocks are shown as small rectangles labeled with a capital I. Logic blocks are shown as taller rectangles labeled with an abbreviated version of their site type, RAMB or DSP. Resources drawn with a white background are unused. The IVT output does not depict individual sites this way.

Figure 13 below shows a horizontal fence passing through a Block RAM column at two different vertical coordinates. No matter where the fence passes through the Block RAM, that entire Block RAM logic block must be unused. However most of the interconnect blocks can be used.

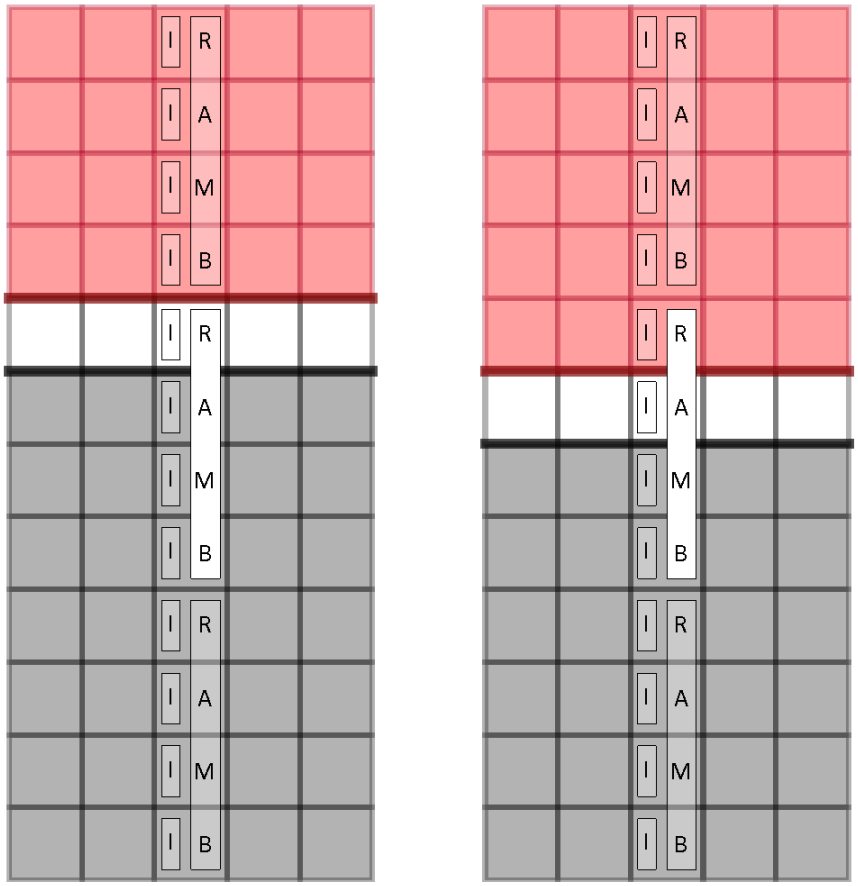

Figure 13: Horizontal fences passing through a Block RAM column (RAMB8, RAMB16, etc.)

Figure 14 below shows a fence that has a vertical section in a Block RAM column. The Block RAM sites that the fence passes through cannot be used, even though only a portion of the bottom Block RAM is needed for the fence.

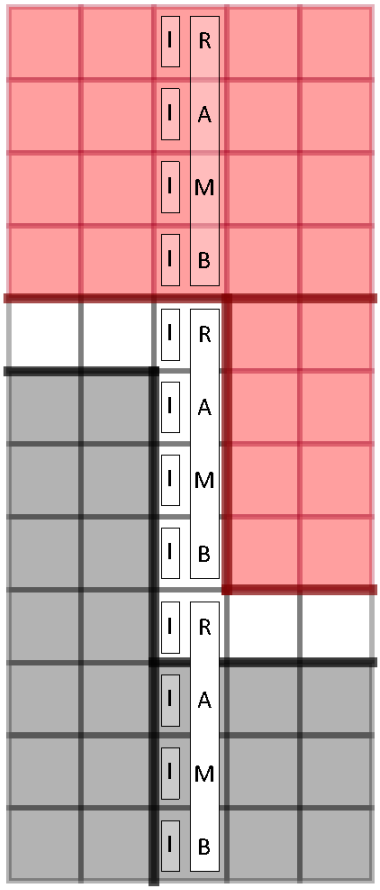

Figure 14: A fence with a vertical section in a Block RAM column (RAMB8, RAMB16, etc.)

Figure 15 below shows DSP sites in a vertical fence. Due to the design of the DSP tile carry chain, two adjacent DSP tiles in a column are required to isolate one region from another. In the figure, the top DSP site in the red region is shown as used, therefore the two below it must be unused. However, if the top site in the red region was not used, then the bottom site in the black region could be used.

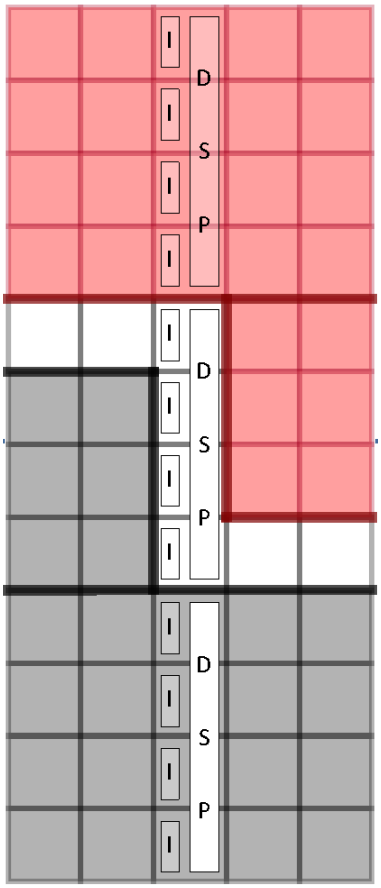

Figure 15: A fence with a vertical section in a DSP column (DSP48, DSP48A1, etc.)

