Computeren inderst inde

DM534

Rolf Fagerberg

Bits

Information = valg mellem forskellig muligheder.

Simpleste situation: valg mellem to muligheder. Kald dem 0 og 1. Denne valgmulighed kaldes en bit.

Bits

Information = valg mellem forskellig muligheder.

Simpleste situation: valg mellem to muligheder. Kald dem 0 og 1. Denne valgmulighed kaldes en bit.

Fysisk repræsentation i elektroniske computere:

- 0 = ingen strøm i ledning
- 1 = strøm i ledning

(Harddisk: magnetisering i stedet for strøm)

Bits

Information = valg mellem forskellig muligheder.

Simpleste situation: valg mellem to muligheder. Kald dem 0 og 1. Denne valgmulighed kaldes en bit.

Fysisk repræsentation i elektroniske computere:

- 0 = ingen strøm i ledning
- 1 = strøm i ledning

(Harddisk: magnetisering i stedet for strøm)

Større samling information: brug flere bits:

$$1011001100111010$$

16 bits = 2^{16} muligheder, f.eks.

Beregning

Beregning = ny information fra gammel.

Dvs. nye bits fra gamle.

Beregning

Beregning = ny information fra gammel.

Dvs. nye bits fra gamle.

For én bit ind (x) og én bit ud (y) er mulighederne:

x	y
0	0
1	0

Konstant 0

x	y
0	1
1	1

Konstant 1

x	y
0	0
1	1

Identiteten

x	y
0	1
1	0

Negering (NOT)

Beregning

Beregning = ny information fra gammel.

Dvs. nye bits fra gamle.

For én bit ind (x) og én bit ud (y) er mulighederne:

x	y
0	0
1	0

Konstant 0

x	y
0	1
1	1

Konstant 1

x	y
0	0
1	1

Identiteten

x	y
0	1
1	0

Negering (NOT)

[Der er 2 forskellige input (rækker), og derfor $2^2 = 4$ forskellige muligheder for valg af output (sidste søjle)]

Beregning

Beregning = ny information fra gammel.

Dvs. nye bits fra gamle.

For én bit ind (x) og én bit ud (y) er mulighederne:

x	y
0	0
1	0

Konstant 0

x	y
0	1
1	1

Konstant 1

x	y
0	0
1	1

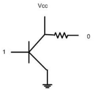
Identiteten

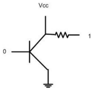
x	y
0	1
1	0

Negering (NOT)

[Der er 2 forskellige input (rækker), og derfor $2^2 = 4$ forskellige muligheder for valg af output (sidste søjle)]

De tre første er trivielle at lave i elektriske kredsløb (forbind output til jord, forbind output til strøm, forbind output til input). NOT kan laves med transistorer og andre simple elektroniske komponenter.

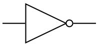




NOT-gate

En kredsløbsdel, som implementerer NOT kaldes en NOT-*gate*.

Symbol for en NOT-gate i et kredsløb:



Sandhedstabel over NOTs funktion:

x_1	$\mathrm{NOT}(x_1)$
0	1
1	0

Matematisk notation for NOT:

$$\neg 0 = 1; \quad \neg 1 = 0;$$

Beregning

Beregning = ny information fra gammel.

Dvs. nye bits fra gamle.

Beregning

Beregning = ny information fra gammel.

Dvs. nye bits fra gamle.

For to bits ind (x_1 og x_2) og én bit ud (y) er mulighederne:

x_1	x_2	y
0	0	0
0	1	0
1	0	0
1	1	0

x_1	x_2	y
0	0	1
0	1	0
1	0	0
1	1	0

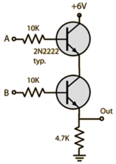
x_1	x_2	y
0	0	0
0	1	1
1	0	0
1	1	0

. . .

x_1	x_2	y
0	0	1
0	1	1
1	0	1
1	1	1

[Der er $2^2 = 4$ forskellige input (rækker), og derfor $4^2 = 16$ forskellige muligheder for valg af output (sidste søjle)]

Vi skal lære navnene (AND, OR, NAND, . . . for en del af disse. De kan også laves med transistorer og andre simple elektroniske komponente. Her er en AND:



AND-gate

En kredsløbsdel, som implementerer AND kaldes en AND-*gate*.

Symbol for en AND-gate i et kredsløb:

Sandhedstabel over ANDs funktion:

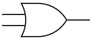
x_1	x_2	$\text{AND}(x_1, x_2)$
0	0	0
0	1	0
1	0	0
1	1	1

Matematisk notation for AND:

$$0 \wedge 0 = 0;\ 0 \wedge 1 = 0;\ 1 \wedge 0 = 0;\ 1 \wedge 1 = 1;$$

OR-gate

Symbol for en OR-gate i et kredsløb:



Sandhedstabel over ORs funktion:

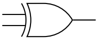
x_1	x_2	$\mathrm{OR}(x_1, x_2)$
0	0	0
0	1	1
1	0	1
1	1	1

Matematisk notation for OR:

$$0 \vee 0 = 0;\ 0 \vee 1 = 1;\ 1 \vee 0 = 1;\ 1 \vee 1 = 1;$$

XOR-gate

Symbol for en XOR-gate i et kredsløb:



