cs113 Lab2: By Amuldeep Dhillon

Text written to file build.sh

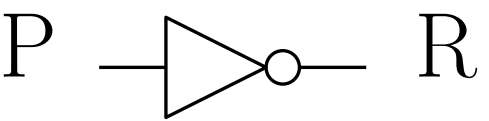
```
doctex lab.doc
pptexenv latex lab.tex
dvipdf lab.dvi
```

Bourne Shell

```
chmod 777 build.sh
./build.sh
```

cs113 Lab2: Not Gate

P	$\sim P$
0	1
1	0



NOT gate

$R \equiv \sim P$

SML

```
exception Error;
fun NOT(0) = 1 | NOT(1) = 0 | NOT(_) = raise Error;
val truthValues1 = [(0),(1)];
map NOT truthValues1;
```

```
> exception Error;
fun NOT(0) = 1 | NOT(1) = 0 | NOT(_) = raise Error;
val truthValues1 = [(0),(1)];
map NOT truthValues1;
exception Error
> val NOT = fn: int -> int
> val truthValues1 = [0, 1]: int list
> val it = [1, 0]: int list
```

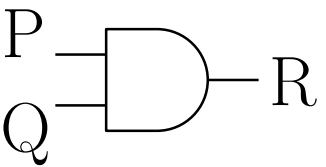
cs113 Lab2: And Gate

P	Q	$P \wedge Q$
0	0	0
0	1	0
1	0	0
1	1	1

SML

```
exception Error;
fun AND(0,_) = 0 | AND(1,x) = x | AND(_,_) = raise Error;
val truthValues2 = [(0,0),(0,1),(1,0),(1,1)];
map AND truthValues2;
```

```
> exception Error;
fun AND(0,_) = 0 | AND(1,x) = x | AND(_,_) = raise Error;
val truthValues2 = [(0,0),(0,1),(1,0),(1,1)];
map AND truthValues2;
exception Error
> val AND = fn: int * int -> int
> val truthValues2 = [(0, 0), (0, 1), (1, 0), (1, 1)]: (int * int) list
> val it = [0, 0, 0, 1]: int list
```



AND gate

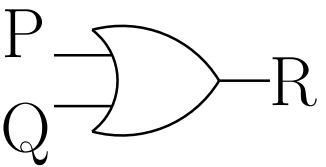
$$R \equiv P \wedge Q$$

P	Q	$P \vee Q$
0	0	0
0	1	1
1	0	1
1	1	1

SML

$exception\ Error;$
$fun\ OR(1,\_) = 1 \mid OR(0,x) = x \mid OR(\_,\_) = raise\ Error;$
$val\ truthValues2 = [(0,0),(0,1),(1,0),(1,1)];$
$map\ OR\ truthValues2;$

```
> exception Error;
fun OR(1,_) = 1 | OR(0,x) = x | OR(_,_) = raise Error;
val truthValues2 = [(0,0),(0,1),(1,0),(1,1)];
map OR truthValues2;
exception Error
> val OR = fn: int * int -> int
> val truthValues2 = [(0, 0), (0, 1), (1, 0), (1, 1)]: (int * int) list
> val it = [0, 1, 1, 1]: int list
```

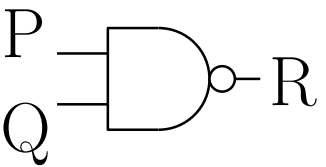


OR gate

$$R \equiv P \vee Q$$

cs113 Lab2: Nand Gate

P	Q	$P \mid Q$
0	0	1
0	1	1
1	0	1
1	1	0



$$R \equiv P \mid Q$$

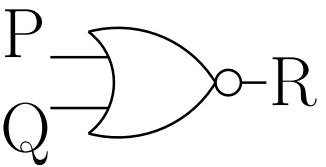
SML

```
exception Error;
fun NAND(0,_) = 1 | NAND(1,x) = NOT(x) | NAND(_,_) = raise Error;
val truthValues2 = [(0,0),(0,1),(1,0),(1,1)];
map NAND truthValues2;
```

```
> exception Error;
fun NAND(0,_) = 1 | NAND(1,x) = NOT(x) | NAND(_,_) = raise Error;
val truthValues2 = [(0,0),(0,1),(1,0),(1,1)];
map NAND truthValues2;
exception Error
> val NAND = fn: int * int -> int
> val truthValues2 = [(0, 0), (0, 1), (1, 0), (1, 1)]: (int * int) list
> val it = [1, 1, 1, 0]: int list
```

