

High-level Modeling/Simulation in Orchestra

- Design is done at higher levels of abstraction (Java and OHDL)
- Orchestra environment provides for cycle-accurate simulations
 - Enhanced visibility and debugging tools
 - Allows you to watch the condition of the configurable logic and see state transitions
- Fast, event-driven simulation engine
 - Quick debug cycles
- Significantly reduces the development time and cost

The screenshot displays the Orchestra simulation environment. On the left, a waveform viewer shows signals such as reset M6, RDn, RXFn, USBData, Enable K3, Led_Le, LED_A, LED_B, LED_C, LED_D, CLK_EN, USB_bus, USBDataIn1, USBDataIn2, and USBCount. The central panel shows the 'Reconfigurable Fabric Current State' with a diagram of a fabric and a table of future transitions. The right panel shows the 'Status' of various LEDs and a 'Continue Simula...' button. The bottom panel shows the Java code for the LEDCounter class.

```
public void setDoneLED() {
    LED();
    reset.setIcon(0);
} else {
    reset.setIcon(1+OFF);
}

public void setFadedLED(boolean en) {
    LED(en);
    enable.setIcon(jcrom);
}
```

Next State	Transition Trigger	Delay
II	Transition when USBData = 50	0 ns
III	Transition when USBData = 51	0 ns
IV	Transition when USBData = 52	0 ns
V	Transition when USBData = 68	0 ns

Output - ConfigurableLEDCounter (run)

```
Orchestra Message (Warning): Category - Bus Connection Value Conversion, Cause - Bus Value of '0' Replaced With '0'
- LEDControl.java (86): Class - LEDCounterProject.configurableLEDdevice.LEDControl, Method - enableClick

Orchestra Message (Warning): Category - Bus Connection Value Conversion, Cause - Bus Value of '0' Replaced With '0'
- LEDControl.java (86): Class - LEDCounterProject.configurableLEDdevice.LEDControl, Method - enableClick

Orchestra Message (Warning): Category - Bus Connection Value Conversion, Cause - Bus Value of '0' Replaced With '0'
- LEDControl.java (86): Class - LEDCounterProject.configurableLEDdevice.LEDControl, Method - enableClick

Orchestra Message (Warning): Category - Bus Connection Value Conversion, Cause - Bus Value of '0' Replaced With '0'
- LEDControl.java (86): Class - LEDCounterProject.configurableLEDdevice.LEDControl, Method - enableClick
```

OHDL to VHDL Translation

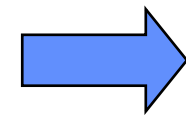
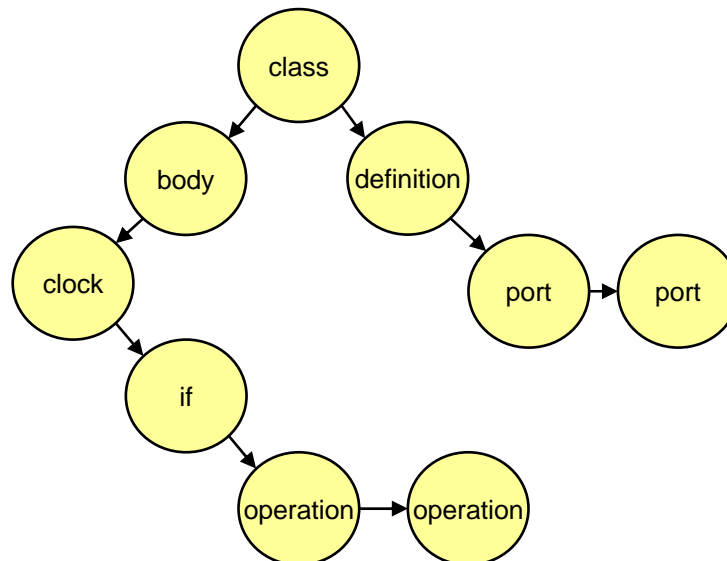
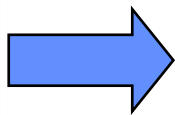
- **Modules within the reconfigurable fabric are translated to VHDL from OHDL**
 - File is parsed into a tree structure containing all the necessary information
 - Tree structure is run through a VHDL generation process to generate the file
- **OHDL and VHDL modules are behaviorally identical**
- **Hooks included to allow for translation to any other HDL or ML**

File.java

```

class Behavior {
  Behavior() {}
  Behavior(int value) {
    mValue = value;
  }
  Behavior(int value, int counter) {
    mValue = value;
    mCounter = counter;
  }
  public int getBehavior() {
    return mValue;
  }
  public void setBehavior(int value) {
    mValue = value;
  }
  public int getCounter() {
    return mCounter;
  }
  public void setCounter(int counter) {
    mCounter = counter;
  }
  public void incrementCounter() {
    mCounter++;
  }
  public void decrementCounter() {
    mCounter--;
  }
  public void resetCounter() {
    mCounter = 0;
  }
}

