Homework #1 Due Today at 11:59pm

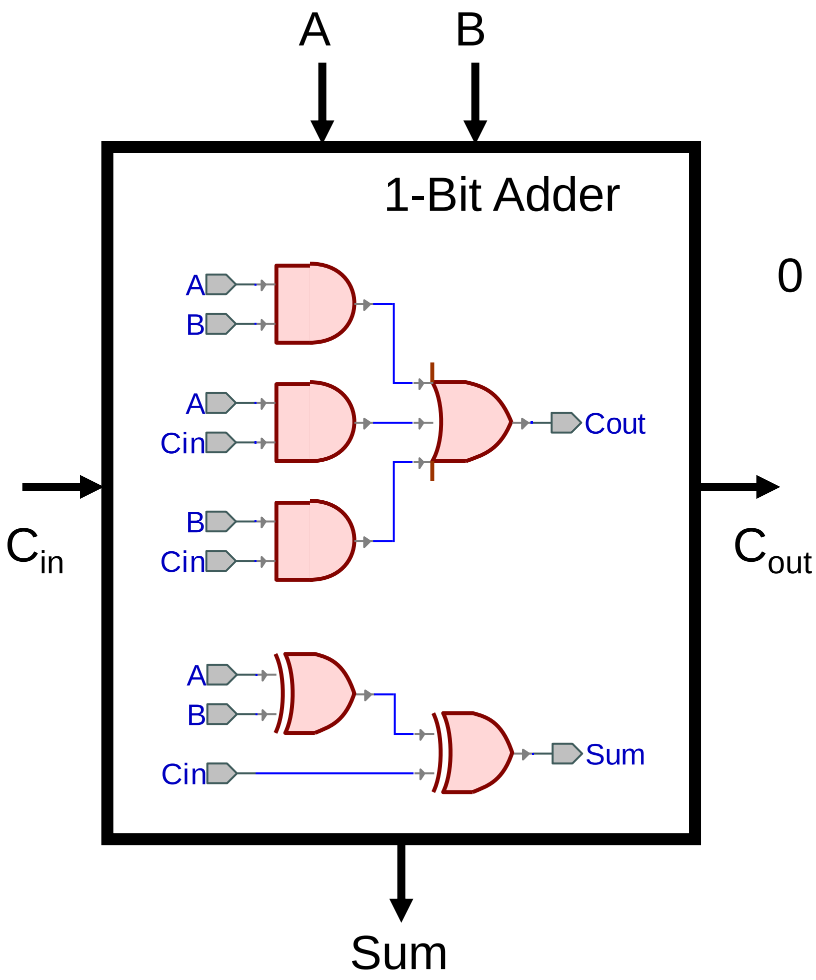
Your Gradescope account is created by your UW/CSE email address

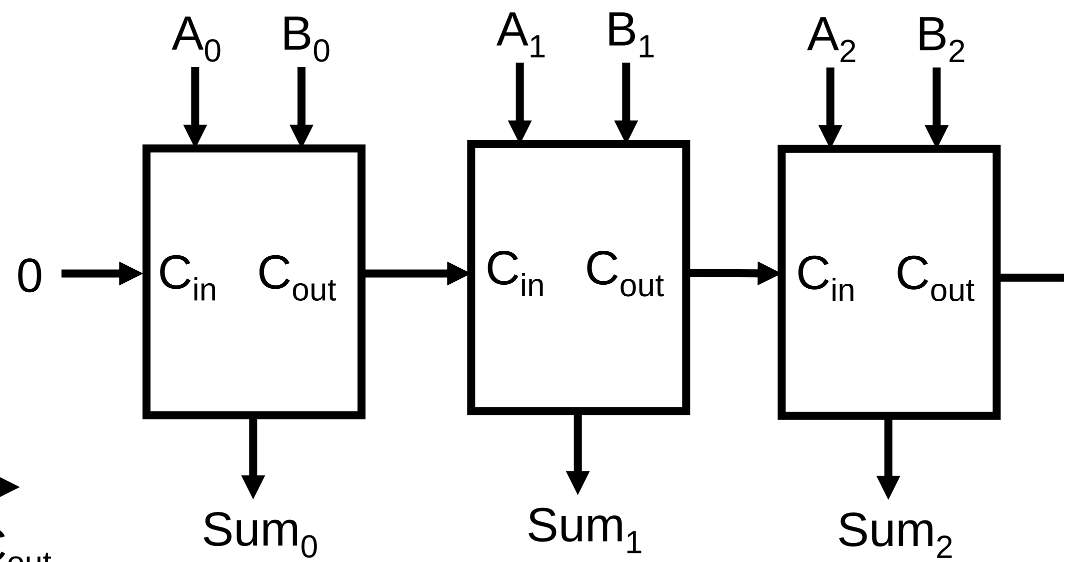
Homework #2 will be posted today and it is due next Friday

