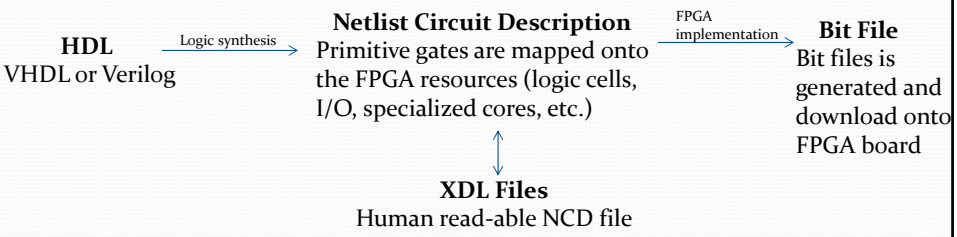
Xilinx Design Language

Bill Jason P. Tomas
University of Nevada- Las Vegas
Dept. of Electrical and Computer Engineering

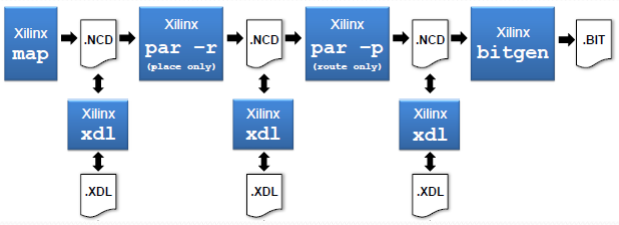
What is Xilinx Design Language?

- XDL is a human-readable ASCII format compatible with the more widely used NCD (Netlist Circuit Description). XDL and NCD files are both native Xilinx netlist formats for describing and representing FPGA designs.



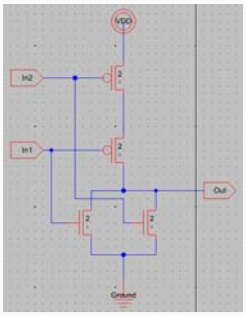
XDL Designs

- XDL is able to represent designs that are:
  - Mapped (unplaced and unrouted)
  - Partially placed and unrouted
  - Partially placed and routed
  - Fully placed and unrouted
  - Contain hard macros and instances of hard macros
  - A hard macro definition



What is a netlist?

- A netlist is a text representation of a circuit diagram or schematic (textual or schematic). The netlist can be generated on any level of the design process whether it be on the transistor level or gate level.



Schematic

```
*** SPICE deck for cell NOR-1{sch} from library NOR
*** Created on Wed Oct 19, 2011 19:39:26
*** Last revised on Sun Oct 23, 2011 17:00:21
*** Written on Sun Oct 23, 2011 17:13:37 by Electric VLSI Design System,
*version 9.00
*** Layout tech: mocmos, foundry MOSIS
*** UC SPICE *** , MIN_RESIST 4.0, MIN_CAPAC 0.1FF

.global gnd vdd

*** TOP LEVEL CELL: NOR-1{sch}
Mnmos@0 Out In1 gnd gnd CMOSN L=0.6um W=0.9um
Mnmos@1 Out In2 gnd gnd CMOSN L=0.6um W=0.9um
Mpmos@0 vdd In1 net@0 vdd CMOSP L=0.6um W=3.9um
Mpmos@1 net@0 In2 Out vdd CMOSP L=0.6um W=3.9um

* Spice Code nodes in cell cell 'NOR-1{sch}'
vdd vdd 0 dc 5
vin In1 0 dc 0 pulse(5 0 1ns 2ns 2ns 10ns 20ns)
.tran 0 100ns 1ns
.include C5_TT.txt
.END
```

Spice Deck Netlist

HDL → Netlist → Map & Route → Hardware

RTL Coding → Synthesis optimization → NCD/XDL File → 110001 11000 1110 Bit stream

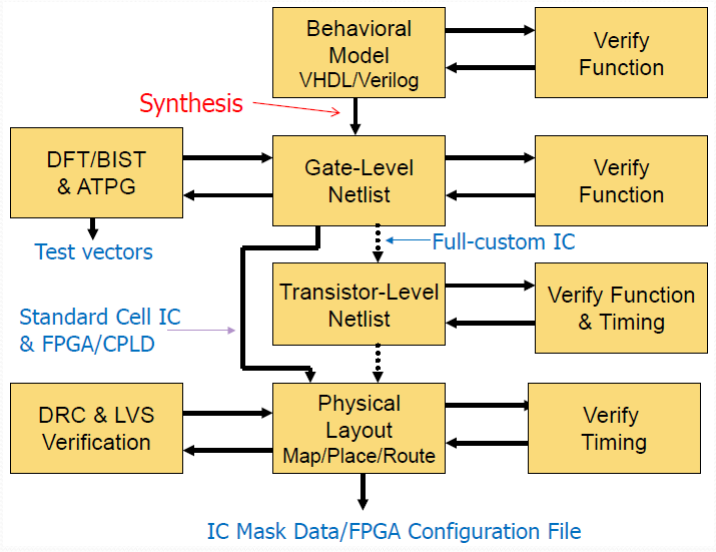
Synthesis optimization → Gate Level → 110001 11000 1110 Bit stream

