The FHDL System

Simulation and Specification

FHDL

Florida

Hardware

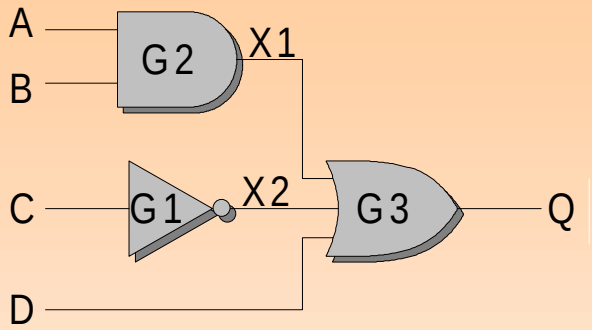
Design

Language

Components

- Gate-Level Simulation
  - Simple Gates
  - Functional Blocks
  - Rudimentary ROM & PLA Support
  - Hierarchical Circuits
  - Basic State Machine Support
- High-Level ROM Specifications
- High-Level PLA Specifications
- Macros

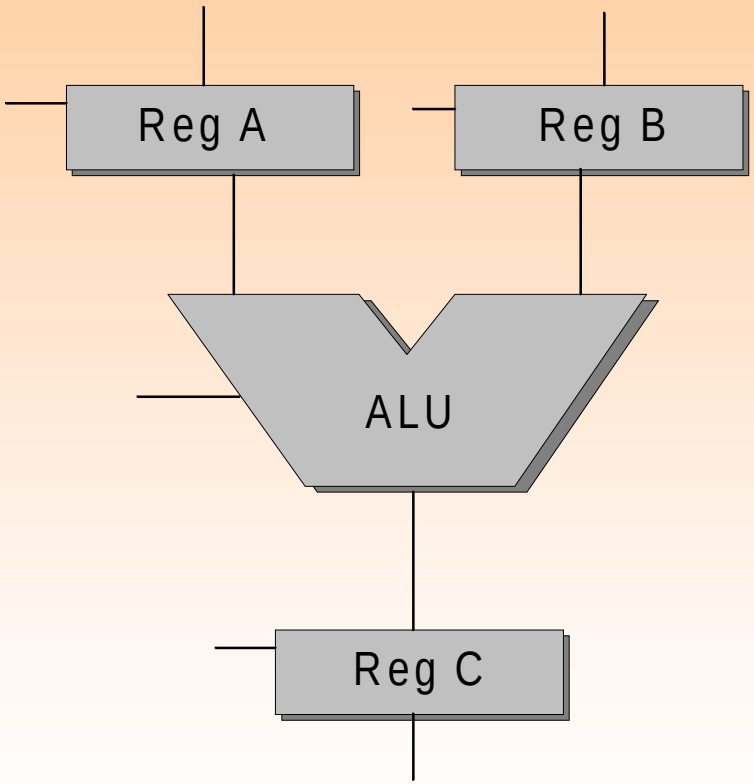
Gate-Level Specifications



Example: circuit

	inputs	A,B,C,D
	outputs	Q
G1:	and	(A,B),X1
G2:	not	C,X2
G3	or	(X1,X2,D),Q
	endcircuit	

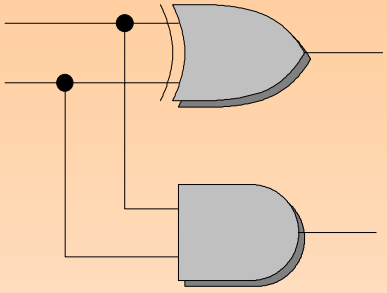
Functional Blocks: Diagram



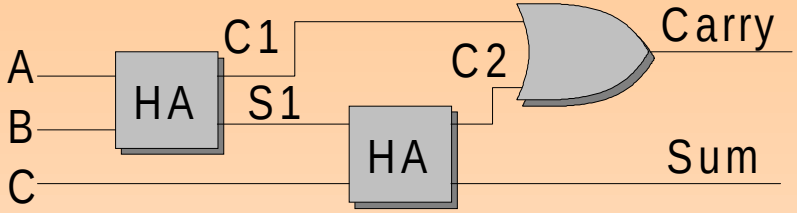
Functional Blocks: FHDL

```
Sample:     circuit
            inputs      InputA,LoadA,InputB,LoadB,ALUControl,LoadC,Clock
            outputs     OutputC

RegA:       register    (InputA,LoadA,Clock),(ALULeft),clock=yes
            wire        InputA,width=32
            wire        ALULeft,width=32
RegB:       register    (InputB,LoadB,Clock),(ALURight),clock=yes
            wire        InputB,width=32
            wire        ALURight,width=32
MainALU:    alu         (ALULeft,ALURight,ALUControl),(InputC),
                        control=(carryin,cenable),
                        function=(0,and,4,or,8,xor)
            wire        ALUControl,width=6
RegC:       register    (InputC,LoadC,Clock),(OutputC),clock=yes
            wire        InputC,width=32
            wire        OutputC,width=32

            endcircuit
```

