



Introduction to Digital Design

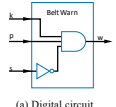
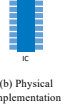
Week 14: Physical Implementation on ICs

Yao Zheng
University of Hawaii at Manoa
Department of Electrical and Computer Engineering

1

Overview

- Many ways to get from design to physical implementation
 - Manufactured IC technologies
 - Full-custom IC
 - Decide on every transistor and wire
 - Semi-custom IC
 - Transistor details pre-designed
 - Gate array: Just wire existing gates
 - Standard cell: Place pre-designed cells and wire them
 - FPGAs
 - Fully programmable
 - Other technologies
 - Logic ICs, PLDs
 - Numerous tradeoffs among technologies, must choose best for given project
 - Trend towards programmable ICs

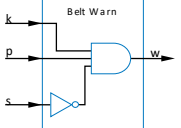
2

2

Introduction

7.1

- A digital circuit design is just an idea, perhaps drawn out
- Need to implement the circuit on a physical device
 - How do we get from *design* to *IC* (integrated circuit, aka chip)?



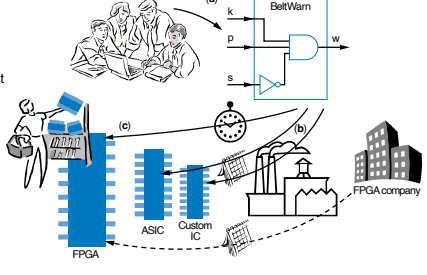

(a) Digital circuit design (b) Physical implementation on an IC

3

3

IC Types, Design Flows

- Many IC types
 - Some fast but expensive
 - Others cheaper but slower
- Types also differ in design flow
 - Some long time
 - Others off-the-shelf
- Now discuss popular types



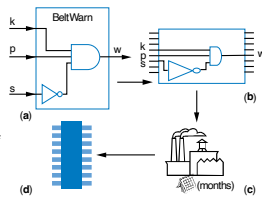
4

4

Manufactured IC Technologies

7.2

- Designer can manufacture a new IC
 - Months of time, millions of dollars
- (1) Full-custom IC**
 - Convert design to *layout*: Describes location/size of every transistor on IC
 - Typically created by CAD tools
 - Send to fabrication plant (fab) to convert layout to actual IC
 - Photographic, laser, chemical equipment
 - Hard!
 - Fab setup costs ("non-recurring engineering", or *NRE*) high—millions of dollars
 - Long fab time (months)
 - Error prone (several "respins")
 - Uncommon
 - Only special ICs that demand the very best performance or the very smallest size/power

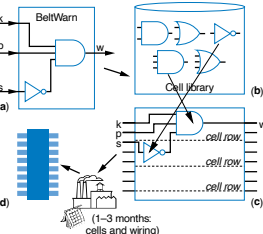


5

5

Manufactured IC Technologies—Standard Cell ASIC

- (2) Semicustom IC (ASIC)**
 - "Application-specific" IC
- (2a) Standard cell ASIC**
 - Pre-layed-out standard-sized "cells" exist in library
 - Designer instantiates cells into pre-defined rows, and connects
 - Vs. full custom
 - Con: Bigger/slower circuit
 - Pro: Easier/faster to design/manufacture

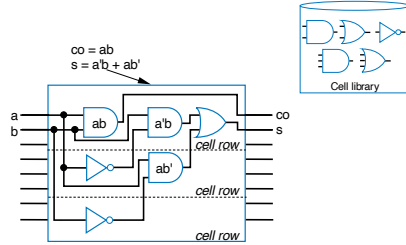


6

6

Manufactured IC Technologies—Standard Cell ASIC

- Example: Mapping half-adder to standard cell ASIC

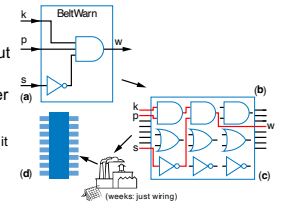


7

Manufactured IC Technologies—Gate Array ASIC

- (2b) Gate array ASIC

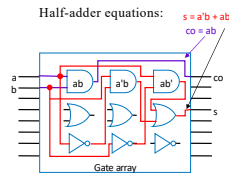
- "Structured" ASIC
- Array of gates already layed out on chip
- Just need to wire them together
- Vs. standard cell ASIC
 - Con: Even bigger/slower circuit
 - Pro: Even easier/faster to design/manufacture
- Very popular