cs113 Lab2: Nor Gate

P	Q	$P \downarrow Q$
0	0	1
0	1	0
1	0	0
1	1	0



NOR gate

$$R \equiv P \downarrow Q$$

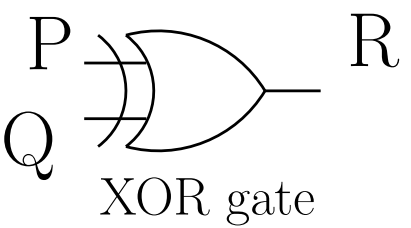
SML

```
exception Error;
fun NOR(1,_) = 0 | NOR(0,x) = NOT(x) | NOR(_,_) = raise Error;
val truthValues2 = [(0,0),(0,1),(1,0),(1,1)];
map NOR truthValues2;
```

```
> exception Error;
fun NOR(1,_) = 0 | NOR(0,x) = NOT(x) | NOR(_,_) = raise Error;
val truthValues2 = [(0,0),(0,1),(1,0),(1,1)];
map NOR truthValues2;
exception Error
> val NOR = fn: int * int -> int
> val truthValues2 = [(0, 0), (0, 1), (1, 0), (1, 1)]: (int * int) list
> val it = [1, 0, 0, 0]: int list
```

cs113 Lab2: Xor Gate

P	Q	$P \oplus Q$
0	0	0
0	1	1
1	0	1
1	1	0



SML

```
exception Error;
fun XOR(0,x) = x | XOR(1,x) = NOT(x) | XOR(_,_) = raise Error;
val truthValues2 = [(0,0),(0,1),(1,0),(1,1)];
map XOR truthValues2;
```

$$R \equiv P \oplus Q$$

```
> exception Error;
fun XOR(0,x) = x | XOR(1,x) = NOT(x) | XOR(_,_) = raise Error;
val truthValues2 = [(0,0),(0,1),(1,0),(1,1)];
map XOR truthValues2;
exception Error
> val XOR = fn: int * int -> int
> val truthValues2 = [(0, 0), (0, 1), (1, 0), (1, 1)]: (int * int) list
> val it = [0, 1, 1, 0]: int list
```

cs113 Lab2: Problem 2.1

- Example 2.2 truth table and circuit

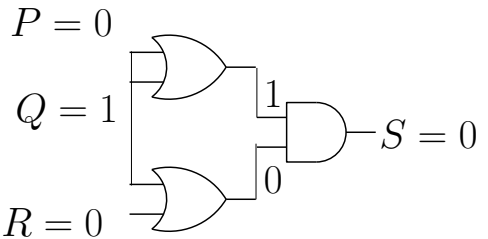
P	Q	R	$((P \vee Q) \wedge (P \vee R))$
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

```
> fun ex22(P,Q,R) = AND(OR(P,Q),OR(P,R));
val truthValues3 =
[(0,0,0),(0,0,1),(0,1,0),(0,1,1),(1,0,0),(1,0,1),(1,1,0),(1,1,1)];
map ex22 truthValues3;
val ex22 = fn: int * int * int -> int
> val truthValues3 =
    [(0, 0, 0), (0, 0, 1), (0, 1, 0), (0, 1, 1), (1, 0, 0), (1, 0, 1),
     (1, 1, 0), (1, 1, 1)]: (int * int * int) list
> val it = [0, 0, 0, 1, 1, 1, 1, 1]: int list
```

$$S \equiv ((P \vee Q) \wedge (P \vee R))$$

SML

```
fun ex22(P,Q,R) = AND(OR(P,Q),OR(P,R));
val truthValues3 = [(0,0,0),(0,0,1),(0,1,0),(0,1,1),
(1,0,0),(1,0,1),(1,1,0),(1,1,1)];
map ex22 truthValues3;
```



cs113 Lab2: Problem 2.2

- Circuit of $((P \wedge Q) \vee \sim R)$

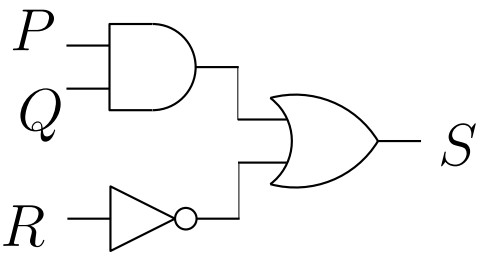
P	Q	R	$((P \wedge Q) \vee \sim R)$
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

```
> fun p22(P,Q,R) = OR(AND(P,Q),NOT(R));
val truthValues3 = [(0,0,0),(0,0,1),(0,1,0),(0,1,1),
(1,0,0),(1,0,1),(1,1,0),(1,1,1)];
map p22 truthValues3;
val p22 = fn: int * int * int -> int
> # val truthValues3 =
   [(0, 0, 0), (0, 0, 1), (0, 1, 0), (0, 1, 1), (1, 0, 0), (1, 0, 1),
    (1, 1, 0), (1, 1, 1)]: (int * int * int) list
> val it = [1, 0, 1, 0, 1, 0, 1, 1]: int list
```

$$S \equiv ((P \wedge Q) \vee \sim R)$$

SML

```
fun p22(P,Q,R) = OR(AND(P,Q),NOT(R));
val truthValues3 = [(0,0,0),(0,0,1),(0,1,0),(0,1,1),
(1,0,0),(1,0,1),(1,1,0),(1,1,1)];
map p22 truthValues3;
```



cs113 Lab2: Problem 2.3

- Match the figure in the Book for Problem 2.3

P	Q	R	S	$(P \wedge Q) \wedge (\sim R)$
1	1	1	0	0
1	1	0	1	1
1	0	1	0	0
1	0	0	0	0
0	1	1	0	0
0	1	0	0	0
0	0	1	0	0
0	0	0	0	0

