Modeling Hardware in Java

William_Kaupinis@eightolives.com
Jan 25, 2010

Abstract

A method of describing hardware design has been developed using the Java software language. The resulting Java package, com.eightolives.Hardware, and its application programming interface (API) provides a framework allowing other software to capture, modify, analyze, simulate and synthesize the design.

Why Java?

- Java is an established object-oriented language with excellent tool, library, and industry support
- Java compiles to byte-code which can execute on a variety of hardware and operating system platforms without change
- Works with other Java tools from eightolives
  - Workspaces Desktop
  - DesignToolLite

Requirements

- The model should be consistent with hardware descriptions in VHDL
- Models support simulation

Key Classes

- DesignObject is a base class that stores name, attributes, comments
- Element is a sub-class similar to a VHDL entity
  - Contains I/O Ports similar to a VHDL Port
  - Ports have a connected signal and an internal signal
- Architecture is a sub-class similar to a VHDL architecture
  - It contains Signals and instances of Elements

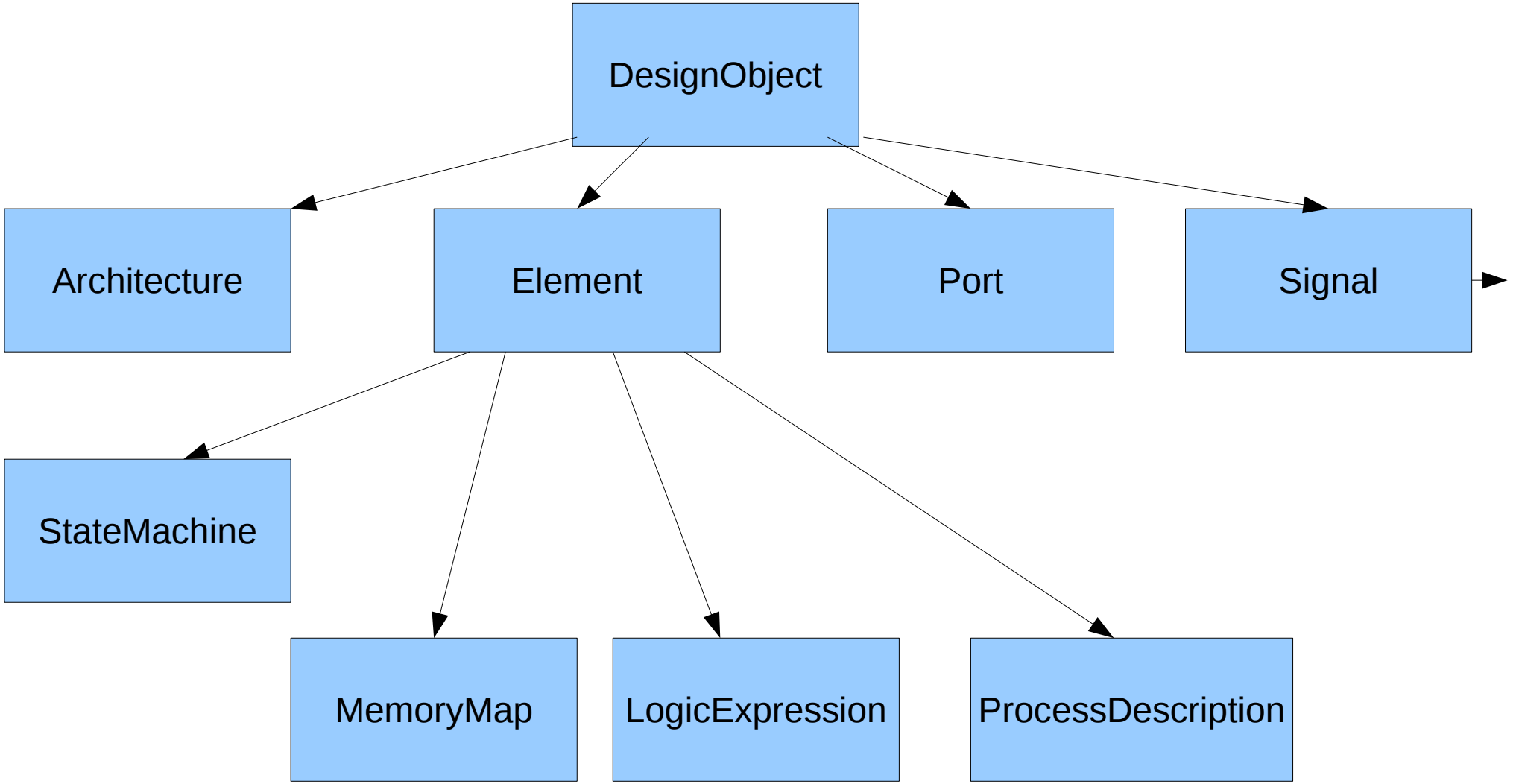
Special Elements

- StateMachine contains StateConditions and StateActions that represent a state machine
- MemoryMap contains MapEntries that define address decoding spaces for reading or writing
- ProcessDescription captures a VHDL process as an Element block
- LogicExpression captures a logic equation as an Element

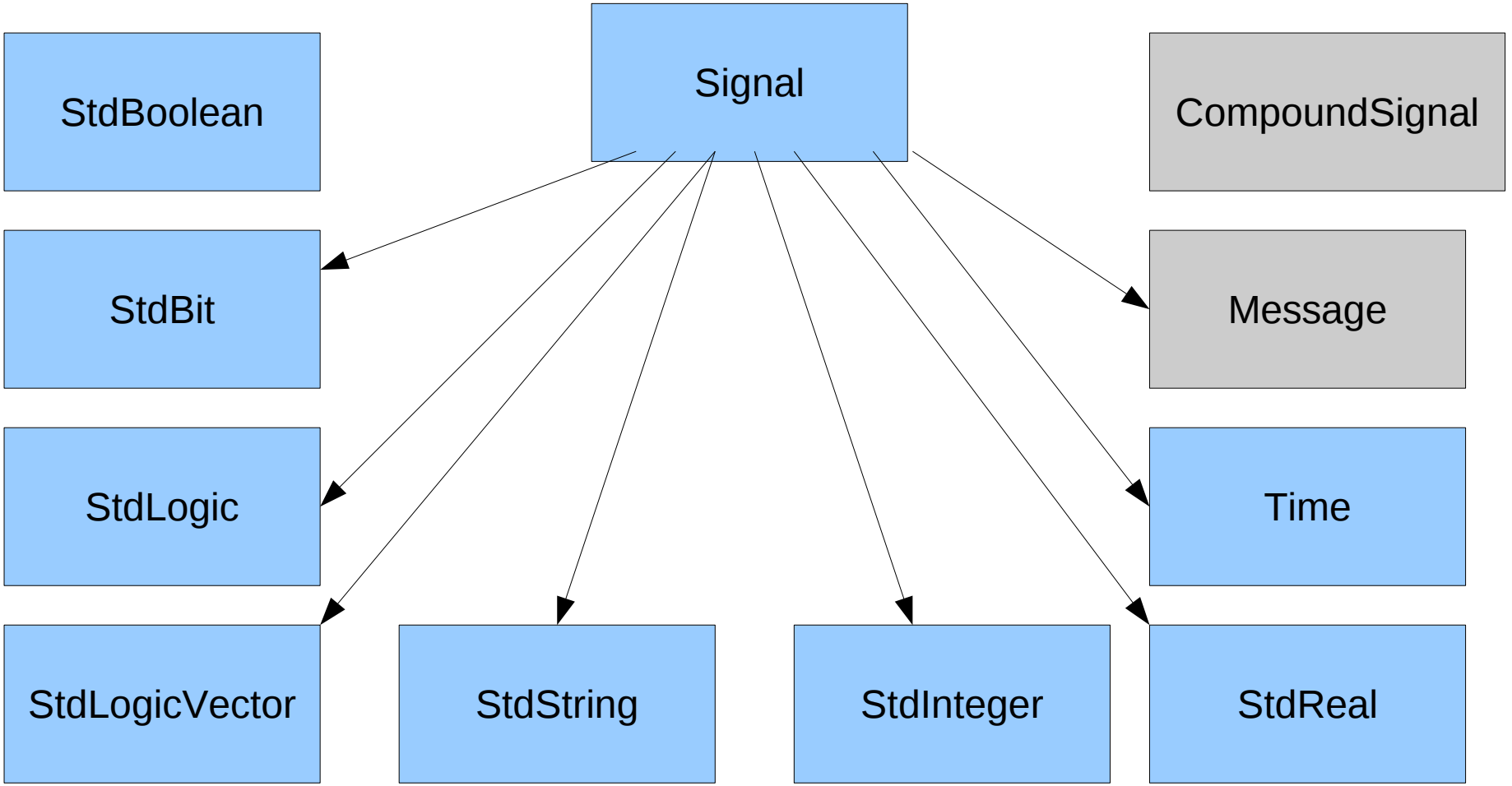
Flavors of Signal

- Signal represents an interconnection of Ports and is a DesignObject that has several sub-classes
  - StdBit
  - StdLogic
  - StdLogicVector
  - StdBoolean
  - StdInteger – Java long
  - StdReal – Java Double
  - StdString
  - Time
  - Message – not implemented

