Tassadaq Hussain

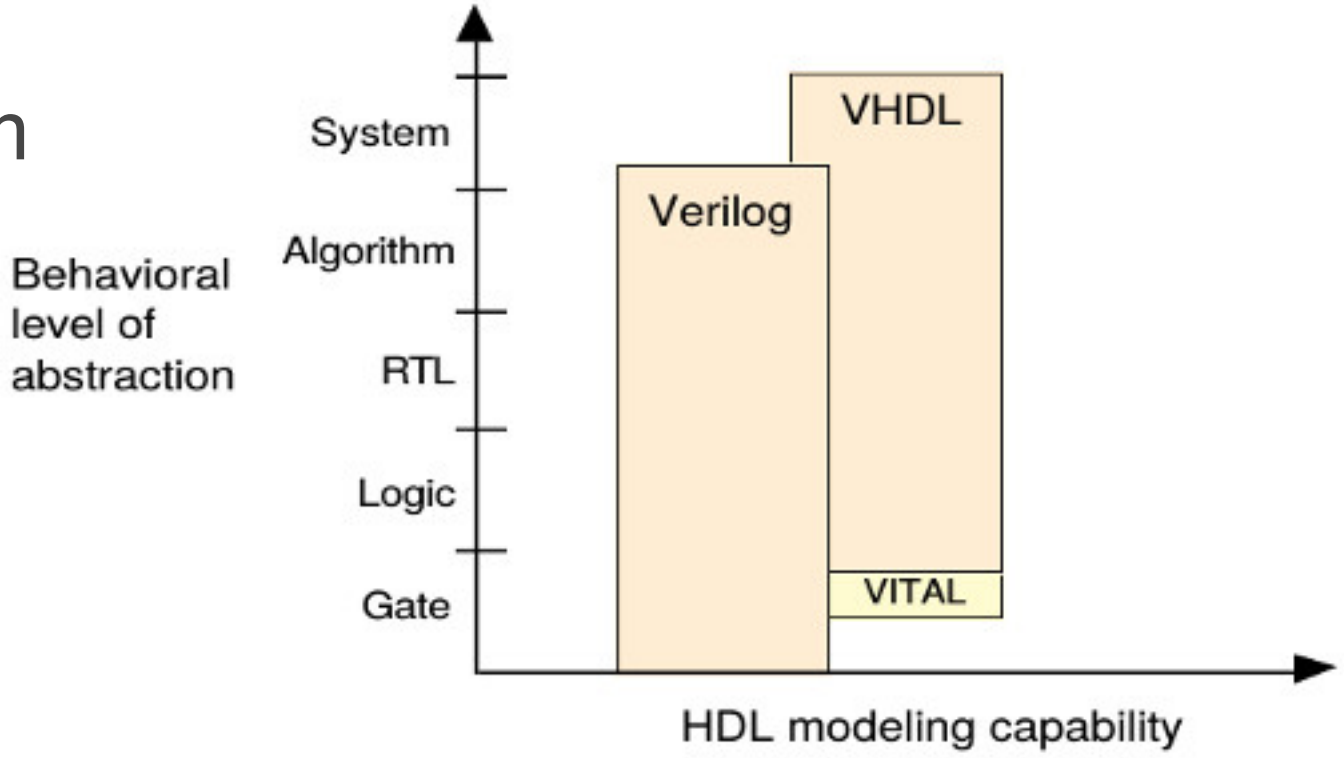
VLSI Design

Gate Level Modeling

Outline

Verilog

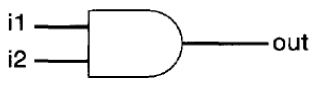
Modelsim



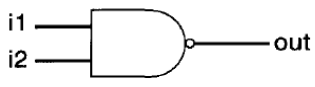
Objectives

- Identify logic gate primitives provided in Verilog.
- Understand instantiation of gates, gate symbols and truth tables for and/or and buf/not type gates.
- Understand how to construct a Verilog description from the logic diagram of the circuit.
- Describe rise, fall, and turn-off delays in the gate-level design.
- Explain min, max, and type delays in the gate-level design.