The DCM and PLL tiles cannot be used as a vertical fence, however, an unused DCM or PLL can appear in a horizontal fence.

8.8 The User Tile Diagram File Structure

The diagrams generated by IVT use XML syntax and Scalable Vector Graphics elements. A complete description of XML and SVG is beyond the scope of this document, but briefly, the output is a fully nested expression of structured data. For the most part, the data elements correspond to geometric primitives, such as rectangles, lines, and text strings. All have associated geometric coordinates and style attributes such as color or typeface. Graphical elements are rendered in the order they occur in the file from back to front. That is to say, later elements are drawn over earlier elements.

A tile can be described in two lines of text, a square and a label. For example,

```
<rect style="fill:#e10000;fill-opacity:0.4;stroke:black;stroke-
  opacity:.1;stroke-width:.1" x="54" y="48" width="1" height="1"
  rx="0" ry="0" />
```

and later,

```
<text x="54.05" y="48.66" stroke="none" font-size="0.25" font-
  family="Helvetica">54,151</text>
```


The astute reader will notice that the coordinates listed in the label do not match the geometric coordinates of the diagram. The reason for this is that only interconnect blocks are shown. Logic tiles and internal tiles in between them are not shown.

> It is possible to quickly modify the diagrams generated by IVT using the search and replace feature of a text editor. It is possible to quickly remove unwanted labels, change the opacity of the colors, or change the style of the diagram to match related documentation.

8.9 Checking I/O-Related Isolation

IVT performs several checks related to I/O isolation. The same conditions are checked in UCF mode and NCD mode.

8.9.1 I/O Buffers

Two isolated designs may not use adjacent I/O buffers. Adjacent even/odd pairs of I/O buffers share an interconnect block. A difference of 1 in Y-coordinate is not sufficient to guarantee the requirement is met. Two I/O buffers from distinct isolation groups must be separated by at least 4 if the lesser Y coordinate is even, and separated by at least 3 if the lesser Y coordinate is odd. In the Figure 15 below, IOB_X1Y20 is considered adjacent to IOB_X1Y21, IOB_X1Y22, and IOB_X1Y23. Likewise, IOB_X1Y21 is adjacent to IOB_X1Y20, IOB_X1Y22, and IOB_X1Y23. (None of the figures in this paper and none of the diagrams generated by IVT are associated with physical dimensions.)

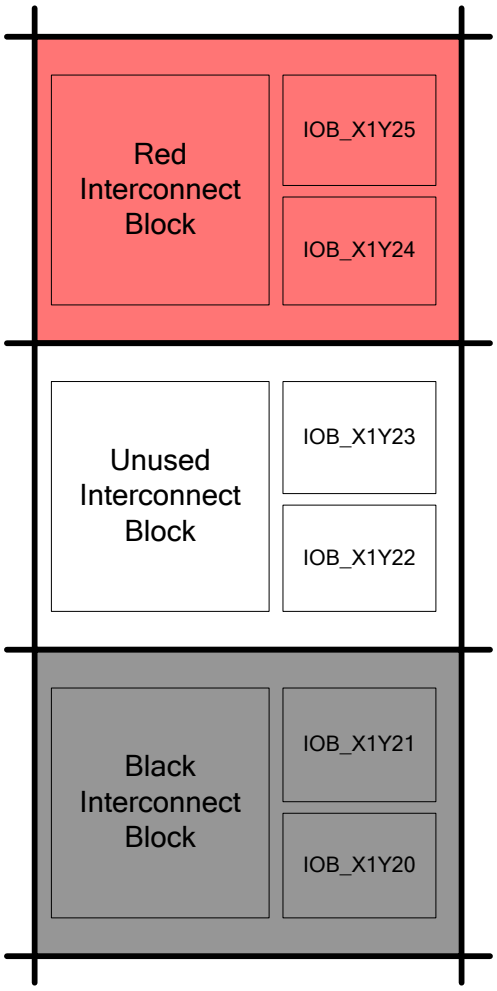

Figure 16: Pairs of I/O buffers share an interconnect block

8.9.1 Package Pins

No pair of package pins from distinct isolation groups shall be adjacent. Note that pins (or bumps) on the package may not be laid out like the die. One reason for this is that not every I/O buffer is bonded out on every package.

8.9.2 I/O Banks

Every I/O buffer is a member of a single I/O bank. To check this, IVT sweeps through all used I/O buffers. The first isolated I/O buffer encountered in each bank is stored as a representative of the isolation group that occupies that bank. If subsequent I/O buffers

have differing isolation groups than the first representative, an example violation is reported.

9 The IVT Report

The text output report always begins with provenance information including the input arguments and the IVT version. The structure of the report is indicated by indentation. The report ends with the elapsed time, unless a fatal error occurs.

The intermediate sections of the report vary based on the mode, command line parameters, and presence or absence of particular features in the design.

9.1 The UCF-Mode Report

A typical UCF mode report contains the following sections.

Provenance – includes the date, IVT version, ISE version used for the run, ISE version against which IVT was compiled, the location of the ISE installation, the command line, the current working directory, the output report location, and the part/package.

Area Range Constraints – for each isolation group, lists the corresponding area groups and associated site ranges used to define the floorplan of the design.

Package Pins, I/O Buffers, and I/O Banks – the pin assignments of the design are presented with coordinates, I/O banks, isolation groups, and the net names.

Pin Isolation Summary – lists the number of isolation violations for die pins (I/O buffers), package pins, and I/O banks.

Area Group Separation – lists the distance in tiles between area ranges from distinct isolation groups.

Area Fault Summary – lists the number of area ranges from distinct isolation groups that are not separated by an adequate fence.

Isolation Verification Summary – lists the total number of constraints violated, reports completion, and reports the elapsed time to perform the analysis and generate the report.

9.2 The NCD-Mode Report

A typical NCD mode report contains the following sections.