TA Office Hours

TA	Office hours	Room
Sam Castle	Wed, 12:00-1:00	CSE 021
Jiechen Chen	Tue, 4:00-5:00	CSE 218
Rebecca Leslie	Mon, 8:30-9:30	CSE 218
Evan McCarty	Tue, 11:30-12:30	CSE 220
Tim Oleskiw	Tue, 3:00-4:00	CSE 218
Spencer Peters	Tue, 1:00-2:00	CSE 218
Robert Weber	Wed, 3:30-4:30	CSE 678 (except Oct 21st at CSE 110)
Ian Zhu	Thu, 4:30-5:30	CSE 021

a 2-bit ripple-carry adder





Fall 2015

Lecture 5: Canonical forms and predicate logic

Given a truth table:

1. Write the Boolean expression
2. Minimize the Boolean expression
3. Draw as gates
4. Map to available gates

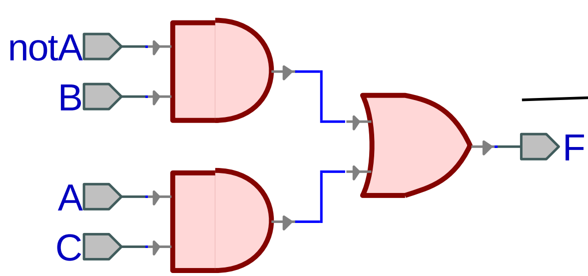
A	B	C	F
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

1

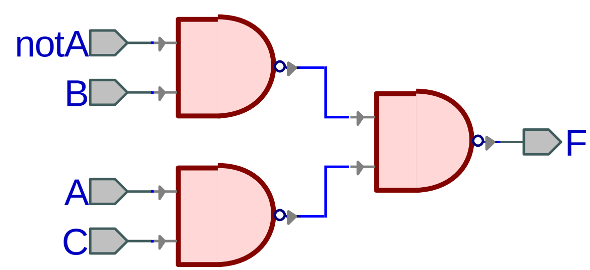
$$F = A'BC' + A'BC + AB'C + ABC$$
$$= A'B(C' + C) + AC(B' + B)$$
$$= A'B + AC$$

2

3



4



- Truth table is the unique signature of a Boolean function
- The same truth table can have many gate realizations
  - we've seen this already
  - depends on how good we are at Boolean simplification
- **Canonical forms**
  - standard forms for a Boolean expression
  - we all come up with the same expression

- also known as Disjunctive Normal Form (DNF)
- also known as minterm expansion

F = *001* *011* *101* *110* *111*

$F = A'B'C + A'BC + AB'C + ABC' + ABC$

A	B	C	F	F′
0	0	0	0	1
0	0	1	1	0
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	0

Product term (or minterm)

- ANDed product of literals – input combination for which output is true
- each variable appears exactly once, true or inverted (but not both)

A	B	C	minterms
0	0	0	A′B′C′
0	0	1	A′B′C
0	1	0	A′BC′
0	1	1	A′BC
1	0	0	AB′C′
1	0	1	AB′C
1	1	0	ABC′
1	1	1	ABC

F in canonical form:

$$F(A, B, C) = A'B'C + A'BC + AB'C + ABC' + ABC$$

canonical form ≠ minimal form

$$\begin{aligned} F(A, B, C) &= A'B'C + A'BC + AB'C + ABC + ABC' \\ &= (A'B' + A'B + AB' + AB)C + ABC' \\ &= ((A' + A)(B' + B))C + ABC' \\ &= C + ABC' \\ &= ABC' + C \\ &= AB + C \end{aligned}$$

- Also known as Conjunctive Normal Form (CNF)
- Also known as maxterm expansion

F = *000* *010* *100*

$F = (A + B + C)\ (A + B' + C)\ (A' + B + C)$

A	B	C	F	F′
0	0	0	0	1
0	0	1	1	0
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	0

Complement of function in sum-of-products form:

– $F' = A'B'C' + A'BC' + AB'C'$

Complement again and apply de Morgan's and get the product-of-sums form:

– $(F')' = (A'B'C' + A'BC' + AB'C')'$
– $F = (A + B + C)\,(A + B' + C)\,(A' + B + C)$

Sum term (or maxterm)

- ORed sum of literals – input combination for which output is false
- each variable appears exactly once, true or inverted (but not both)

A	B	C	maxterms
0	0	0	$A+B+C$
0	0	1	$A+B+C'$
0	1	0	$A+B'+C$
0	1	1	$A+B'+C'$
1	0	0	$A'+B+C$
1	0	1	$A'+B+C'$
1	1	0	$A'+B'+C$
1	1	1	$A'+B'+C'$

F in canonical form:

$$F(A, B, C) = (A + B + C)\,(A + B' + C)\,(A' + B + C)$$

canonical form ≠ minimal form

$$\begin{aligned} F(A, B, C) &= (A + B + C)\,(A + B' + C)\,(A' + B + C) \\ &= (A + B + C)\,(A + B' + C) \\ &\quad\ (A + B + C)\,(A' + B + C) \\ &= (A + C)\,(B + C) \end{aligned}$$

- Propositional Logic
  - If Pikachu doesn’t wear pants, then he flies on Bieber’s jet unless Taylor is feeling lonely.