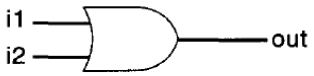
Gate Level Modeling



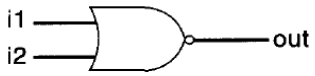
and



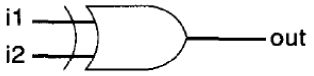
nand



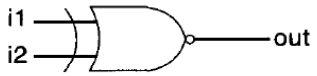
or



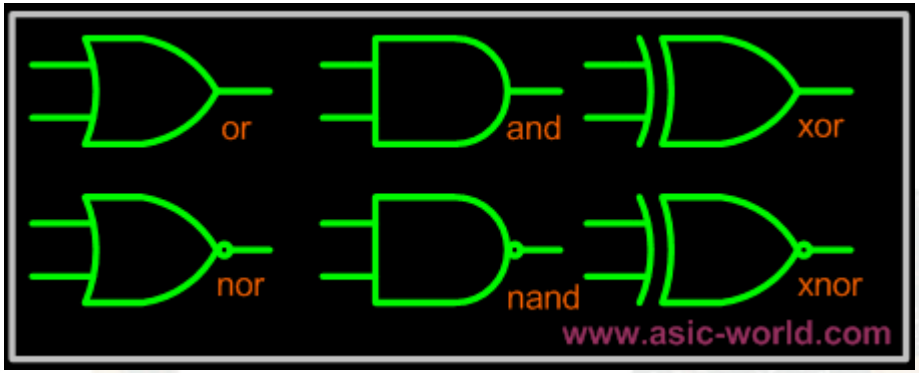
nor



xor



xnor




```
and       or       xor
nand      nor      xnor
```

Basic Gates Instantiation

```
and a1(OUT, IN1, IN2);
nand na1(OUT, IN1, IN2);
or or1(OUT, IN1, IN2);
nor nor1(OUT, IN1, IN2);
xor x1(OUT, IN1, IN2);
xnor nx1(OUT, IN1, IN2);
```

Types of Gates

And/Or Gates

Buff/Not

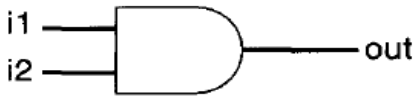
Logic Tables for NMOS and PMOS

nmos data	control 0	1	x	z
0	z	0	L	L
1	z	1	H	H
x	z	x	x	x
z	z	z	z	z

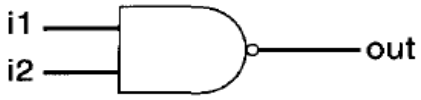
pmos data	control 0	1	x	z
0	0	z	L	L
1	1	z	H	H
x	x	z	x	x
z	z	z	z	z

And/Or Gates

```
and       or       xor
nand      nor      xnor
```

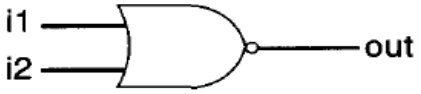


and



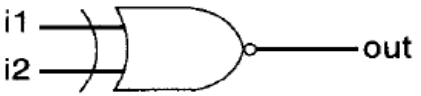
nand

or



nor

xor



xnor

Basic Gates

```
// basic gate instantiations.
and a1(OUT, IN1, IN2);
nand na1(OUT, IN1, IN2);
or or1(OUT, IN1, IN2);
nor nor1(OUT, IN1, IN2);
xor x1(OUT, IN1, IN2);
xnor nx1(OUT, IN1, IN2);

// More than two inputs; 3 input nand gate
nand na1_3inp(OUT, IN1, IN2, IN3);

// gate instantiation without instance name
and (OUT, IN1, IN2); // legal gate instantiation
```