8

Manufactured IC Technologies—Gate Array ASIC

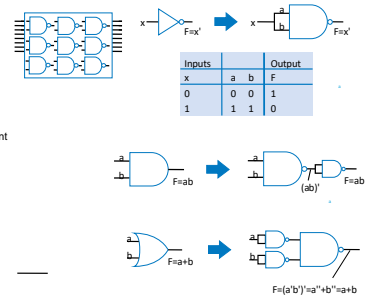
- Example: Mapping a half-adder to a gate array ASIC



9

Implementing Circuits Using NAND Gates Only

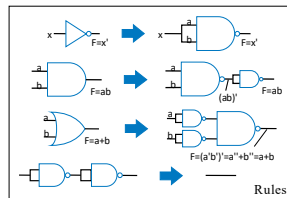
- Recall NAND/NOR more efficient than AND/OR
 - Gate array may have NANDs only
 - Std cell more efficient using NANDs
- NAND is a *universal gate*
 - Any circuit can be mapped to NANDs only
 - Each of AND, OR, and NOT can be converted to equivalent circuit of NANDs
- Convert AND/OR/NOT circuit to NAND-only circuit using mapping rules
 - After converting, remove double inversions



10

Implementing Circuits Using NAND Gates Only

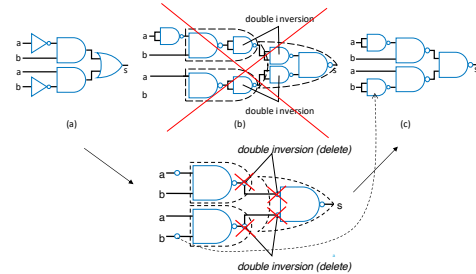
- Example: Half-adder



11

Implementing Circuits Using NAND Gates Only

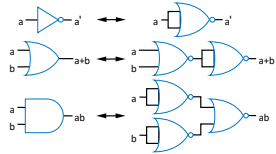
- Shortcut when converting by hand
 - Use inversion bubbles rather than drawing inverters as 2-input NAND
 - Then remove double inversions as before



12

Implementing Circuits Using NOR Gates Only

- NOR gate is also universal
- Converting AND/OR/NOT to NOR done using similar rules

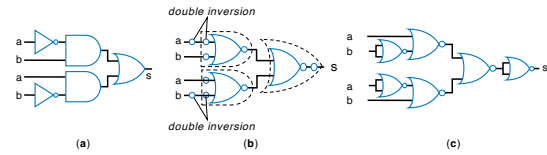


13

13

Implementing Circuits Using NOR Gates Only

- Example: Half adder

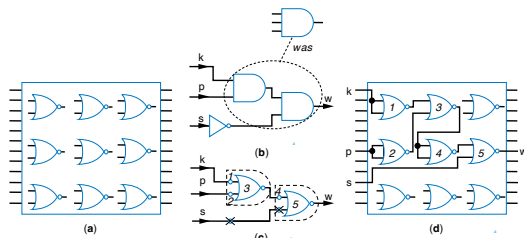


14

14

Implementing Circuits Using NOR Gates Only

- Example: Seat belt warning light on a NOR-based gate array
- Note: if using 2-input NOR gates, first convert AND/OR gates to 2-inputs



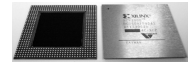
15

15

Off-the-Shelf Programmable IC Type-FPGA

7.3

- Manufactured IC technologies require months to fabricate
 - Also large (million dollar) NRE costs
- Programmable ICs are pre-manufactured
 - User just downloads bits into device, in just seconds
 - Slower/bigger/more-power than manufactured ICs
 - But get it today, and no NRE costs
- Popular programmable IC-FPGA
 - "Field-programmable gate array"
 - Developed late 1980s
 - Though no "gate array" inside
 - Named when gate arrays were popular in 1980s
 - Programmable in the "field" (e.g. your lab) rather than requiring a fab

