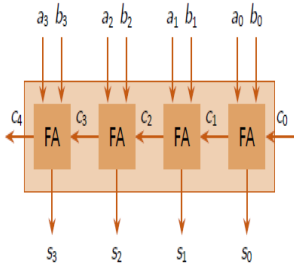


# PES UNIVERSITY 3<sup>rd</sup> SEMESTER


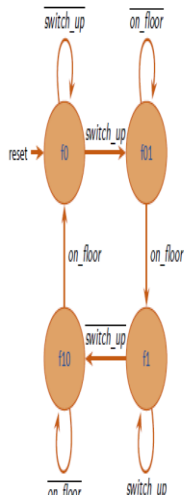
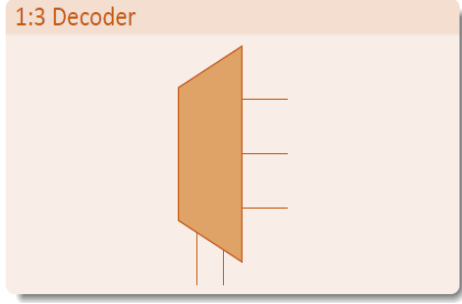
## Errata for DIGITAL DESIGN AND COMPUTER ORGANISATION Slides Uploaded to PESU ACADEMY Portal

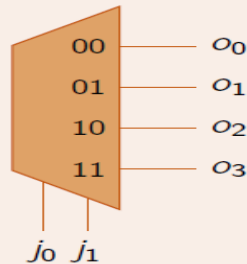
Errata Last updated 1<sup>st</sup> December 2020

**This list is a work in progress. Some of the following corrections may be revised, and additional corrections will probably be added.**

Sl. No	Lecture Number	Content in the slide	To be Corrected as
1.	UNIT3 Lecture 31 Slide 34	<p><b>CARRY-LOOKAHEAD AND PREFIX ADDERS - 1</b> <b>Carry-Lookahead Adder Time Estimate</b></p> <ul style="list-style-type: none"> <li>Carry formulas for 4-bit carry-lookahead adder: <ul style="list-style-type: none"> <li><math>C_1 = g_0 + p_0 C_0</math></li> <li><math>C_2 = g_1 + p_1 g_0 + p_1 p_0 C_0</math></li> <li><math>C_3 = g_2 + p_2 g_1 + p_2 p_1 g_0 + p_2 p_1 p_0 C_0</math></li> <li><math>C_4 = g_3 + p_3 g_2 + p_3 p_2 g_1 + p_3 p_2 p_1 g_0 + p_3 p_2 p_1 p_0 C_0</math></li> </ul> </li> <li>Critical path delay for a carry-lookahead adder is composed of: <ul style="list-style-type: none"> <li><math>p</math> and <math>g</math> computation <ul style="list-style-type: none"> <li><math>t_g</math> time required for two input AND/OR gate</li> </ul> </li> <li>carry computation delay <ul style="list-style-type: none"> <li>time required for <math>C_i</math> depends on <math>i</math></li> </ul> </li> <li>sum computation <ul style="list-style-type: none"> <li><math>2t_g</math> time required for three input XOR</li> </ul> </li> </ul> </li> <li>Carry computation delay: for an <math>n</math>-bit carry-lookahead adder, longest delay is for <math>C_{n-1}</math>, which is composed of: <ul style="list-style-type: none"> <li>Delay for the minterm <math>p_{n-2} p_{n-3} \dots p_0 C_0</math> <ul style="list-style-type: none"> <li><math>n</math> input AND gate requires <math>\lceil \log_2(n-1) \rceil t_g</math> time</li> </ul> </li> <li>Delay for the OR of all minterms <ul style="list-style-type: none"> <li>OR of <math>n</math> inputs requires <math>\lceil \log_2(n-1) \rceil t_g</math> time</li> </ul> </li> </ul> </li> <li>So total time required (critical path delay) for an <math>n</math>-bit carry-lookahead adder: <math>2\lceil \log_2(n-1) \rceil t_g + 3t_g</math></li> </ul>	n input and gate requires $\lceil \log_2 n \rceil$ time (n instead of n-1). so total time is $2\lceil \log_2 n \rceil t_g + 3t_g$ .
2.	UNIT3 Lecture 31 Slide 5	<p><b>CARRY-LOOKAHEAD AND PREFIX ADDERS - 1</b> <b>Ripple Carry Adder Area and Time</b></p>  <ul style="list-style-type: none"> <li>Area requirements: <ul style="list-style-type: none"> <li>Each full adder contains five two input gates (for carry) and one three input gate (for sum)</li> <li>So each full adder occupies <math>7a_g</math> area</li> <li>An <math>n</math>-bit ripple carry adder thus occupies <math>7na_g</math> area</li> </ul> </li> <li>Time requirements: For an <math>n</math>-bit ripple carry adder, critical path delay is composed of: <ul style="list-style-type: none"> <li>Propagation delay from <math>C_0</math> to <math>C_{n-1}</math> <ul style="list-style-type: none"> <li>Signal passes through two gates in each of the <math>n-1</math> stages</li> <li><math>2(n-1)t_g</math> time required</li> </ul> </li> <li>Sum computation <ul style="list-style-type: none"> <li><math>2t_g</math> time required for three input XOR gate</li> </ul> </li> </ul> </li> <li>An <math>n</math>-bit ripple carry adder thus occupies <math>2nt_g</math> time</li> </ul>	signal passes through 3 gates in each stage so delay is $3(n-1)t_g$