Provenance – includes the date, IVT version, ISE version used for the run, ISE version against which IVT was compiled, the location of the ISE installation, the command line, the current working directory, the output report location, the part/package, and the name of the combined design.

Input Designs – lists the isolation groups and associated design blocks or partial NCD files.

Uncategorized User Global Nets – lists nets (signals) in the design that are above the isolated modules in the design hierarchy and may connect isolated functions. All such nets must be examined for their impact on the data separation and independence requirements of the system. Ideally, the only nets listed in this section would be nets

specifically intended to connect isolated functions. In practice some global clock signals appear here as well.

In the report, uncategorized global nets are said to be “found in multiple isolation groups”. This is an artifact of the original implementation of IVT in which multiple partial NCD files were used to specify isolation groups. Global resources are duplicated in all the partial NCD files.

Categorized Nets – has many possible subsections corresponding to various categories of nets that for one reason or another are benign with respect to isolation. Examples include constants, global clocks, trusted bus macros, and clocking inserted automatically to mitigate negative-bias temperature instabilities. See section Net Categories Related to Isolation above.

Trusted Bus Macros – lists all the instances of Trusted Bus Macros in the design and the nets connected to them.

Area Range Constraints – for each isolation group, lists the corresponding area groups and associated site ranges used to define the floorplan of the design.

Net Fault-Cost Violations (Failing Paths) – for Virtex-4 designs, lists pairs of putatively isolated nets that cannot be shown to be sufficiently isolated with fault-cost-based routing search. Examples of low cost paths between isolated nets are listed. Several customer designs have produced isolation violations, however in all cases, the violations turned out to be due to insufficient information in the cost function, not an actual vulnerability.

Tiles with Net Content Violations – lists the contents of all tiles that contain isolated logic or routing from more than one isolation group.

Tiles in the Fence Containing Nets – lists the contents of tiles that are outside all the isolated area ranges. This list is advisory. It is permissible for inter-region signals and clocking to exist in the fence. Akin to the list of “Uncategorized Global Nets,” nets in tiles that are outside of all isolation groups must be vetted for their impact on data separation and independence requirements. This section provides low level details of the specific nodes and wire segments used to implement routing in the fence.

Tiles in the Fence Containing Programming – lists tiles in the fence that are associated with comps and therefore are not entirely unused. All such tiles must be examined for their impact on the data separation and independence requirements of the system.

Tiles in the Fence Containing Used PIPs – lists tiles containing nodes that have the potential to connect to other nodes. Recall PIP stands for Programmable Interconnect Point. For example, suppose a certain type of node spans five horizontal tiles called A, B, C, D, and E. Further suppose that this node has PIPs in tiles A, C, and E. It is allowed to use this node to connect two isolated regions provided the only tiles in the fence are B or D.

Tiles with Net Adjacency Violations – lists all pairs of tiles are adjacent and contain isolated logic or routing resources from distinct isolation groups. In other words, pairs of tiles that should be separated by a fence tile.

Package Pins, I/O Buffers, and I/O Banks – the pin assignments of the design are presented with coordinates, I/O banks, isolation groups, and the net names.

I/O Buffer Isolation Violations – lists pairs of I/O buffers from distinct isolation groups that are adjacent on the die.

I/O Bank Violations – lists examples of pins from distinct isolation groups that are members of a single I/O bank.

Package Pin Isolation Violations – lists pairs of pins from distinct isolation groups that are adjacent on the package.

Isolation Verification Summary – lists the total number of constraints violated by category, reports completion, and reports the elapsed time to perform the analysis and generate the report. For each section of the report containing violations, there is a line in the summary with a tally of the violations in that section.

An example of an NCD summary is given below.

```
Section 15 - Isolation Verification Summary

  Net Isolation
      Net Isolation Violations Found: 248
      Net Isolation Violations Reported: 2
      Routes Examined: 43381

  Tile Adjacency
      Net Adjacency Violations: 71
      Logic Adjacency Violations: 0

  Tile Content
      Net Content Violations: 382
      Logic Content Violations: 0

  Inter-region Signals
      Inter-region Net PIP Violations: 0
      Inter-region Load Violations: 2

  Special Fence Rules
      DSP Violations: 0

  I/O Isolation
      I/O Buffer Isolation Violations: 0
      Package Pin Isolation Violations: 3
      Bank Isolation Violations: 0

  NCD Isolation Verification Summary
      Total isolation violations: 458
      Unrouted nets: 0
```

The Net Isolation section refers to Virtex-4 fault-cost-based isolation analysis. Tile Adjacency refers to violations arising from isolated logic or nets lacking a sufficient fence from other isolated logic or nets. Tile Content refers to lack of isolation within a single tile. Inter-region Signals refers to signals that are not isolated but connect modules that are isolated. I/O Isolation refers to adjacency restrictions for both I/O buffers and pins on the package as well as I/O bank occupancy. The final summary gives the total number of violations and the number of unrouted nets. Although

unrouted nets are not isolation violations, their presence almost always indicates serious problems in the FPGA design.

9.3 UCF Mode: Verifying the Pin Choices and Isolation Region Boundaries

The User Constraint File (UCF) specifies the correspondence between signals within the design and physical pins used to transmit those signals outside the FPGA. It also defines geometric regions of the FPGA to contain various logical modules of the design. The UCF is completed early in the design cycle to allow work to begin on the circuit board that will eventually contain the FPGA. Typically, the final placed and routed NCD file is not available when the circuit board design commences.

IVT verifies several aspects of isolated design with only a UCF file and a PIG file to avoid costly rework of the circuit board.

Command line argument syntax for UCF checking is:

```
-device device
-package package
-group isolation_group area_group
[-spacing clbs]
[-pig pin_isolation_groups[.pig]]
[-output output[.rpt]]
[-nopin]
[-verbose]
[-f argument_file]
constraints[.ucf]
```

For example,

```
% ivt -device xc4vlx60 -package ff1152 -group Red AG_Red
  -group Black AG_Black -pig pin_groups.pig crypto.ucf
```

Due to its length, the command example is presented on multiple lines, but it would be entered as a single line at an interactive command prompt. When arguments are supplied in a file however, they may be spread across multiple lines and the file may contain comments. Any text that follows a number sign (#) is considered a comment and ignored by IVT.