Gate Level ↔ Transistor Level (Equivalence check)

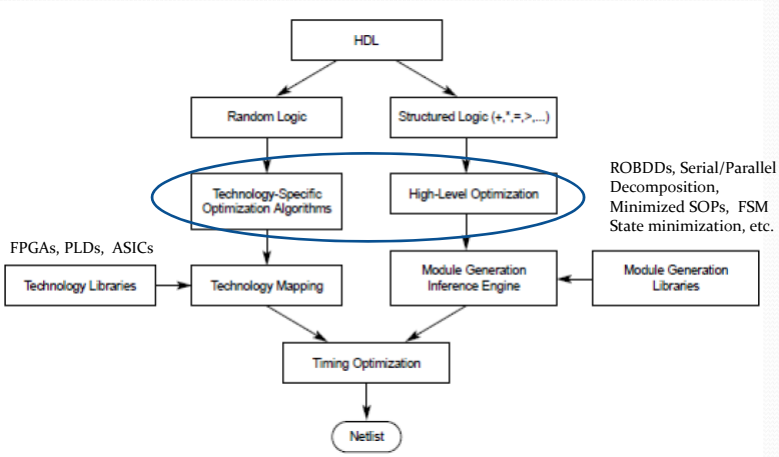
Transistor Level → 110001 11000 1110 Bit stream

→ Hardware Implementation (FPGA / ASIC)

Digital Design Flow



Synthesis Design Flow



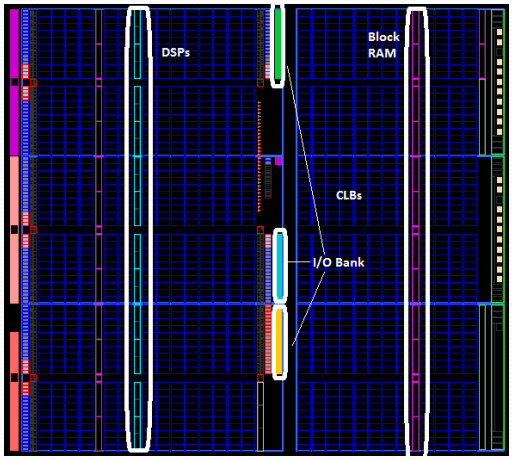
Available Resources via XDL

- XDL reports (XDLRC files) can be generated to give a description of all FPGA resources and their connections between each other.
- These reports vary from a few megabytes (smaller devices) up to several gigabytes (newer devices).
- Uses Xilinx tile structure of the FPGA, consisting of CLBs, I/O tiles, Configurable Interconnects, and specialized cores (block RAM, power PCs, DSPs).

Xilinx Tile Map (Virtex-5)

FPGAs consists of an array of tiles of different types:

-CLBs
-Interconnects
-Clocks
-BRAMs
-I/O
-Specialized Cores



FPGA Global Coordinates

Tile Type Specific Coordinates

Spartan-6 XDLRC (Bechkhoff)

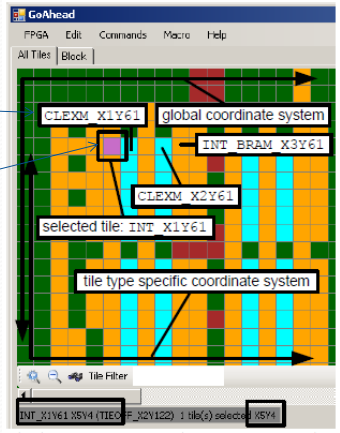
Header describes the device, and the number of nodes (73 x 62)

Tile INT_X61Y61 connects CLB CLEXL_X1Y61 With other CLBs

Global coordinates

tile type coordinates

```
# header
(xdl_resource_report v0.2 xc6slx16csg324-3 spartan6
# dimension
(tiles 73 62
 ...
 # configurable logic block with two slices
 (tile 4 6 CLEXL_X1Y61 CLEXL 2
   (primitive_site SLICE_X0Y61 SLICEL internal 45
    (pinwire A1 input L_A1)
    ...
    (primitive_site SLICE_X1Y61 SLICEX internal 43
    ...
    (pinwire D output XX_D)
  ...
 # interconnect tile
 (tile 4 5 INT_X1Y61 INT 1
  ...
   (wire EE2B0 2
     (conn CLEXM_X2Y61 CLEXM_EE2M0)
     (conn INT_BRAM_X3Y61 EE2E0)
   ...
   # switch matrxi multiplexers
   (pip INT_X1Y61 EE2E0 -> EE2B0)
   (pip INT_X1Y61 EE4E0 -> EE2B0)
  (pip INT_X1Y61 EL1E_S0 -> LOGICIN_B9)
   ...
# summary
(summary tiles=4526 sites=5378 sitedefs=46
   numpins=157962 numpips=5782505))
```