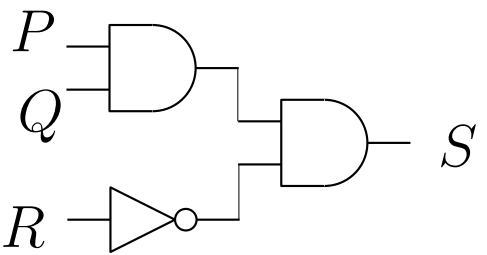
- Since the last two columns are identical (Left from textbook), they are equivalent

```
> fun p23(P,Q,R) = AND(AND(P,Q),NOT(R));
val truthValues3 = [(0,0,0),(0,0,1),(0,1,0),(0,1,1),
(1,0,0),(1,0,1),(1,1,0),(1,1,1)];
map p23 truthValues3;
val p23 = fn: int * int * int -> int
> # val truthValues3 =
    [(0, 0, 0), (0, 0, 1), (0, 1, 0), (0, 1, 1), (1, 0, 0), (1, 0, 1),
     (1, 1, 0), (1, 1, 1)]: (int * int * int) list
> val it = [0, 0, 0, 0, 0, 0, 1, 0]: int list
```

$$S \equiv ((P \wedge Q) \wedge \sim R)$$

SML

```
fun p23(P,Q,R) = AND(AND(P,Q),NOT(R));
val truthValues3 = [(0,0,0),(0,0,1),(0,1,0),(0,1,1),
(1,0,0),(1,0,1),(1,1,0),(1,1,1)];
map p23 truthValues3;
```



cs113 Lab2: Problem 2.7

- Create the Truth Table for the given Circuit for Problem 2.7

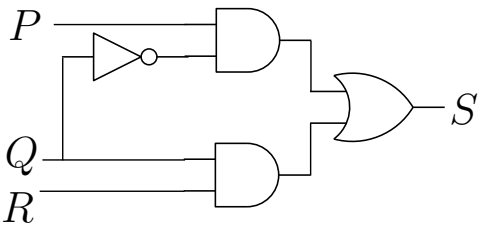
P	Q	R	$(P \wedge \sim Q) \vee (Q \wedge R)$
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

```
> fun p27(P,Q,R) = OR(AND(P,NOT(Q)),AND(Q,R));
val truthValues3 = [(0,0,0),(0,0,1),(0,1,0),(0,1,1),
(1,0,0),(1,0,1),(1,1,0),(1,1,1)];
map p27 truthValues3;
val p27 = fn: int * int * int -> int
> # val truthValues3 =
    [(0, 0, 0), (0, 0, 1), (0, 1, 0), (0, 1, 1), (1, 0, 0), (1, 0, 1),
     (1, 1, 0), (1, 1, 1)]: (int * int * int) list
> val it = [0, 0, 0, 1, 1, 1, 0, 1]: int list
```

$$S \equiv (P \wedge \sim Q) \vee (Q \wedge R)$$

SML

```
fun p27(P,Q,R) = OR(AND(P,NOT(Q)),AND(Q,R));
val truthValues3 = [(0,0,0),(0,0,1),(0,1,0),(0,1,1),
(1,0,0),(1,0,1),(1,1,0),(1,1,1)];
map p27 truthValues3;
```



cs113 Lab2: Problem 2.8

- Create the Truth Table for the exclusive nor for Problem 2.8

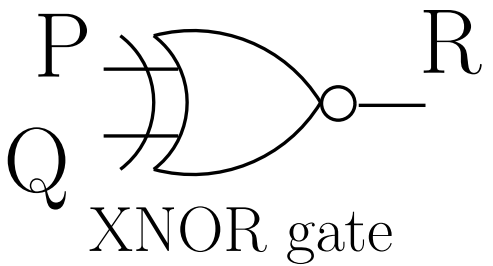
P	Q	$\sim (P \oplus Q)$
0	0	1
0	1	0
1	0	0
1	1	1

```
> fun p28(P,Q) = NOT(XOR(P,Q));
val truthValues2 = [(0,0),(0,1),(1,0),(1,1)];
map p28 truthValues2;
val p28 = fn: int * int -> int
> val truthValues2 = [(0, 0), (0, 1), (1, 0), (1, 1)]: (int * int) list
> val it = [1, 0, 0, 1]: int list
```