9.4 NCD Mode: Verifying the Placed and Routed NCD Design

The NCD files of a design contain the complete placement and routing information, including names of blocks, pins, and nets. IVT performs verification of pins and routing in the NCD files.

Command line argument syntax for NCD checking is:

```
-group isolation_group partial_reconfiguration_module[.ncd]
[-faults number]
[-output output[.rpt]]
[-nopin]
[-verbose]
[-f argument_file]
combined_design[.ncd]
```

For example,

```
% ivt -group Red red.ncd -group Black black.ncd combined.ncd
```

Since the NCD files contain complete information about the design, there is no need to specify the device, the package, nor the mapping from pins to area groups.

> The modules specified need not be blocks directly below the top level of the design hierarchy; however, the blocks must be disjoint.

10 Application Notes

Table 5 below lists documentation available through the Xilinx Single Chip Crypto Lounge website (registration required).

Table 5: Xilinx Single Chip Crypto development application notes

FPGA Family	Xilinx ISE Version	Application Note	Development Flow	Comments
Spartan-6	12.2	XAPP1104	Trusted Routing	No patch required.
Spartan-6	12.3–12.4	XAPP1104	Trusted Routing	Patch required for 12.3 and 12.4.
Virtex-5	11.4–11.5	XAPP1135/XAPP1134	Trusted Routing	XAPP1135 – reference design XAPP1134 – SCC design rules
Virtex-5	12.1–12.4	XAPP1105/XAPP1134	Trusted Routing	XAPP1105 – reference design XAPP1134 – SCC design rules No patches required
Virtex-4	9.2	XAPP984	DEPRECATED FLOW	No longer supported; refer to XAPP1134 instead
Virtex-4	11.1–11.3	N/A Use XAPP1135 for reference but use	Trusted Bus Macros	Refer to XAPP984 for general Virtex-4 SCC flow

		Trusted Bus Macros		
Virtex-4	11.4– 11.5	N/A Use XAPP1135 for reference but use Trusted Bus Macros	Trusted Bus Macros	Refer to XAPP984 for general Virtex-4 SCC flow

11 References

1. **Corbett, John D.** *The Isolation Design Flow for Fault-Tolerant Systems.* [White Paper] San Jose : Xilinx, Inc., 2012.
2. *FPGA-Based Single Chip Cryptographic Solution (U).* **McLean, Mark; Moore, Jason.** 2006. Proceeding of the SDR 06 Technical Conference and Product Exposition. http://www.sdrforum.org/pages/sdr06/sdr06_papers/1.3/1.3-04.pdf.
3. **RTCA, Inc.** *DO-254, Design Assurance Guidance for Airborne Electronic Hardware (AEH).* SC-180. 2000. DO-254. RTCA DO-254.
4. **International Electrotechnical Commission.** *IEC 61508, Functional safety of electrical/electronic/programmable electronic safety-related.* 2010. ISBN 978-2-88910-524-3.
5. **Le Mauff, Joël and Elliott, Jeff.** *Meeting DO-254 and ED-80 Guidelines When Using Xilinx FPGAs.* Xilinx, Inc. 2009. White Paper. WP332.
6. **ISO.** *ISO 14971:2007(E), Medical devices — Application of risk management to medical devices.* 2007. ISO 14971:2007(E).
7. **Hu, Ching and Zain, Suhail.** *NSEU Mitigation in Avionics Applications.* Xilinx, Inc. 2010. Application Note. XAPP1073.
8. **Bridgford, Brendan, Charmichael, Carl and Tseng, Chen Wei.** *Single-Event Upset Mitigation Selection Guide.* Xilinx, Inc. 2008. Application Note. XAPP987.
9. Scalable Vector Graphics “Scalable Vector Graphics” (SVG) 1.1 Specification. [Online] 1.1, 2009. http://www.w3.org/TR/2003/REC-SVG11-20030114/.

12 Definitions

Many of the terms below are used in a specialized sense in this document. The glossary below may be helpful to readers who are not already well versed in Xilinx terminology related to FGPA architecture and configuration or to a lesser extent, search algorithms.

arc – a representation of a connection between two wires in the software device model of the FPGA hardware. Whether or not an arc is used by a design is usually determined by the state of a corresponding configuration memory cell however there are some arcs present in the software device model that do not correspond to hardware structures. An arc usually corresponds to a pass transistor; however, there are arcs in the device model that correspond to multiple transistors, and arcs that do not correspond to any transistors.

area group – a design implementation constraint that enables partitioning of an FPGA design into physical regions for mapping, packing, placement, and routing.

area group range – a list of rectangular regions identifying a subset of the device resources in an FPGA. It is defined as a list of resource pairs. Each pair defines two opposite corners of an included rectangular region.

depth first search – an algorithm (systematic method) for exploring a network of connected entities. The IVT searches starting from every putatively isolated net in an FGPA design for potential routes to other isolated nets. The search explores one route as far as possible before backtracking and exploring a different route. This process is continued exhaustively. The search is limited to routes of less than a given cost.

device model – the data and data structures that describe the potential programming of a specific model of an FPGA. The device model specifies the capacity of the device and all of the features that can be configured by an FPGA design. The device model is highly abstracted from the FPGA hardware schematics. Only aspects of the FPGA hardware that are programmable are represented. Although it is theoretically possible that a programmable feature of an FPGA might not be represented in the device model (for example, if testing show the feature to be unreliable), in practice this is not done because it would make it impossible to perform the tests that would validate or invalidate the feature.

fault-cost function – a function that provides a lower bound on the number of faults required to connect one wire with another assuming those wires were not previously connected.

fence tile – a site free of isolated routing or logic used to separate two or more tiles containing logic or routing from distinct isolation groups.

graph – in mathematics, a structure consisting of a set of vertices, and a set of edges. An edge is an ordered pair of vertices in the graph. In the software device model, the terms node and arc are synonyms for vertex and edge, respectively.