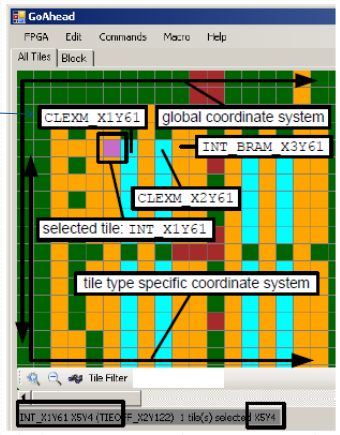
Spartan-6 XDLRC

node

Subnodes of CLB: 2 slices: sliceL (logic) & sliceX (no carry-chain)

```
# header
(xdl_resource_report v0.2 xc6slx16csg324-3 spartan6
# dimension
(tiles 73 62
 ...
 # configurable logic block with two slices
 (tile 4 6 CLEXL_X1Y61 CLEXL 2
   (primitive_site SLICE_X0Y61 SLICEL internal 45
    (pinwire A1 input L_A1)
    ...
    (primitive_site SLICE_X1Y61 SLICEX internal 43
    ...
    (pinwire D output XX_D)
  ...
 # interconnect tile
 (tile 4 5 INT_X1Y61 INT 1
   ...
   (wire EE2B0 2
     (conn CLEXM_X2Y61 CLEXM_EE2M0)
     (conn INT_BRAM_X3Y61 EE2E0)
    ...
   # switch matrxi multiplexers
   (pip INT_X1Y61 EE2E0 -> EE2B0)
   (pip INT_X1Y61 EE4E0 -> EE2B0)
 (pip INT_X1Y61 EL1E_S0 -> LOGICIN_B9)
  ...
# summary
(summary tiles=4526 sites=5378 sitedefs=46
   numpins=157962 numpips=5782505))
```

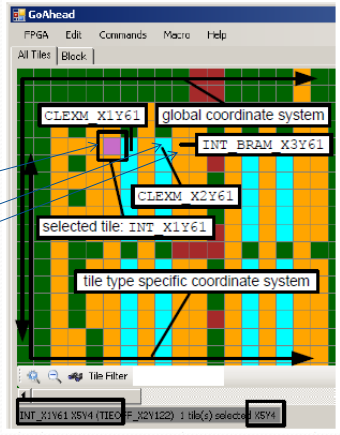
Pinwire's represent input and outputs to the site (varies by component)



Spartan-6 XDLRC

Wire EE2B0 connects the interconnect tile INT_X1Y61 with CLEXM (CLB) and INT_BRAM (block RAM)

```
# header
(xdl_resource_report v0.2 xc6slx16csg324-3 spartan6
# dimension
(tiles 73 62
 ...
 # configurable logic block with two slices
 (tile 4 6 CLEXL_X1Y61 CLEXL 2
   (primitive_site SLICE_X0Y61 SLICEL internal 45
    (pinwire A1 input L_A1)
    ...
    (primitive_site SLICE_X1Y61 SLICEX internal 43
    ...
    (pinwire D output XX_D)
  ...
 # interconnect tile
 (tile 4 5 INT_X1Y61 INT 1
   ...
   (wire EE2B0 2
     (conn CLEXM_X2Y61 CLEXM_EE2M0)
     (conn INT_BRAM_X3Y61 EE2E0)
    ...
   # switch matrxi multiplexers
   (pip INT_X1Y61 EE2E0 -> EE2B0)
   (pip INT_X1Y61 EE4E0 -> EE2B0)
 (pip INT_X1Y61 EL1E_S0 -> LOGICIN_B9)
  ...
# summary
(summary tiles=4526 sites=5378 sitedefs=46
   numpins=157962 numpips=5782505))
```



Spartan-6 XDLRC

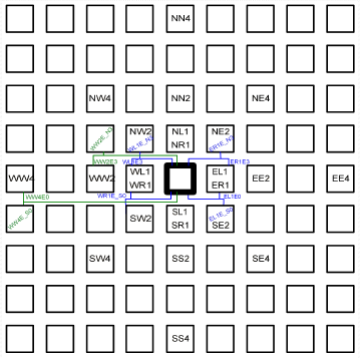
```
# header
(xdl_resource_report v0.2 xc6slx16csg324-3 spartan6
# dimension
(tiles 73 62
 ...
 # configurable logic block with two slices
 (tile 4 6 CLEXL_X1Y61 CLEXL 2
   (primitive_site SLICE_X0Y61 SLICEL internal 45
    (pinwire A1 input L_A1)
    ...
    (primitive_site SLICE_X1Y61 SLICEX internal 43
    ...
    (pinwire D output XX_D)
 ...
 # interconnect tile
 (tile 4 5 INT_X1Y61 INT 1
  ...
  (wire EE2B0 2
    (conn CLEXM_X2Y61 CLEXM_EE2M0)
    (conn INT_BRAM_X3Y61 EE2E0)
  ...
  # switch matrxi multiplexers
  (pip INT_X1Y61 EE2E0 -> EE2B0)
  (pip INT_X1Y61 EE4E0 -> EE2B0)
(pip INT_X1Y61 EL1E_S0 -> LOGICIN_B9)
 ...
# summary
(summary tiles=4526 sites=5378 sitedefs=46
  numpins=157962 numpips=5782505))
```

# of tiles the interconnect tile connects to

Direction (EE = east; NN = north, etc.)