Truth Table of Gates

and	i1 = 0	i1 = 1	i1 = x	i1 = z
i2 = 0	0	0	0	0
i2 = 1	0	1	x	x
i2 = x	0	x	x	x
i2 = z	0	x	x	x

nand	i1 = 0	i1 = 1	i1 = x	i1 = z
i2 = 0	1	1	1	1
i2 = 1	1	0	x	x
i2 = x	1	x	x	x
i2 = z	1	x	x	x

or	i1 = 0	i1 = 1	i1 = x	i1 = z
i2 = 0	0	1	x	x
i2 = 1	1	1	1	1
i2 = x	x	1	x	x
i2 = z	x	1	x	x

nor	i1 = 0	i1 = 1	i1 = x	i1 = z
i2 = 0	1	0	x	x
i2 = 1	0	0	0	0
i2 = x	x	0	x	x
i2 = z	x	0	x	x

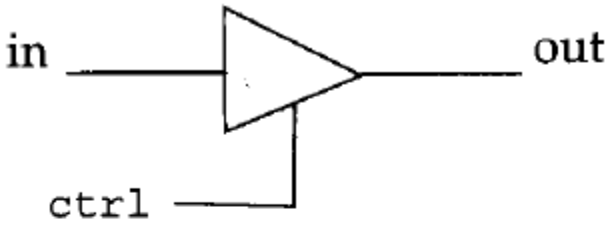
xor	i1 = 0	i1 = 1	i1 = x	i1 = z
i2 = 0	0	1	x	x
i2 = 1	1	0	x	x
i2 = x	x	x	x	x
i2 = z	x	x	x	x

xnor	i1 = 0	i1 = 1	i1 = x	i1 = z
i2 = 0	1	0	x	x
i2 = 1	0	1	x	x
i2 = x	x	x	x	x
i2 = z	x	x	x	x

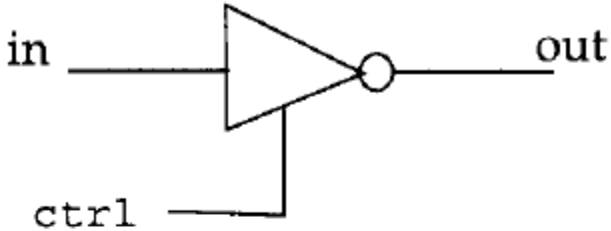
Buf/Not Gates

```
// basic gate instantiations.
buf b1(OUT1, IN);
not n1(OUT1, IN);

// More than two outputs
buf b1_2out(OUT1, OUT2, IN);

// gate instantiation without instance name
not (OUT1, IN); // legal gate instantiation
```