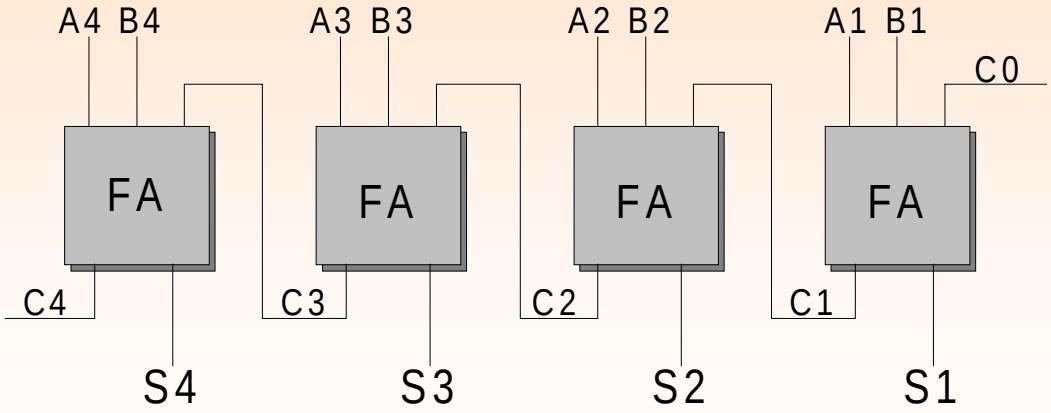
Hierarchical Specifications



Half Adder



Full Adder



Four-Bit Adder

Hierarchical FHDL

```
HalfAdder: circuit
           Inputs     A,B
           Outputs    S,C
G1:        xor        (A,B),S
G2:        and        (A,B),C
           endcircuit
```

```
FullAdder: circuit
           Inputs     A,B,C
           Outputs    Sum,Carry
SS1:       HalfAdder  (A,B),(S1,C1)
SS2:       HalfAdder  (C,S1),(Sum,C2)
G3:        or         (C1,C2),Carry
           endcircuit
```

```
FourBit:   circuit
           Inputs     A4,A3,A2,A1,B4,B3,B2,B1,C0
           Outputs    S4,S3,S2,S1,C4
SS1:       FullAdder  (A1,B1,C0),(S1,C1)
SS2:       FullAdder  (A2,B2,C1),(S2,C2)
SS3:       FullAdder  (A3,B3,C2),(S3,C3)
SS4:       FullAdder  (A4,B4,C3),(S4,C4)
           endcircuit
```

High-Level ROM Code

```
start:      cont     loadr,loadd,loadq,loadl,clrlast,muxin
            cont     0->ovflval,loadovfl,loadctr
            cjump    hibit0,setovfl
repeat:     cont     shift
            cont     decr
            cont     loadr,loadl,loadlast
            cjump    ctrnz,repeat
            cjump    lastz,halt
            cont     loadr
            jump     halt
setovfl: cont        1->ovflval,loadovfl
halt:       jump     halt,jobdone
```

High-Level PLA Code

```
abx:      pla
i1:       field    type=input,position=0,width=3
i2:       field    type=input,position=3,width=1
a:        field    position=0,width=3
i1=2:     word     7->a
i1=3&i2=1: word 6->a
i1=1|i1=5: word          2->a
i2=0:     word     3->a
          endpla
```

Winfhdl

- Interactive, Runs on MS Windows 3.1
- Runs on Windows 95
- Full-Featured Text Editor
- Graphical Schematic Editor
- Based on VBX FHDL Components
- Easily Expandable
- Latest User Interface Do-Dads
- Full Driver Language Support

WinFHDL Teaching Techniques

- Use the "Include" feature of ROM language
  - Define your own microprogramming language
  - Automatically include your sequencer
- Use the Standard Cell Libraries
  - Define your own standard cells: e.g. MSI Comp.
  - Define macros for more complex cells
  - Define graphical cells for schematic capture
- Save as VHDL for other tools