# of tiles spanned

Wires are uni-directional with a beginning, middle, and end (B,M,E)



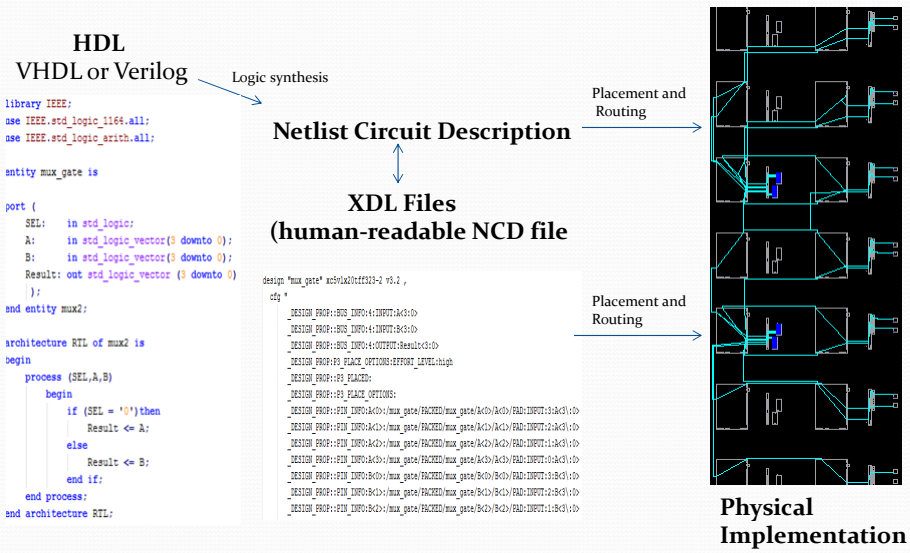
Overview of XDLRC Syntax

- **Tiles**: Building blocks of Xilinx FPGAs, which are arranged in a 2D array of tiles. Each tile is declared with a "tile" directive, followed by a row and column index (global), type (CLB, DSP, BRAM), tile specific coordinates, and the number of primitive sites in the tiles. Tiles can have three sub-nodes: primitives, wires, and PIPs.
- **Primitive Sites:** Location on the FPGA that allows for an instance of that primitive type (ex: multiple slice types for a single CLB). Given with a unique name (ex. SLICE_X9Y127) and type (SLICEL). They also contain pinwires which describe the name and direction of pins in the primitive site.

Overview of XDLRC Syntax

- **Wire:** Routing resource that exists in the tile that may have zero or more connections leaving the tile. These connections are described using the tile name, and wire name of that tile denoting connectivity. These connections are not programmable, however the PIPs which control the direction of the bit data can be controlled.
- **PIP:** Unidirectional routing resource used within the net to complete routing connections between sources and destinations. Described using an existing tile and two wires with a connection between them.

Digital System Design Flow in Xilinx



XDL System Design Representation

- XDL can also be used to implement a complete design of a system as an XDL netlist description.
- NCD files (generated by Xilinx ISE) can be converted to XDL files and vice-versa with internal commands in ISE.
- Describe implementation of a system after map and routing has been performed, and can include placement information of primitive sites, and their routing in terms of switch matrix connections.
- Although the system and resource description share common syntactical elements, they differ in their structure.

XDL Syntax – Design Statement

Design statements (1 per design) contain global information which includes the name of the design and the intended Xilinx FPGA device. In can also contain a list of attributes in a "cfg" body, which allows the user to configure a certain aspect of the design