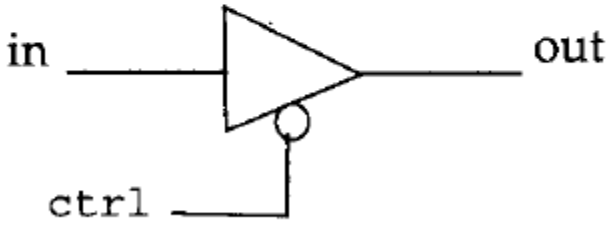
Bufif / Notif



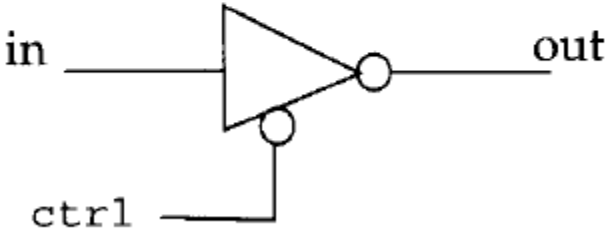
bufif1



notif1



bufif0

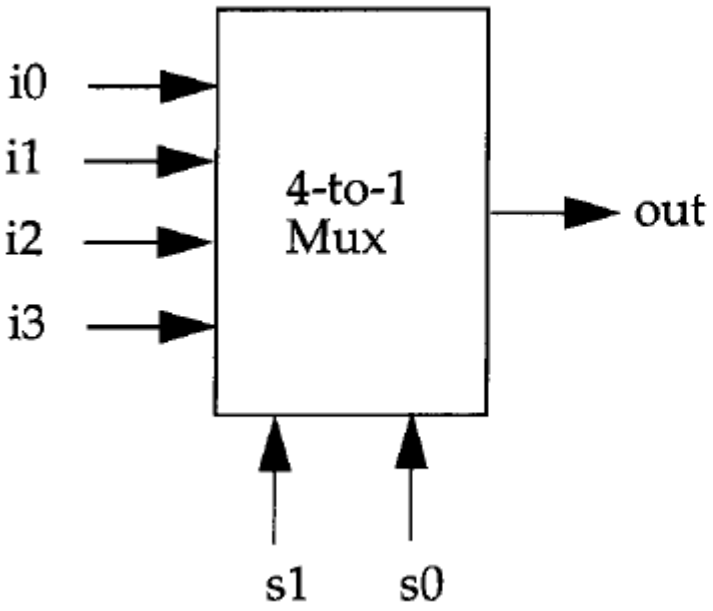


notif0

Bufif / Notif

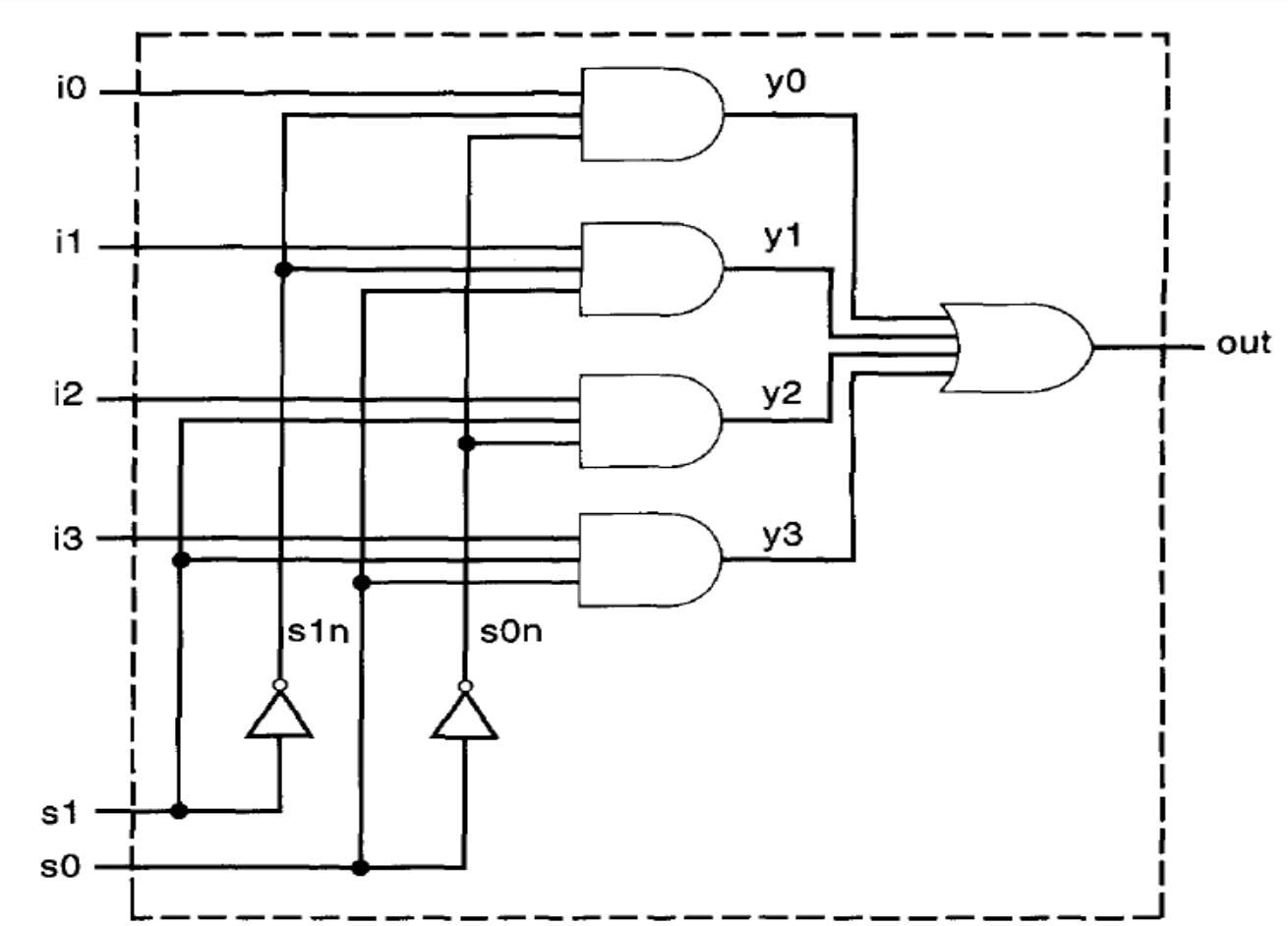
```
//Instantiation of bufif gates.
bufif1 b1 (out, in, ctrl);
bufif0 b0 (out, in, ctrl);

//Instantiation of notif gates
notif1 n1 (out, in, ctrl);
notif0 n0 (out, in, ctrl);
```

