

Timing Analysis, Decoders, Multiplexers for CSE 140-S

Chester Holtz

Based on SP21 slides

March 30, 2022

Overview

- ▶ Timing analysis
- ▶ Decoder
- ▶ Multiplexer

Sequential Networks & Timing Analysis

Definitions

- ▶ Setup time t_{setup} Time before the clock data that must be stable (i.e. not change)
 - ▶ Setup time violation occurs if the input signal D does not settle to a stable value at least t_{setup} before the clock edge.
- ▶ Hold time t_{hold} Time after the clock edge that data must be stable
 - ▶ Hold time violation occurs if the input signal D does not remain unchanged at least t_{hold} after the clock edge.

Sequential Networks & Timing Analysis

Definitions

- ▶ Setup time t_{setup} Time before the clock data that must be stable (i.e. not change)
 - ▶ Setup time violation depends on maximum delay through the combinational logic
- ▶ Hold time t_{hold} Time after the clock edge that data must be stable
 - ▶ Hold time violation depends on the minimum delay through the combinational logic

Sequential Networks & Timing Analysis

Timing constraints

- ▶ Skew constraint

$$t_{\text{skew}} = \text{CLK1} - \text{CLK2}$$

- ▶ Setup time constraint

$$T_C \geq t_{pcq} + t_{pd} + t_{\text{setup}} - t_{\text{skew}}$$

- ▶ Hold time constraint

$$t_{ccq} + t_{cd} > t_{\text{hold}} + t_{\text{skew}}$$

Circuit Properties	
Flip-Flops	Gates
t_{setup} t_{hold} t_{skew} t_{pcq} t_{ccq}	t_{pd} (prop. delay) t_{cd} (contam. delay)

Sequential Networks & Timing Analysis

Timing constraints

- ▶ Skew constraint

$$t_{\text{skew}} = \text{CLK1} - \text{CLK2}$$

- ▶ Setup time constraint

$$T_C \geq t_{pcq} + t_{pd} + t_{\text{setup}} - t_{\text{skew}}$$

- ▶ Hold time constraint

$$t_{ccq} + t_{cd} > t_{\text{hold}} + t_{\text{skew}}$$

Circuit Properties	
Flip-Flops	Gates
t_{setup} t_{hold} t_{skew} t_{pcq} t_{ccq}	t_{pd} (prop. delay) t_{cd} (contam. delay)

Sequential Networks & Timing Analysis

Timing constraints

- ▶ Skew constraint

$$t_{\text{skew}} = \text{CLK1} - \text{CLK2}$$

- ▶ Setup time constraint

$$T_C \geq t_{pcq} + t_{pd} + t_{\text{setup}} - t_{\text{skew}}$$

- ▶ Hold time constraint

$$t_{ccq} + t_{cd} > t_{\text{hold}} + t_{\text{skew}}$$

Circuit Properties	
Flip-Flops	Gates
t_{setup}	t_{pd} (prop. delay) t_{cd} (contam. delay)
t_{hold}	
t_{skew}	
t_{pcq}	
t_{ccq}	

Examples given a circuit:

- ▶ What is the maximum clock frequency?
- ▶ Clock skew tolerance s.t. hold time constraint satisfied?

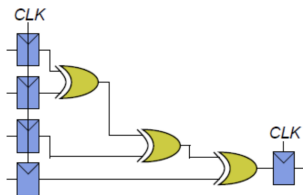
Sequential Network & Timing Analysis Example

Setup time constraint

$$T_C \geq t_{pcq} + t_{pd} + t_{setup} - t_{skew}$$

Hold time constraint

$$t_{ccq} + t_{cd} > t_{hold} + t_{skew}$$



FA'10 Midterm2 Q.4

Circuit Properties	
Flip-Flops	Gates
$t_{setup} = 60ps$	$t_{pd} = 100ps$ (prop. delay) $t_{cd} = 55ps$ (contam. delay)
$t_{hold} = 20ps$	
$t_{pcq} = 70ps$	
$t_{ccq} = 50$	

1. If there is no clock skew, what is the *maximum operating frequency*?
2. How much clock skew can the circuit tolerate before *hold-time violation*?