3.	UNIT5 Lecture 53 Slide 10	<h3 style="text-align: center;">SYSTOLIC ARRAY MATRIX MULTIPLY</h3> <div style="text-align: center;"> <p><b>Systolic Array Example:</b></p> <p><b>3x3 Systolic Array Matrix Multiplication</b></p> <ul style="list-style-type: none"> <li>Processors arranged in a 2-D grid</li> <li>Each processor accumulates one element of the product</li> </ul> <p style="text-align: right;">EECC756 - Shaaban</p> <p style="font-size: small;">Example source: <a href="http://www.cs.bmc.edu/courses/2001/spring/cs156/">http://www.cs.bmc.edu/courses/2001/spring/cs156/</a> 48 lec #3 Spring 2003 3-11-2003</p> </div>	<p>The total time needed for computation (of the matrix C) with the systolic array will be equal to <math>3n-2</math> clock cycles to perform <math>n \times n</math> matrix multiplication.</p>
4.	UNIT4 Lecture 38 Slide 18	<h3 style="text-align: center;">WALLACE TREE MULTIPLIER</h3> <h4 style="text-align: center;">Logic to Add Three Numbers</h4> <div style="text-align: right;"> </div> <ul style="list-style-type: none"> <li>What logic is required to add three numbers? <ul style="list-style-type: none"> <li>One <math>n</math>-bit adder</li> <li>One <math>(n+1)</math>-bit adder</li> </ul> </li> <li>How long does it take? <ul style="list-style-type: none"> <li><math>n</math>-bit ripple carry adder takes <math>nt_{FA}</math></li> <li>So <math>(2n+1)t_{FA}</math> time for ripple carry adders</li> <li>Even if parallel prefix adder used, factor of two remains</li> </ul> </li> <li>Can we do better? <ul style="list-style-type: none"> <li>Can we add three numbers in only slightly longer than time required to add two numbers?</li> </ul> </li> </ul>	<p>The time required to add three Numbers using ripple carry Adders is incorrectly mentioned as <b><math>(2n + 1)t_{FA}</math></b></p> <p><b>The correct time using ripple carry Adders is <math>(n + 2) t_{FA}</math></b></p>
5.	UNIT3 Lecture 34 Slide 16	<h3 style="text-align: center;">CARRY-LOOKAHEAD AND PREFIX ADDERS - 4</h3> <h4 style="text-align: center;">Associative Ripple Carry?</h4> <div style="text-align: center;"> <p>Ripple Carry Adder</p> </div> <ul style="list-style-type: none"> <li><math>c_{i+1} = ab + bc_i + c_i a</math></li> <li>Generate and Propagate: <ul style="list-style-type: none"> <li><math>g_i</math> carry generated in position <math>i</math></li> <li><math>p_i</math> carry propagated in position <math>i</math></li> <li><math>g_{0:i}</math> carry generated in positions 0 to <math>i</math></li> <li><math>p_{0:i}</math> carry propagated in positions 0 to <math>i</math></li> </ul> </li> <li><math>g_i = a_i b_i</math></li> <li><math>p_i = a_i + b_i</math></li> <li><math>g_{0:i+1} = g_i + p_i g_{0:i}</math></li> <li><math>p_{0:i+1} = p_i p_{0:i}</math></li> </ul>	<p>In the figure the labels <math>p_{0:i}</math> and <math>g_{0:i}</math> need to be interchanged.</p>