Design Name

Virtex-5 LXT Device

```
design "mux_gate" xc5vlx20tff323-2 v3.2 ,
  cfg "
       _DESIGN_PROP::BUS_INFO:4:INPUT:A<3:0>
       _DESIGN_PROP::BUS_INFO:4:INPUT:B<3:0>
       _DESIGN_PROP::BUS_INFO:4:OUTPUT:Result<3:0>
       _DESIGN_PROP:P3_PLACE_OPTIONS:EFFORT_LEVEL:high
       _DESIGN_PROP::P3_PLACED:
       _DESIGN_PROP::P3_PLACE_OPTIONS:
       _DESIGN_PROP::PIN_INFO:A<0>:/mux_gate/PACKED/mux_gate/A<0>/A<0>/PAD:INPUT:3:A<3\:0>
       _DESIGN_PROP::PIN_INFO:A<1>:/mux_gate/PACKED/mux_gate/A<1>/A<1>/PAD:INPUT:2:A<3\:0>
       _DESIGN_PROP::PIN_INFO:A<2>:/mux_gate/PACKED/mux_gate/A<2>/A<2>/PAD:INPUT:1:A<3\:0>
       _DESIGN_PROP::PIN_INFO:A<3>:/mux_gate/PACKED/mux_gate/A<3>/A<3>/PAD:INPUT:0:A<3\:0>
       _DESIGN_PROP::PIN_INFO:B<0>:/mux_gate/PACKED/mux_gate/B<0>/B<0>/PAD:INPUT:3:B<3\:0>
       _DESIGN_PROP::PIN_INFO:B<1>:/mux_gate/PACKED/mux_gate/B<1>/B<1>/PAD:INPUT:2:B<3\:0>
       _DESIGN_PROP::PIN_INFO:B<2>:/mux_gate/PACKED/mux_gate/B<2>/B<2>/PAD:INPUT:1:B<3\:0>
       _DESIGN_PROP::PIN_INFO:B<3>:/mux_gate/PACKED/mux_gate/B<3>/B<3>/PAD:INPUT:0:B<3\:0>
       _DESIGN_PROP::PIN_INFO:Result<0>:/mux_gate/PACKED/mux_gate/Result<0>/Result<0>/PAD:OUTPUT:3:Result<3\:0>
       _DESIGN_PROP::PIN_INFO:Result<1>:/mux_gate/PACKED/mux_gate/Result<1>/Result<1>/PAD:OUTPUT:2:Result<3\:0>
       _DESIGN_PROP::PIN_INFO:Result<2>:/mux_gate/PACKED/mux_gate/Result<2>/Result<2>/PAD:OUTPUT:1:Result<3\:0>
```

XDL Syntax – Module Statement

```
module "moduleName" "anchorInstanceName" , cfg "_SYSTEM_MACRO::FALSE" ;
  port "portName1" "anchorInstanceName" "F2";
  port "portName2" "anotherInstanceInTheModule" "F4";
...
  inst "anchorInstanceName" "SLICEL", placed CLB_X14Y4 SLICE_X23Y8 , …
...
  net "aNetInsideTheModule" , …
...
endmodule "moduleName";
```

Modules are collections of instances and nets which can be described as hard macros if the instances are placed and nets are routed. A module will have a list of ports that determine the interface of the hard macro or module and each module will have its own list of instances and nets to describe the logic inside.

XDL Syntax – Instance Statement

The instance statement, which begins with the keyword "inst", is an instance of an FPGA primitive which can be placed or unplaced depending if a tile and primitive site location are specified. The instance also has a primitive type (such as SLICEL, SLICEM, etc). Instances are configured with a cfg string which is a list of attributes that define LUT content, and other functionality.

```
inst "Result<0>" "IOB",placed CIOB_X17Y29 R1  ,
  cfg " DIFFI_INUSED::#OFF DIFF_TERM::#OFF IMUX::#OFF OUSED::0 PADOUTUSED::#OFF
       PULLTYPE::#OFF TUSED::#OFF OUTBUF:Result_0_OBUF: PAD:Result<0>:
         DRIVE::12  OSTANDARD::LVCMOS25  SLEW::SLOW "
  ;
inst "Result<1>" "IOB",placed CIOB_X17Y28 U1  ,
  cfg " DIFFI_INUSED::#OFF DIFF_TERM::#OFF IMUX::#OFF OUSED::0 PADOUTUSED::#OFF
       PULLTYPE::#OFF TUSED::#OFF OUTBUF:Result_1_OBUF: PAD:Result<1>:
         DRIVE::12  OSTANDARD::LVCMOS25  SLEW::SLOW "
  ;
inst "Result<2>" "IOB",placed CIOB_X17Y24 U3  ,
  cfg " DIFFI_INUSED::#OFF DIFF_TERM::#OFF IMUX::#OFF OUSED::0 PADOUTUSED::#OFF
       PULLTYPE::#OFF TUSED::#OFF OUTBUF:Result_2_OBUF: PAD:Result<2>:
         DRIVE::12  OSTANDARD::LVCMOS25  SLEW::SLOW "
  ;
inst "Result<3>" "IOB",placed CIOB_X17Y23 P4  ,
  cfg " DIFFI_INUSED::#OFF DIFF_TERM::#OFF IMUX::#OFF OUSED::0 PADOUTUSED::#OFF
       PULLTYPE::#OFF TUSED::#OFF OUTBUF:Result_3_OBUF: PAD:Result<3>:
         DRIVE::12  OSTANDARD::LVCMOS25  SLEW::SLOW "
```

Different XDL Applications

- Implementing bus macros
- Constraining routing
- Remapping on-chip clocks
- Homogenous placement and routing
- Netlist Relocation
- Design Merging