Class Structure

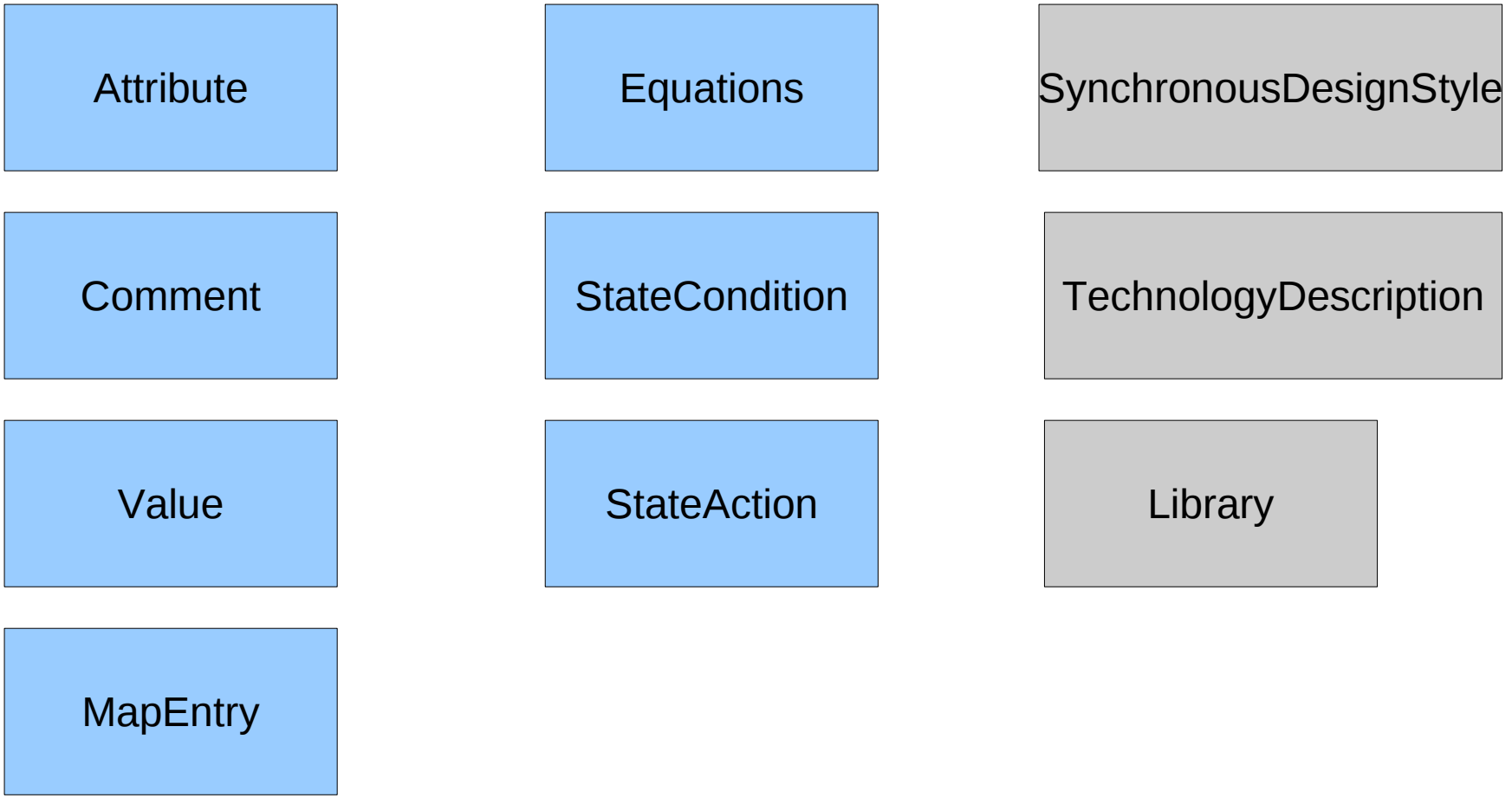


Signal Sub-Classes



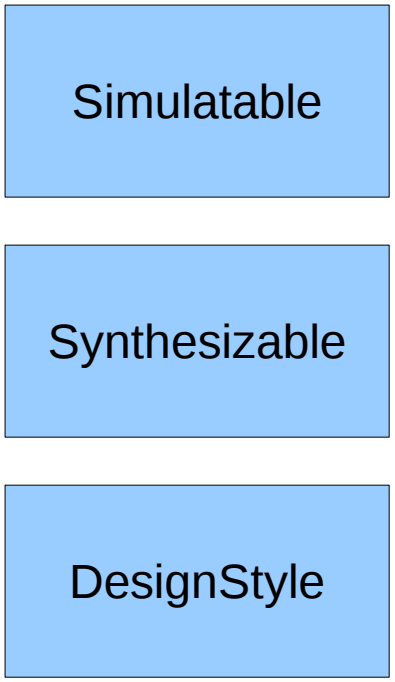
Note: Grayed items not complete in Version 1

Other Classes



Note: Grayed items not complete in Version 1

Interfaces



Some Example Methods

- Element
  - addPort
  - addNewStdLogicSignalPort
  - addArchitecture
  - addGeneric
  - getPort
  - getLabel
  - getInstance (clone)
  - getParent
  - flatten
  - union
  - intersect
  - uniinitialize
- Architecture
  - addElement
  - addInstance
  - addExpression
  - addSignal
  - addUniqueSignal
  - connect
  - getElementByLabel
  - getSignal
  - getSignals
  - getElements
  - getExpressions
  - removeSignal
  - removeElement

GET methods typically have SET counterparts

Some Example Methods

- Port
  - addConnection
  - getConnectedSignal
  - isClock
  - getClockPeriod
  - getClockReference
  - getSetupTime, getHoldTime
  - getDelayTime
  - getIOH, getIOL
  - getIIH, getIIL
  - getPinNumber
  - reduceToSignal
- Signal
  - addConnection
  - isClock
  - elevateToPort
  - setTo
  - endSimTick
  - evaluate
  - getSimulationQueue
  - getInitialValue
  - getParent

GET methods typically have SET counterparts

Some Example Methods

- StdLogic
  - nand
  - nor
  - and
  - or
  - not
  - xor
  - xnor
  - to-01x
- StdLogicVector
  - hasRange
  - getLimit1
  - getLimit2
  - getDirection
  - getSubSignals
  - getSubSignal

Using the Package

- The Workspaces Desktop and DesignToolLite tools use the Hardware package to store designs input from HDL code or user GUI interaction
  - The tools allow Javascript script or interactive access to the API