6.	UNIT3 Lecture 31 Slide 5	<ul style="list-style-type: none"><li>Time requirements: For an <math>n</math>-bit ripple carry adder, critical path delay is composed of:<ul style="list-style-type: none"><li>Propagation delay from <math>c_0</math> to <math>c_{n-1}</math><ul style="list-style-type: none"><li>Signal passes through two gates in each of the <math>n - 1</math> stages</li><li><math>2(n - 1)t_g</math> time required</li></ul></li><li>Sum computation<ul style="list-style-type: none"><li><math>2t_g</math> time required for three input XOR gate</li></ul></li></ul></li><li>An <math>n</math>-bit ripple carry adder thus occupies <math>2nt_g</math> time</li></ul>	Propagation delay from $c_0$ to $c_{n-1}$ should be $3(n-1)t_g$ instead of $2(n-1)t_g$ because although the carry signal passes through two gates in each stage, one of the gates has three inputs which counts as two 2 input gates..																																																																																																																																																																																				
7.	UNIT3 Lecture 31 Slide 34,35	<p><b>CARRY-LOOKAHEAD AND PREFIX ADDERS - 1</b> <b>Performance Comparison</b></p> <ul style="list-style-type: none"><li>Area and time estimates for <math>n</math>-bit adders:</li></ul> <table><thead><tr><th></th><th>Area</th><th>Time</th></tr></thead><tbody><tr><td>Ripple carry</td><td><math>7na_g</math></td><td><math>2nt_g</math></td></tr><tr><td>Carry-lookahead</td><td><math>(n^2 + 5n)a_g</math></td><td><math>2\lceil \log_2(n - 1) \rceil t_g + 3t_g</math></td></tr></tbody></table> <ul style="list-style-type: none"><li>Compared to the ripple carry adder's linear delay increase with size, the carry-lookahead adder delay increase only logarithmically, resulting in dramatically faster adders</li><li>However, the area of the carry-lookahead adder increase quadratically with size</li><li>Is there an adder design that retains the carry-lookahead adder's speed but has significantly lesser area?</li></ul>		Area	Time	Ripple carry	$7na_g$	$2nt_g$	Carry-lookahead	$(n^2 + 5n)a_g$	$2\lceil \log_2(n - 1) \rceil t_g + 3t_g$	 $\log_2(n-1)$ should be replaced by $\log_2(n)$ .																																																																																																																																																																											
	Area	Time																																																																																																																																																																																					
Ripple carry	$7na_g$	$2nt_g$																																																																																																																																																																																					
Carry-lookahead	$(n^2 + 5n)a_g$	$2\lceil \log_2(n - 1) \rceil t_g + 3t_g$																																																																																																																																																																																					
8.	UNIT3 Lecture 24 Slide 42 till Slide 83	 <p><b>Elevator Example State Transition Table</b></p> <table><thead><tr><th colspan="4">Current State</th><th colspan="2">Inputs</th><th colspan="4">Next State</th></tr><tr><th><math>s_3</math></th><th><math>s_2</math></th><th><math>s_1</math></th><th><math>s_0</math></th><th>switch_up</th><th>on_floor</th><th><math>s'_3</math></th><th><math>s'_2</math></th><th><math>s'_1</math></th><th><math>s'_0</math></th></tr></thead><tbody><tr><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>1</td><td>0</td><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>1</td><td>0</td><td>1</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>0</td><td>1</td><td>1</td><td>0</td><td>1</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td><td>0</td><td>1</td><td>1</td><td>0</td><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>0</td><td>0</td><td>0</td><td>1</td></tr></tbody></table>	Current State				Inputs		Next State				$s_3$	$s_2$	$s_1$	$s_0$	switch_up	on_floor	$s'_3$	$s'_2$	$s'_1$	$s'_0$	0	0	0	1	0	0	0	0	0	1	0	0	0	1	0	1	0	0	0	1	0	0	0	1	1	0	0	0	1	0	0	0	0	1	1	1	0	0	1	0	0	0	1	0	0	0	0	0	1	0	0	0	1	0	0	1	0	1	0	0	0	0	1	0	1	0	0	0	1	0	0	0	1	0	1	1	0	1	0	0	0	1	0	0	0	0	0	1	0	0	0	1	0	0	0	1	1	0	0	0	0	1	0	0	1	0	0	1	0	0	0	1	0	0	1	1	0	1	0	0	1	0	0	0	0	0	0	1	0	0	1	0	0	0	0	1	0	0	0	1	1	0	0	0	1	0	1	0	0	0	1	0	0	0	1	1	0	0	0	1	on_first should be equal to 1 in state 3(0100) instead of the incorrectly mentioned on_first should be equal to 1 in state 4(1000) 'lift_down' be 1 in state 4 (1000) according to the state diagram
Current State				Inputs		Next State																																																																																																																																																																																	
$s_3$	$s_2$	$s_1$	$s_0$	switch_up	on_floor	$s'_3$	$s'_2$	$s'_1$	$s'_0$																																																																																																																																																																														
0	0	0	1	0	0	0	0	0	1																																																																																																																																																																														
0	0	0	1	0	1	0	0	0	1																																																																																																																																																																														
0	0	0	1	1	0	0	0	1	0																																																																																																																																																																														
0	0	0	1	1	1	0	0	1	0																																																																																																																																																																														
0	0	1	0	0	0	0	0	1	0																																																																																																																																																																														
0	0	1	0	0	1	0	1	0	0																																																																																																																																																																														
0	0	1	0	1	0	0	0	1	0																																																																																																																																																																														
0	0	1	0	1	1	0	1	0	0																																																																																																																																																																														
0	1	0	0	0	0	0	1	0	0																																																																																																																																																																														
0	1	0	0	0	1	1	0	0	0																																																																																																																																																																														
0	1	0	0	1	0	0	1	0	0																																																																																																																																																																														
0	1	0	0	1	1	0	1	0	0																																																																																																																																																																														
1	0	0	0	0	0	0	1	0	0																																																																																																																																																																														
1	0	0	0	0	1	0	0	0	1																																																																																																																																																																														
1	0	0	0	1	0	1	0	0	0																																																																																																																																																																														
1	0	0	0	1	1	0	0	0	1																																																																																																																																																																														
9.	UNIT 2, Lecture 14 Slide 20	<p><b>1:3 Decoder</b></p> 	To be corrected as 2:3 decoder																																																																																																																																																																																				