Sandhedstabel over XORs funktion:

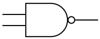
x_1	x_2	XOR(x_1, x_2)
0	0	0
0	1	1
1	0	1
1	1	0

Matematisk notation for XOR:

$$0 \oplus 0 = 0;\ \ 0 \oplus 1 = 1;\ \ 1 \oplus 0 = 1;\ \ 1 \oplus 1 = 0;$$

NAND-gate

Symbol for en NAND-gate i et kredsløb:



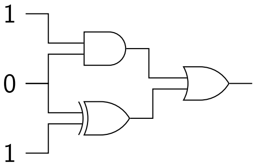
Sandhedstabel over NANDs funktion:

x_1	x_2	NAND(x_1, x_2)
0	0	1
0	1	1
1	0	1
1	1	0

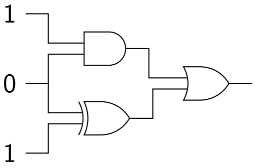
Matematisk notation for NAND:

$$0 \text{ nand } 0 = 1; \; 0 \text{ nand } 1 = 1; \; 1 \text{ nand } 0 = 1; \; 1 \text{ nand } 1 = 0;$$

Eksempel på kredsløb

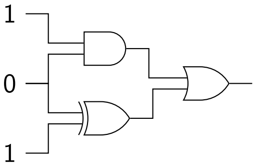


Eksempel på kredsløb



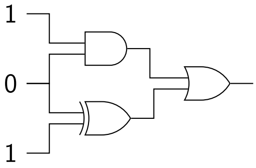
Hvad er gates?

Eksempel på kredsløb



Hvad er gates? Gates er: AND, XOR og OR.

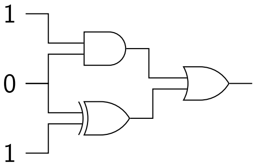
Eksempel på kredsløb



Hvad er gates? Gates er: AND, XOR og OR.

Hvad er output?

Eksempel på kredsløb



Hvad er gates? Gates er: AND, XOR og OR.

Hvad er output? Output er: 1.

Alle sandhedstabeller

Bemærk at man med AND, OR og NOT kan lave kredsløb som beregner enhver ønsket sandhedstabel (dvs. enhver ønsket beregning):

- Finde de rækker (input) i tabellen, hvor y (output) er lig 1.
- For hver række (input) brug AND og NOT til at lave et kredsløb, som giver output 1 for dette og kun dette input (se eksemplet på næste side for metoden).
- Sæt disse kredsløb sammen med en masse OR.

Det resulterende kredsløb giver output 1 præcis for de rækker (input), hvor sandhedstabellen har 1 som output.

Et eksempel gives på næste side, med fire inputs x_1, x_2, x_3 og x_4. Det udtrykkes med matematisk notation, men kan nemt konverteres til et kredsløb med fire input ledninger x_1, x_2, x_3 og x_4 (brug AND-gates for $\wedge$, NOT-gates for $\neg$ og OR-gates for $\vee$).

Eksempel

I følgende sandhedstabel er der to rækker med output (y) lig 1:

x_1	x_2	x_3	x_4	y
0	0	0	0	0
1	0	0	0	0
$\vdots$	$\vdots$	$\vdots$	$\vdots$	0
0	1	1	0	1
$\vdots$	$\vdots$	$\vdots$	$\vdots$	0
1	0	1	0	1
$\vdots$	$\vdots$	$\vdots$	$\vdots$	0

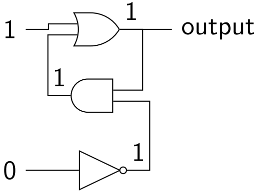
Følgende to udtryk har værdi 1 præcis når input (x_1, x_2, x_3, x_4) er $(0, 1, 1, 0)$, henholdsvis $(1, 0, 1, 0)$:

$$(\neg x_1) \wedge x_2 \wedge x_3 \wedge (\neg x_4)$$
$$x_1 \wedge (\neg x_2) \wedge x_3 \wedge (\neg x_4)$$

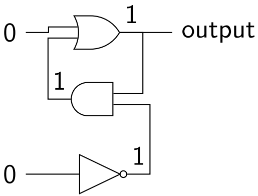
Derfor har følgende udtryk værdi 1 præcis for de input hvor sandhedstabellen har 1 i output:

$$((\neg x_1) \wedge x_2 \wedge x_3 \wedge (\neg x_4)) \vee (x_1 \wedge (\neg x_2) \wedge x_3 \wedge (\neg x_4))$$

Flip flop

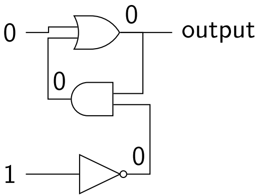


Flip flop



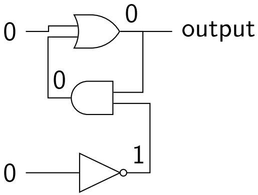
Bemærk at output er stabilt.

Flip flop



Vi kan ændre output.

Flip flop



Bemærk at output er stabilt.

Alt i alt: vi kan lave kredsløb som kan *gemme* en ønsket bit (sålænge der er strøm).

CPU'er

CPU'er er bygget op af sådanne kredse.