Digital Logic Circuit Simulator

Group Members:
Meghanad Shingate (09307608)
Nirbhay Rane (09307905)
Bharat Kumar (09307904)

Group : 13
AE663 Final Presentation
(Google code repository - http://code.google.com/p/pydlcs/)

November 28, 2011

Problem Statement

Implementation of Logic Circuit Simulator using Python

- Basic logic gates (e.g. AND, OR, NOR, etc.)
- Combinational Logic circuits (e.g. Half adder, Full adder, Mux/Demux etc.)
- Sequential Logic circuits (e.g Flip-Flop, Counters, Shift registers etc.)
- Digital signal sources (e.g. clock source)
- Simulation of any circuit designed using above elements

Implementation Details - Class Structure

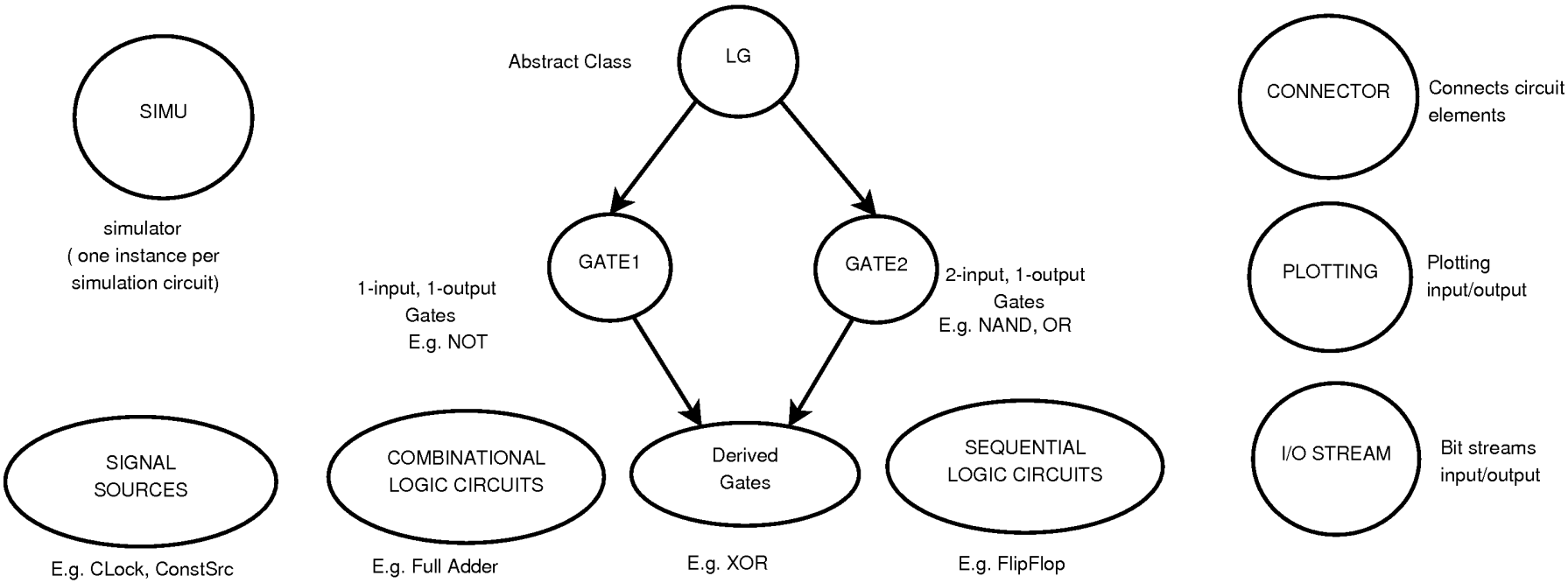

Figure: Class Structure

Implementation Details - Simulator Model

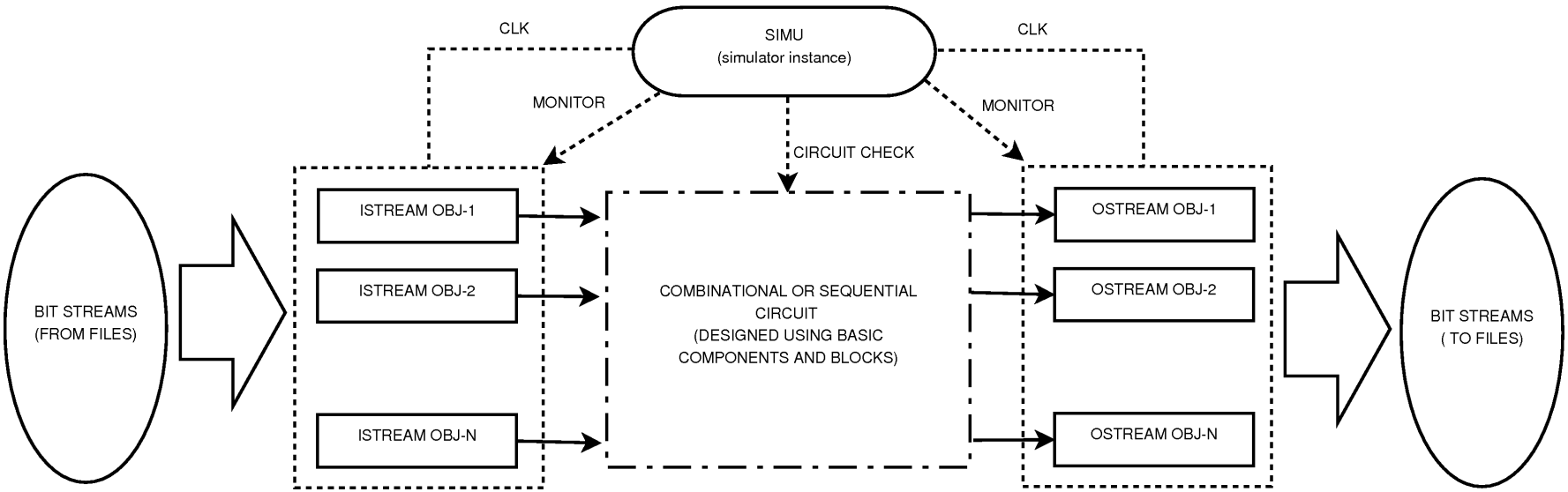

Figure: Simulator model

Implementation Details - Simulator Model

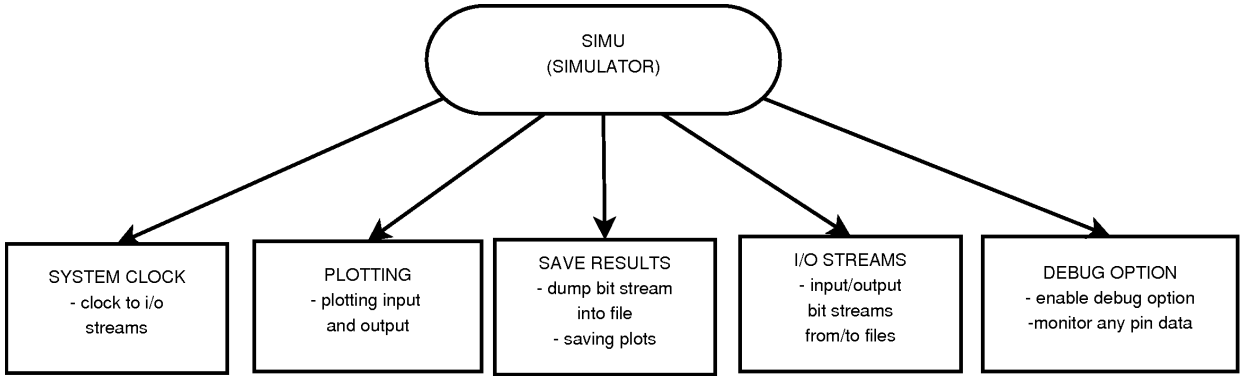

Figure: *SIMU* Class details

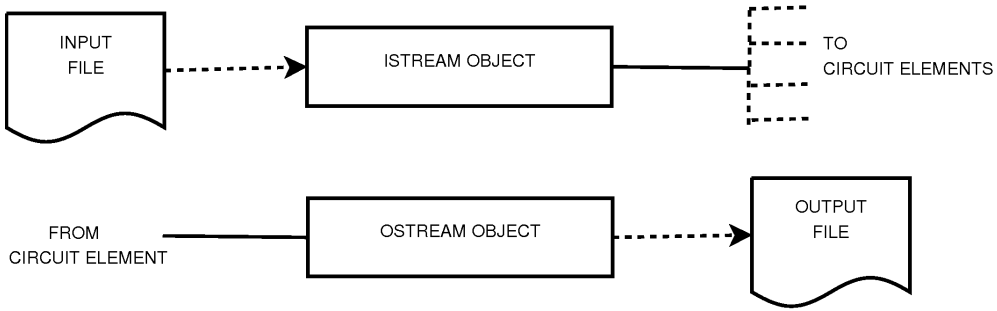

Figure: *i/o stream* model

Coding Details - Basic Gates

Logic Gate Class - Abstract Class

```
1 # CLASS : LOGIC GATE (LG)
2 class LG :
3     def __init__ (self, name) :