SML

```
fun p28(P,Q) = NOT(XOR(P,Q));
val truthValues2 = [(0,0),(0,1),(1,0),(1,1)];
map p28 truthValues2;
```

$S \equiv \sim (P \oplus Q)$



cs113 Lab2: Problem 2.10

- Create the Circuit for the given Boolean Function for Problem 2.10

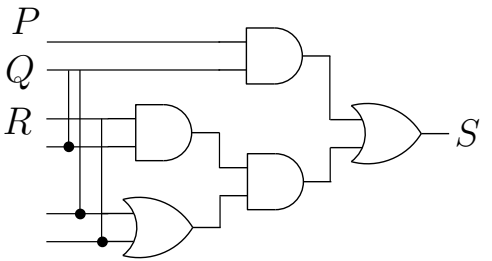
P	Q	R	$((P \wedge Q) \vee ((Q \wedge R) \wedge (Q \vee R)))$
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

```
> fun p210(P,Q,R) = OR(AND(P,Q),AND(AND(Q,R),OR(Q,R)));
val truthValues3 = [(0,0,0),(0,0,1),(0,1,0),(0,1,1),
(1,0,0),(1,0,1),(1,1,0),(1,1,1)];
map p210 truthValues3;
val p210 = fn: int * int * int -> int
> # val truthValues3 =
    [(0, 0, 0), (0, 0, 1), (0, 1, 0), (0, 1, 1), (1, 0, 0), (1, 0, 1),
     (1, 1, 0), (1, 1, 1)]: (int * int * int) list
> val it = [0, 0, 0, 1, 0, 0, 1, 1]: int list
```

$$S \equiv ((P \wedge Q) \vee ((Q \wedge R) \wedge (Q \vee R)))$$

SML

```
fun p210(P,Q,R) = OR(AND(P,Q),AND(AND(Q,R),OR(Q,R)));
val truthValues3 = [(0,0,0),(0,0,1),(0,1,0),(0,1,1),
(1,0,0),(1,0,1),(1,1,0),(1,1,1)];
map p210 truthValues3;
```



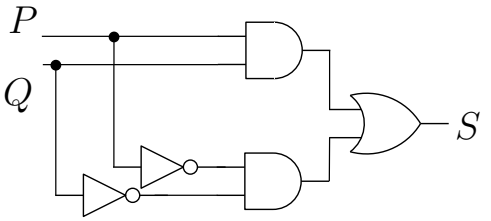
cs113 Lab2: Problem 2.14

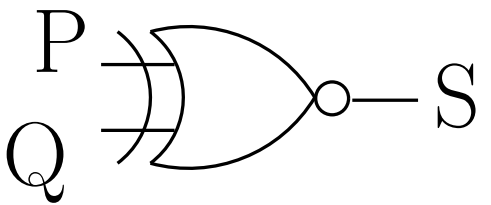
- Prove the two Circuit are equivalent for Problem 2.14

P	Q	$((P \wedge Q) \vee (\sim P \wedge \sim Q))$
0	0	1
0	1	0
1	0	0
1	1	1

P	Q	$\sim (P \oplus Q)$
0	0	1
0	1	0
1	0	0
1	1	1

- Since the last column of both tables are identical the two expressions are equivalent





cs113 Lab2: Problem 2.14 (Cont.)

- Prove the two Circuits are equivalent for Problem 2.14

SML

```
fun p214a(P,Q) = OR(AND(P,Q),AND(NOT(P),NOT(Q)));
val truthValues2 = [(0,0),(0,1),(1,0),(1,1)];
map p214a truthValues2;
```

```
> fun p214a(P,Q) = OR(AND(P,Q),AND(NOT(P),NOT(Q)));
val truthValues2 = [(0,0),(0,1),(1,0),(1,1)];
map p214a truthValues2;
val p214a = fn: int * int -> int
> val truthValues2 = [(0, 0), (0, 1), (1, 0), (1, 1)]: (int * int) list
> val it = [1, 0, 0, 1]: int list
```

$$(P \wedge Q) \vee (\sim P \wedge \sim Q) \equiv \sim (P \oplus Q)$$

```
fun p214b(P,Q) = NOT(XOR(P,Q));
val truthValues2 = [(0,0),(0,1),(1,0),(1,1)];
map p214b truthValues2;
```

```
> fun p214b(P,Q) = NOT(XOR(P,Q));
val truthValues2 = [(0,0),(0,1),(1,0),(1,1)];
map p214b truthValues2;
val p214b = fn: int * int -> int
> val truthValues2 = [(0, 0), (0, 1), (1, 0), (1, 1)]: (int * int) list
> val it = [1, 0, 0, 1]: int list
```