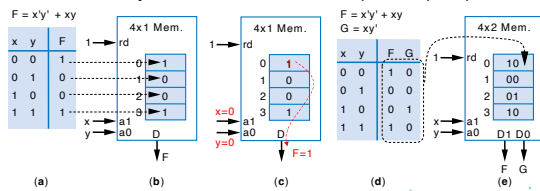


16

16

FPGA Internals: Lookup Tables (LUTs)

- Basic idea: Memory can implement combinational logic
 - Ex: 2-address memory can implement 2-input logic
 - 1-bit wide memory – 1 function; 2-bits wide – 2 functions
- Such memory in FPGA known as *lookup table (LUT)*

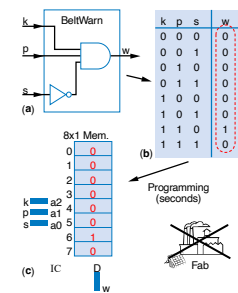


17

17

Mapping a Combinational Circuit to a LUT

- Example: Seat-belt warning light (again)

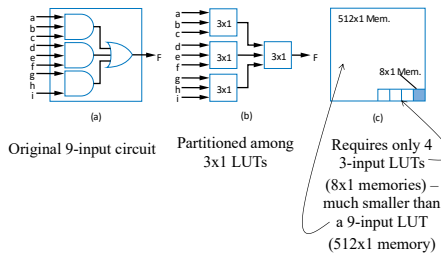


18

18

FPGAs More Efficient With Numerous Small LUTs

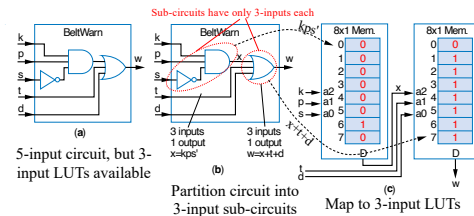
- Lookup tables become inefficient for more inputs
 - 3 inputs → only 8 words 8 inputs → 256 words 16 inputs → 65,536 words!
- FPGAs thus have numerous small (3, 4, 5, or even 6-input) LUTs
 - If circuit has more inputs, must partition circuit among LUTs
 - Ex: 9-input circuit more efficient on 8x1 mems rather than 512x1



19

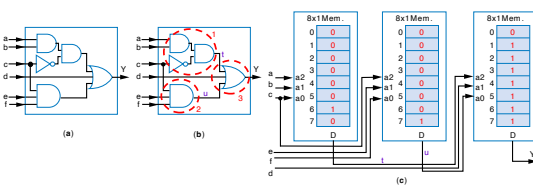
Circuits Must be Partitioned among Small LUTs

- Example: Extended seat-belt warning light system
 - (Assume for now we can create any wires to/between LUTs)



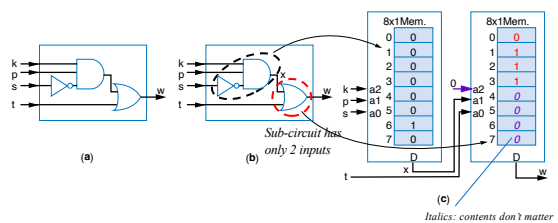
20

Mapping a Circuit to 3x1 LUTs



21

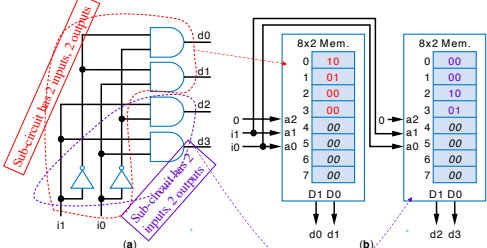
Underutilized LUTs are Common



22

Mapping to 3x2 LUTs

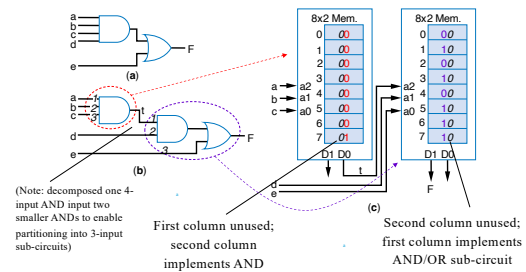
- Example: Mapping a 2x4 decoder to 3-input 2-output LUTs