4         self.name = name
5     def evaluate (self) : return
```

Coding Details - Basic Gates

Logic Gate Class - Abstract Class

```
1 # CLASS : LOGIC GATE (LG)
2 class LG :
3     def __init__ (self, name) :
4         self.name = name
5     def evaluate (self) : return
```

AND Gate

```
1  # ==================================
2  # CLASS : GATE2 (2 INPUTS, 1 OUTPUT GATES)
3  #==================================
4  class Gate2 (LG) :          # two input gates. Inputs A and B. Output C.
5      def __init__ (self, name) :
6          LG.__init__ (self, name)
7          self.A = Connector(self,'A', activates=1)
8          self.B = Connector(self,'B', activates=1)
9          self.C = Connector(self,'C')
10
11 # ============================
12 # CLASS : AND GATE (And)
13 # ============================
14 class And (Gate2) :        # two input AND Gate
15     def __init__ (self, name) :
16         Gate2.__init__ (self, name)
17     def evaluate (self) : self.C.set(self.A.value and self.B.value)
```

Coding Details - Derived Gates

XOR Gate

```
1  # =========================
2  # CLASS : XOR GATE (Xor)
3  # =========================
4  class Xor (Gate2) :
5      def __init__ (self, name) :
6          Gate2.__init__ (self, name)
7          self.A1 = And("A1") # See circuit drawing to follow connections
8          self.A2 = And("A2")
9          self.I1 = Not("I1")
10         self.I2 = Not("I2")
11         self.O1 = Or ("O1")
12         self.A.connect    ([ self.A1.A, self.I2.A])
13         self.B.connect    ([ self.I1.A, self.A2.A])
14         self.I1.B.connect ([ self.A1.B ])
15         self.I2.B.connect ([ self.A2.B ])
16         self.A1.C.connect ([ self.O1.A ])
17         self.A2.C.connect ([ self.O1.B ])
18         self.O1.C.connect ([ self.C ])
```

Coding Details - Sequential Element

T Flip Flop

```
1  # ================
2  # T - FlipFlop
3  # ================
4
5  class TFlipFlop (LG):
6
7   def __init__(self, name):
8    LG.__init__(self,name)
9    self.T = Connector(self,'T')
10   self.Q = Connector(self,'Q')
11   self.C = Connector(self,'C', activates = 1)
12   self.Q.value = 0
13   self.prev = 0
14
15  def evaluate (self):
16   if (not self.C.value) and self.prev and self.T.value: # clock drop and T=1
17    self.Q.set(not self.Q.value)
18   self.prev = self.C.value
```

