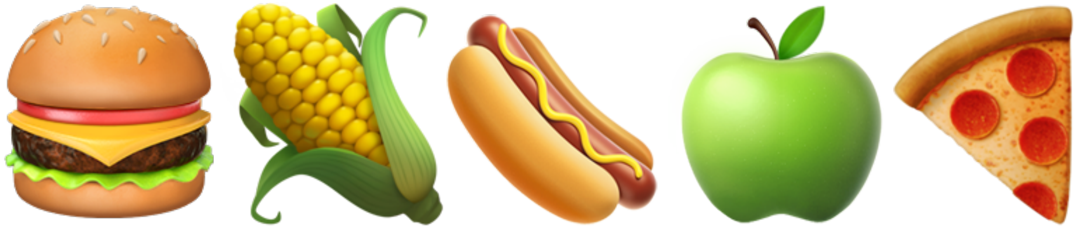
# Logic Design Example

or

# What's for Lunch Today?

ENCE260: Computer Architecture Topic 6



# What's for Lunch Today?

- Deciding what to have for lunch each day can be a real challenge.
- Fortunately, we can use the combinational logic design process to help us clarify our thoughts and automate the process…


Ciaran Moore
ciaran.moore@canterbury.ac.nz

# What's for Lunch Today?

- Specifications:
  - Choosing lunch is hard work, so on days when I've had a big breakfast I don't even bother…
    …I only have lunch when I'm *not full* from breakfast.


Ciaran Moore
ciaran.moore@canterbury.ac.nz

# What's for Lunch Today?

- Specifications:
  - Buying lunch each day can get expensive…
    …so I try to choose food that is *not too pricey*!


Ciaran Moore
ciaran.moore@canterbury.ac.nz

# What's for Lunch Today?

- Specifications:
  - But if I see something that looks really *good,* I'll order it even if it is a little more expensive.


Ciaran Moore
ciaran.moore@canterbury.ac.nz

# Design Process

- I won't have lunch if I'm still full from breakfast.
- If I do have lunch, I'll eat anything that looks good or is not expensive.

## 1. Truth Table

| full | good | pricey | lunch |
|---|---|---|---|
| 0 | 0 | 0 | |
| 0 | 0 | 1 | |
| 0 | 1 | 0 | |
| 0 | 1 | 1 | |
| 1 | 0 | 0 | |
| 1 | 0 | 1 | |
| 1 | 1 | 0 | |
| 1 | 1 | 1 | |



Ciaran Moore
ciaran.moore@canterbury.ac.nz

# Design Process

- I won't have lunch if I'm still full from breakfast.
- If I do have lunch, I'll eat anything that looks good or is not expensive.

## 1. Truth Table

| full | good | pricey | lunch |
|---|---|---|---|
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 0 |


Ciaran Moore
ciaran.moore@canterbury.ac.nz

# Design Process

## 2. Boolean Expression

1. Write down the combination of inputs that give an output of 1.

2. Combine these terms into a *Sum-of-Products* expression.

| full | good | pricey | lunch |
|---|---|---|---|
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 0 |


Ciaran Moore
ciaran.moore@canterbury.ac.nz

# Design Process

## 2. Boolean Expression

1. Write down the combination of inputs that give an output of 1.

$$(\overline{full} \cdot \overline{good} \cdot \overline{pricey})$$

$$(\overline{full} \cdot good \cdot \overline{pricey})$$

$$(\overline{full} \cdot good \cdot pricey)$$

| full | good | pricey | lunch |
|---|---|---|---|
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 0 |



Ciaran Moore
ciaran.moore@canterbury.ac.nz

# Design Process

## 2. Boolean Expression

2. Combine these terms into a *Sum-of-Products* expression.

$$lunch = (\overline{full} \cdot \overline{good} \cdot \overline{pricey}) + (\overline{full} \cdot good \cdot \overline{pricey}) + (\overline{full} \cdot good \cdot pricey)$$

| full | good | pricey | lunch |
|---|---|---|---|
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 0 |