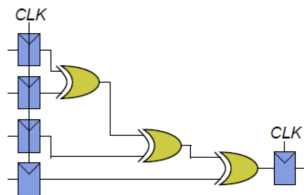
Sequential Network & Timing Analysis Example

Setup time constraint

$$T_C \geq t_{pcq} + t_{pd} + t_{setup} - t_{skew}$$

Hold time constraint

$$t_{ccq} + t_{cd} > t_{hold} + t_{skew}$$



FA'10 Midterm2 Q.4

1. If there is no clock skew, what is the *maximum operating frequency*?

Circuit Properties	
Flip-Flops	Gates
$t_{setup} = 60ps$ $t_{hold} = 20ps$ $t_{pcq} = 70ps$ $t_{ccq} = 50$	$t_{pd} = 100ps$ (prop. delay) $t_{cd} = 55ps$ (contam. delay)

$$\begin{aligned} T_C &\geq t_{pcq} + t_{pd} + t_{setup} - t_{skew} \\ &\geq 70 + 3 \cdot 100 + 60 = 430 \text{ ps} \end{aligned}$$

$$F = \frac{1}{T_C} = 2.33 \text{ GHz}$$

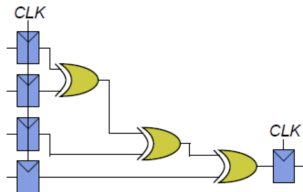
Sequential Network & Timing Analysis Example

Setup time constraint

$$T_C \geq t_{pcq} + t_{pd} + t_{setup} - t_{skew}$$

Hold time constraint

$$t_{ccq} + t_{cd} > t_{hold} + t_{skew}$$



FA'10 Midterm2 Q.4

2. How much clock skew can the circuit tolerate before *hold-time violation*?

Circuit Properties	
Flip-Flops	Gates
$t_{setup} = 60ps$	
$t_{hold} = 20ps$	
$t_{pcq} = 70ps$	$t_{pd} = 100ps$ (prop. delay)
$t_{ccq} = 50$	$t_{cd} = 55ps$ (contam. delay)

$$\begin{aligned} T_{ccq} + T_{cd} &\geq T_{hold} + T_{skew} \\ T_{skew} &\leq 50 + 1 \cdot 55 - 20 \\ &\leq 85 \text{ ps} \end{aligned}$$

Mealy and Moore Machines

- ▶ Mealy machine: general
 - ▶ $y_i(t) = f_i(X(t), S(t))$
- ▶ Moore machine: output is independent of current input
 - ▶ $y_i(t) = f_i(S(t))$
 - ▶ $S_i(t+1) = g_i(X(t), S(t))$

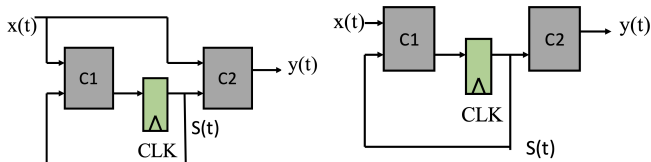
Mealy and Moore Machines

- ▶ Mealy machine: general
 - ▶ $y_i(t) = f_i(X(t), S(t))$
- ▶ Moore machine: output is independent of current input
 - ▶ $y_i(t) = f_i(S(t))$
 - ▶ $S_i(t+1) = g_i(X(t), S(t))$

Mealy and Moore Machines

- ▶ Mealy machine: general
 - ▶ $y_i(t) = f_i(X(t), S(t))$
- ▶ Moore machine: output is independent of current input
 - ▶ $y_i(t) = f_i(S(t))$
 - ▶ $S_i(t + 1) = g_i(X(t), S(t))$

Which is which?