In this context, the term node is not to be confused with a “technology node” that relates to the physical dimensions of features in the layout. To make matters more confusing, although it is customary in mathematics to depict nodes as circles or boxes and arcs as lines connecting nodes, this representation is inverted with respect to

electrical schematics. In a circuit schematic, lines between boxes represent wire and boxes represent gates or lower level schematics. The correspondence between circuit schematics and graphs is counter-intuitive: graph nodes correspond to circuit wires; and switches correspond to arcs. A switch that is closed indicates the presence of a connection. In graph terminology, the presence of a connection is an arc (or edge).

interconnect block – a common hard IP block providing a programmable switching matrix connecting programmable logic elements of virtually all types to routing resources. In the end user documentation, the interconnect blocks are collectively referred to as the Global Switching Matrix. Usually individual interconnect blocks are not referred to in the documentation, but might occasionally be referred to as switch boxes. There are approximately 200 wires in an interconnect block that are available for routing a design in Virtex-4 and Virtex-5.

inter-region signal, IRS – a non-isolated net with one source and one load typically used to connect one isolated function to another, though sometimes used to connect one port of an isolated function to another port of the same isolated function. An IRS is not permitted to use routing resources containing programmable interconnect points (PIPs) in fence tiles.

isolation – free from unintended influence. For the case of routing, the degree isolation is measured by the number of switch failures required to established an unintended signal path between isolated circuits. For the case of floorplanning, isolation is determined by the presence of a “fence” of tiles free of isolated logic and routing between isolated portions of the design.

isolated function, isolated module – a portion of the user design that is intended to be isolated.

isolated region, isolation region – a collection of tiles defined by area range constraints that can be used when implementing an isolated function.

I/O buffer – a circuit in an FPGA that controls the behavior of the input/output pins on the FPGA package. An I/O buffer controls various communication-related settings such as whether an associated pin is connected internally to an input circuit or an output circuit; or selects the voltage level expected by the pin, etc.

I/O bank – a circuit that is shared among a collection of I/O buffers in an FPGA for settings and signals common to the collection.

NCD – a Xilinx design file that can describe the programming of a Xilinx FPGA at the logical level, but may also contain complete placement and routing information. IVT requires NCD files with complete placement and routing information.

net – a named collection of routing resources that create signal paths among a collection of logic elements. A net may span levels in the design hierarchy.

node – a representation of a wire in the software device model. Note that a node may branch out to connect more than two points. See also graph above.

package pin – a conductor on the outside of an FPGA package used for powering or interfacing with the FPGA, typically shaped like a short wire protruding perpendicularly from the chip package or shaped like a ball.

path – a hypothetical collection of unused routing resources that are examined for potential isolation violations.

placement – the assignment of a logical function to specific hardware resources.

Pin Isolation Group (PIG) – a file that specifies the correspondence between package pins and isolation groups use to aid IVT in checking the preliminary design along with the User Constraint File.

Programmable Interconnect Point (PIP) – a switch connecting a node with a port of a logic function or another node.

route – a path a signal can follow within an FPGA, especially as represented by a sequence of arcs that connects one node to another.

site – a physical location in the FPGA tile array that can be referenced in the User Constraint File, such as a SLICE, or RAMB16, etc.

search algorithm – a systematic method to find objects of interest—in this case, low paths connecting regions of a circuit.

signal – a net that does not cross block hierarchy boundaries.

switch matrix – customer terminology for an interconnect block.

Trusted Bus Macro – a soft IP design module used to provide trusted interconnect in isolated designs.

Trusted Routing – routing that connects isolated functions using routing resources with no programmable interconnection points within fence tiles. Unlike Trusted Bus Macros, Trusted Routing is generated automatically without manual placement.

undirected graph – in mathematics, an undirected graph is a symmetric relation defined by a set of vertices and a set of edges. An edge is an unordered pair of vertices from the graph representing a connection between the vertices. See also graph above.

User Constraint File – a design file that specifies constraints the Xilinx design software should maintain when it converts the user's design from a high-level description to a file that can be used to program the configuration memory. Typically, the constraints describe aspects of the design such as pin assignments, electrical properties of I/O signals, and timing characteristics, but not the logic of the design.

wire – a conductive path in a chip along which signals or power flow. A wire is the hardware that implements the node abstraction. The term wire is also used in the software device model for a portion of a node that occupies exactly one tile. Informally, the term wire segment is used to emphasize the geometric aspect of a wire in this context.

13 Error, Warning, and Isolation Violation Messages

In addition to isolation violations, IVT reports three kinds of error conditions:

- Argument Error – the parameters supplied are incomplete, inconsistent, or invalid.
- Input Errors – the input files are inaccessible or invalid.
- Internal Errors – a condition occurred for which there is no reasonable basis upon which to analyze the inputs.

Several conditions can trigger errors: obvious conditions such as incorrect file names, problems in the input files, and undefined conditions in the code. The presence of any error indicates that isolation has not been verified. Errors unrelated to the user's design end with the sentence, "Contact Xilinx for assistance." All other errors are intended to be self-explanatory. The error numbers are intended to uniquely identify the location in the code at which the error condition was reported.

IVT exists with a non-zero exit status when any errors are detected.

All errors and isolation violations are listed below.

13.1 Argument Errors

The following errors are generated when the parameters supplied are incomplete, inconsistent, or invalid. When any of the errors below is generated, no information can be inferred about the validity of the isolation of the input design.

Argument Error:1: At least two distinct -group command line arguments are required for isolation analysis.

Argument Error:2: Both device and package must be specified for user constraints file (.ucf) checking.

Argument Error:3: Device <*device*> was not found.
Check the spelling of the -device parameter.

Argument Error:14: Package <*package*> was not found.
Check the spelling of the -package parameter.