Ciaran Moore
ciaran.moore@canterbury.ac.nz

# Design Process

## 3. Simplify

- Use Boolean algebra to reduce the number of terms

$$lunch =$$
$$(\overline{full} \cdot \overline{good} \cdot \overline{pricey}) + (\overline{full} \cdot good \cdot \overline{pricey}) + (\overline{full} \cdot good \cdot pricey)$$

$$lunch = \overline{full} \cdot$$


Ciaran Moore
ciaran.moore@canterbury.ac.nz

# Design Process

## 3. Simplify

- Use Boolean algebra to reduce the number of terms

$$lunch =$$
$$(\overline{full} \cdot \overline{good} \cdot \overline{pricey}) + (\overline{full} \cdot good \cdot \overline{pricey}) + (\overline{full} \cdot good \cdot pricey)$$

$$lunch = \overline{full} \cdot$$


Ciaran Moore
ciaran.moore@canterbury.ac.nz

# Design Process

## 3. Simplify

- Use Boolean algebra to reduce the number of terms

$$lunch = (\overline{full} \cdot \overline{good} \cdot \overline{pricey}) + (\overline{full} \cdot good \cdot \overline{pricey}) + (\overline{full} \cdot good \cdot pricey)$$

$$lunch = \overline{full} \cdot (good + \overline{pricey})$$


Ciaran Moore
ciaran.moore@canterbury.ac.nz

# Design Process

## 3. Simplify

- Use Boolean algebra to reduce the number of terms

$$lunch = (\overline{full} \cdot \overline{good} \cdot \overline{pricey}) + (\overline{full} \cdot good \cdot \overline{pricey}) + (\overline{full} \cdot good \cdot pricey)$$

$$lunch = \overline{full} \cdot (good + \overline{pricey})$$


Ciaran Moore
ciaran.moore@canterbury.ac.nz

# Design Process

4. Verify

- Karnaugh Map:

| $lunch$ | $\overline{good} \cdot \overline{pricey}$ | $\overline{good} \cdot pricey$ | $good \cdot pricey$ | $good \cdot \overline{pricey}$ |
|---|---|---|---|---|
| $\overline{full}$ | | | | |
| $full$ | | | | |

| full | good | pricey | lunch |
|---|---|---|---|
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 0 |



Ciaran Moore
ciaran.moore@canterbury.ac.nz

# Design Process

4. Verify

- Karnaugh Map:

| $lunch$ | $\overline{good} \cdot \overline{pricey}$ | $\overline{good} \cdot pricey$ | $good \cdot pricey$ | $good \cdot \overline{pricey}$ |
|---|---|---|---|---|
| $\overline{full}$ | 1 | 0 | 1 | 1 |
| $full$ | 0 | 0 | 0 | 0 |

| full | good | pricey | lunch |
|---|---|---|---|
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 0 |

$$lunch = \overline{full} \cdot good + \overline{full} \cdot \overline{pricey}$$

$$lunch = \overline{full} \cdot (good + \overline{pricey})$$



Ciaran Moore
ciaran.moore@canterbury.ac.nz

# Design Process

## 4. Verify

| full | good | pricey | lunch |
|---|---|---|---|
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 0 |