- Predicate Logic
  - If x, y, and z are positive integers, then $x^3 + y^3 \neq z^3$.

Predicate or Propositional Function

– A function that returns a truth value, e.g.,

“x is a cat”

“x is prime”

“student x has taken course y”

“$x > y$”

“$x + y = z$” or Sum(x, y, z)

“$5 < x$”

Predicates will have variables or constants as arguments.

We must specify a "domain of discourse", which is the possible things we're talking about.

"x is a cat"
(e.g., mammals)

"x is prime"
(e.g., positive whole numbers)

student x has taken course y"
(e.g., students and courses)

$\forall x \, P(x)$

P(x) is true for **every** x in the domain

read as "**for all x, P of x**"

$\exists x \, P(x)$

There is an x in the domain for which P(x) is true

read as "**there exists x, P of x**"

- $\exists x\ \mathrm{Even}(x)$
- $\forall x\ \mathrm{Odd}(x)$
- $\forall x\ (\mathrm{Even}(x) \vee \mathrm{Odd}(x))$
- $\exists x\ (\mathrm{Even}(x) \wedge \mathrm{Odd}(x))$
- $\forall x\ \mathrm{Greater}(x+1, x)$
- $\exists x\ (\mathrm{Even}(x) \wedge \mathrm{Prime}(x))$

Domain:
Positive Integers

Even(x)
Odd(x)
Prime(x)
Greater(x,y)
(or "$x>y$")
Equal(x,y)
(or "$x=y$")
Sum(x,y,z)
(or "$z=x+y$")

- $\forall x\ \exists y$ Greater (y, x)

- $\forall x\ \exists y$ Greater (x, y)

- $\forall x\ \exists y$ (Greater(y, x) $\wedge$ Prime(y))

- $\forall x$ (Prime(x) $\rightarrow$ (Equal(x, 2) $\vee$ Odd(x))

- $\exists x\ \exists y$ (Sum(x, 2, y) $\wedge$ Prime(x) $\wedge$ Prime(y))

Domain:
Positive Integers

Even(x)
Odd(x)
Prime(x)
Greater(x,y)
(or "$x>y$")
Equal(x,y)
(or "$x=y$")
Sum(x,y,z)
(or "z=x+y")

	Prev	Now
• $\forall x\ \exists y$ Greater (y, x)	T	
• $\forall x\ \exists y$ Greater (x, y)	F	

Domain:
All integers

Even(x)
Odd(x)
Prime(x)
Greater(x,y)
(or "x>y")
Equal(x,y)
(or "x=y")
Sum(x,y,z)
(or "z=x+y")

Domain of quantifiers is important!

Cat(x)
Red(x)
LikesTofu(x)

- "Red cats like tofu"

- "Some red cats don't like tofu"

- not every positive integer is prime
- some positive integer is not prime
- prime numbers do not exist
- every positive integer is not prime

$\forall x$ PurpleFruit(x)

Domain:
Fruit

PurpleFruit(x)

Which one is equal to $\neg\forall x$ PurpleFruit(x)?

- $\exists x$ PurpleFruit(x)?

- $\exists x$ $\neg$PurpleFruit(x)?

$$\boxed{\begin{array}{l} \neg\forall x \; P(x) \equiv \exists x \, \neg P(x) \\ \neg\exists x \; P(x) \equiv \forall x \, \neg P(x) \end{array}}$$

$$\boxed{\begin{aligned} \neg\forall x\ P(x) &\equiv \exists x\ \neg P(x) \\ \neg\exists x\ P(x) &\equiv \forall x\ \neg P(x) \end{aligned}}$$

"There is no largest integer."

$$\begin{aligned} &\neg\exists x\ \forall y\ (x \geq y) \\ \equiv\ &\forall x\ \neg\forall y\ (x \geq y) \\ \equiv\ &\forall x\ \exists y\ \neg(x \geq y) \\ \equiv\ &\forall x\ \exists y\ (y > x) \end{aligned}$$

"For every integer there is a larger integer."

example: Notlargest(x) $\equiv \exists y$ Greater (y, x)
$\equiv \exists z$ Greater (z, x)

truth value:

doesn't depend on y or z "bound variables"

does depend on x "free variable"

quantifiers only act on free variables of the formula they quantify

$\forall x\, (\exists y\, (P(x, y) \rightarrow \forall x\, Q(y, x)))$

$\exists x\ (P(x) \land Q(x))$ vs. $\exists x\ P(x) \land \exists x\ Q(x)$