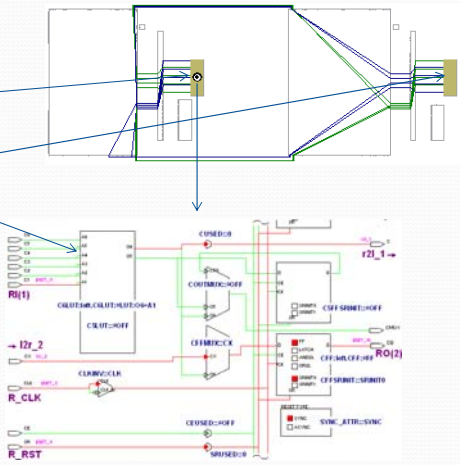
Implementing Bus Macros

- Bus macros are used to implement the interfaces between the static system and the reconfigurable modules that will be swapped at run-time.
  - Requirement for a static-only system in the physical implementation of a reconfigurable system is **binding of the partial module entity signals to a set of predefined wires of the FPGA fabric**. This separates partial resources from resources used to implement the static system, allowing run-time reconfigurable systems to be implemented.

Bus Macro Implementation in a Spartan-6 CLB (Beckhoff)

```
design "S6BusMacro.ncd" xc6slx16cpg196-2 v3.2 ;
module "S6BM", "left" cfg "_SYSTEM_MACRO::FALSE";
# I/O ports (4 I/Os per direction&side+CLK&reset)
  port "LI(0)" "left" "D1";
  port "LI(1)" "left" "A1";
  ...
# component instantiations
inst "left" SLICEX,placed CLEXM_X8Y33 SLICE_X11Y33,
 cfg "A6LUT:left.A6LUT:#LUT:O6=A1 AFFMUX::AX
      AUSED::0 AFFSRINIT::SRINIT0 AFF:left.AFF:#FF
      ...
      D6LUT:left.D6LUT:#LUT:O6=A1 DFFMUX::DX
      DUSED::0 DFFSRINIT::SRINIT0 DFF:left.DFF:#FF
      SRUSED::0 SYNC_ATTR::SYNC CLKINV::CLK ";
inst "right" SLICEX,placed CLEXL_X9Y33 SLICE_X13Y33,
 cfg "A6LUT:right.A6LUT:#LUT:O6=A1 ... ";

# dummy nets for I/O pins
net "LI(0)", inpin "left" "D1", ;
  ...
# nets between left and right slice
net "l2r_0" ,
  outpin "left" D ,  inpin "right" AX ,
  pip CLEXL_X9Y33 CLEXL_LOGICIN_B6 -> XX_AX ,
  pip CLEXM_X8Y33 X_D -> CLEXM_LOGICOUT9 ,
  pip INT_X8Y33 LOGICOUT9 -> ER1B0 ,
  pip INT_X9Y33 ER1E0 -> LOGICIN_B6 ,   ;
...
endmodule "S6BM" ;
```



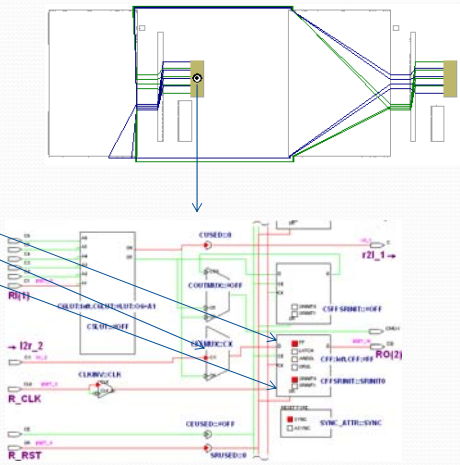
Macro's define the configuration of the system, and for this case: the output of the LUT, definition of the FF attributes (SR, synchronous, init value), and the outputs of the CLB (one through a FF, the other the output from the LUT)

Bus Macro Implementation in a Spartan-6 CLB

```
design "S6BusMacro.ncd" xc6slx16cpg196-2 v3.2 ;
module "S6BM", "left" cfg "_SYSTEM_MACRO::FALSE";
# I/O ports (4 I/Os per direction&side+CLK&reset)
  port "LI(0)" "left" "D1";
  port "LI(1)" "left" "A1";
  ...
# component instantiations
inst "left" SLICEX,placed CLEXM_X8Y33 SLICE_X11Y33,
 cfg "A6LUT:left.A6LUT:#LUT:O6=A1 AFFMUX::AX
      AUSED::0 AFFSRINIT::SRINIT0 AFF:left.AFF:#FF
      ...
      D6LUT:left.D6LUT:#LUT:O6=A1 DFFMUX::DX
      DUSED::0 DFFSRINIT::SRINIT0 DFF:left.DFF:#FF
      SRUSED::0 SYNC_ATTR::SYNC CLKINV::CLK ";
inst "right" SLICEX,placed CLEXL_X9Y33 SLICE_X13Y33,
 cfg "A6LUT:right.A6LUT:#LUT:O6=A1 ... ";

# dummy nets for I/O pins
net "LI(0)", inpin "left" "D1", ;
  ...
# nets between left and right slice
net "l2r_0" ,
  outpin "left" D ,  inpin "right" AX ,
  pip CLEXL_X9Y33 CLEXL_LOGICIN_B6 -> XX_AX ,
  pip CLEXM_X8Y33 X_D -> CLEXM_LOGICOUT9 ,
  pip INT_X8Y33 LOGICOUT9 -> ER1B0 ,
  pip INT_X9Y33 ER1E0 -> LOGICIN_B6 ,   ;
...
endmodule "S6BM" ;
```