The FHDL System

Simulation and Specification

FHDL

Florida

Hardware

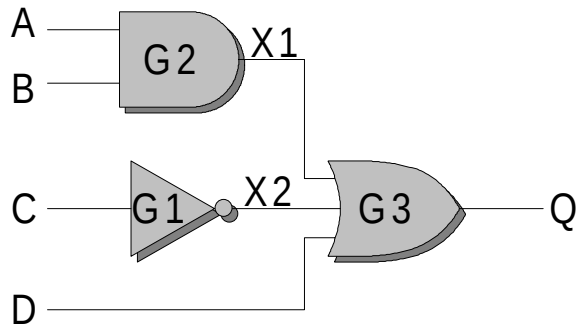
Design

Language

Components

- Gate-Level Simulation
  - Simple Gates
  - Functional Blocks
  - Rudimentary ROM & PLA Support
  - Hierarchical Circuits
  - Basic State Machine Support
- High-Level ROM Specifications
- High-Level PLA Specifications
- Macros

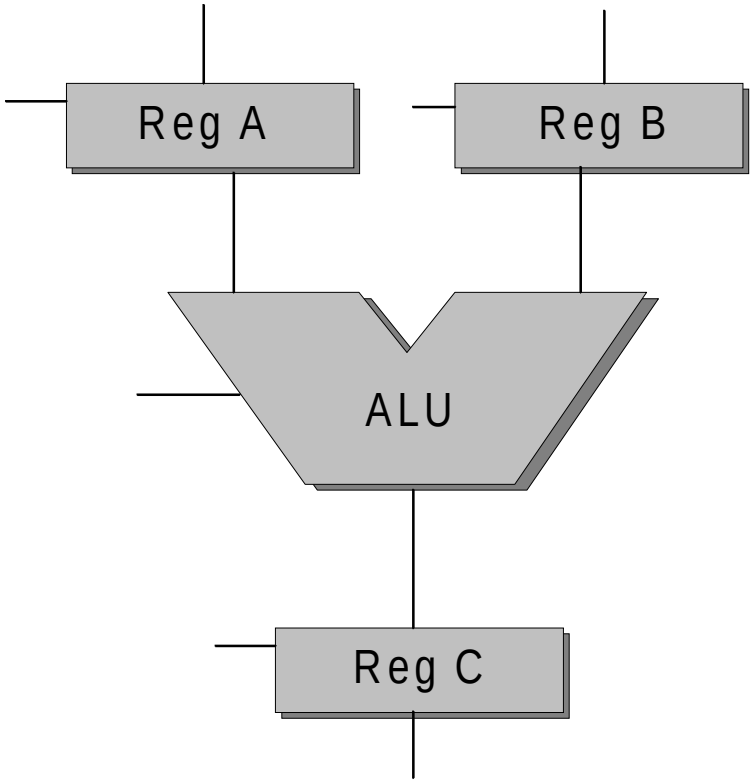
Gate-Level Specifications



Example: circuit

	inputs	A,B,C,D
	outputs	Q
G1:	and	(A,B),X1
G2:	not	C,X2
G3	or	(X1,X2,D),Q
	endcircuit	

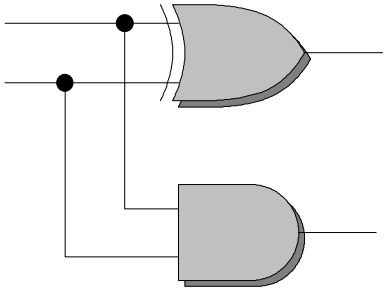
Functional Blocks: Diagram



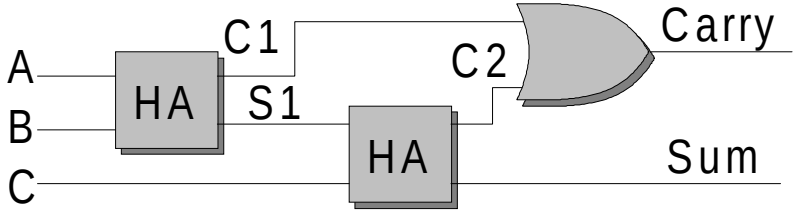
Functional Blocks: FHDL

```
Sample:     circuit
            inputs      InputA,LoadA,InputB,LoadB,ALUControl,LoadC,Clock
            outputs     OutputC

RegA:       register    (InputA,LoadA,Clock),(ALULeft),clock=yes
            wire        InputA,width=32
            wire        ALULeft,width=32
RegB:       register    (InputB,LoadB,Clock),(ALURight),clock=yes
            wire        InputB,width=32
            wire        ALURight,width=32
MainALU:    alu         (ALULeft,ALURight,ALUControl),(InputC),
                        control=(carryin,cenable),
                        function=(0,and,4,or,8,xor)
            wire        ALUControl,width=6
RegC:       register    (InputC,LoadC,Clock),(OutputC),clock=yes
            wire        InputC,width=32
            wire        OutputC,width=32

            endcircuit
```