$$lunch = \overline{full} \cdot (good + \overline{pricey})$$


Ciaran Moore
ciaran.moore@canterbury.ac.nz

# Design Process

## 5. Combinational Circuit

1. Place a gate for each Boolean operator:

$$lunch = \overline{full} \cdot (good + \overline{pricey})$$



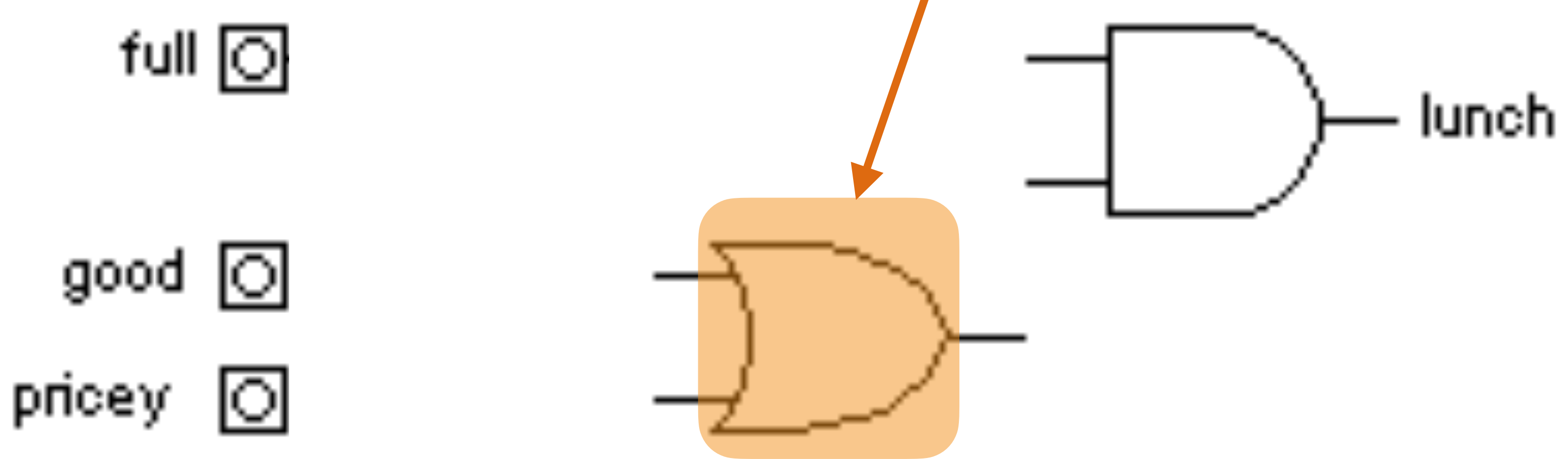
Ciaran Moore
ciaran.moore@canterbury.ac.nz

# Design Process

## 5. Combinational Circuit

1. Place a gate for each Boolean operator:

$$lunch = \overline{full} \cdot (good \boxed{+} \overline{pricey})$$





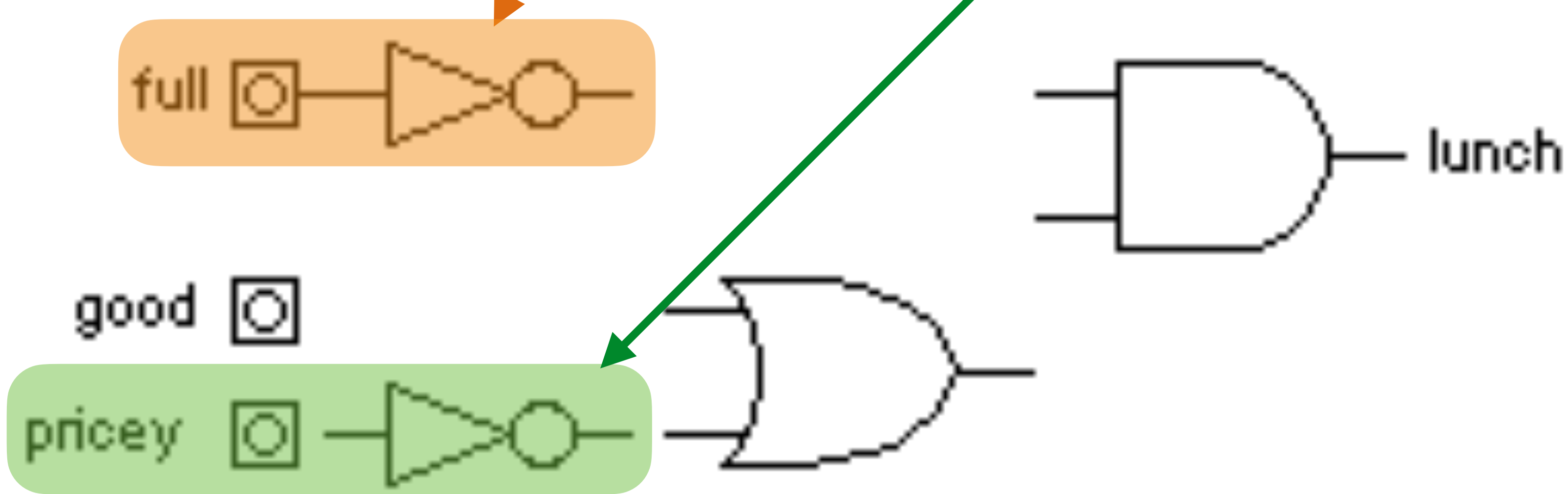
Ciaran Moore
ciaran.moore@canterbury.ac.nz