Router is in route-through mode in which the input variable is passed through the LUT and logic expression is evaluated



Bus Macro Implementation

Macro File
(NMC format)

or

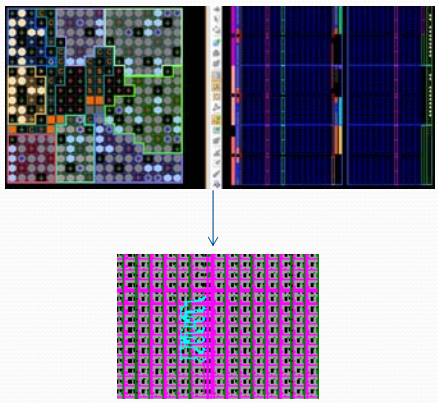
FPGA-editor script
(insert ports)

```
unselect -all
select  net "LI(0)"
select  net "LI(1)"
...
delete

unselect -all
select pin SLICE_X13Y33.CLK
add extpin
post attr pin SLICE_X13Y33.CLK
setattr pin SLICE_X13Y33.CLK
external_name R_CLK
unpost pin "SLICE_X13Y33.CLK"

unselect -all
select pin SLICE_X13Y33.B1
add extpin
post attr pin SLICE_X13Y33.B1
setattr pin SLICE_X13Y33.B1  external_name RI(0)
unpost pin "SLICE_X13Y33.B1"
```

Routing
UCF file

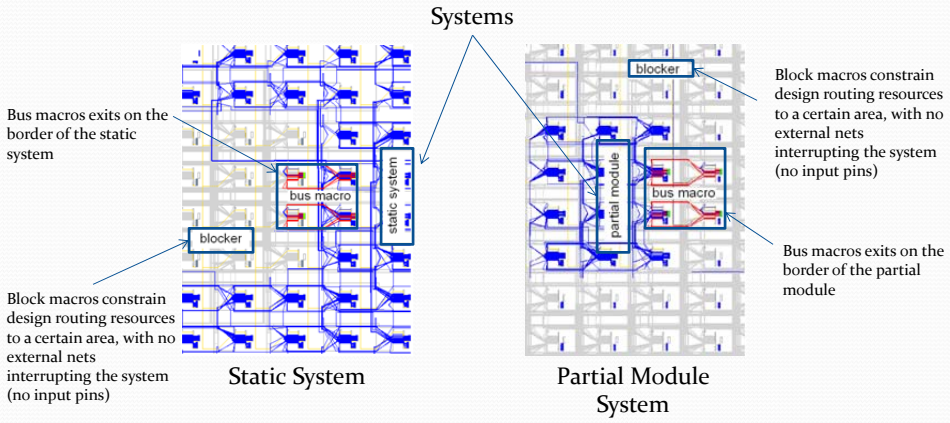


Routing Constraints

To implement a reconfigurable system using the bus macro approach, the macros have to be placed on the border between the static system and a partial module. This is accomplished by creating constraints in the macro file. For more in-depth floor planning there are *area group* and *prohibit* constraint functions in ISE which groups designs in a certain area of the FPGA or prohibits it respectively, but these functions do not aid in routing the necessary nets and wires.

A solution is to use block macros, which are generated to occupy a definable set of routing resources. A block macro consists of primitive instantiations acting as drivers, which are the starting points of antenna nets, that have no input pins, but contain PIP statements.

Routing Constraints Using Block Macros



Clock Remapping of a System

- With XDL, it is possible to remap a reconfigurable module or any other part of a system to another clock without affecting the rest of the system.
- Can be used in a component-based design in which a fully placed and routed module from one system can move to another, even if they have been implemented using different clock networks. → Operation of modules can swap between two clock domains, since no dedicated clock domain is required to control the clock frequency of a module.