10.	UNIT 2 Lecture 14 Slide 12-16	<div>1:4 Decoder</div> <div>1:4 decoder symbol:</div> 	To be corrected as 2:4 decoder																																																															
11.	UNIT1, Lecture 3, Slide 12	<table><tr><th>a</th><th>b</th><th>c</th><th>y</th></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td><td>1</td></tr></table> <div>Truth table</div>	a	b	c	y	0	0	0	0	0	0	1	0	0	1	0	0	0	1	1	1	1	0	0	0	1	0	1	1	1	1	0	1	1	1	1	1	<table><tr><th>a</th><th>b</th><th>c</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	a	b	c	0	0	0	0	0	1	0	1	0	0	1	1	1	0	0	1	0	1	1	1	0	1	1	1
a	b	c	y																																																															
0	0	0	0																																																															
0	0	1	0																																																															
0	1	0	0																																																															
0	1	1	1																																																															
1	0	0	0																																																															
1	0	1	1																																																															
1	1	0	1																																																															
1	1	1	1																																																															
a	b	c																																																																
0	0	0																																																																
0	0	1																																																																
0	1	0																																																																
0	1	1																																																																
1	0	0																																																																
1	0	1																																																																
1	1	0																																																																
1	1	1																																																																
12.	UNIT1, Lecture 4, Slide 13	<div>Boolean Algebra</div> <div>1 Set {0, 1}</div> <div>2 Operations AND, OR, NOT</div> <div>3 Identity elements 0 (for AND), 1 (for OR)</div> <div>4 Laws/Identities Commutative, associative, distributive, ...</div>	<div>Set f0; 1g</div> <div>Operations AND, OR, NOT</div> <div>Identity elements 1 (for AND), 0 (for OR)</div> <div>Laws/Identities Commutative, associative, distributive:</div>																																																															

13. UNIT1,  
Lecture 8  
Slide 4 till  
Slide  
16,Also  
same error  
in Slide 40

<i>a</i>	<i>b</i>	<i>c</i>	<i>d</i>	<i>y</i>
0	0	0	0	1
0	0	0	0	0
0	0	1	0	1
0	0	1	0	0
0	1	0	0	0
0	1	0	0	1
0	1	1	1	0
0	1	1	1	1
1	0	0	1	1
1	0	0	1	0
1	0	1	0	1
1	0	1	0	0
1	1	0	1	0
1	1	0	1	1
1	1	1	1	0
1	1	1	1	1

Four Input Truth Table

a	b	c	d
0	0	0	0
0	0	0	1
0	0	1	0
0	0	1	1
0	1	0	0
0	1	0	1
0	1	1	0
0	1	1	1
1	0	0	0
1	0	0	1
1	0	1	0
1	0	1	1
1	1	0	0
1	1	0	1
1	1	1	0
1	1	1	1

The min terms for table are  $F(a, b, c, d) = \sum(0, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15)$