# Design Process

## 5. Combinational Circuit

1. Place a gate for each Boolean operator:

$$lunch = \overline{full} \cdot (good + \overline{pricey})$$





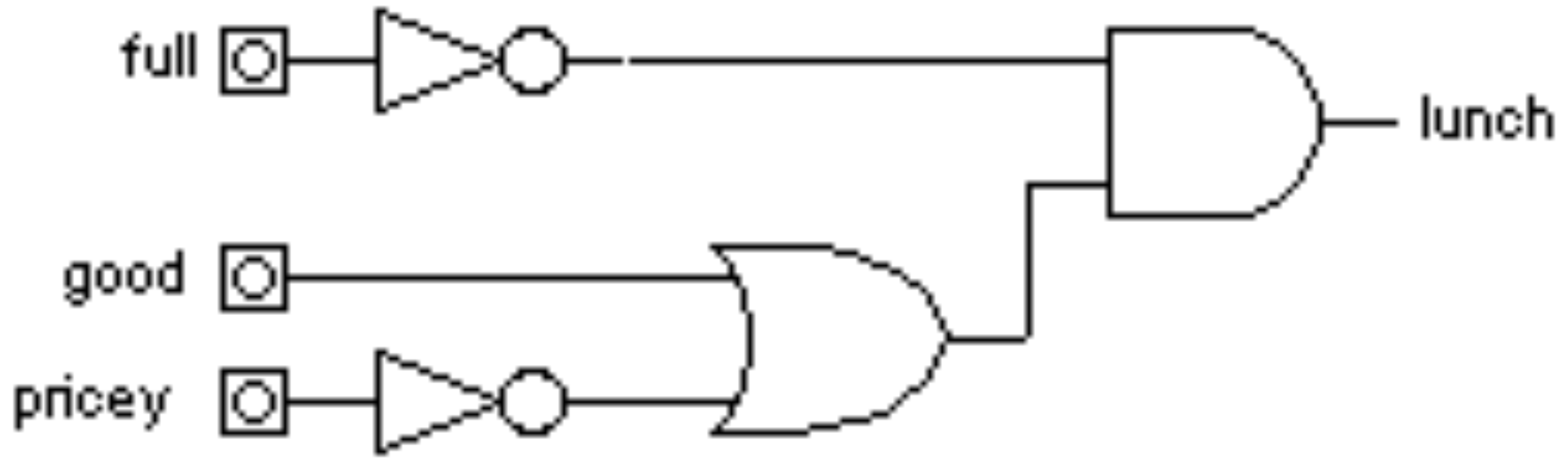

Ciaran Moore
ciaran.moore@canterbury.ac.nz

# Design Process

## 5. Combinational Circuit

2. Connect inputs to outputs through the gates:

$$lunch = \overline{full} \cdot (good + \overline{pricey})$$





Ciaran Moore
ciaran.moore@canterbury.ac.nz

# Summary

1. Define the function using a **truth table**
2. Convert the truth table to a **Boolean expression**
3. **Simplify** the expression
4. Verify the expression using a **Karnaugh map**
5. Map Boolean operators to **logic gates**


Ciaran Moore
ciaran.moore@canterbury.ac.nz