Object-Oriented Programming
Homework#2 review

CSIE Department, NTUT
Woei-Kae Chen
UML Review

- Association (knows a)
- Dependency (uses a)
- Composition (has a)
- Aggregation (has a)
- Inheritance (is a)
- Class template
Homework #2 Class Diagram 1

- Works, but...
  - difficult to extend

Necessary: keep main() clean.
Better, but…
– How to construct circuits?

Abstract class

**simulator**
- circuit : vector<gate *>
  + loadCircuit()
  + simulate()

Must be pointer

vector class is not shown
OK for circuits, but…
– What about I/O pins of the circuit?

Use abstract class to simplify drawing

Separate getOutput() and simulate() for efficiency

Association

Pure
Controller ⇔ Coordinator
- Controller: one class (simulator) do all the work
- Coordinator: give responsibilities to other classes (gates)

Why NOT `vector<int>`?

<table>
<thead>
<tr>
<th>simulator</th>
</tr>
</thead>
<tbody>
<tr>
<td>- circuit : vector&lt;gate *&gt;</td>
</tr>
<tr>
<td>+ loadCircuit()</td>
</tr>
<tr>
<td>+ simulate()</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>gate</th>
</tr>
</thead>
<tbody>
<tr>
<td>- ipin : vector&lt;int&gt;</td>
</tr>
<tr>
<td>+ addInputPin()</td>
</tr>
<tr>
<td>+ getOutput()</td>
</tr>
<tr>
<td>+ simulate()</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>gateOR</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>gateAND</th>
</tr>
</thead>
</table>

| gateNOT |
Homework #2 Class Diagram 4

Separate I/O pins from circuit

For iPin only

Non-pure

oPin may be omitted

simulator
- ipin : vector<gate *>
- olin : vector<gate *>
- circuit : vector<gate *>
+ loadCircuit()
+ simulate()

 gate
- ipin : vector<gate *>
+ addInputPin()
+ getOutput()
+ simulate()
+ setInput()

OR

iPin

oPin

 gateOR

gateAND

gateNOT
Works, but…

- iP in “is a” gate?
- oPin “is a” gate?

Concrete class

For iP in only
A good choice
- Works for now, but…
This is a quick and dirty fix

Use device instead of gate
Homework #2 Class Diagram 7

- Structurally better, but...
  - May be over killing for now

```
simulator
- ipin : vector<IOPin *>
- opin : vector<IOPin *>
- circuit : vector<gate *>
+ loadCircuit()
+ simulate()
```

```
device
- ipin : vector<device *>
+ addInputPin()
+ getOutput()
+ simulate()
```

```
IOPin
```

```
gate
```

```
iPin
```

```
oPin
```

```
gateOR
gateAND
gateNOT
```

Pure