Conversion from Mealy to Moore

Algorithm

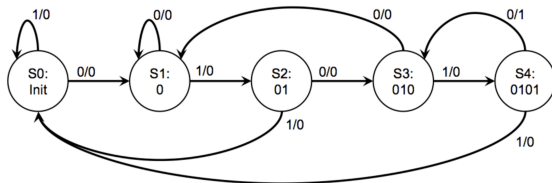
1. Identify distinct (NS, y) pair
2. Replace each distinct (NS, y) pair with distinct new states
3. Insert rows of present state=new states
4. Append each present state with its output y .

Conversion from Mealy to Moore

Algorithm

1. Identify distinct (NS, y) pair
2. Replace each distinct (NS, y) pair with distinct new states
3. Insert rows of present state=new states
4. Append each present state with its output y .

A pattern recognizer produces output $y = 1$ when $x(t - 4, t) = 01010$



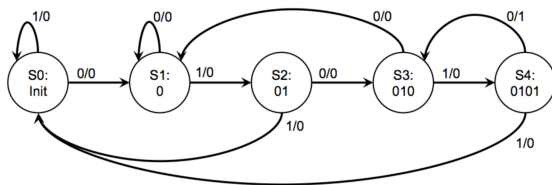
	Input	
	$x = 0$	$x = 1$
S_0		
S_1		
PS S_2		
S_3		
S_4		

Conversion from Mealy to Moore

Algorithm

1. Identify distinct (NS, y) pair
2. Replace each distinct (NS, y) pair with distinct new states
3. Insert rows of present state=new states
4. Append each present state with its output y .

A pattern recognizer produces output $y = 1$ when $x(t-4, t) = 01010$



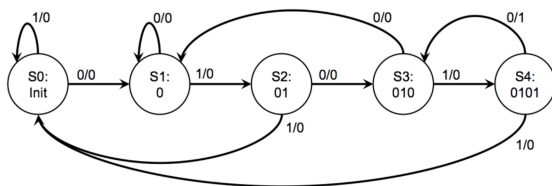
	Input	
	$x = 0$	$x = 1$
S_0	$(S_1, 0)$	$(S_0, 0)$
S_1	$(S_1, 0)$	$(S_2, 0)$
PS S_2	$(S_3, 0)$	$(S_0, 0)$
S_3	$(S_1, 0)$	$(S_4, 0)$
S_4	$(S_3, 1)$	$(S_0, 0)$

Conversion from Mealy to Moore

Algorithm

1. Identify distinct (NS, y) pair
2. Replace each distinct (NS, y) pair with distinct new states
3. Insert rows of present state=new states
4. Append each present state with its output y .

A pattern recognizer produces output $y = 1$ when $x(t-4, t) = 01010$



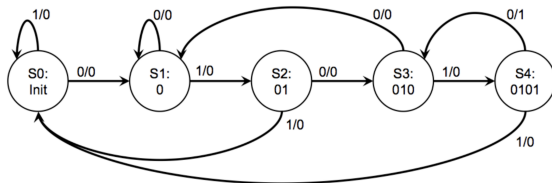
	Input	
	$x = 0$	$x = 1$
S_0	$(S_1, 0)$	$(S_0, 0)$
S_1	$(S_1, 0)$	$(S_2, 0)$
PS S_2	$(S_3, 0)$	$(S_0, 0)$
S_3	$(S_1, 0)$	$(S_4, 0)$
S_4	$(S_3, 1)$	$(S_0, 0)$

Conversion from Mealy to Moore

Algorithm

1. Identify distinct (NS, y) pair
2. Replace each distinct (NS, y) pair with distinct new states
3. Insert rows of present state=new states
4. Append each present state with its output y .