Argument Error:15: The -nopin option and the -pig option are mutually exclusive. The -nopin option suppresses pin analysis and obviates the pin isolation group (PIG) file.

Argument Error:17: Unable to open output file <*file name*>.
Check the file path syntax, the permissions settings of the target directory, and the free space on the device.

Argument Error:18: The word "<*reserved label*>" cannot be used as a group label.
The words 'combined' and 'ignored' are reserved.

Argument Error:65: Area group <*group*> is associated with more than one isolation group. An area group cannot appear more than once in the command line arguments.

Argument Error:77: Architecture <*architecture name*> is not supported.
Supported architectures are:
- Automotive Artix7
- Artix7
- Artix7 Low Voltage
- Automotive Spartan6
- Automotive Virtex4
- Automotive Zynq
- Kintex7
- Kintex7 Low Voltage
- Defense-Grade Artix7
- Defense-Grade Artix7 Low Voltage
- Defense-Grade Kintex7
- Defense-Grade Kintex7 Low Voltage
- Space-Grade Virtex-4QV
- Defense-Grade Spartan-6Q
- Defense-Grade Spartan-6Q Lower Power
- Defense-Grade Virtex-4Q
- Defense-Grade Virtex-5Q
- Defense-Grade Virtex-6Q
- Defense-Grade Virtex-6Q Lower Power
- Defense-Grade Virtex7
- Defense-Grade Zynq
- Spartan6
- Spartan6 Lower Power
- Virtex4
- Virtex5
- Virtex6
- Virtex6 Lower Power
- Virtex7
- Zynq

Argument Error:105: Unable to open output file <*file name*>.
Check the file path syntax, the permissions settings
of the target directory, and the free space on the device.

Argument Error:135: Device <*device name*> is not supported.
Supported architectures are:
- Automotive Artix7
- Artix7
- Artix7 Low Voltage
- Automotive Spartan6
- Automotive Virtex4
- Automotive Zynq
- Kintex7
- Kintex7 Low Voltage
- Defense-Grade Artix7
- Defense-Grade Artix7 Low Voltage
- Defense-Grade Kintex7

Defense-Grade Kintex7 Low Voltage
Space-Grade Virtex-4QV
Defense-Grade Spartan-6Q
Defense-Grade Spartan-6Q Lower Power
Defense-Grade Virtex-4Q
Defense-Grade Virtex-5Q
Defense-Grade Virtex-6Q
Defense-Grade Virtex-6Q Lower Power
Defense-Grade Virtex7
Defense-Grade Zynq
Spartan6
Spartan6 Lower Power
Virtex4
Virtex5
Virtex6
Virtex6 Lower Power
Virtex7
Zynq

Argument Error:136: The -f switch must be followed by a file name.

13.2 Input Errors

The following errors are generated when one of the input files supplied is incomplete, inconsistent, or invalid. When any of the errors below is generated, either a file name is incorrect, or there is a problem with the file itself. In the latter case, it may be advisable to reexamine EDA tool reports for errors and warnings.

Input Error:4: Encountered a pad on pin <*pin*> with no cell.
Check that the design as placed and routed without
errors or warnings.

Input Error:5: The file <*file name*>
did not load successfully.

Input Error:7: Failed to load design <*file name*>.

Input Error:16: Unable to load <*file name*>.
Check the file path, and file permissions.

Input Error:25: More than one site was associated with LOC constraint:
<*constraint identifier*>
Check for duplicated LOC constraints, wild cards,
missing delimiters, or other syntax errors.

Input Error:44: Net <*net*> is not fully placed and routed.
Isolation analysis cannot be completed without complete routing
information. Refer to the Place & Route Report of your synthesis tool.

Input Error:54: Failed to load design <*file name*>.

Input Error:55: Unable to load <*file name*>.
Check the file path, and file permissions.

Input Error:56: Isolation group <*group*> in the pin isolation group file was not defined in the arguments.
[Group <*group*> is an area group name.]

Input Error:57: Group <*group*> was redefined in the pin isolation group file at line <*line number*>.
Each isolation group must only be defined once.

Input Error:58: Duplicate pin net name <*net*> found in pin isolation group file at line <*line number*>.
Each pin must be listed at most once.

Input Error:62: The combined input design was not fully placed and routed.
Review the implementation report and correct any errors preventing the design from being fully implemented.

Error:64: Unable to open <*file name*>.
Check the file path, and file permissions.

Input Error:67: Expecting an isolation group in pin isolation group file at line <*line number* >.
Check for a missing semicolon, quotation mark or comment delimiter.

Input Error:68: Expecting a net name in pin isolation group file at line <*line number* >.
Check for a missing semicolon, quotation mark or comment delimiter.

Input Error:69: Unclosed isolation group in pin isolation group file at line <*line number* >.
Missing END statement.

Input Error:72: Group NCD file <*file name*> has more than one implemented block below top.
An NCD file that defines an isolation group cannot be merged nor complete. Consider supplying the module name instead of an NCD file path.

Input Error:73: No user visible modules were found in <*file name*>.
Check that the correct NCD file was supplied.

Input Error:74: '<*name*>' is not a module name in <*file name*>.
The modules found were:
<*module name list*>

> Only modules directly below the top level are listed because IVT is unable to determine which of the submodules might be desirable candidates for isolation analysis.

Input Error:76: The device family specified in <*file name*> does not match the one specified in <*file name*>.
The device in all design files must match.

Error:95: Unable to obtain area range constraints for module <module name>
Either this module is not isolated and therefore should not have
associated -group parameters on the command line or isolation-related
design constraints were not defined correctly.

Input Error:101: Unable to find user tile <*site name*>.

Input Error:102: Unclosed isolation group in pin
isolation group file at line <*line number*>.
Check for a missing semicolon, quotation mark or comment delimiter.

Input Error:125: Unable to obtain package information.
The input NCD file is not valid.

Input Error:131: Group <*group*> is an area group name.

Input Error:147: The Zynq Processing Subsystem site was not included
in the area range constraints (floorplan). Xilinx does not
support this configuration mode in production.