Example Circuit Simulation - Circuit

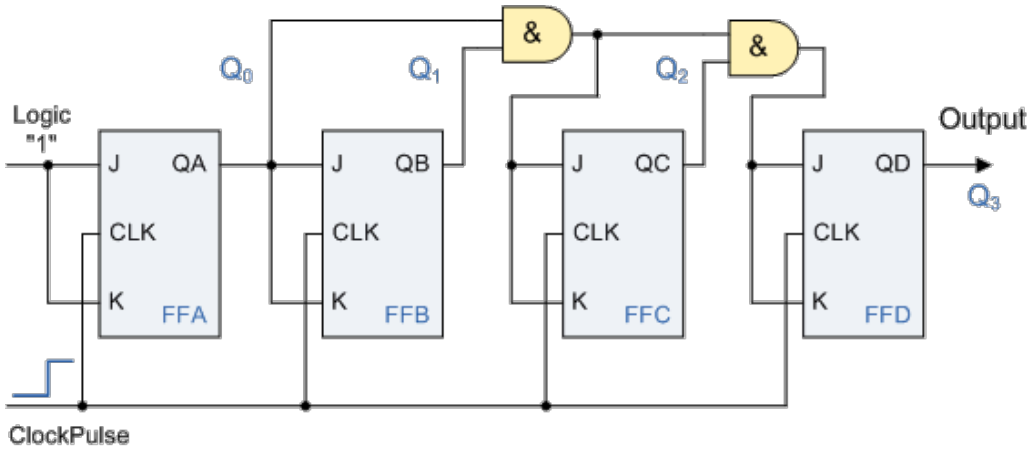

Figure: Example - 4 Bit synchronous binary counter

Example Circuit Simulation - Code

```
# A 4 Bit Syncronous Binary Counter using JK Flip Flpos
from libpydlcs import *

sim = SIMU('sim1',start = 0,plots = 1, debug =1, pclk = 1 , step =0, clocks=33)

C0 = ConstSrc('CS',value =1)

O1 = Ostream('OUT1', stream = 1)
O2 = Ostream('OUT2', stream = 1)
O3 = Ostream('OUT3', stream = 1)
O4 = Ostream('OUT4', stream = 1)

JK1 = JKFlipFlop('JK1')
JK2 = JKFlipFlop('JK2')
JK3 = JKFlipFlop('JK3')
JK4 = JKFlipFlop('JK4')

A1 = And('A1')
A2 = And('A2')

sim.clk_out.connect([JK4.C, O4.clk_in, JK3.C, O3.clk_in, JK2.C, O2.clk_in,C0.clk_in, JK1.C, O1.clk_in])
C0.data_out.connect([JK1.J, JK1.K])
JK1.Q.connect([JK2.J, JK2.K, A1.A, O1.data_in])
JK2.Q.connect([A1.B, O2.data_in])
A1.C.connect([JK3.J, JK3.K, A2.A])
JK3.Q.connect([A2.B, O3.data_in])
A2.C.connect([JK4.J, JK4.K])
JK4.Q.connect([O4.data_in])

sim.addplot([O1.data, O2.data, O3.data, O4.data])
sim.addpname(["Q1", "Q2", "Q3", "Q4"])

sim.start = 1
sim.simulate()
```