23

More Mapping Issues

- Gate has more inputs than does LUT → Decompose gate first
- Sub-circuit has fewer outputs than LUT → Just don't use output



24

FPGA Internals: Switch Matrices

- Previous slides had hardwired connections between LUTs
- Instead, want to program the connections too
- Use switch matrices (also known as programmable interconnect)
 - Simple mux-based version – each output can be set to any of the four inputs just by programming its 2-bit configuration memory

(a) (b)

25

Ex: FPGA with Switch Matrix

- Mapping the extended seatbelt warning light circuit onto an FPGA with a switch matrix

(a) (b)

These bits establish the desired connections

26

Configurable Logic Blocks (CLBs)

- Include flip-flops to support sequential circuits
- Muxes programmed to output registered or non-registered LUT output

(a) (b)

27

Sequential Circuit Mapped to FPGA

(a) (b)

28

FPGA Internals: Overall Architecture

- Consists of hundreds or thousands of CLBs and switch matrices (SMs) arranged in regular pattern on a chip

Represents channel with tens of wires

Connections for just one CLB shown, but all CLBs are obviously connected to channels

29

Programming an FPGA

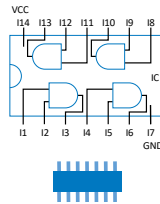
- All configuration memory bits are connected as one big shift register
 - Known as scan chain
- Shift in the "bit file" of the desired circuit

Bit file contents: 00000010 01010101 1 1 00 01 10 00001111 01101111 0 0

30

Other Off-the-Shelf IC Types

- Off-the-shelf logic (SSI) IC
 - Logic IC has a few gates, connected to IC's pins
 - Known as Small Scale Integration (SSI)
 - Popular logic IC series: 7400
 - Originally developed 1960s
 - Back then, each IC cost \$1000
 - Today, costs just tens of cents



7.4

31

7400-Series Logic ICs

TABLE 7.1: Commonly used 7400-series ICs.

Part	Description	Pins
74LS00	Four 2-input NAND	14
74LS02	Four 2-input NOR	14
74LS04	Six inverters	14
74LS08	Four 2-input AND	14
74LS10	Three 3-input NAND	14
74LS11	Three 3-input AND	14
74LS14	Six inverters (Schmitt trigger)	14
74LS20	Two 4-input NAND	14
74LS27	Three 3-input NOR	14
74LS30	One 8-input NAND	14
74LS32	Four 2-input OR	14
74LS74	Two D flip-flop, positive edge triggered, with preset and reset	14
74LS83	4-bit binary full-adder	16
74LS85	4-bit magnitude comparator	16

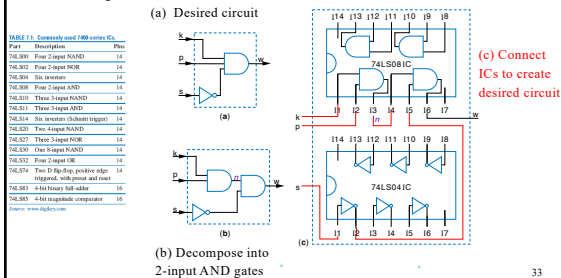
Source: www.digkey.com



32

Using Logic ICs

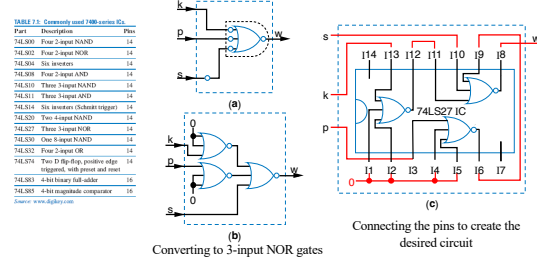
- Example: Seat belt warning light using off-the-shelf 7400 ICs
 - Option 1: Use one 74LS08 IC having 2-input AND gates, and one 74LS04 IC having inverters