A pattern recognizer produces output $y = 1$ when $x(t - 4, t) = 01010$



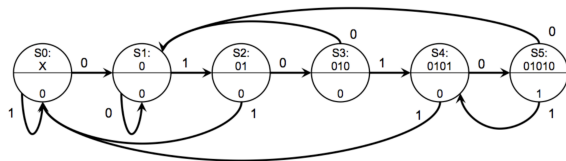
	Input Out		
	x=0	x=1	y
PS	S_0	S_1 S_0	0
	S_1	S_1 S_2	0
	S_2	S_3 S_0	0
	S_3	S_1 S_4	
	S_4	S_5 S_0	
	S_5		

Conversion from Mealy to Moore

Algorithm

1. Identify distinct (NS, y) pair
2. Replace each distinct (NS, y) pair with distinct new states
3. Insert rows of present state=new states
4. Append each present state with its output y .

A pattern recognizer produces output $y = 1$ when $x(t - 4, t) = 01010$



	Input Output		
	x=0	x=1	y
S_0	S_1	S_0	0
S_1	S_1	S_2	0
S_2	S_3	S_0	0
S_3	S_1	S_4	0
S_4	S_5	S_0	0
S_5	S_1	S_4	1

Decoders

Definitions

- ▶ Digital module, N inputs, 2^N outputs
- ▶ One-hot outputs: only one output HIGH (1) at most

Decoders

Definitions

- ▶ Digital module, N inputs, 2^N outputs
- ▶ One-hot outputs: only one output HIGH (1) at most

Can be used to implement Boolean functions

- ▶ Decoder produces minterms when $E=1$.
- ▶ Can use an OR gate to collect the minterms to cover the On-set.
- ▶ For the Don't Care-Set, we can just ignore the terms

Decoders

Example

$$f(a, b, c, d) = \sum m(12, 15)$$

Implement f using a 4 – 16 decoder and OR gates

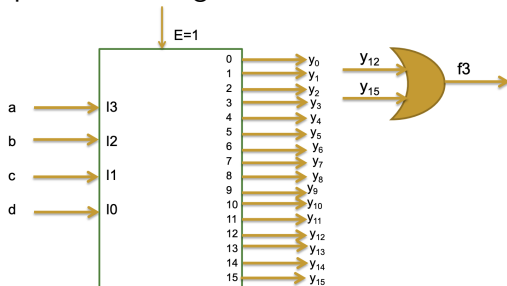
	0
	1
	2
I3	3
	4
I2	5
	6
	7
I1	8
	9
I0	10
	11
	12
	13
	14
	15

Decoders

Example

$$f(a, b, c, d) = \sum m(12, 15)$$

Implement f using a 4 – 16 decoder and OR gates



Decoders

Example

$$f(a, b, c, d) = \sum m(12, 15)$$

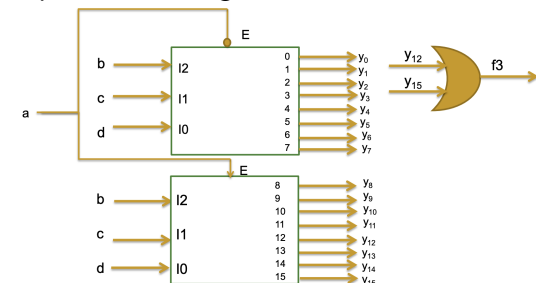
Implement f using 3 – 8 decoders and OR gates

Decoders

Example

$$f(a, b, c, d) = \sum m(12, 15)$$

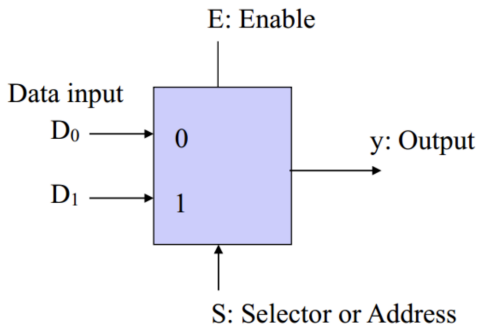
Implement f using 3 – 8 decoders and OR gates