Input Error:151: Isolation group: <*group*>
has no area range constraints.
Compare the area group spelling on the command line to the UCF.

13.3 Internal Errors

An internal error indicates a condition occurred for which there is no reasonable basis upon which to analyze the inputs. IVT contains many internal consistency checks intended to guard against the IVT developer's mistakes or EDA tool defects. These messages are not expected to be meaningful without the context provided by the IVT source code. In practice, these errors have been very rare. It is best to contact Xilinx for assistance, if at all possible; however, in extremely rare cases, changing tools, tool versions or development platforms may help.

Internal Error:6: Unable to locate wire <*wire*> in device <*device*>.
Contact Xilinx for assistance.

Internal Error:10: Invalid regular expression syntax at i=<*pattern index*>:
<*regular expression error*>
Contact Xilinx for assistance.

Internal Error:12: Node <*node*> is not connected to node <*wire*>.
Contact Xilinx for assistance.

Internal Error:13: Nodes (aka wires) from last arc do not contain last node in path.
Contact Xilinx for assistance.

Internal Error:19: Unable to obtain Cs_ResSpec_TypeName.
Contact Xilinx for assistance.

Internal Error:20: Data from combined group was not found.
Contact Xilinx for assistance.

Internal Error:21: Area group not in expected form: <*constraint identifier*>.
Contact Xilinx for assistance.

Internal Error:22: Failed to enable test arcs.
Contact Xilinx for assistance.

Internal Error:23: I/O buffer name not in expected form: <*name*>.
Contact Xilinx for assistance.

Internal Error:24: I/O buffer name regular expression is incorrect: <*name*>.
Contact Xilinx for assistance.

Internal Error:26: Pin <*pin*> has unexpected functional type or hierarchical level.
Functional Block Type = <*type*>
Hierarchical Level = <*level*>
Group cannot be determined.
Contact Xilinx for assistance.

Internal Error:27: PinGroup::Visit() called on null.
Contact Xilinx for assistance.

Internal Error:28: ReportFault called on empty path.
Contact Xilinx for assistance.

Internal Error:29: Unable to obtain constraint manager.
Contact Xilinx for assistance.

Internal Error:30: Unable to process LOC constraint.
Contact Xilinx for assistance.

Internal Error:31: Unable to process range constraint.
Contact Xilinx for assistance.

Internal Error:32: Area group not in expected form: <*constraint identifier*>.
Contact Xilinx for assistance.

Internal Error:34: Data from combined group not found.
Contact Xilinx for assistance.

Internal Error:35: Line <*line number*> in pin isolation group file
matches netPat but has the wrong number of fields.
Contact Xilinx for assistance.

Internal Error:36: Found script line that matches arcPat but result.size() is <*match count*>, not 3.
Contact Xilinx for assistance.

Internal Error:37: Found script line that matches nodePat but result.size() is <*match count*>, not 2.
Contact Xilinx for assistance.

Internal Error:38: Inconsistent results from GetArcNodes().
Node <*node*> not connected to arc <*arc*>.
Contact Xilinx for assistance.

Internal Error:39: Invalid package pin for net <*net*> in group <*group*>.
Contact Xilinx for assistance.

Internal Error:40: Found net <*net*> with no group.
Contact Xilinx for assistance.

Internal Error:41: No file name found for net <*net*>.
Contact Xilinx for assistance.

Internal Error:42: pkgMap was not initialized.
Contact Xilinx for assistance.

Internal Error:43: Unable to get I/O buffer from package pin on net <*net*>.
Contact Xilinx for assistance.

Internal Error:59: Net <*net*> has no nodes in common with <arc>
Net <*net*> contains the following arcs:
<*arc list*>
Contact Xilinx for assistance.

Internal Error:60: GetNext() returned undefined node.
Contact Xilinx for assistance.

Internal Error:70: Uncaught exception.
Contact Xilinx for assistance.

Internal Error:71: Unable to get net for pin <*pin*>.
Contact Xilinx for assistance.

Internal Error:79: First node in path <*node*> is not part of any routed net.
Contact Xilinx for assistance.

Internal Error:80: NetChecker::ReportFault called on null start or end net.
Contact Xilinx for assistance.

Internal Error:81: Data from group <*group*> was not found.
Contact Xilinx for assistance.

Internal Error:82: Invalid RevArc direction.
Contact Xilinx for assistance.

Internal Error:85: A fatal error has occurred in the UCF parser:
<*error message*>
Line: <*line number*>
<*statement*>
Contact Xilinx for assistance.

Internal Error:93: Invalid regular expression.
Contact Xilinx for assistance.

Internal Error:94: Invalid regular expression.
Contact Xilinx for assistance.

Internal Error:97: Invalid cell passed to IsInterconnectTile.
Contact Xilinx for assistance.

Internal Error:98: Non-tile cell passed to IsInterconnectTile.
Contact Xilinx for assistance.

Internal Error:99: Device model problem: undefined cell.
Contact Xilinx for assistance.

Internal Error:103: About to overwrite pkgMap entry.
Contact Xilinx for assistance.

Internal Error:104: Data from group 'combined' was not found.
Contact Xilinx for assistance.

Internal Error:106: Tile cell definition is not valid.
Contact Xilinx for assistance.

Internal Error:107: Unable to find <*cell*> in cell-to-package map.
Contact Xilinx for assistance.

Internal Error:110: Unable to find area range data. Code: <*error code*>
Contact Xilinx for assistance.

Internal Error:116: Unable to parse range constraint. Code: <*error code*>
Contact Xilinx for assistance.

Internal Error:117: Unable to find tile (site) named <*site name*>.
Contact Xilinx for assistance.

Internal Error:121: FindNprSiteRangeListText failed to find area group name for module <*module name*>.
Contact Xilinx for assistance.

Internal Error:124: No cell for tile <*cell count*>.
Contact Xilinx for assistance.

Error:127: GetIntTile() for site <*site*> returned an invalid cell.
Contact Xilinx for assistance.