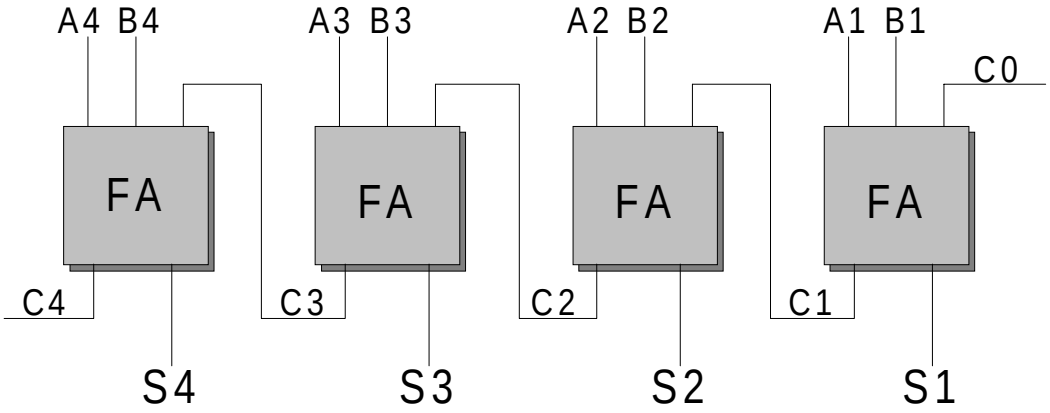
Hierarchical Specifications



Half Adder



Full Adder



Four-Bit Adder

Hierarchical FHDL

```
HalfAdder: circuit
           Inputs     A,B
           Outputs    S,C
G1:        xor        (A,B),S
G2:        and        (A,B),C
           endcircuit
```

```
FullAdder: circuit
           Inputs     A,B,C
           Outputs    Sum,Carry
SS1:       HalfAdder  (A,B),(S1,C1)
SS2:       HalfAdder  (C,S1),(Sum,C2)
G3:        or         (C1,C2),Carry
           endcircuit
```

```
FourBit:   circuit
           Inputs     A4,A3,A2,A1,B4,B3,B2,B1,C0
           Outputs    S4,S3,S2,S1,C4
SS1:       FullAdder  (A1,B1,C0),(S1,C1)
SS2:       FullAdder  (A2,B2,C1),(S2,C2)
SS3:       FullAdder  (A3,B3,C2),(S3,C3)
SS4:       FullAdder  (A4,B4,C3),(S4,C4)
           endcircuit
```

High-Level ROM Code

```
start:      cont     loadr,loadd,loadq,loadl,clrlast,muxin
            cont     0->ovflval,loadovfl,loadctr
            cjump    hibit0,setovfl
repeat:     cont     shift
            cont     decr
            cont     loadr,loadl,loadlast
            cjump    ctrnz,repeat
            cjump    lastz,halt
            cont     loadr
            jump     halt
setovfl: cont        1->ovflval,loadovfl
halt:       jump     halt,jobdone
```

High-Level PLA Code

```
abx:     pla
i1:      field    type=input,position=0,width=3
i2:      field    type=input,position=3,width=1
a:       field    position=0,width=3
i1=2:    word     7->a
i1=3&i2=1: word 6->a
i1=1|i1=5: word          2->a
i2=0:    word     3->a
         endpla
```

Winfhdl

- Interactive, Runs on MS Windows 3.1
- Runs on Windows 95
- Full-Featured Text Editor
- Graphical Schematic Editor
- Based on VBX FHDL Components
- Easily Expandable
- Latest User Interface Do-Dads
- Full Driver Language Support

WinFHDL Teaching Techniques

- Use the “Include” feature of ROM language
  - Define your own microprogramming language
  - Automatically include your sequencer
- Use the Standard Cell Libraries
  - Define your own standard cells: e.g. MSI Comp.
  - Define macros for more complex cells
  - Define graphical cells for schematic capture
- Save as VHDL for other tools