Multiplexers

Definitions

- ▶ Selects one of N inputs to connect to the output
- ▶ $\log_2(N)$ -bit select input – control input



Multiplexers

Example

$$f(a, b, c) = \sum m(0, 3, 5, 7) + \sum d(6)$$

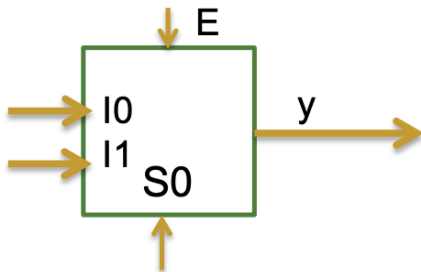
implement the function using a minimal network of 2:1 multiplexers.

Multiplexers

Example

$f(a, b, c)$

id	<i>a</i>	<i>b</i>	<i>c</i>	<i>f</i>
0	0	0	0	1
1	0	0	1	0
2	0	1	0	0
3	0	1	1	1
4	1	0	0	0
5	1	0	1	1
6	1	1	0	x
7	1	1	1	1



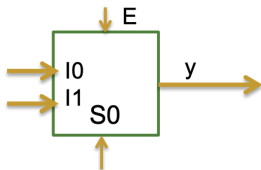
Multiplexers

Example

$$f(a, b, c) = \sum m(0, 3, 5, 7) + \sum d(6)$$

implement the function using a minimal network of 2:1 multiplexers.

id	a	b	c	f
0	0	0	0	1
1	0	0	1	0
2	0	1	0	0
3	0	1	1	1
4	1	0	0	0
5	1	0	1	1
6	1	1	0	x
7	1	1	1	1



bc	00	01	10	11	
a=0	1	0	0	1	$I0 = d0(bc)$
a=1	0	1	x	1	$I1 = d1(bc)$

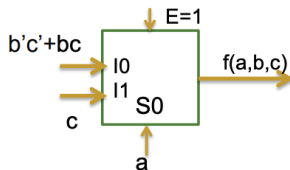
Multiplexers

Example

$$f(a, b, c) = \sum m(0, 3, 5, 7) + \sum d(6)$$

implement the function using a minimal network of 2:1 multiplexers.

bc	00	01	10	11	
a=0	1	0	0	1	$l0 = d0(bc)$
a=1	0	1	x	1	$l1 = d1(bc)$



c\b	0	1
0	1	0
1	0	1

c\b	0	1
0	1	0
1	0	1

Multiplexers

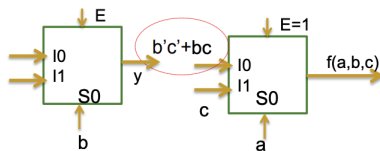
Example

$$f(a, b, c) = \sum m(0, 3, 5, 7) + \sum d(6)$$

implement the function using a minimal network of 2:1 multiplexers.

c\b	0	1
0	1	0
1	0	1

c\b	0	1
0	1	0
1	0	1



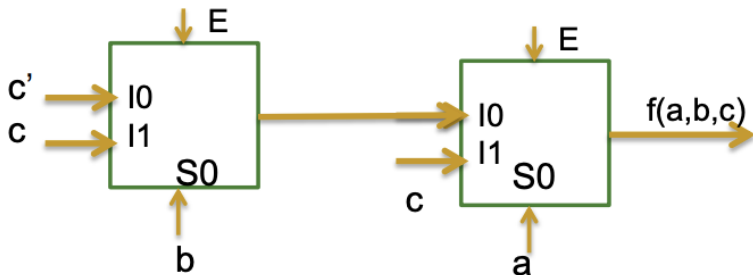
c	0	1	
b=0	1	0	I0 = c'
b=1	0	1	I1 = c

Multiplexers

Example

$$f(a, b, c) = \sum m(0, 3, 5, 7) + \sum d(6)$$

implement the function using a minimal network of 2:1 multiplexers.



Good Luck!