Internal Error:130: Unable to categorize net <*net*>.
Source:
Group: <*group*>
Block: <*block*>
Type: <*type*>
Pin: <*pin*>
Load:
Group: <*group*>
Block: <*block*>
Type: <*type*>
Pin: <*pin*>
Contact Xilinx for assistance.

Internal Error:132: Unable to find logic configuration information.
Contact Xilinx for assistance.

13.4 Warnings

A warning indicates that an ambiguous condition was encountered and that IVT is ignoring some portion of the input. IVT does not have enough information to determine whether there is a valid reason to ignore the condition or definitively report an error condition. With the exception of the PIG file, all the inputs to IVT contain more information than IVT analyzes. In some cases it should be obvious to the designer whether or not a warning is significant. In other cases, the warning indicates that a

condition assumed to be true by the IVT developer is not always true, but the condition is probably not critical to successful isolation analysis.

Warning:11: No isolation group was found for area group <*group*>.
Check that the command line parameters match the input design files.

Warning:45: Isolation group <$group_1$> declared in pin isolation group file does not match isolation group <$group_2$> inferred from area group for pin <*pin*> on net <*net*>.
This inconsistency between the supplied PIG file and the UCF file should be investigated as it may indicate invalid inputs were supplied.

Warning:47: No isolation group was specified for pin <*pin*> on net <*net*>.

Warning:49 Pin <*pin*> appears unconnected.
Check the ngdbuild report for related errors or warnings.

Warning:51: Unable to determine area group for range constraint:
<constraint>
Contact Xilinx for assistance.

Warning:52: Unable to find isolation group for area group <*group*>.
If the area group listed is needed for isolation analysis, this condition must be corrected.

Warning:75: Unable to find isolation group for pin <*pin*>.
Check the ngdbuild report for related errors or warnings.

Warning: 78: Net <*net*> appears more than once in group <*group*>.
Check the command line arguments for duplicated file names.

Warning:86: UCF parser indicated a non-fatal error condition:
<*message*>
Line: <*line number*>
<*statement*>

Warning: 100: Unable to find both tileMin and tileMax.
<*group*>:<*site type*>
Contact Xilinx for assistance.

Warning:108: The package pin below was listed more than once.

Warning:109: The I/O buffer below was listed more than once.

Warning:120: Unable to obtain placement information for block <*block*>.
Verify that the design is fully placed and routed.

Warning:121: No tile for site <*site*>.
Contact Xilinx for assistance.

Warning:146: Unexpected site type: <*site type*> for int: <*int*>

This message should only appear when running IVT on unsupported devices such as the Virtex-7 SSIT devices.

13.5 Isolation Violations

When IVT detects a violation of the isolation rules, one or more of the messages below are reported. To achieve the utmost reliability, the conditions reported below should be corrected.

Isolation Violation:63: Isolation groups <*group*$_1$> and <*group*$_2$> are adjacent.
Range <*range*$_1$> from area group <*group*$_1$> conflicts with
range <*range*$_2$> from area group <*group*$_2$>.

Isolation Violation:63: Isolation groups <*group*$_1$> and <*group*$_2$> overlap.
Range <*range*$_1$> from area group <*group*$_1$> conflicts with
range <*range*$_2$> from area group <*group*$_2$>.

Isolation Violation:112: The pairs of tiles listed below contain nets from multiple isolation groups, but lack an adequate fence between them.

Isolation Violation:114: One or more tiles contain nets from multiple isolation groups.

Isolation Violation:122: No I/O buffers from distinct isolation groups shall be adjacent.

Isolation Violation:123: No pins on the package from distinct isolation groups shall be adjacent.

Isolation Violation:124: No I/O buffers from distinct isolation groups shall share an I/O bank.

Isolation Violation:128: Inter-region signal <*net*> has loads in more than one isolation group.

Isolation Violation:129: The following tiles are configured to use wires that contain programmable interconnect points (PIPs) in the fence. Such wires are not allowed. Check that constraints needed for isolation are present.

Isolation Violation:133: According to current fault costs, the paths below could be enabled with fewer than the required number of faults.

[The list below is intended to be representative, not complete. For each pair of isolated nets that cannot be shown to be isolated, the path between them with minimum cost is reported. Due to the relative abundance of routing resources in the Virtex-4 FPGA, when any failing paths exist, there are usually many variations with equal cost.]

Isolation Violation:134: The tiles below are used to separate isolated nets or logic and have non-default configurations.

Isolation Violation:137: The following sites cannot be used as a fence.
<*site_type*> Site: <*site_name*> has neighbors in multiple isolation groups.
Group: <*group*> Tile: <*tile*>

Isolation Violation:141: Isolated CMT logic sites require all adjacent
CMT sites to be included in matching isolation groups. The following
sites are either not included in isolated area ranges or are included
in non-matching isolated ranges.
<*sites*>

Isolation Violation:142: Used CMT logic sites must be isolated.
The following tiles are not part of isolated area ranges:
<*tiles*>

Isolation Violation:143: The Config Center sites cannot be used by more than one
isolation group. The complete list of config center sites is:
BSCAN, CAPTURE, DCIRESET, DNA_PORT, EFUSE_USR,
FRAME_ECC, ICAP, STARTUP, USR_ACCESS, and XADC.

Group: <*group*> Tile: <*tile*>

Isolation Violation:144: The Config Center sites cannot be used as a fence.
The complete list of config center sites is:
BSCAN, CAPTURE, DCIRESET, DNA_PORT, EFUSE_USR,
FRAME_ECC, ICAP, STARTUP, USR_ACCESS, and XADC.

Group: <*group*> Tile: <*tile*>

Isolation Violation:148: 7 Series GTX tiles within a single clock
region cannot occupy multiple isolation groups.

Isolation Violation:149: The clock buffer column is not allowed
as a fence.

Isolation Violation:150: The config center column is not allowed
as a fence.

1

14 Index

I

L

M

N

P

R