33

Using Logic ICs

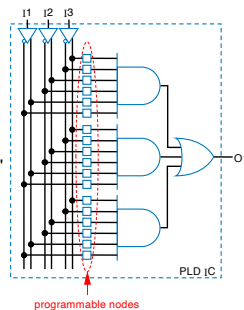
- Example: Seat belt warning light using off-the-shelf 7400 ICs
 - Option 2: Use a single 74LS27 IC having 3-input NOR gates



34

SPLD

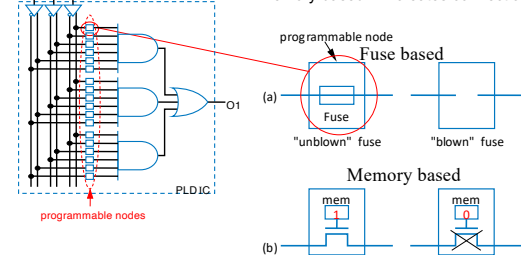
- Simple Programmable Logic Devices (SPLDs)
 - Developed 1970s (thus, pre-dates FPGAs)
 - Prefabricated IC with large AND-OR structure
 - Connections can be "programmed" to create custom circuit
 - Programmable circuit shown can implement any 3-input function of up to 3 terms
 - e.g., $F = abc + a'c'$



35

Programmable Nodes in an SPLD

- Fuse based – "blown" fuse removes connection
- Memory based – 1 creates connection



36

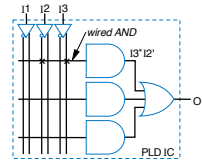
33

36

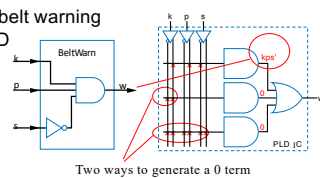
PLD Drawings and PLD Implementation Example

- Common way of drawing PLD connections:

- Uses one wire to represent all inputs of an AND
- Uses "x" to represent connection
 - Crossing wires are not connected unless "x" is present



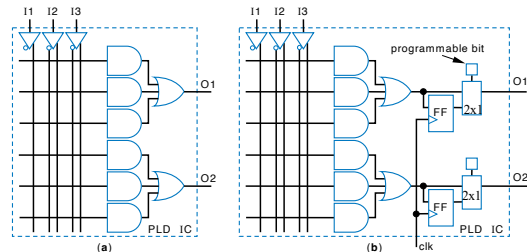
- Example: Seat belt warning light using SPLD



37

37

PLD Extensions



(a) Two-output PLD

(b) PLD with programmable registered outputs

38

38

More on PLDs

- Originally (1970s) known as Programmable Logic Array – PLA
 - Had programmable AND and OR arrays
- AMD created "Programmable Array Logic" – "PAL" (trademark)
 - Only AND array was programmable (fuse based)
- Lattice Semiconductor Corp. created "Generic Array Logic" – "GAL" (trademark)
 - Memory based
- As IC capacities increased, companies put multiple PLD structures on one chip, interconnecting them
 - Became known as *Complex PLDs (CPLD)*, and older PLDs became known as *Simple PLDs (SPLD)*
- GENERALLY SPEAKING, difference of SPLDs vs. CPLDs vs. FPGAs:
 - SPLD: tens to hundreds of gates, and usually non-volatile (saves bits without power)
 - CPLD: thousands of gates, and usually non-volatile
 - FPGA: tens of thousands of gates and more, and usually volatile (but no reason why couldn't be non-volatile)

39

39

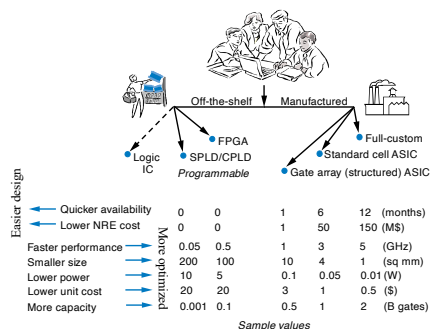
FPGA-to-Structured-ASIC

- FPGA sometimes used as ASIC prototype
 - Typical flow
 - (1) Implement user circuit on FPGA and test
 - (2) Implement user circuit on ASIC (large NRE cost)
 - FPGA-to-structured-ASIC flow
 - (1) Implement user circuit on FPGA and test
 - (2) Implement *FPGA* on ASIC
 - ASIC reflects FPGA structure, NOT the user's circuit structure
 - But remove programmability—LUTs and switch matrices are "hardwired"
 - ASICs lower layers prefabricated, only top layers remaining
 - Less chance of problems (ASIC is similar to FPGA, fewer changes)
 - Results in less NRE cost and less time to manufacture
 - But slower/bigger than if implement user circuit on ASIC directly