- You can create Java models of hardware such as model part libraries for simulation or for models for hardware/software co-simulation

Example Model 2 Input NAND Gate (1 of 2)

```
// nand2.java

import com.eightolives.Hardware.*;


public class nand2 extends
com.eightolives.Hardware.Element

{

private Architecture a = null;

private StdLogic A = null;

private StdLogic B = null;

private StdLogic Y = null;

private Time Td = null;

// internal variables

private char iY = 'U';
```

```
/** The constructor.

*/

public nand2()

{

super();

setName("nand2");

setComment("2 input NAND gate ");

initInterfaces();

initArchitecture();

Attribute att = new
Attribute("PRIMITIVE", "true");

setAttribute(att);

} // end of constructor
```

Example Model 2 Input NAND Gate (2 of 2)

```
public void initInterfaces()
{
Td = new Time( "Td", "3 ns");
Td.setComment("propagation delay");
addGeneric(Td);
A = addNewStdLogicSignalPort("A", Port.IN, " ");
B = addNewStdLogicSignalPort("B", Port.IN, " ");
Y = addNewStdLogicSignalPort("Y", Port.OUT, " ");
}

public void initArchitecture()
{
a = addNewArchitecture( "primitive");
}
```

```
public void uninitialize(SimulationQueue sq)
{
super.uninitialize(sq);
// initiatialize any internal variables here
iY = 'U';
}

public void execute()
{
iY = A.nand(B);
Y.setTo(iY, Td);
}

} // end of class
```

Summary

- A Java software package has been created that represents hardware for purposes of design analysis, expression and simulation
- The package is used in the eightolives' Workspaces Desktop and DesignToolLite tools.