Example

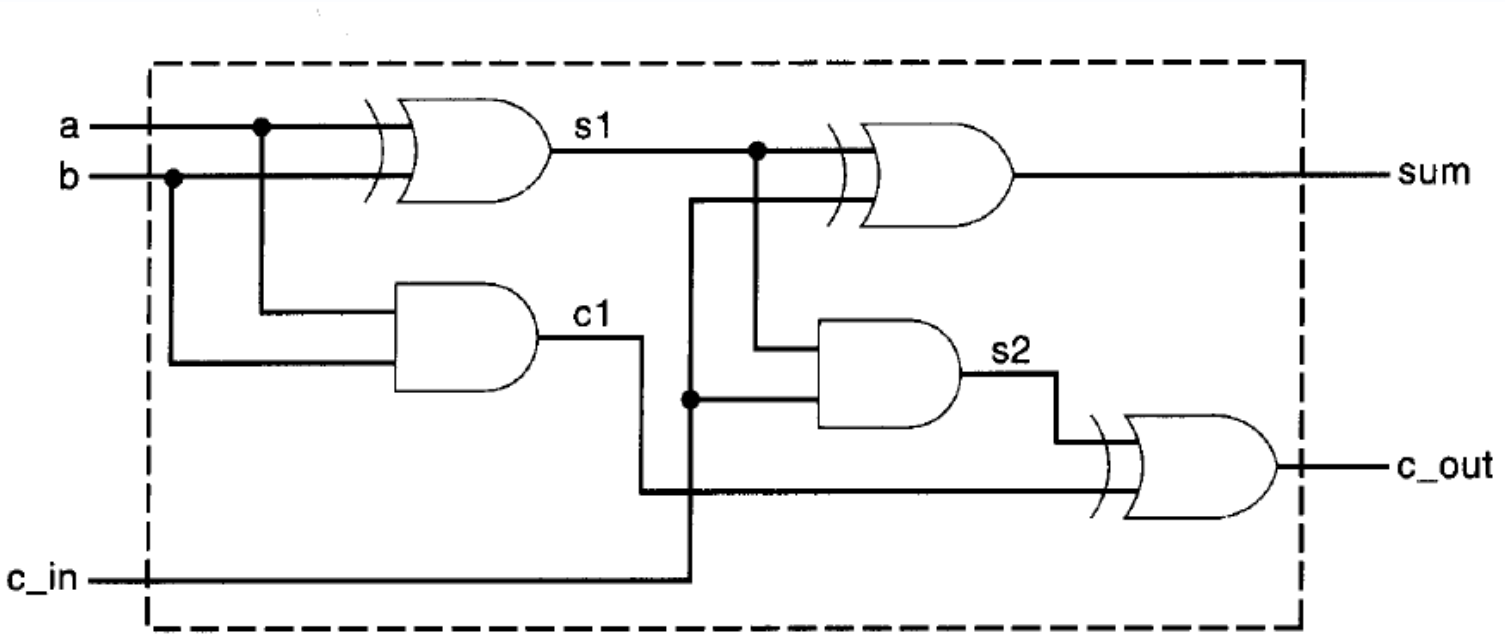


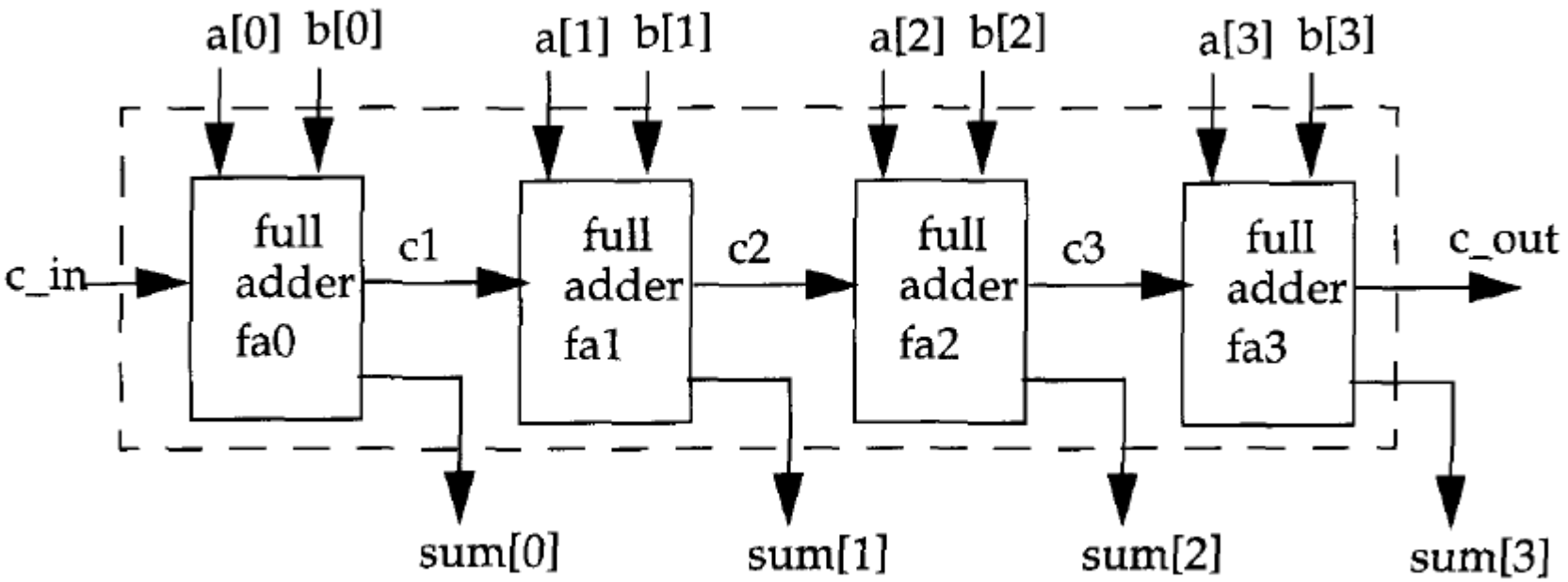
s1	s0	out
0	0	I0
0	1	I1
1	0	I2
1	1	I3

Gate Level Diagram



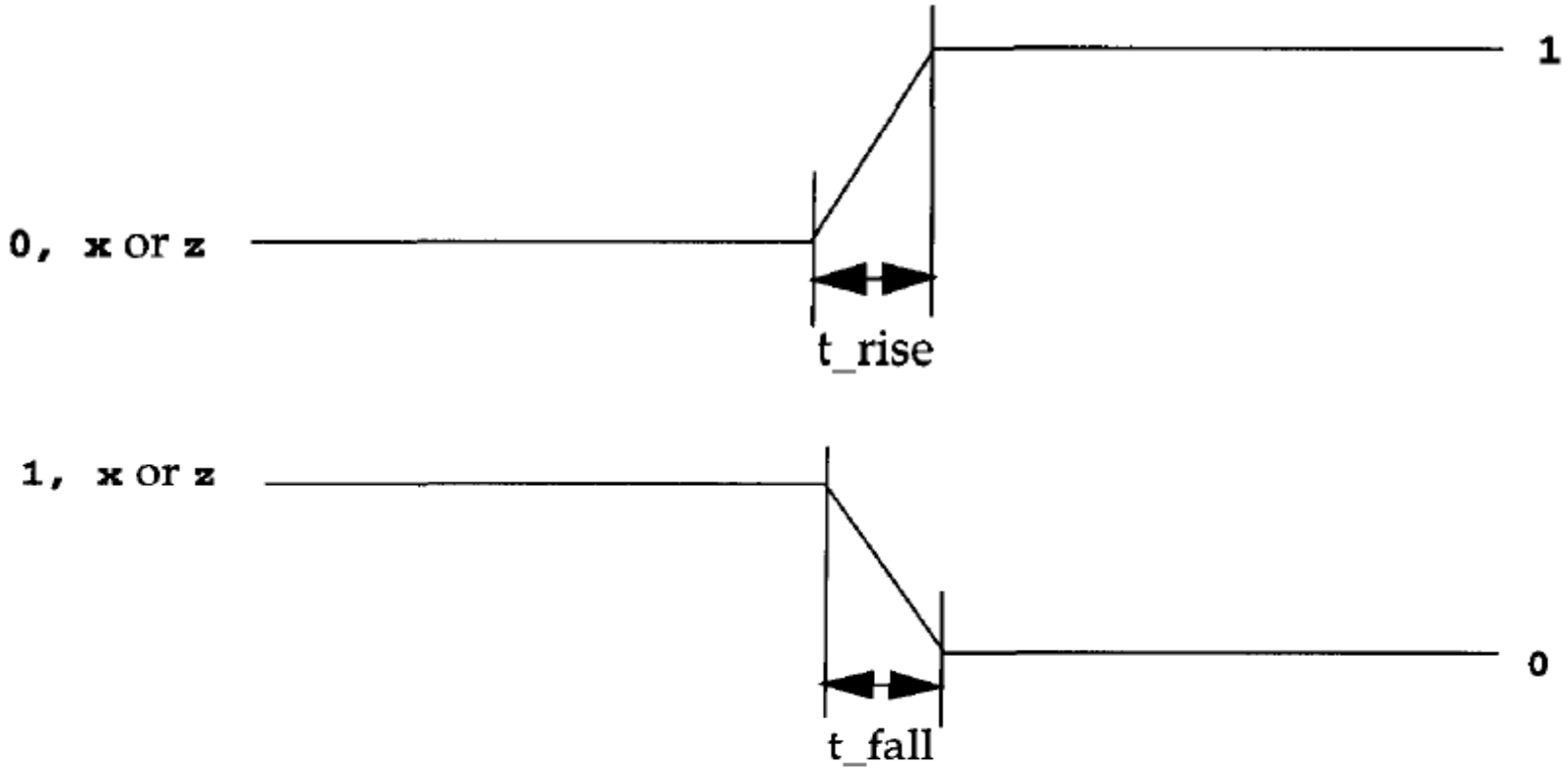
4 Bit Adder




a[0] b[0]
a[1] b[1]
a[2] b[2]
a[3] b[3]
c_in
full adder fa0
c1
full adder fa1
c2
full adder fa2
c3
full adder fa3
c_out
sum[0]
sum[1]
sum[2]
sum[3]

Gate Delays

Rise, Fall, and Turn-off Delays



Min/Typ/Max Value

```
// One delay
// if +mindelays, delay= 4
// if +typdelays, delay= 5
// if +maxdelays, delay= 6
and #(4:5:6) a1(out, i1, i2);

// Two delays
// if +mindelays, rise= 3, fall= 5, turn-off = min(3,5)
// if +typdelays, rise= 4, fall= 6, turn-off = min(4,6)
// if +maxdelays, rise= 5, fall= 7, turn-off = min(5,7)
and #(3:4:5, 5:6:7) a2(out, i1, i2);

// Three delays
// if +mindelays, rise= 2 fall= 3 turn-off = 4
// if +typdelays, rise= 3 fall= 4 turn-off = 5
// if +maxdelays, rise= 4 fall= 5 turn-off = 6
and #(2:3:4, 3:4:5, 4:5:6) a3(out, i1,i2);
```

Turn-off Delay

The Turn-off delay is associated with a gate output transition to z from another value (0, 1, x).