40

40

IC Tradeoffs, Trends, and Comparisons



41

41

Choose an IC Type for Each Project

- Project A involves putting the circuit into 100 million mobile phones; encryption speed must be 2.5 GHz, and each chip can be priced up to \$5.
 - Only IC type with at least 2.5 GHz speed is standard cell ASIC. The \$50 million in NRE cost can be amortized over the 100 million chips by adding just \$0.50 to the price of each chip, which when added to the \$1 unit cost results in a price of \$1.50 per chip, much less than the limit of \$5.
 - Use standard cell ASICs.
- Project B involves putting the circuit into 10,000 medical devices; encryption speed must be 1 MHz, and each chip can be priced up to \$50.
 - All three IC types meet speed requirement of 1 MHz. The \$50 million of NRE for a standard cell ASIC amortized over 10,000 chips would involve adding \$5,000 to the price of each chip, which clearly exceeds the limit of \$50 per chip. Even the \$1 million of NRE for a gate array ASIC would require adding \$100 to the price of each chip, which is still too much. Fortunately, the FPGA has no NRE cost, and a unit cost of \$20, which is less than the \$50 limit per chip.
 - Use FPGAs.
- Project C involves putting the circuit into 100,000 automobiles; encryption speed must be 10 MHz, and each chip can be priced up to \$10.
 - All three IC types meet speed requirement of 10 MHz. Amortizing standard cell NRE would result in too high a chip price. Amortizing the gate array ASIC NRE of \$1 million over 100,000 chips would add \$10 per chip, which when added to the \$1 unit cost would result in \$11 per chip, slightly exceeding the \$10 per chip limit. However, the unit cost per FPGA chip is \$20. Thus, none of the three IC types meets project C's price per chip requirement, but the gate array IC type comes very close.
 - Use gate array.

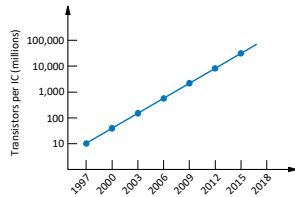
(Choose from among standard cell ASIC, gate array ASIC, or FPGA IC types only, use metric values from previous slide.)

42

42

Key Trend in Implementation Technologies

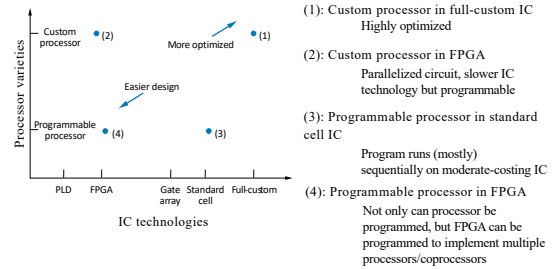
- Transistors per IC doubling every 18 months for past three decades
 - Known as "Moore's Law"
 - Tremendous implications – applications infeasible at one time due to outrageous processing requirements become feasible a few years later
 - Can Moore's Law continue?



43

43

Technology Comparisons

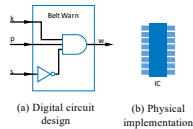


44

44

Summary

- Many ways to get from design to physical implementation
 - Manufactured IC technologies
 - Full-custom IC
 - Decide on every transistor and wire
 - Semi-custom IC
 - Transistor details pre-designed
 - Gate array: Just wire existing gates
 - Standard cell: Place pre-designed cells and wire them
 - FPGAs
 - Fully programmable
 - Other technologies
 - Logic ICs, PLDs
 - Numerous tradeoffs among technologies, must choose best for given project
 - Trend towards programmable ICs



45

45