Clock Remapping of a System

To remap a clock, the clock input statements have to be moved to a new clock net

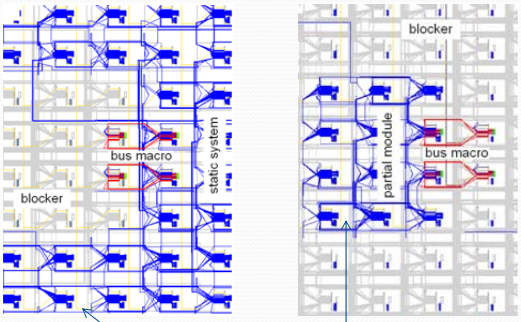
New clock output can be re-routed to a different domain or different clock net

```
net "clk_100" ,
  outpin "clk_generator/PLL1_CLK_BUFG_INST" O,
  inpin "Hex2Bin_1/HighReg<3>" CLK ,
  ...
# add blocker clock inputs to clock net
  inpin "SLICE_X10Y22" CLK ,
  inpin "SLICE_X10Y21" CLK ,
  ...
```

Dummy clock inputs block original mapping

Original clock wire

Original statements from first clock map (switch matrix settings, clock MUX from global clock tree) must first be removed. New clock routes can be done in FPGA editor (missing PIP statements will be added) without disrupting the original clock tree.

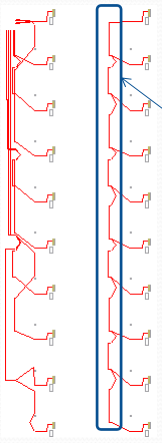


Clock net is highlighted yellow

Homogenous Place and Route

Xilinx tools provide no option to generate homogenously arranged physical implementations of any part of a system. Even if primitives are placed in a certain area, the routing will be irregular.

However, some applications need homogenous routing such as on-FPGA communication architectures for reconfigurable systems or time-to-digital converters. In a time-to-digital converter, a propogation delay of a signal is used to measure pulses faster than the bus clock speed, which is achieved by connected the signal to a several FFs which sample on the same bus clock. Homogenous routing ensures linear latency increase from FF to FF.



Time to Digital Circuit Routing

```
net "delay_net" ,
 outpin S0 C, inpin S1 C3,inpin S2 C5,...
 pip CLBLL_X19Y99 L_C->SITE_LOGIC_OUTS10,
 pip INT_X19Y99 LOGIC_OUTS10->SR2BEG1,
 pip INT_X19Y98 SR2MID1 -> SW2BEG1,
 pip INT_X19Y97 SW2MID1 -> SR2BEG1,
 pip INT_X19Y96 SR2MID1 -> SW2BEG1,
 pip INT_X19Y95 SW2MID1 -> SR2BEG1,
 pip INT_X19Y94 SR2MID1 -> SW2BEG1,
 pip INT_X19Y98 SR2MID1 -> IMUX_B8,
 pip CLBLL_X19Y98 SITE_IMUX_B8 -> L_C3,
 pip INT_X19Y97 SW2MID1 -> IMUX_B9,
 pip CLBLL_X19Y97 SITE_IMUX_B9 -> L_C5,
 ...
```

PIPs for the south routing (SR and SW directions) route to the same wire.

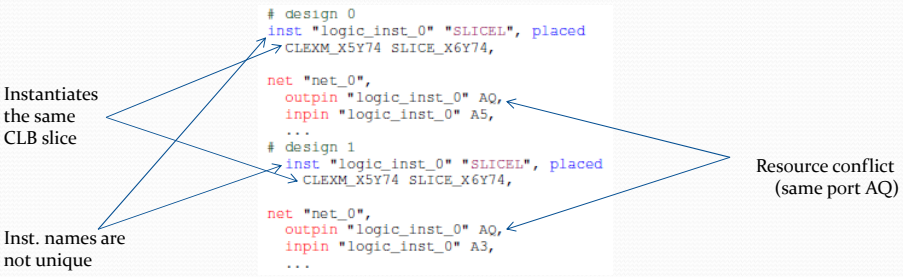
Netlist Relocation

- With a fully placed and routed netlist, each components XY coordinates can be relocated by a given offset, thus making it possible to relocate the entire netlist.
  - Must have knowledge of resource layout since netlist cannot be arbitrarily relocated anywhere on the FPGA.
  - Each tile type has a specific XY coordinate system in addition to the global coordinate system, thus relocating a netlist's tile types can affect the design routing.

Design Merging

- Two XDL netlists can be merged into a single design (provided sufficient resource on the FPGA).
- In the case of disjointed netlist resources, the XDL design can be concantenated
- Ability to merge functionality of differetn FPGAs into a larger one.
- In order to merge any design, XDL takes advantage of the possiblity to assign symbolic identifier to instances.

Design Merging



Can be applied to modules as well
In order to merge the two designs:
-prefix each identifier in each design with a design name
-remove placement information from instances
-remove routing information for the nets

XDL Cons

- Different ISE versions have different adaptations of the Xilinx design flow. Errors in mapping and macro port specifications have been found in the latest release.
- No formal documentations available for new Xilinx FPGA devices
  - Changes and/or additions in XDL not directly specified (only format for each syntax is given)
- Hierarchy for macros should be avoided
  - All macros should be instantiated in the top-level design
  - When instantiating macros in a sub hierarchy, some tools might fail to implement the design

Questions?