Example Circuit Simulation - Plots

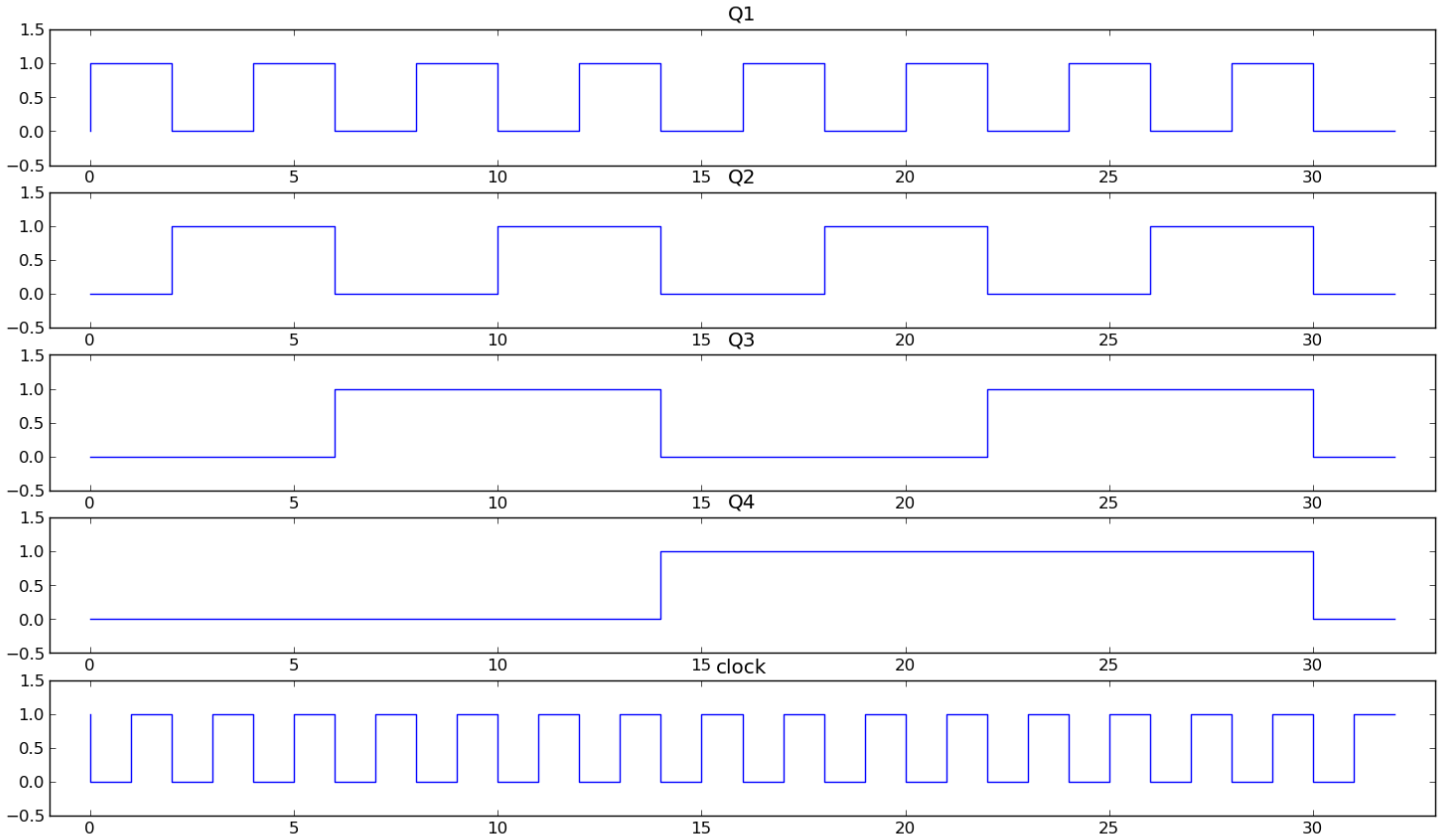

Figure: Example - 4 Bit synchronous binary counter plot

Project Status & future work

Project Status

- Project status - Complete as per proposal
- Work hours put - approximately 25 hours per person

Project Status & future work

Project Status

- Project status - Complete as per proposal
- Work hours put - approximately 25 hours per person

Future Work

- GUI is not provided completely, so this could be potential future work.
- Feedback that activates the evaluation function is not possible in current release, this problem could be rectified.

Thank You !