```



File.vhd

```

LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
USE IEEE.STD_LOGIC_ARITH.ALL;
USE IEEE.STD_LOGIC_VECTOR.ALL;

-- Used to declare KLUZE protocols (if needed)
LIBRARY KLUZE;

entity BehaviorJava is
  port (
    clk : IN STD_LOGIC;
    reset : IN STD_LOGIC;
    val : IN STD_LOGIC;
    counter : OUT STD_LOGIC_VECTOR(31 downto 0);
  );
end BehaviorJava;

architecture Behavioral of BehaviorJava is
  -- IEEE KLUZE DECLARATION
  signal behaviorCounter : STD_LOGIC_VECTOR(31 downto 0);
  signal value : STD_LOGIC_VECTOR(31 downto 0);

begin
  BEHAVIOR_PROCESS: process(clk)
  begin
    if rising_edge(clk) then
      if reset = '1' then
        behaviorCounter <= x"00000000";
      end if;
      value <= val;
      counter <= x"00000000";
    end if;
  end process;

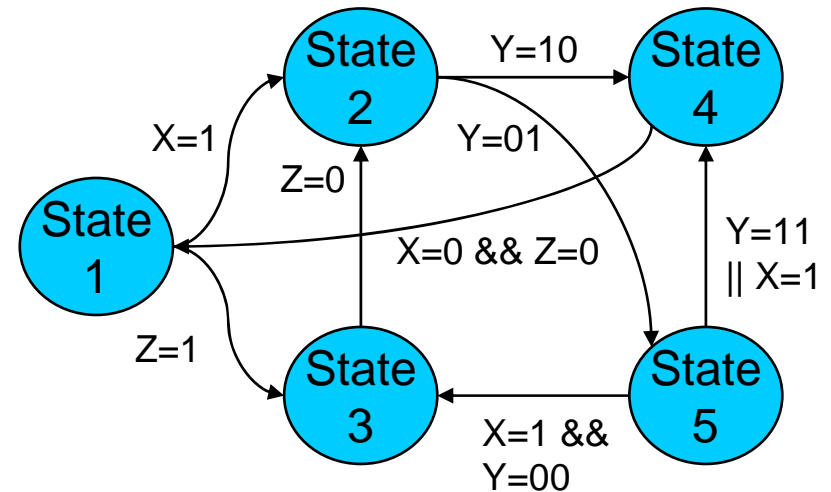
  if (behaviorCounter = x"00000000") then
    counter <= x"00000000";
  else
    counter <= counter + x"00000001";
  end if;
end Behavioral;

```

Configuration Controller Generation

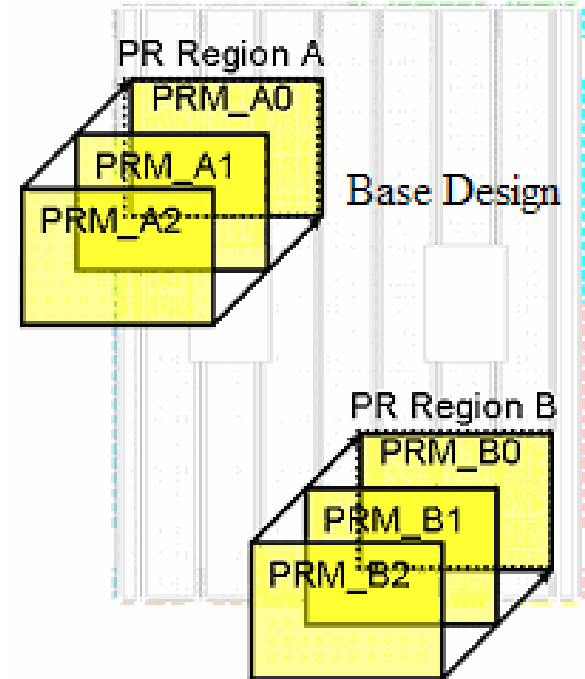
- In software, the state table must be defined
 - Determine which modules exist in which states
 - Create a unique identifier for each state as well as a priority
- Also in software, the transition table must be defined
 - What signals cause transitions from each state
 - What state to enter due to each transition
- These tables define the configuration controller and allow simulation of partial reconfiguration
- Hardware behavior during reconfiguration is user-controlled
- The tools will generate a hardware-equivalent of the configuration controller in VHDL

	Hardware Module			
State	A	B	C	D
1	0	0	1	1
2	0	1	1	0
3	1	0	0	1
4	1	1	0	0
5	0	1	0	0



Region Partitioning/Floorplanning

- Knowing the state table, the tools optimally allocate sets of hardware modules to share distinct regions
 - Requirements:
 - Modules that share a region must not exist at the same time in the state table
 - Modules that share an output port must exist in the same region
- Once the regions have been partitioned, they must be physically constrained onto the chip
 - Define the size, shape, and placement of the region
 - Must be large enough to accommodate the biggest module
 - Must be shaped and placed to allow the design to meet timing
 - Must obey internal FPGA placement restrictions
 - Frame Boundaries
 - Embedded I/O
 - RAMB16/DSP48 Column Boundaries
 - Must also locate and constrain bus macros





Partial Bitstream Generation

- Vendor/FPGA specific step
- Special PR Service Pack
 - Early access flow not available to the general public at this time
- Vendor design flow accessed through generated script files
- The tools generate:
 - Partial bitstreams for each reconfigurable module
 - Total bitstreams for each individual state
 - Blanking bitstreams for each reconfigurable region
- Timing is checked for each individual state
- Bitstreams are parsed and downloaded to a bank of non-volatile flash memories